

University *of Ljubljana*
Faculty *of Computer and Information Science*




Zbornik

## Digitalna forenzika

# Seminarske naloge, 2013/14

Ljubljana, 2014

Zbornik
*Digitalna forenzika, Seminarske naloge 2013/2014*

Editors: Andrej Brodnik, Svetlana Nikić, študenti

Ljubljana : Univerza v Ljubljani, Fakulteta za računalništvo in informatiko, 2014.

© 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.

Digitalna forenzika je znanost, katere naloga je zbiranje, shranjevanje, iskanje, analiza in dokumentacija digitalnih dokazov, prispevala pa je tudi k varnejši digitalni tehnologiji. Čeprav je digitalna forenzika pravzaprav le veja forenzične znanosti, vseeno pokriva ogromno področje, kamor med drugim spadajo računalniška forenzika, forenzika mobilnih naprav, omrežna forenzika in forenzika podatkovnih baz. V okviru predmeta Digitalna forenzika je vsak študent sodeloval pri izdelavi seminarske naloge izbrane tematike, ki spada pod eno izmed prej omenjenih vej forenzike. Vsaka seminarska naloga predstavlja del znanja, ki ga je prispevala skupina študentov, delo vseh študentov pa je združeno v tem zborniku.

Delo Petra Dolenca in Gašperja Žgajnarja se osredotoča na poškodovane DEFLATE datoteke, njihovo obnavljanje in rekonstrukcijo.
Rok Bajec in Jan Robas opisujeta kako klasificirati delčke datotek z uporabo empiričnega pristopa.
V članku Anžeta Rezlja in Janeza Bindasa je opisano delovanje diskov SSD in njihov negativen vpliv na digitalno forenziko.
Članek avtorjev Tomaža Tomažiča in Žige Zupanca govori o tehnologiji IP in o analizi podatkov, ki so pridobljeni s klici z uporabo omenjene tehnologije.
Tjaša Saje, Katerina Bashova in Žan Anderle so se v svojem delu osredotočili na Dropbox, ter na podatke, ki po uporabi ostanejo na uporabnikovem računalniku s stališča forenzike.
Članek Mihe Mohorčiča, Roka Povšiča in Marka Škrjanca predstavi, kako preprečiti nepooblaščeno rabo mobilnega telefona s pomočjo biometričnih podatkov in s tem povečati varnost mobilnih telefonov.
Sandi Šemrov in Saša Makorič sta se osredotočila na vprašanje, kako ponarediti SMS sporočilo ter kako ugotoviti, če je sporočilo na mobilni napravi ponarejeno.
V prispevku Grege Gašperšiča in Karmen Bezlaj so opisane metode za učinkovito izsledite naslova IP napadalca v primeru napada DoS.
Tadej Jagodnik in Jan Češnjevar sta predstavila metode za vzorčenje omrežnega prometa in algoritme za detekcijo anomalij v omrežju.
Avtorji Leon Noe Jovan, David Novak in Dejan Petrović v svojem članku opisujejo delovanje botnetov in uporabo samoozdravljivih sistemov kot učinkovito zaščito pred njimi.
Delo Roka Gomiščka, Igorja Lalića in Jona Premika predstavi vplivna vozlišča v kompleksnih omrežjih in prikaže, kako jih odkriti.
Avtorji Jovan Buragev, Branislav Todorov in Bojana Todorović v svojem delu prikažejo, kako se zaščititi pred vdori, ki so izvedeni preko omrežja.
Članek Tomaža Borštnika, Darka Božidarja in Gregorja Čepina ponuja vpogled v postopek forenzične analize omrežja in najpogostejše oblike zlonamernih programskih oprem.

V delu Nejca Sajeta, Nejca Severja in Anžeta Sodje je povzeto delovanje, implementacija in analiza forenzičnega orodja FROST, ki je uporabljeno v platformi OpenStack.
Peter Miklavčič v svojem članku obravnava forenzično analizo brezžičnih lokalnih omrežij in njihove ranljivosti.
Nejc Bizjak in Dean Črnigoj sta predstavila, kako rekonstruirati poizvedbe iz dnevniških datotek podatkovnega sistema InnoDB.
Avtorja Blaž Meden in Tine Mislej v svojem prispevku prikažeta forenzične sledi, ki nastanejo pri uporabi brskalnikov v zasebnem načinu.
V članku Zlatka Hrnčića in Blaža Jeršana je opisan postopek anti-forenzike oziroma uničevanja dokazov na mobilnih napravah.
Prispevek Mateja Andolška, Andraža Pajtlerja in Antona Semprimožnika opisuje prototip poslovnega sistema za nadzor pametnih mobilnih telefonov.
V delu avtorjev Andraža Gregorčiča in Jerneja Grosarja so predstavljenje faze digitalnega preiskovanja in tehnike za zajem pomnilniške slike.
Avtorja Gregor Cimerman in Vlada Semenova v svojem članku ponudita vpogled v postopek forenzične preiskave pri brodolomu potniške ladje Costa Concordia.
Delo Tomaža Bartola, Jerneja Jerina in Tadeja Vodopivca se sooča z analizo digitalnih oseb v preiskavah spletnega kriminala.
V prispevku Matevža Černeta in Klemna Marolta je opisan postopek, kako oceniti velikost človeka na podlagi posnetka nadzorne kamere.
Članek Svetlane Nikić, Anžeta Škerjanca in Nejca Škerjanca predstavi koncept triaže podatkov in nov model za selektivno iskanje digitalnih dokazov.

Zaradi njene obširnosti se je pri digitalni forenziki večkrat potrebno nasloniti na delo drugih. Ta zbornik je dobro izhodišče za vse, ki jih zanima digitalna forenzika in tiste, ki se z njo ukvarjajo pa jih bolj podrobno zanima katera izmed izbranih tematik.

Luka Golinar

# Abstracts

1. **Izpopolnjeno obnavljanje in rekonstrukcija poškodovanih DEFLATE datotek**

This article is about recovering compressed files which were compressed with the algorithm DEFLATE. It explains how this algorithm works and how the damaged compressed files can be reconstructed. The article shows through example how text data can be reconstructed by investigators when the DEFLATE algorithm is used. In the last part the authors talk about ZipRec tool which is one of the tools that use this algorithm.

2. **Klasifikacija delčkov datotek**

The article explains investigation on damaged files where forensic investigators need to find out the type or content of the file. Damaged file is a file that is missing some parts (files without their signature, name, end of the file etc.). This article explains the evaluation of this investigation and what kind of ideas experts give on finding some patterns matching the files if there is only start part, middle part or end part of the file. The architecture of some files and their most significant properties are also explained.

3. **SSD: Začetek konca trenutne prakse v digitalni forenziki?**

Forensic investigation on SSD disk is the main theme of the article. How the old methods for investigation on HDD disks are not completely relevant for the new "SSD" disk. It explains how SSD works and what the differences from HDD are. At the end we can read about some tests on SSD disks. The results show how SSD works on different operation systems and how successful the forensic investigations on this type of disks are.

4. **Analiza podatkov VoIP klicev**

This article is about Voice over Internet Protocol (VoIP). The article begins with overall picture of VoIP, what it is and what this protocol is made for. How VoIP works is explained in more detail and what are advantages and disadvantages of this kind of communication. The authors explain how Skype works since it is one of the most famous applications that use VoIP communications.

5. **Dropbox analysis: Data remnants on user machines**

Quick & Choo's research of data remnants on cloud storage is the base for this article. By setting up experimentation on Virtual Machines this article shows the investigators what kind of tools are the most useful in certain conditions. The whole idea is how to investigate cloud storage, how to find out its owner and how to recover data from it.

6. **Prevent unauthorized use of mobile phone using biometric data**
   **Analysis of the use of biometric systems on mobile devices**

Today almost every person owns a mobile device. The modern technology and existence of smart phones allows us to use them for many different purposes and we can store important data on mobile devices. This kind of important data should be somehow secured. Biometric, hardware security and combination of them is used for securing the devices from undesirable intrusions and malicious usage of the important data. This kind of security is described in the main part of this

article. From the biometric aspect the authors mentioned face, fingerprint, body, eyes and voice recognition. From the hardware aspect we have the unique chargers for mobile devices and RFID. Combination of these two kinds of security produces the two methods of security: ePet and BioID. In the second part of the article we find interesting research by the authors about face recognition and its security level. The article ends with forensic examination of mobile devices and existence of data that can be important in forensic investigations.

**7. Falsifying the messages SMS**

Investigation about falsifying SMS messages was conducted through real research by the authors of the article. Two devices were used in research: Nokia 6021 and Android Smartphone along with other software and hardware equipment designed for falsifying SMS messages. On each device tests were made both within the limits of the device, and outside the limits of device using this equipment. At the end of the article we find the elements that needed to be checked by the investigators to find out if the SMS message is falsified or not.

**8. Analysis of systems for tracing an IP address**

DoS and DDos intrusions are very often today. To detect the source of the attack the investigators should use some advanced methods. It is well known that every device operating in the global network (internet) must have its own IP address, so the best way to detect the intruder is to find out their IP address and trace them. Tracing is described trough three views: basic principles of tracing, processing methods and location (Source group method and Network group method). Two methods are presented: DMP and DFM, how they work, what are their main advantages, their analysis and the results of their usage.

**9. The impact of sampling on algorithms for detecting anomalies in the network traffic**

Sampling data transferred trough network is used to reduce the processing requirements of network devices. In this article the impact of data sampling on algorithms for detecting anomalies in network traffic is researched including how it impacts detection of malicious activity through the network. Intrusion detection systems use different kind of algorithms to detect the anomalies in network traffic and this article is focused on Threshold Random Walk (TRW), Time Access Pattern Scheme (TAPS) and Entropy-Based Scan Detection. Techniques of sampling are described and analyses of algorithms for anomalies detection are made. Based on research and comparison of the algorithms the conclusions are given. The main conclusion leads to the main point of the article and that is that the sampling of data has very big influence on working of algorithms.

**10. Protection against botnets**

BOTNET, a network of private computers infected with malicious software and controlled as a group without the owner's knowledge is the main topic of discussion. Its usage and topology are described. Milestones through years are listed influenced by known botnets like Torpig, Virut, Zeus, Storm and many others. Protection against this kind of intrusions should be present and known in today's networks. A possible answer to advanced botnets today is self-healing system – a way of presenting computer systems and network environment in which they operate. Self-healing systems have their own architecture and a way of working against this kind of intrusions.

**11. Detecting Influential Spreaders in Complex, Dynamic Networks**

The main point of this article are the influential nodes in complex dynamic networks like today's social networks Facebook, Linkedin, well known news portals and many others. How this nodes influence other nodes, how they can be used for malicious activities and how to find the most influential nodes are the main focus of the article. Two methods are presented with algorithms and explanations on how to find those influential nodes in networks. Second part of the article presents practical experiment which tests the techniques of the presented methods.

**12. Network intrusion investigation – preparation and challenges**

Network intrusion can be defined as unauthorized intrusion into network where one or more devices are operating. Most of the intrusions have malicious intent under systems in network and intruders are motivated for some reason. Owners and administrators of computer networks are faced with this problem and this article introduces the types of attacks, how to prevent the system from intrusions, finding evidence and fingerprints if some intrusion happened in network and good practices to keep the evidence admissible in front of the court.

**13. Procedure of network forensic analysis**

This article is based on Zurich University's research which was conducted by experts in computer security field. The description of malicious software in the beginning gives us clear vision of types of malicious software that produce different types of data that can be inspected. The research is based on analysis of traffic in custom made network system. Snort, Nessus and OpenVas are the intrusion detection systems which are used in the research and each system is shortly described. Along these, two more procedures have been used: checking the users black lists of the network and Google search engine. Whole network was scanned with these systems which resulted in reports and improvements about vulnerability of the network. Four most known malicious software types are described from the incidents and reports of the researchers. In the last two parts we find out more about characteristics of good data package signatures and how to recognize the malicious software.

**14. Forensics theory in cloud computing and FROST system in practices**

Idea of this article is cloud computing presented through the OpenStack platform and use of FROST system as a tool for forensic analysis for cloud platform. FROST is described through its requirements and specifications, architecture and implementation. Its weak sides are also presented and assessments of the FROST operations with all data that could be provided for forensic investigation are given. Before passing to the forensic part of FROST we can read about analysis of some related scientific articles. Last part of the article is related to the cloud forensics. Little introduction is given at first and then the services that can be used for investigation and information about gaining needed data is described.

**15. Wireless Local Area Network Forensics**

Nowadays wireless data traffic is used more and more in everyday life. At the beginning this article talks about weak and strong security mechanisms of wireless networks. On the weak side we have Hiding SSID, Filtering MAC and IP addresses, WEP protection of data traffic and WIFI protected setups and among strong mechanisms we have WPA2 protection and EAP. As most used home device for sharing wireless device router is analyzed, how it's functioning and its main architecture. We also find out what kind of data router keeps that can be important for further forensic investigation.

16. **Rekonstrukcija manipuliranih poizvedb iz InnoDB Redo Log datotek**

Investigation of database transactions is the main topic of this article. In the field of database management systems, InnoDB is a useful storage engine built for MySQL. Manipulation of the database can cause more damage than imagined. Attackers try to leave as little trace as possible behind them which is why RedoLog file is their target. InnoDB keeps log of all changes made in the database in Redo log file and by manipulating this file attackers are able to hide their traces. This kind of attacks are explained in the article, how forensic investigators examine the evidence from Redo log file, how they recover already manipulated data from it and how they find the changes made in the database.

17. **Forenzične sledi pri uporabi brskalnikov v zasebnem ali prenosnem načinu**

This article talks about browsers and what kind of attacks can be made through them. The authors analyze the most famous browsers; they explain how they work (browsing) and what are the advantages and disadvantages from forensic investigation's point of view. This analysis also includes portable browsers, and comparison with the regular ones.

18. **Android anti-forensics through a local paradigm**

The main topic of the article is Android mobile device examination inflicted by anti-forensic techniques. Anti-forensic techniques are techniques trying to make investigation unsuccessful, or more precisely shadowing the evidence to not be found by the investigators. This can make the investigation very hard especially on mobile devices, where this kind of technique is new and not well known to forensic investigators. In the article some anti-forensic tools and techniques on Android mobile devices are described including how Android mobile devices can be investigated.

19. **Android forensics**

This article gives us an overall picture of examining Android phones. It also describes some of the tools for monitoring Android smart phones. The article focuses on DroidWatch tool, which automatically captures data from Android smart phone and makes collecting data from it much easier which helps forensic investigators.

20. **Platforma za ocenjevanje forenzičnih orodji za kreiranje slike delovnega pomnilnika**

This article explains the process of digital investigation in forensics: explaining how the process works, showing some techniques for making memory image from computer and working on that image. Bochs x86 emulator is used for testing memory image taken from the computer. In the end of the article the authors describe other software solutions and hardware ways of capturing memory image.

21. **Brodolom potniške ladije Costa Concordia**

The article talks about VDR (Voyage Data Recorder) systems which play an important role in forensic investigations after shipwreck.

The most common causes for accident that can lead to shipwreck are weather conditions but for "Costa Concordia" those did not play a role.

In the first part of the article, the authors explain what VDR is, what it stands for and what kind of information it collects. This part of the article also contains an explanation of when and by whom VDR systems were invented.

In the second part of the article the VDR system and the seven parts which are plugged into it are explained in more detail. They explain how VDR collects the data from the navigation bridge and what kind of problems can occur with older cruisers on this field. We find out what NMEA (National Marine Electronics Association) data format is through one general example and one from "Costa Concordia" case. In the last part of this section they talk about problems in forensic examination while investigating VDR systems, and what kind of problems investigators had before IMO (International Marine Organization). Since the VDR from "Costa Concordia" was found in perfect condition and the data was not damaged, the authors added a counter example "Maersk Kendal" where significant data and evidence were destroyed.

The last part talks about forensic investigation in "Costa Concordia", giving us information about VDR system that was found, what kind of data was used by the system and how the data was examined by forensic investigators.

**22. Kdo sem jaz? Analiza digitalnih oseb v preiskavah spletnega kriminala**

This article is about Isis tool, a tool for recognizing digital personality. We can see that nowadays almost everything that happens in real world can also happen in digital world. Online bullying is just one of those cases. Furthermore some techniques for detecting organizations which act from hidden networks are described.

**23. Assessment of the size of humans on the basis of track surveillance cameras**

Height of a suspect in investigation has a major role in finding his identity. Today investigators have many methods of extracting the height of a person. In this article main topic is assessment of the size of person based on surveillance cameras. Two methods are analyzed, SVM and QTM. Through the method's research it is shown how the body posture and way of moving of the person are affecting the results of the algorithms.

**24. Triage and a new model for digital selective search evidence**

In digital forensics many devices and tools can be used to obtain evidence and some model for standardization is needed. Triage is used mainly to intensify the process of investigation through limiting objects that are used. Through limitation negative sides appear and new triage model is needed. Through analysis of proposed triage model we are given a clear picture of what the idea of triage model is. Based on research, authors made some corrections of the proposed approach with ideas how to improve it. It's difficult to have strong defined standardization of triage model because of different laws in different countries and many other factors. But it is good to have some procedure in process of investigating devices.

Branislav Todorov, Jovan Buragev

# Kazalo

**Digitalna forenzika na diskih** ..... 1

Izpopolnjeno obnavljanje in rekonstrukcija poškodovanih DEFLATE datotek ..... 2
*Peter Dolenc, Gašper Žgajnar*

Klasifikacija delčkov datotek ..... 9
*Rok Bajec, Jan Robas*

SSD: Začetek konca trenutne prakse v digitalni forenziki? ..... 16
*Anže Rezelj, Janez Bindas*

Analiza podatkov VoIP klicev ..... 27
*Tomaž Tomažič, Žiga Zupanec*

Dropbox analysis: Data remnants on user machines ..... 35
*Tjaša Saje, Katerina Bashova, Žan Anderle*

**Digitalna forenzika na omrežjih** ..... 42

Preprečevanje nepooblaščene rabe mobilnega telefona s pomočjo biometričnih podatkov ..... 43
*Miha Mohorčič, Rok Povšič, Marko Škrjanec*

Ponarejanje sporočil SMS ..... 52
*Sandi Šemrov, Saša Makorić*

Analiza sistemov za izsledite IP naslova ..... 60
*Grega Gašperšič, Karmen Bezlaj*

Vpliv vzorčenja na algoritme za detekcijo anomalij v omrežnem prometu ..... 68
*Tadej Jagodnik, Jan Češnjevar*

Zaščita pred botneti ..... 75
*Leon Noe Jovan, David Novak, Dejan Petrović*

Detecting Influential Spreaders in Complex, Dynamic Networks ..... 82
*Jon Premik, Igor Lalilć, Rok Gomišček*

Network intrusion investigation – Preparation and Challenges ..... 86
*Jovan Buragev, Branislav Todorov*

Postopek forenzične analize omrežja ..... 100
*Gregor Čepin, Darko Božidar, Tomaž Borštnik*

Forenzika v oblačnem računalništvu v teoriji in praksi s sistemom FROST ..... 108
*Nejc Sever, Nejc Saje, Anže Sodja*

Forenzika brezžičnih lokalnih omrežij ..... 116
*Peter Miklavčič*

**Digitalna forenzika na sistemih** ..... 122

Rekonstrukcija manipuliranih poizvedb iz InnoDB Redo Log datotek ..... 123
*Nejc Bizjak, Dean Črnigoj*

Forenzične sledi pri uporabi brskalnikov v zasebnem ali prenosnem načinu ..... 130
*Blaž Meden, Tine Mislej*

Android anti-forensics through a local paradigm ..... 139
*Blaž Jeršan, Zlatko Hrnčić*

Android forensics ..... 145
*Andraž Pajtler, Anton Semprimožnik, Matej Andolšek*

Platforma za ocenjevanje forenzičnih orodji za kreiranje slike delovnega pomnilnika ..... 154
*Jernej Grosar, Andraž Gregorčič*

**Druga področja digitalne forenzike** ..... 164

Brodolom potniške ladije Costa Concordia ..... 165
*Gregor Cimerman, Vlada Semenova*

Kdo sem jaz? Analiza digitalnih oseb v preiskavah spletnega kriminala ..... 172
*Jernej Jerin, Tomaž Bartol, Tadej Vodopivec*

Ocenjevanje velikosti človeka na podlagi posnetka nadzorne kamere ..... 181
*Matevž Černe, Klemen Marolt*

Triaža in nov model za selektivno iskanje digitalnih dokazov ..... 188
*Nejc Škerjanc, Anže Škerjanc, Svetlana Nikić*

# Digitalna forenzika na diskih

# Izpopolnjeno obnavljanje in rekonstrukcija poškodovanih DEFLATE datotek

Peter Dolenc
Fakulteta za računalništvo in informatiko
Tržaška cesta 25
Ljubljana, Slovenija
pe.dolenc@gmail.com

Gašper Žgajnar
Fakulteta za računalništvo in informatiko
Tržaška cesta 25
Ljubljana, Slovenija
gzgajnar@gmail.com

## ABSTRACT
V tem članku so predstavljeni pristopi k rekonstrukciji poškodovanega arhiva, ki je bil stisnjen z uporabo algoritma DEFLATE. Rekonstrukcija obsega rekonstrukcijo, ki jo je možno narediti s poznavanjem algoritma DEFLATE. Tudi delovanej algortima DEFLATE, ki se sicer najpogosteje uporablja za stiskanje datotek je predstavljeno. Nadaljna rekonstrukcija obsega rekonstrukcijo s pomočjo znanja o samem jeziku (za tekstovne datoteke). Predstavljeno je tudi orodje ki implementira omenjene algoritme za rekonstrukcijo.

## Categories and Subject Descriptors
E. 5 [**Data**]: Backup/recovery

## General Terms
Algorithms, Languages

## Keywords
DEFLATE, ZipRec, Huffman, recovery, reconstruction, archive

## 1. UVOD
V tem članku so predstavljeni pristopi k rekonstrukciji poškodovanega arhiva. Obravnava se tipičen primer iz digitalne forenzike, ko imamo opravka z arhivom, katerega delca nismo mogli pridobiti iz trdega diska (na primer zaradi fragmentacije). Osnova za pristope je bil članek *Improved recovery and reconstruction of DEFLATEd files* [1], v katerem se obravnava več različnih korakov rekonstrukcije takega arhiva. Začetni koraki rekonstrukcije v večini temeljijo na poznavanju delovanja najbolj uporabljenega algoritma za stiskanje datotek DEFLATE. Če poznamo kako so stisnjeni podatki shranjeni lahko omilimo posledice izgube dela arhiva. Zato je v prvem poglavju predstavljen tudi sam algoritem DEFLATE. Končni korak rekonstrukcije pa se osredotoča na rekonstrukcijo tekstovnih datotek. Z uporabo znanj lingvistike, poznavanja jezika teksta in uporabo slovarjev lahko rekonstrukcijo pripeljemo še korak dlje in z dokaj zanesljivo verjetnostjo rekonstruiramo poškodovane dele besedila. Na koncu članka je predstavljeno tudi orodje, ki implementira vse opisane algoritme za rekonstrukcijo.

## 2. ALGORITEM DEFLATE
DEFLATE [3] (PKWare Inc, 2012) je algoritem za stiskanje podatkov in deluje kot kombinacija algoritma LZ77 in Huffmanovega kodiranja. Zgoščevanje poteka v dveh korakih. Najprej se podvojeni nizi zamenjajo s kazalci na referenčni niz, nato pa se vsak simbol zamenja z uteženim glede na pogostost uporabe.

DEFLATE tok podatkov je sestavljen iz množice zaporednih blokov. Vsak se prične s 3-bitno glavo, ki določa eno izmed treh možnih zgoščevalnih oblik [5]:

- Najprej LZ77 in nato Huffmanovo kodiranje z uporabo statičnih dreves (oblika drevesa je splošno določena in znana zgoščevalniku (ang. compressor) in raztezalniku (ang. decompressor); drevesa ni potrebno podati podatkom)
- Najprej LZ77 in nato Huffmanovo kodiranje z uporabo dinamičnih dreves (oblika drevesa je odvisna od podatkov in izračunanih entropij; drevesi je potrebno podati podatkom)
- Brez zgoščevanja; uporablja se pri že stisnjenih podatkih. Pri ponovnem stiskanju bi se količina podatkov dejansko povečala, tako pa se razširijo minimalno - le za dolžino glave bloka. Dolžina bloka je omejena na 64 kB.

Če želimo zamenjati način zgoščevanja, je potrebno trenutni blok zaključiti in nov naačin zgoščevanja pričeti z novim.

### 2.1 Zgoščevanje LZ77
Zgoščevanje LZ77 je algoritem za brezizgubno stiskanje podatkov, ki temelji na uporabi slovarja. Algoritem išče zaporedja podatkov, ki se ponavljajo. Pri LZ77 slovar predstavlja drseče okno velikosti 32 kB, kar pomeni, da se stiskalnik (ang. compressor) in raztezalnik (ang. decompressor) na vsaki točki v podatkih sklicujeta na znake iz predhodnik 32768 bitov. Ko se tako najde znak ali zaporedje znakov, ki so enaki nizu v drsečem oknu, se to zaporedje zamenja z dvema številoma - dolžino zaporedja in razdaljo znotraj drsečega okna, kjer se nahaja to zaporedje [6].

## 2.2 Huffmanovo kodiranje

Huffmanovo kodiranje je algoritem za brezizgubno stiskanje podatkov, ki za kodiranje izkorišča entropijo. Ideja algoritma temelji na pogostosti uporabe posameznega elementa iz določene abecede. Pogostejši naj uporabljajo krajše kode od manj pogostih elementov.

Algoritem najprej dodeli vsakemu elementu utež, ki je fiksno določena, če se uporablja statična drevesa, ali pa se jo izračuna (v primeru dinamičnih dreves). V splošnem pa to število predstavlja relativno frekvenco pojavljanja elementa znotraj podatkov.

Nato se zgradi binarno drevo. Izmed vseh elementov, izberemo dva z najmanjšo utežjo in ju dodamo v drevo (v levi in desni list). Nato ju združimo, seštejemo njuni uteži in ju vrnemo med preostale elemente. Postopek ponavljamo dokler ne združimo vseh elementov v drevo.

Klasična oblika Huffmanovega kodiranja lahko generira različne oblike dreves pri istih vhodnih podatkih. Zato DEFLATE uporablja Huffmana z dvema dodatnima praviloma:

1. Elementi s krajšo kodo se nahajajo v levih listih
2. Če sta elementa enake dolžine, je v levem listu tisti, ki se nahaja prej po abecednem redu.

Pri danih podatkih in s pripadajočimi dolžinami kodov, je z upoštevanjem teh dveh pravil tako možno zgraditi le eno obliko drevesa. Za rekonstrukcijo drevesa je tako dovolj, da pošljemo le dolžine kodov.

DEFLATE uporablja dve različni Huffmanovi drevesi. Prvo je sestavljeno iz podatkov in dolžin ponavljajočih sekvenc. Drugo pa je namenjeno razdaljam znotraj drsnega okna (ang. sliding windows). Kombinacije podatkov in dolžin sekvenc se v istem drevesu uporablja, ker se vsaka koda prične z dejanskim simbolom ali pa z dolžino sekvence. Če dekodirnik odkrije podatek, ga zapiše v izhodni niz. Če pa se pojavi koda za dolžino sekvence, ji nato vedno sledi še razdalja in iz njiju se nato izračuna izhodni niz.

Dolžina ponavljajoče sekvence je lahko velikosti od 3 do 258 bajtov. Največja dolžina drsečega okna je 32 kB. Kljub temu da so podatki strukturirani v bloke, se lahko naslavlja ponavljajoče nize med različnimi bloki, vse dokler je razdalja krajša od 32768.

Če je napaka v zakodiranem nizu podatkov, se pri raztezanju lahko zgodi, da se bajt razpakira v napačen znak ali pa je skopirano napačno zaporedje znakov iz drsečega okna. V obeh primerih se napaka prenaša naprej.

# 3. ZIPREC

Orodje za obnavljanje podatkov iz poškodovanih ZIP arhivov in nizov podatkov zapakiranih z algoritmom DEFLATE.

ZipRec 1.0 [8] avtomatsko zazna napako v nizu le v primeru, ko je vsaj 128 zaporednih enakih bajtov; verjetnost, da pride do takega niza pa je zelo majhna. Narejenih je bilo že veliko različnih načinov za detekicjo napake v splošnem, po večini neuspešno. Razvija pa se tudi uporabniški vmesnik, ki bo omogoča ročno označevanje poškodovanih območji.

Vsi opisani postopki za rekonstrukcijo poškodovanih DEFLATE datotek, so implementirani v verziji 1.0.

# 4. POVEZANA DELA

Čeprav je DEFLATE zelo razširjen in se uporablja v ZIP arhivih, večini dokumentov temelječih na XML (kot so npr. docx, pptx, OpenDocument, OOXML, EPUB ...), na omrežnih protokolih, je bilo storjenih malo raziskav o obnovi podatkov v primeru, ko je poškodovan tok zapakirahni podatkov.

Obstajajo orodja, ki znajo razpakirati nepoškodovane dele arhivov, a redki poskušajo popraviti in razpakirati poškodovane. Programsko orodje Gzip Recovery Toolkit skuša preskočiti poškodovane predele s tehniko skeniranja posameznih bajtov vse dokler ne najde zlib koda, ki ga lahko uspešno razpakira. Ta način deluje s katero koli DEFLATE datoteko, ampak potrebuje vstavljanje sinhronizacijskih točk (značilnih za zlib) med stiskanjem podatkov.

Park et al. (2008) je razvil orodje, ki omogoča obnovo paketov, ki za stiskanje uporabljajo statično Huffmanovo tabelo in ne vsebujejo nizov, ki kažejo na neznane bajte pred tem paketom. Pred tem je razvil tudi orodje za obnovo delnih datotek z uporabo hitrejšega povratnega skeniranja za iskanje prvih nepoškodovanih DEFLATE paketov v datoteki. Dodal je tudi metodo za obnovo temelječo na besednih unigramih in bajtnih trigramih. Ker algoritem med iskanjem veljavnih paketov ne omejuje sklicevanja na neznane bajte, je bila velika večina paketov s statičnimi Huffmanovimi tabelami spuščenih.

# 5. REKONSTRUKCIJA PRIZADETEGA ARHIVA

## 5.1 Odkrivanje napake

Ko pride do napake znotraj paketa, je raztezalniku (ang. decompressor) še vedno na voljo velika količina uporabnih informacij. Najpomembnejši sta Huffmanovi drevesi in kazalci na ponavljajoče nize. Z uporabo Huffmanovih dveh dreves lahko niz bitov razdelimo na posamezne Huffmanove simbole, na kateri koli točki, kjer se prejšnji simbol konča. Če se raztezanje prične na sredi datoteke, je vsebina okna zgodovine neznana, zato je potrebno vse bajte, ki se kažejo na to okno, označiti kot neznane. Ko vemo katera obomočja bajtov v nizu DEFLATE so napačna, raztezalnik lahko spusti te predele in okno zgodovine počisti pred ponovnim zagonom raztezalnika. Na žalost pa je težko zaznati obomočja kjer je prišlo do napake. Običajno raztezalnik niti ne ugotovi, da je prišlo do napake, vse dokler se raztezanje ne zaključi in se izračuna napačna CRC vrednsot. Med preizkušanjem je le v 4% bila odkrita napaka v nizu podatkov.

Ko se odkrije mesto napake, obstaja težava kje ponovno začeti z raztezanjem. V splošnem za vsakim bitom obstaja možnost, da je nadaljevanje toka veljavno. Ker je vsak Huffmanov kod lahko dolg največ 48 bitov, tako obstaja 48 različnih začetnih točk, od katerih pa velika večina privede do napačnega izhodnega toka bajtov. Da lahko najdemo pravo mesto, je potrebno upoštevati lastnost Huffmanovega koda,

```
Bitstream:   1 0 1 0 1 1 0 1 1 1 0 0 1 1 ..
Candidate 1: 1-0
Candidate 2:   0-1-0-1 1-0 1-1-1-0
Candidate 3:     1-0 1-1-0
Candidate 4:       0-1-1 0-1-1 1-0
Resynchronization:                 0 1 1 ..
```

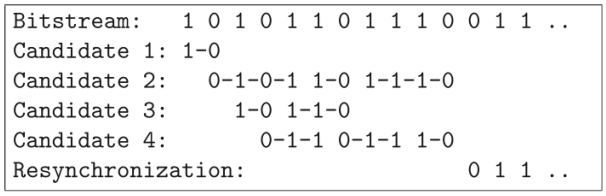


**Figure 1: Resinhronizacija niza DEFLATE po napaki.**

ki je zasnovan tako, da enolično razdeli niz bitov. Tako poizkušamo podatke razpakirati od izbranega mesta z odmiki od 0 do 48. Pravilna točka resinhronizacije je tista, kjer se izhod raztezanja ponovi oziroma to pomeni da smo začeli raztezanje prav na začetku zakodiranega znaka. Od te točke naprej so vse nadaljne razširitve pravilne. Mediana pri konvergenci je manjša od 50 bajtov in med testiranji ni nikoli presegla 150 bajtov.

Potem, ko se ugotovita začetek napake in resinhronizacijska točka (ang. resynchronization point), razširjanje poteka v dveh korakih. Najprej se razpakirajo podatki do mesta napake, nato se izpusti vse bite do resinhronizacijske točke, kjer se premično okno posodobi – mesta s poškodovanimi biti se označijo kot neznana. Od tu naprej se nadaljuje normalno razširjanje.

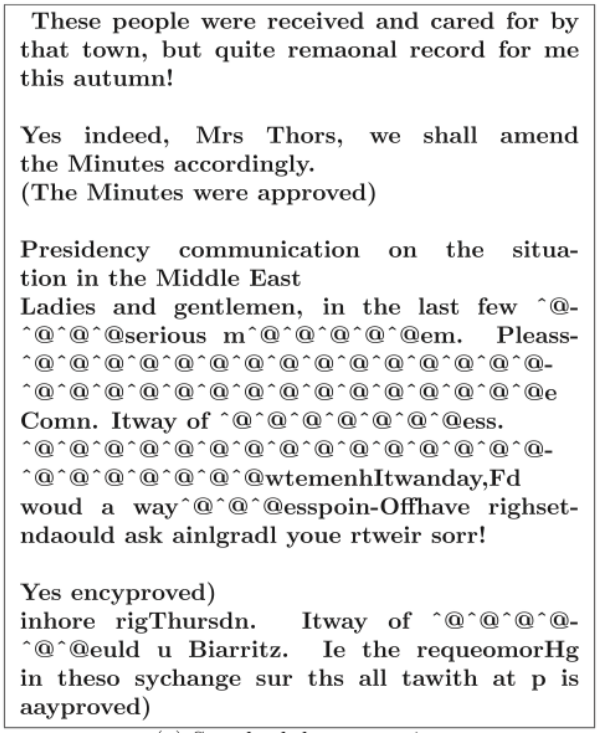
Ker algoritem LZ77 uporablja okno zgodovine za stiskanje večjih ponavljajočih se segmentov tako, da v stisnjenem bloku navedemo referenco na okno zgodovine, to pomeni da lahko tudi nadaljnji bloki vsebujejo referenco na poškodovano območje kar bi privedlo do vključevanja nepravilnih podatkov v razširjeno datoteko. Zato je pomembno da se poškodovano območje označi in da se zamenjava ponavljajočih segmentov ne izvrši dokler rekonstrukcija poškodovanega območja ni končana.

Slika 2 prikazuje rezultat razširjanja datoteke z uporabo Info-ZIP projekta "unzip"verzije 6.00. Napačni NUL bajti v zapakirani datoteki se razširijo kot vrednost $\hat{@}$, ker se kopirajo iz napačnih mest v zaporedju, s tem pa se ustvarijo tudi novi, nepravilni simboli. V tem primeru se kopirajo tudi biti, ki se nahajajo celo pred začetkom datoteke, kar predstavlja jasen znak, da zaporedje vsebuje napako. Če bi bila datoteka poškodovana na mestu bolj oddaljenim od začetka, napaka ne bi bila tako jasna.

Slika 3 prikazuje rezultat razširjanja iste datoteke z uporabo orodja ZipRec 1.0. Ker ve, da je bila napaka, se lahko poškodovanemu delu izogne in identificira tiste bajte, ki so rezultat nedoločenih kopij. Vendar pa s tem pridemo do velikega števila neznanih bajtov, kar zmanjšuje uporabnost rezultata.

## 5.2 Rekonstrukcija podatkov

Pri rekonstrukciji podatkov lahko izkoristimo dejstvo, da ima lahko vsak neznan bajt v izhodnem nizu povezavo na po večkrat ponovljen niz. Tako je za določene skupine bajtov znano, da so enake, kar v povezavi s kontekstualnimi namigi iz posameznih jezikovnih modelov, omogoča sklepanje njihovih pravih vrednosti.



**These people were received and cared for by that town, but quite remaonal record for me this autumn!**

**Yes indeed, Mrs Thors, we shall amend the Minutes accordingly.**
**(The Minutes were approved)**

**Presidency communication on the situation in the Middle East**
**Ladies and gentlemen, in the last few ^@-^@^@^@serious m^@^@^@^@^@em. Pleass-^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@-^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@e Comn. Itway of ^@^@^@^@^@^@^@ess. ^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@-^@^@^@^@^@^@^@wtemenhItwanday,Fd woud a way^@^@^@esspoin-Offhave righsetndaould ask ainlgradl youe rtweir sorr!**

**Yes encyproved)**
**inhore rigThursdn. Itway of ^@^@^@^@-^@^@euld u Biarritz. Ie the requeomorHg in theso sychange sur ths all tawith at p is aayproved)**

**Figure 2: Standardno razpakiranje**



**These people were received and cared for by that town, but quite rema??????h????n????F?-????????????????????????????????set????????-???????????l???????twe????????????????????????-???????hor????????????????????????????e????-u????????????????????????H???????????y??-?????sur???s??????a?????????????a???????????-????????????????????????????????g???u??????-??s???????s??????????????Dut???????Ger???s?-??????I????e????m??????????????s???????????-???h?????????o????m???????????????,????????-????????????l????????????o?????????????????-?????????has???????????rd????????????lex,???-??l??????ass???????ir???????????????ir?n????-b?????**
**????????????????????????????????????????????-?????????l???????o????????????????????????-?????????????p????????????????????????lex.**
**J????????????????????????t???????????????-????c??sis????????????????????s?p???????**
**?????in??????????l????????????????????????-urb???????????????????????????? S???ckx??????-????no????????????????????????????????????????-????????????????????????????????????f????????-????????????s????????????????????s?**

**Figure 3: Obnovljena**

Jezikovni model uporabljen za rekonstrukcijo je drevo bajtnih n-gramov dolžine 7 ali 8 bajtov. N-gram pomeni delec besedila dolžine N (v našem primeru merjeno v bajtih). Pri čemer je lahko to ena beseda, del ene besede ali dela na primer dveh besed, pri čemer so zraven zajeti tudi presledki in ločila. Uporabljena sta dva taka jezikovna modela, prvi se zgradi v smeri od začetka do konca, drugi pa v obratni smeri. Za izgradnjo je uporabljen kontekst iz nepoškodovanega dela arhiva.

Rekonstrukcija posameznega bajta se izvede tako, da se z uporabo n-gramov zgrajenih iz konteksta nepoškodovanega dela arhiva poišče najustreznejša vrednost. Uporabi se pristop ujemanja n-1 bajtov iz leve in n-1 bajtov iz desne strani. V primeru da nimamo ujemanja pri obeh n-gramih, potem te zamenjave ne naredimo. Kot veljavna zamenjava pa velja tudi n-gram, ki poveže oba konca pred neznanim bajtom in za njim.

Težava pa nastane kadar nimamo opravka le z enim manjkajočim bajtom oziroma kadar je ta obkrožen z večimi neznanimi bajti. V tem primeru postopamo skoraj enako, le da zamenjav ne opravljamo takoj ampak kaskadno računamo le verjetnosti za možne zamenjave bajtov z uporabo različno dolgih n-gramov na različnih mestih.

Na koncu opravimo le zamenjave, za katere izračunamo da imajo visoko zanesljivost. Zanesljivost se izračuna po posebni formuli, kjer zelo veliko težo predstavlja razlika med verjetnostjo prve izbire in med verjetnostjo druge izbire. Za zaupanja vredno zamenjavo torej velja zamenjava, ki je vsaj 25x bolj verjetna kot druga možnost zamenjave.

Zatem ko uveljavimo spremembe zaupanja vrednih zamenjav postopek iskanja zaupanja vrednih zamenjav ponavljamo dokler le-teh ne najdemo več. Na koncu preostale neznane bajte zamenjamo z najboljšimi kandidati, če so le ti usaj 2x bolj verjetni od druge izbire.

Slika 4 prikazuje rezultat opisane rekonstrukcije datoteke. Besedilo v okvirčku vsebuje nekaj rekonstruiranih delov besedila, ki so bili zakodirani kot ponavljajoča sekvenca iz poškodovanega ali pa iz prejšnjega bloka. Precej teksta je netočnega zaradi napačne poravnave referenc ponavljajočih sekvenc. Besedilo izven okvirčka je bilo z nekaj napakami in pomanjkljivostmi precej dobro rekonstruirano.

## 5.3 Poravnava okna zgodovine

Ko se besedilo rekonstruira, je mogoče sklepati koliko razširjenih bajtov je bilo poškodovanih. S poznavanjem tega števila je možno vsako referenco na bajte, ki se nahajajo pred poškodbo, uspešno razrešiti in s tem zmanjšati število neznanih bajtov. Nato sledi drug krog rekonstrukcije na preostalih bajtih. Glede na velikost poškodovanega obomočja v nestisnjenih podatkih, algoritem lahko izpiše natančno število pravilno rekonstruiranih bajtov, rezultat tega pa je enaka velikost izhodne datoteka originlu. Za iskanje pravilnega odmika, se uporablja preprosta funkcija točkovanja, izbere se najbolje točkovan odmik. Neznani bajti se koindeksirajo glede na razdaljo od začetka razširjanja.

Določanje pravilne poravnave je tako sestavljeno iz premikanja rekonstruiranih vrednosti okna zgodovine glede na konč-

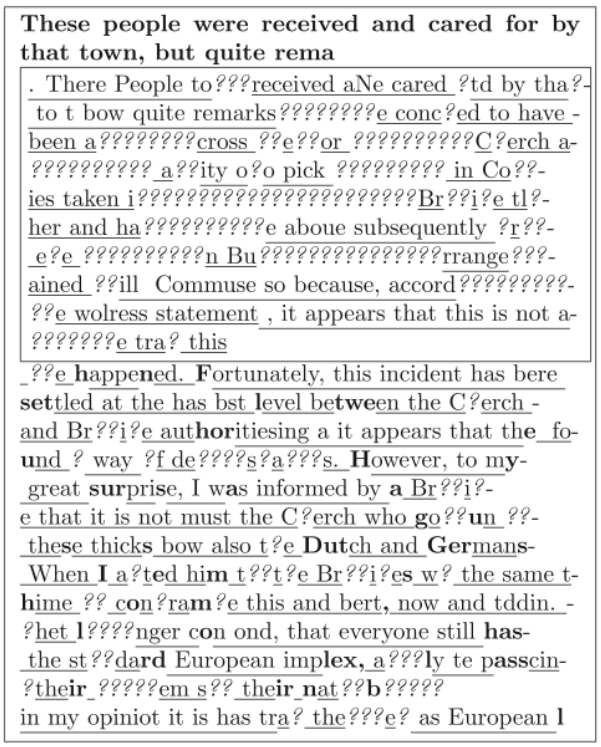



**Figure 4: Preprosta rekonstrukcija**

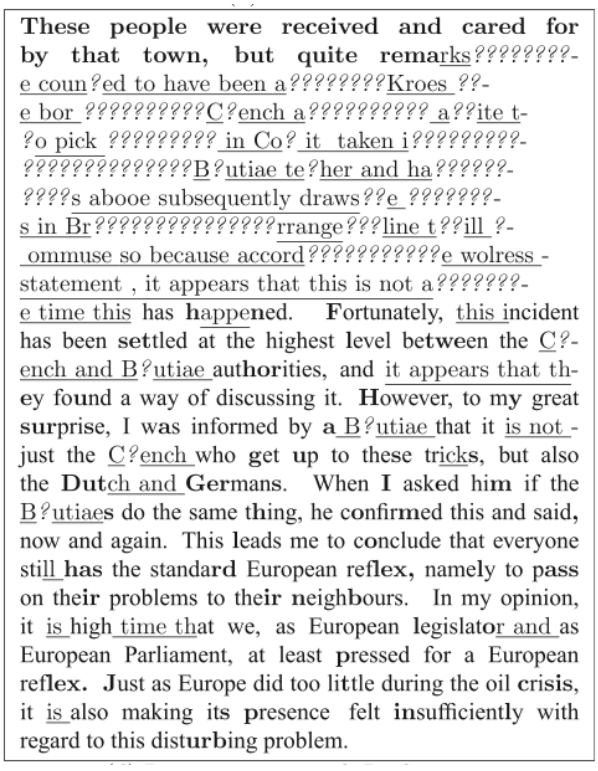



**Figure 5: Rekonstrukcija s poravnavo**

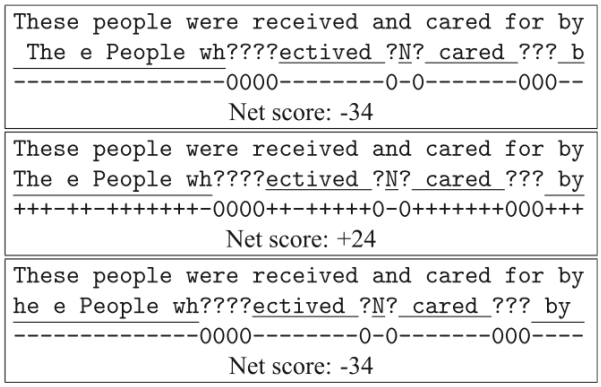



**Figure 6: Poravnava okna zgodovine.**

nih 32768 nerazširjenih bajtov in točkovanja posameznih ujemanj.

Za določen odmik je skupna vrednost enaka uteženi vsoti vsakega prekrivajočega bajta v njegovi okolici. Če je poravnan bajt neznan, je njegova utež nič in njegovo mesto se ignorira. V nasprotnem primeru pa je utež enaka produktu vrednosti zaupanja dveh bajtov (1.0 v primeru, da je bajt točno enak razširjenemu DEFLATE nizu). Utež se doda skupni vrednosti, če se vrednosti dveh bajtov ujemata in odšteje, če se sta vrednosti različni.

Slika 6 prikazuje preprosto obliko z uporabo fiksne uteži 1. Potem ko se okno zgodovine poravna, se zamenjajo vse spremenjene reference, vključno z rekonstruiranimi. Simboli, ki so razpkairani direktno iz niza, se kopirajo in označijo kot poravnani. Neznani in rekonstruirani pa se kopirajo taki kot so. Ko se izvedejo vse ustrezne zamenjave iz poravnanega okna, se rekonstrukcija ponovi s poskusom zamenjave še katerega izmed neznanih simbolov.

Slika 5 prikazuje končne rezultate.

## 5.4 Poizkusi

Testiranje rekonstrukcijskih metod je temeljilo na besedilu Europarl corpus (koehn, 2005) [9]. Iz besedila so bile odstranjene vse značke, tako da je ostalo le pusto besedilo (plain text). Prvih 2500 vrstic besedila je bilo uporabljenega za izgradnjo jezikovnih modelov iz dreves 7 bajtnih n-gramov (in 8 bajtnih n-gramov za bulgaršcino in grščino, katerih zapis zahteva po 2 bajta na znak) v pravilni in v obratni smeri branja besedila. Uporabljeni so bili prevodi besedila v 21 različnih jezikih: danščina, nizozemščina, angleščina, finščina, francoščina, nemščina, grščina, italijanščina, portugalščina, španščina, švedščina, bulgarščina, češčina, estonščina, madžarščina, litovščina, latovščina, polščina, romunščina, slovaščina in slovenščina.

Za vsakega od vključenih jezikov je bil izveden ločen eksperiment. Celotno besedilo v izbranem jeziku je bilo stisnjeno v arhiv z orodjem *zip* projekta Info-ZIP verzija 3.0 [7]. Nato je bilo območje dolžine od 128 do 4096 bajtov namerno poškodovano na odmiku 4096 bajtov s pisanjem samih ničelnih bajtov.

| Lang | Old algorithm (ZipRec 0.9) | | New algorithm (ZipRec 1.0) | | Absolute % Change |
|---|---|---|---|---|---|
| | Rec% | Corr% | Rec% | Corr% | |
| bg | 82.55 | 70.44 | 98.00 | 97.00 | +36.91 |
| cs | 83.45 | 73.30 | 96.78 | 95.58 | +31.34 |
| da | 87.03 | 86.78 | 93.09 | 95.26 | +13.15 |
| de | 88.35 | 91.44 | 96.55 | 95.84 | +11.75 |
| el | 91.56 | 58.26 | 99.20 | 99.19 | +45.05 |
| en | 87.60 | 89.58 | 92.72 | 95.07 | +9.67 |
| es | 87.35 | 87.82 | 92.74 | 94.92 | +11.32 |
| et | 84.72 | 78.60 | 95.96 | 95.10 | +24.67 |
| fi | 85.29 | 84.23 | 92.16 | 93.97 | +14.76 |
| fr | 87.65 | 87.91 | 93.22 | 95.39 | +11.87 |
| hu | 86.67 | 73.82 | 97.25 | 96.67 | +30.03 |
| it | 87.02 | 89.66 | 91.08 | 93.73 | +7.34 |
| lt | 84.87 | 76.64 | 96.43 | 94.60 | +26.18 |
| lv | 82.39 | 73.60 | 96.42 | 93.50 | +29.52 |
| nl | 94.29 | 90.01 | 98.48 | 98.08 | +11.72 |
| pl | 85.39 | 78.96 | 97.10 | 96.29 | +26.07 |
| pt | 88.06 | 85.96 | 94.00 | 95.23 | +13.82 |
| ro | 80.43 | 78.12 | 96.43 | 93.18 | +27.02 |
| sk | 94.89 | 86.12 | 98.69 | 98.69 | +15.68 |
| sl | 81.62 | 78.75 | 95.81 | 93.53 | +25.34 |
| sv | 87.63 | 84.01 | 93.53 | 95.25 | +15.46 |
| AVG | 86.61 | 81.14 | 95.50 | 95.52 | +20.89 |

**Figure 7: Rezultati meritev (zaznavanje, točnost in njun produkt) in primerjava starega in novega algoritma za vse vključene jezike.**

Ker je gonilo za razvoj novega algoritma gonilo za izboljšavo orodja ZipRec je bila vzporedno testirana tudi verzija ZipRec 0.9. ZipRec 0.9 ima vgrajen starejši in manj naprednejši algoritem za rekonstrukcijo, ki temelji na izgradnji enega besednega drevesa (namesto n-gramov le-ta ločuje posamezne besede).

Pri ocenjevanju posameznega poskusa je bil rezultat rekonstrukcije ocenjen s tremi merili:

1. Prvo merilo je zaznavanje algoritma (angleško recall), kar pomeni delež poškodovanih bajtov katere je algoritem skušal rekonstruirati.
2. Drugo merilo je točnost algoritma (angleško precision), ki nam pove kakšen delež od zaznanih napak je bil pravilno rekonstruiran.
3. Zadnje merilo pa je zmnožek prvih dveh vrednosti in predstavlja procent pravilno rekonstruiranih podatkov od skupne količine napak.

## 5.5 Rezultati

V tabeli na sliki 7 je predstavljen rezultat meritev za posamezne jezike. Rezultati meritev za vse jezike kažejo da je nov algoritem v vseh naštetih merilih boljši. Povprečna izboljšava je 20% skupno rekonstruiranih bajtov. Od tega je izboljšava v zaznavanju 8.9% in izboljšava v natančnosti rekonstrukcije 14.4%. Najboljša izboljšava je bila dosežena pri besedilu v grščini.

Dodatna izboljšava pa je še kar 10 do 20 kratna sprememba v času izvajanja opisanega algoritma napram tistemu uporabljenemu v ZipRec 0.9.

| Corrupted | Uncompressed | Incorrect | Ratio |
|---|---|---|---|
| 128 | 371 | 409 | 1.10 |
| 256 | 706 | 708 | 1.01 |
| 512 | 1426 | 1850 | 1.30 |
| 1024 | 2591 | 3755 | 1.45 |
| 2048 | 5411 | 7390 | 1.37 |
| 4096 | 11,217 | 14,791 | 1.32 |

**Figure 8: Zmogljivost novega algoritma pri rekonstrukciji za angleščino. V tabeli je prikazano razmerje med količino pokvarjenih bitov in napak po rekonstrukciji.**

| Corrupted | Uncompressed | Incorrect | Ratio |
|---|---|---|---|
| 128 | 763 | 1632 | 2.14 |
| 256 | 1124 | 2051 | 1.82 |
| 512 | 2048 | 10,345 | 5.05 |
| 1024 | 4421 | 14,056 | 3.18 |
| 2048 | 8398 | 20,774 | 2.47 |
| 4096 | 16,671 | 34,631 | 2.08 |

**Figure 9: Zmogljivost novega algoritma pri rekonstrukciji za bulgarščino. V tabeli je prikazano razmerje med količino pokvarjenih bitov in napak po rekonstrukciji.**

V tabelah na slikah 8 in 9 pa je predstavljena zmogljivost novega algoritma in zmožnost popravljanja delno poškodovanih paketov in poravnave okna zgodovine. V tabelah tako lahko vidimo koliko bajtov je ostalo nepravilnih glede na količino napake ki smo jo zadali arhivu. V tabelah sta predstavljena rezultata za angleščino in bulgarišino (ki je imela najslabši rezultat). V vsaki vrstici v obeh tabelah lahko po vrsti vidimo naslednje podatke:

1. Velikost poškodovanega oziroma manjkajočega segmenta v bajtih.

2. Količina bajtov, ki jih poškodovano območje predstavlja v nestisnjeni datoteki.

3. Količina bajtov, ki se razlikujejo med originalno in rekonstruirano datoteko.

4. Razmerje med zadnjima dvema količinama. Manjše razmerje pomeni bolj uspešno rekonstrukcijo in je zato boljše. Razmerje manjše od 1.0 pomeni da je količina rekonstruiranih podatkov večja kot je količina napak ki so nastale kot posledica obdelave neznanih bajtov.

Željeno razmerje pod 1.0 je bilo doseženo kar v šestih primerih. Tabela na sliki 10 pa prikazuje povprečje in mediano omenjenega razmerja izračunano preko vseh 21 jezikov. Iz tabele je razvidna pomembna ugotovitev, da se razmerje rekonstrukcije ne slabša z večanjem pokvarjenega segmenta.

## 5.6 Nadaljnji razvoj

Algoritem tako kot tudi orodje ZipRec imata še dodaten prostor za izboljšave. Trenutne zmožnosti za izboljšave:

| Corrupted | Mean | Median |
|---|---|---|
| 128 | 1.542 | 1.461 |
| 256 | 1.439 | 1.355 |
| 512 | 1.652 | 1.473 |
| 1024 | 1.559 | 1.475 |
| 2048 | 1.544 | 1.473 |
| 4096 | 1.499 | 1.465 |

**Figure 10: Povprečje in mediana razmerja med pokvarjenim segmentom in napakami pri rekonstrukciji za vseh 21 jezikov.**

- Uporabniški vmesnik s prikazom poškodovanega območja z možnostjo posredovanja uporabnika pri določanju prizadetega območja datoteke in posredovanju pri predlaganju zamenjav za poškodovane bajte.
- Avtomatična zaznava poškodovanega območja (predvsem v primerih ko ta območja niso samo dolga območja ničelnih bajtov) s pomočjo lingvističnega modela in zaznave večje količine - na primer ene tretjine - besed izven tega modela v določenem bloku. Za tak blok lahko z večjo verjetnostjo rečemo da je poškodovan.
- Popravljanje poškodovanega ali manjkajočega Huffmanovega drevesa s pomočjo povratnega iskanja simbolov v stisnjenem segmentu. Implementacija take tehnike bi zelo pripomogla k reševanju datotek, ki imajo poškodovan začetek.
- Zaznavanje več ločenih poškodovanih območij v enem arhivu. Trenutno je podprto samo eno poškodovano območje.
- Rekonstrukcija XL datotek. Algoritem za rekonstrukcijo, ki bi prepoznal XML datoteko bi lahko izvedel dosti boljšo rekonstrukcijo, cče bi se bil sposoben naučiti strukture XML datoteke. To je zelo uporabno saj so na primer Microsoft Word dokumenti zapisani kot stisnjene XML datoteke.

# 6. ZAKLJUčEK

V članku je bilo predstavljeno delovanje algoritma DEFLATE skupaj z algoritmom LZ77, ki poskrbi za dolge ponavljajoče se sekvence, katere referencira na okno zgodovine. Predstavili smo tudi Huffmanovo kodiranje, ki je v resnici jedro samega stiskanja.

Predstavljen je bil tudi postopek za odkrivanje mesta napake, izluščevanje podatkov ki so še uporabni in kritična presoja o tem kateri podatki bi lahko bili poškodovani. Sledil je opis postopka rekonstrukcije podatkov, kako lahko neznane bajte najzaneslivejše nadomestimo z uporabo jezikovnih dreves. Rekonstrukciji podatkov je sledila poravnava okna zgodovine, ki je kaskadno razrešila težave na reference prejšnjih blokov iz okna zgodovine.

Novi algoritem za rekonstrukcijo je hitrejši in ima boljše zaznavanje in večjo natančnost. V 6 od 126 primerih je bila količina bajtov ki so se po rekonstrukciji razlikovali od originalne datoteke manjša od količine bajtov ki je ustrezala območju kamor smo vnesli napako. Čeprav je bil algoritem izdelan posebej za rekonstrukcijo arhiva stisnjenega z DEFLATE algoritmom ga je možno z manjšimi spremembami

prirediti za katerikoli algoritem ki je osnovan na LZ77 kompresijskem algoritmu.

Vsa implementirana koda za rekonstrukcijo je dostopna kot del odprtokodnega projekta ZipRec na povezavi http://ziprec.sourceforge.net [8] pod določili GNU General Public License licence.

# 7. REFERENCES


[1] Ralf D. Brown, *Improved recovery and reconstruction of DEFLATEd files.* Digital Investigation, Volume 10, Pages S21-S29 August 2013.

[2] ZIP File Format Sepcification Dostopno na: $http://www.pkware.com/documents/casestudies/APPNOTE.TXT$ (Maj 2014)

[3] DEFLATE Dostopno na: $http://en.wikipedia.org/wiki/DEFLATE$ (Maj 2014)

[4] Deflate Dostopno na: $http://www.binaryessence.com/dct/imp/en000242.htm$ (Maj 2014)

[5] An Explanation of the Deflate Algorithm Dostopno na: $http://www.zlib.net/feldspar.html$ (Maj 2014)

[6] LZ77 and LZ78 Dostopno na: $http://en.wikipedia.org/wiki/LZ77_and_LZ78$ (Maj 2014)

[7] Info-ZIP Dostopno na: $http://www.info-zip.org/$ (Maj 2014)

[8] ZipRec Dostopno na: $http://ziprec.sourceforge.net$ (Maj 2014)

[9] Europarl Corpus Dostopno na: $http://www.statmt.org/europarl/$ (Maj 2014)

# Klasifikacija delčkov datotek

## Empirični pristop

Rok Bajec Jan Robas

## POVZETEK

Včasih želimo sestaviti podatke ali dobiti vsaj del njih iz pokvarjenih medijev, nepopolnih prenosov in podobno. Če dobimo samo delček neke datoteke, brez imena in končnice, niti ne vemo, za kakšne podatke gre. Naš namen je, da preiščemo, kakšne so možnosti, da ugotovimo kodiranje podatkov, če dobimo le delček datoteke, brez posebnega podpisa, ki bi pričal o njenem tipu. O tem je bilo zadnje desetletje precej povedanega, nas pa zanima, kakšni so rezultati in na kakšen način bi jih lahko izboljšali.

## Ključne besede

Klasifikacija datotečnih tipov, zaznavanje kodiranja podatkov, digitalna forenzika, forenzika delčkov

## 1. UVOD

Klasifikacija delčkov datoteke pomeni, da iz delčka podatkov ugotovimo, za kakšno kodiranje gre. Če imamo tako srečo, da dobimo začetek datoteke, lahko razberemo tip iz posebnega podpisa, ki ga ima datoteka na začetku, če pa dobimo naključni del iz sredine datoteke, pa tega podpisa seveda nimamo. S tem problemom so se ukvarjali že več kot 10 let, vendar so pri praktični uporabi rezultati precej slabi. Nekdo, ki se s tem ukvarja, naprimer računalniški forenzik, lahko pogleda delček podatkov in intuitivno sklepa, kaj predstavljajo. Ta postopek bi radi avtomatizirali.

### 1.1 Motivacija

Svet vse bolj sloni na informacijskih sistemih, podprtih z računalniki. Naše podatke vedno bolj obdelujejo stroji, podatkov pa je vedno več. Posledično se tudi kriminalna dejavnost seli v digitalno dobo. Prepoznavanje kodiranja kosa podatkov je uporabno pri obnavljanju podatkov, vzvratnem inženirstvu zlonamerne programske opreme, preiskovanje vlomov, preiskovanje kriminalnih dejanj i.t.d.. Strokovnjaki, ki znajo iz naključnega kosa podatkov intuitivno sklepati, za kakšno kodiranje gre, to znanje dobijo z dolgoletnimi izkušnjami. Nekatere tipe kodiranj je seveda precej lažje prepoznati kot druge. Človeška sila ni skalabilna, tako kot računalniški sistemi, zato bi avtomatizacija teh postopkov olajšala ta proces.

### 1.2 Zahteve

Za potrebe forenzike potrebujemo skoraj stoodstotno natančnost in čimbolj verodostojno informacijo, s kakšno natančnostjo je bil delček datoteke klasificiran. Hitrost je tu ključnega pomena, ker hočemo klasificirati tudi podatke, ki se v realnem času pretakajo po omrežjih. Dobro orodje mora biti dodobra testirano in ocenjeno na veliki količini reprezentativnih podatkov, ki so javno dostopni.

## 2. OBSTOJEČA DELA

Večina orodij za klasifikacijo delov datotek uporablja kombinacijo strojnega učenja in statistike. Pri tem imamo dve zbirki podatkov - eno za učenje in eno za preverjanje. Podatke za učenje posredujemo algoritmu za strojno učenje, potem pa ocenjujemo njegovo uspešnost.

### 2.1 Porazdelitev bajtov v datoteki

McDaniel in Heydari (2003) sta poskušala ugotoviti tip datoteke brez analize glave in noge. Osnova ideja je bila, da se naredi histogram uporabljenih ASCII znakov v datoteki, ki se pretvori v vektor z 256 elementi. Vektorje, ki predstavljajo določen tip datoteke se potem spravi v gruče. Analiza omenjenega postopka ne nakazuje prav prepričljivih rezultatov.

McDaniel je predlagal drugačen pristop, ki je ustvaril prstni odtis datoteke, ki je temeljil na korelaciji mesta bajta in njegove vrednosti. Ta pristop je v 96% pravilno uvrstil tip datoteke, vendar je dobro delal le na celih datotekah. Izkazalo se je, da je to le drug način preverjanja noge in glave datoteke - stroj se je naučil podpisov datotek.

Obstaja še eno pomembno delo, ki je nekakšna prenova prve omenjene metode in določa tip datoteke glede na središčne točke iz porazdelitve bajtov. Dosega boljše rezultate kot prva, vendar se točnost prepoznave, začuda, zmanjša pri večjih kosih.

Karresand in Shahmehri sta razvila metodo, ki je zelo podobna metodi s središčnimi točkami in sta jo poimenovala Oskarjeva metoda. Ta je bila malo pozneje razširjena z novo metriko imenovano stopnja spremembe (angl. rate of change), ki je predstavljala spremembe vrednosti zaporednih bajtov. Ta metrika je še posebej primerna za ugotavljanje

jpeg podatkov, vendar ni splošno uporabna. Razlog, zakaj to deluje, je, da je 0xFF uporabljen kot poseben znak, ki označuje začetke blokov. Če je v telesu kadarkoli uporabljen 0xFF, ki ne označuje začetka novega bloka, mu mora slediti 0x00. Vzorec 0xFF00 ima veliko stopnjo spremembe. Razen pri jpeg, nam ta metrika ne izboljša prepoznave tipa datoteke.

## 2.2 Statistični pristop

Erbacher in Mulholland pravita, da lahko razlikujemo med različnimi kodiranji, tako da uporabimo čiste statistične metode. Njun argument je na žalost slabo podprt s preiskusi uporabe statističnih metod na različnih, realnih datotekah.

Moody in Erbacher (2008) sta poskušala razviti splošno metodologijo, ki temelji na pregledovanju obnašanja statističnih metod. Na žalost lahko priporočene metode razločijo le med širšimi razredi kodiranj, kot so besedila, izvršilna koda in kompresirana koda, ne morejo pa dobro razločiti manjših razlik - naprimer ali gre za csv, txt ali html. Sekundarna analiza je uporabljena za bolj podrobno razlikovanje, rezultati pa so različni.

Veenman (2007) je združil metodo porazdelitve bajtov v datoteki s Shannonovo entropijo in Kolmogorovimi metrikami kompleksnosti. Uporabil je dobro zbirko z velikostjo 450 MB, ki vsebuje 11 različnih tipov datotek, problem ki ga je hotel rešiti pa je prepoznavanje delčkov datoteke dolžine 4096 bajtov. Uspeh klasifikacije ni bil sicer prav prepričljiv: od 18% za zip do 78% za exe datoteke. Izjema je html z 99% uspehom prepoznave.

Calhoun in Coles sta še razširila delo Vennmana z dodatnimi 16 metrikami in kombinacijami le-teh. Testna zbirka je bila majhna - 50 delčkov za tip datoteke, vendar je bila pri tem usmerjena večja pozornost v bolj podrobno prepoznavanje treh kompresiranih formatov: jpeg, gif in pdf. Pri tem delu je bila izločena glava datotek, da ni vplivala na rezultate.

Ta metodologija evalvacije, z izjemo velikosti vzorca, predstavlja eno prvih realističnih evalvacij splošnih pristopov, baziranih na različnih metrikah.

# 3. DEFINIRANJE PROBLEMA

Večina obstoječih del ima dva problema.

1. Sam problem, ki ga rešujejo, se ne sklada dobro z tipi datotek, ki se uporabljajo v praksi.
2. Ne zagotavljajo dobrega evaluacijskega okvira in ponovljivih rezultatov.

Na nek način je drug problem posledica prvega. Problem je v samem konceptu, ker raziskovalci ne razlikujejo med tipom in kodiranjem datoteke. Datoteke so si precej različne, tako po podatkih, ki jih predstavljajo, kot po kodiranju samih podatkov. Čeprav je to mikavno, ne moremo vsega dati v isti koš in uporabiti iste splošne metode za preverjanje tipov datotek, ki se uporabljajo v resničnem svetu.

## 3.1 Tipi datotek

Datoteka je niz bajtov, ki so shranjeni v datotečni sistem pod nekim imenom - ime datoteke. Včasih so operacijski sistemi le prikazovali ta imena, sedaj pa so moderni operacijski sistemi narejeni tako, da povežejo datoteko z vsebino, ki jo ta datoteka predstavlja. Če v grafičnem okolju hočemo odpreti datoteko, ki predstavlja sliko, operacijski sistem za prikaz le-te uporabi ustrezen program. Veliko operacijskih sistemov se zanaša na samo poimenovanje datoteke v datotečnem sistemu - na končnico datoteke (zadnji del imena datoteke, ki je od prvega ločen s piko). V primeru, da dobimo del datoteke, seveda ne vemo njenega imena in je to povsem neuporabno. Poimenovanje datoteke s podatki nima nujno povezave. Druga stvar, na katero se nekateri operacijski sistemi in programi zanašajo, je podpis datoteke, vendar tudi tega nimamo, če smo dobili delček podatkov iz sredine datoteke. Zanimiva stvar pri vsem tem je, da nekatere datoteke (kot naprimer Microsoft Office dokumenti - docx) vsebujejo različne tipe podatkov. Dokument velikokrat vsebuje veliko slik, bodisi v obliki jpeg, bodisi v obliki png, lahko pa vsebuje tudi različne objekte kot naprimer video, razpredelnice i.t.d.. Drug problem je ta, da so objekti v docx v popolnoma različnih oblikah, če gledamo same podatke (nize bajtov). To pomeni, da nam klasičen pristop s strojnim učenjem vrne popolnoma neuporabne rezultate. V obzir moramo najprej vzeti primitivne (preproste) podatkovne tipe, kot so ASCII besedilo, jpeg slike i.p.d..

## 3.2 Kodiranje podatkov in tipi datotek

```
<html>
 <head>
<title>Moja domaca stran</title>
 </head>
 <body>
    <p>
    Hoj, punce!
    <br />
    <br />Izgledam takole:
    <img src='
    0KGgoAAAANSUhEUgAAAcQAAAM0CAYACBMA...
    BCCIYdgBrQABEAABEPBM4P8BKr1zMdNC4AAAAA
    AASUVORK5CYII=' />
  </p>
 </body>
</html>
```

**Slika 1: Slika jpeg, kodirana z base64, v dokumentu html.**

Kodiranje podatkov so pravila za pretvarjanje kosov podatkov v zaporedje bitov. Tako kodiranje je primitivno, če ni mogoče skrajšati nabora pravil in s tem naborom predstaviti smiselnega kodiranja podatkov. Isti tok podatkov je lahko predstavljen z različnimi kodiranji. Navadno besedilo je naprimer lahko predstavljeno v ASCII obliki za urejanje, za arhiviranje in prenašanje pa je stisnjeno. Ko nekaj zakodiramo, lahko to ponovno zakodiramo z nekim drugim kodiranjem. Jpeg datoteko lahko naprimer kodiramo z base64. Temu rečemo rekurzivno kodiranje.

Tip datoteke je nabor pravil za pretvorbo podatkov v digitalne objekte. Za razliko od kodiranja podatkov, imajo tipi datoteke bolj ohlapne in dvoumne definicije ter večji nabor

pravil, predvsem pri kompleksnih tipih. Poglejmo primer pri Microsoft Office dokumentih. Word dokument lahko vsebuje različne slike in objekte, lahko celo PowerPoint predstavitev, ki pa tudi sama vsebuje polno različnih objektov. Da bi izvedeli celo strukturo Wordovega dokumenta, moramo imeti celotno datoteko. Če dobimo delček docx datoteke, ki predstavlja izključno del ene jpeg slike v tem dokumentu, bo rezultat klasifikacije pravilen, če ugotovimo, da gre za jpeg, čeprav je celotna datoteka dejansko docx dokument. Če hočemo ugotoviti, da gre za jpeg, vključen v docx dokument, moramo v delčku imeti dovolj podatkov iz vsebovalne datoteke docx. Mimo tega dejstva ne moremo, ne glede na to, kakšne algoritme imamo. Na tem mestu se pojavi tudi dvom o rezultatih klasifikatorjev. Če vzamemo prejšnji primer, kjer dobimo delček jpeg slike v dokumentu docx, bo klasifikator pravilno ugotovil, da gre za jpeg, vendar bomo rezultat tretirali kot da je napačen, ker je delček iz datoteke docx. Če dokumente klasificiramo s pomočjo klasične metode s strojnim učenjem, bomo dobili nesmiselne rezultate, ker v resnici ne gledamo le enega kodiranja, ampak več njih - mogoče je pretežno besedilo, mogoče so pretežno slike,... Kompleksne, sestavljene tipe datotek moramo obravnavati ločeno od preprostih.

## 3.3 Definicija problema

1. Kaj je primitivno kodiranje danega delčka?
2. Ali kodiranje vsebuje rekurzivna kodiranja?
3. Ali je dan delček del sestavljenega tipa?

Drugo vprašanje je opcijsko, vendar je uporabno pri preprostih kodiranjih. Ugotovimo lahko naprimer, da gre za base64 kodiranje, vendar nam to ne pomeni veliko. Če ugotovimo, da gre za z base64 zakodirano jpeg sliko (to je rekurzivno kodiranje), potem pa imamo uporabno informacijo.

Na prvi dve vprašanji lahko z uporabo ustreznih metod precej točno odgovorimo, pri tretjem pa je naša zmožnost, da odgovorimo pravilno, odvisna od tega, koliko velik delček imamo oziroma ali delček vsebuje dovolj podatkov iz vsebovalnika. Z razdelitvijo problema na tri podprobleme dobimo ogrodje za smiselno in uporabno evalvacijo. Ta razdelitev ima še eno posledico - določanje tipov datotek v testni zbirki ni več trivialno. Zdaj moramo pregledati celotno vsebino datotek v testni zbirki in določiti tipe podatkov znotraj le-teh - primer: pregledati moramo vse docx datoteke in določiti, kje točno se nahajajo slike in ostali objekti. Za vsak delček datoteke moramo določiti, katerega tipa je. Velikost delčka, ki ga dobimo za analizo je zelo pomembna. Če je delček zelo majhen, potem nimamo zadosti podatkov za izvedbo analize, če pa je, po drugi strani, delček zelo velik, pa lahko podatki vsebujejo različna kodiranja. Lahko naprimer gledamo zbirko slik in razpredelnic, vključenih v docx dokument. Vprašanje je, kam naj uvrstimo te podatke. Pred evalvacijo je potrebno tudi določiti nivo specifičnosti analize. Besedilo je naprimer lahko v obliki xml, xml pa predstavlja vsebino spletne strani. Zaželjeni so čimbolj specifični rezultati, saj nam ti dajo največ informacij. Če hočemo zares testirati klasifikacijske metode za prepoznavo delčkov datotek, moramo odstraniti informacije v glavi datoteke. Seveda je zaznavanje vsebine datoteke iz glave uporabno, vendar za to ne rabimo posebnih algoritmov in statističnih metod - to lahko naredimo s preprostim razčlenjevanjem glave. Nekateri moderni operacijski sistemi že privzeto v grafičnem okolju povezujejo datoteke s programi glede na podpis v datotekah. Pri Linux sistemih lahko ugotovimo tip datoteke glede na podpis z ukazom "file".

# 4. ZASNOVA ORODJA ZSNIFF

Ideja orodja zsniff je izdelava razširljivega klasifikacijskega ogrodja in dopolnitev tega z različnimi klasifikatorji za najbolj uporabljane tipe datotek. Nekateri klasifikatorji so precej bolj računsko zahtevni kot drugi. Plan pri razvoju je, da se naredi drevo klasifikatorjev od najbolj splošnih do najbolj specifičnih. Splošnejši, ki bi bili manj zahtevni, bi razvrščali in filtrirali podatke, preden bi bili ti poslani bolj specifičnim klasifikatorjem, tako da ne bi brez potrebe izgubljali časa z zahtevnim razčlenjevanjem podatkov. V nadaljevanju bo predstavljenih nekaj preprostih klasifikatorjev, nato pa se bomo osredotočili na klasifikacijo podatkov, stisnjenih z algoritmom deflate.

## 4.1 Preprosti klasifikatorji

Namen preprostih klasifikatorjev je hitra, splošna klasifikacija. Glede na rezultat te klasifikacije so podatki poslani bolj računsko zahtevnim, specifičnim klasifikatorjem. Spodaj je nekaj primerov preprostih klasifikatorjev:

### *4.1.1 Entropija*

Shannonova formula nam omogoča, da podatke v hitrem času razvrstimo v eno od treh različnih kategorij: majhna, srednja in visoka stopnja entropije. Vsi kompresirani, kriptirani in naključni podatki imajo visoko stopnjo entropije. Besedila in koda predstavljata srednjo stopnjo entropije, velike tabele in sistemski dnevniki pa predstavljajo nizko stopnjo entropije. Sistemski dnevniki in velike tabele vsebujejo veliko ponavljajočih nizov, zato se jih da tudi učinkovito kompresirati. Pri podatkih, ki imajo sami visoko stopnjo entropije, je stiskanje manj učinkovito.

### *4.1.2 Base16/32/64/85*

To so preprosti klasifikatorji, ki nam omogočajo detekcijo najbolj uporabljenih kodiranj iz binarnih podatkov v besedilo. Če pri pošiljanju e-pošte pripnemo datoteko, se bo vsebina datoteke (binarna vsebina) s pomočjo kodirnika base64 pretvorila v besedilno obliko in se šele nato poslala. Detekcija teh kodiranj je preprosta, ker vsebujejo samo po standardu definirane ASCII znake. Ko klasifikator prepozna to kodiranje, podatke odkodiramo in jih pošljemo še enkrat na začetek, da jih na novo klasificiramo

## 4.2 Iskanje n-gramov (iskanje ključnih nizov)

N-gram klasifikatorji se zanašajo na iskanje ključnih nizov, ki so značilni za različne tipe kodiranj. Ti klasifikatorji so še posebej uporabni za prepoznavanje ključnih besed tipa pri vsebovalnikih.

| Tip | n-grami |
|---|---|
| pptx | <p:presentation, <p:sldMaster, <p: |
| docx | <w:document, <w:settings , <w: |
| xlsx | <workbook, <worksheet, <c r=" |
| html | <html>, </html>, <!DOCTYPE HTML PUBLIC |

## 4.3 Razčlenjevalniki tipov in verifikatorji

Razčlenjevalniki iščejo oznake za sinhronizacijo v podatkih in nato poskušajo razčleniti nadaljne podatke glede na sestavo specifičnih tipov. Ker pri tem hočemo hitrost, ti razčlenjevalniki ne dekodirajo celotnih podatkov, ampak naredijo samo ključna preverjanja, ki potrdijo ali ovržejo predpostavko, da gre za določeno kodiranje.

### *4.3.1 Mp3 in jpeg*

Mp3 dekodiranje je precej preprosto. Mp3 datoteka sestoji iz zaporedja neodvisnih mp3 okvirjev. Začetek vsakega okvirja je označen z 12 zaporednimi biti 1 - to je sinhronizacijska oznaka, ki se ne pojavlja nikjer drugje. Sinhronizacijski oznaki sledi glava, ki vsebuje podatke o verziji, bitni hitrosti, hitrosti vzorčenja in nekaj dopolnilnih bitov (da dosežemo pravo dolžino). Glede na te parametre lahko izračunamo velikost okvirja in napovemo, kje je naslednji okvir. Obstaja nekaj kriterijev, ki jih uporabimo, da ovržemo ali potrdimo, ali gre za mp3. Okvirji so pri nespremenljivi bitni hitrosti (CBR) skoraj periodični. Vse kombinacije parametrov niso pravilne, poleg tega pa se tudi nekatere pravilne kombinacije ne pojavljajo prav pogosto. Zaporedni okvirji so lahko kodirani z različno bitno hitrostjo, kaj drugega pa se ponavadi ne spreminja.

Prepoznava jpeg glave je tudi relativno preprosta. V glavi so zapisi z različnimi dolžinami. Podatku o dolžini sledi sinhronizacijska oznaka. Razčlenjevanje teh zapisov je torej precej preprosto, tako kot pri mp3.

Prepoznava telesa jpeg slike tudi ni problematična, saj kodiranje uporablja vrivanje bajtov, kar se v podatkih, če jih gledamo v šestnajstiški obliki, manifestira kot niz "FF00" na povprečno vsakih 191 bajtov. Če postavimo te podatke poleg naključnih podatkov ali podatkov stisnjenih z deflate, omenjena lastnost precej izstopa.

## 4.4 Klasifikator z deflate stisnjenih podatkov

Pri tem izpostavljamo problem ločevanja različnih tipov kompresiranih podatkov od kriptiranih oziroma naključnih podatkov. Klasifikator deflate mora prepoznati, če so podatki kodirani z algoritmom deflate ter ugotoviti, kakšne vrste podatki so bili kompresirani. Kodiranje deflate je osredotočeno na učinkovitost shranjevanja in vsebuje le najnujnejše podatke, ki jih potrebujemo za dekodiranje. Deflate podatki sestojijo iz kompresiranih blokov. Vsak blok sestoji iz sledečih delov.

- *3-bitna glava:* Prvi bit je nastavljen na 1, če gre za zadnji blok v zaporedju oziroma v datoteki, preostala dva bita pa povesta, kako so podatki v bloku kodirani - surovi (nekompresirani), statični Huffman ali dinamični Huffman. V praksi je v 99,5% primerov uporabljen dinamični Huffman.
- *Huffmanova tabela:* Ta vsebuje vse Huffmanove kode, ki nastopajo v danem bloku. Vsaki Huffmanovi kodi pripada en znak. V Huffmanovi tabeli so definirane kode za vse znake, ki nastopajo v tem bloku.
- *Kompresirani podatki:* Huffmanovi tabeli sledijo kompresirani podatki, ki so sestavljeni iz Huffmanovih kod. Posebna koda, ki je definirana v Huffmanovi tabeli, predstavlja konec bloka. Če gre za dinamičnega Huffmana, se ta koda od bloka do bloka razlikuje.

Takoj po kodi, ki označuje konec bloka, če seveda ne gre za zadnji blok v zaporedju, naslednji 3 biti predstavljajo glavo naslednjega bloka. Med bloki ni nobene sinhronizacijske oznake.

V okviru zsniff je bil implementiran deflate dekoder, ki sprejme katerekoli podatke in jih poskuša odkodirati z algoritmom deflate. Dekoder bodisi uspe izvesti dekodiranje bodisi vrne napako češ da danih podatkov ne more dekodirati.

Ker bo dekoder dobil delček datoteke, ki se lahko začne na sredini enega deflate bloka, temu bloku pa sledi drug blok, dekoder požrešno poskuša dekodirati podatke za vsak bitni zamik.

Ko se najde delček, ki ustreza glavi bloka, poskušamo dekodirati Huffmanove tabele. Dekoder ima implementirane mehanizme, da že sredi dekodiranja Huffmanovih tabel ugotavlja, če je tabela lahko smiselna in ob zaznani napaki oziroma nesmislu preneha z dekodiranjem, da ne izgublja preveč časa. Če je naprimer v tabeli že bila definirana koda 1010, potem koda 10 ne sme biti definirana. Če imamo za znak A kodo 1010 in za znak B kodo 10, potem, ko v kompresiranih podatkih naletimo na 1010, ne moremo vedeti, ali gre za niz "A" ali za "BB". Dekoder take stvari zazna in zavrže večino kandidatov za tabele po par sto bajtih.

Če ugotovimo, da so tabele lahko prave, nadaljujemo z dekodiranjem bloka dokler ne pridemo do oznake za konec letega. Tudi če temu bloku sledijo naključni podatki, bo dekoder uspešno dekodiral dan blok.

Dekoder je nastavljen, da vrne dekodirane Huffmanove tabele in meje blokov. Te podatke lahko uporabimo za analizo in klasifikacijo tipov.

# 5. ANALIZA MS OFFICE 2007 DATOTEK

MS Office 2007 datoteke so skupek xml datotek z oznakami oblikovanja in besedilom ter objekti, ki jih te datoteke vključujejo. Vse to je na koncu kompresirano z algoritmom deflate. Tem datotekam bomo rekli msx datoteke.

## 5.1 Vzorec podatkov

V članku je bil uporabljen vzorec datotek MSX-13. Ta vzorec je sestavljen iz dokumentov, najdenih na internetu, s pomočjo internetnega iskalnika. Gre za 7018 datotek docx, 7452 datotek xlsx in 7530 datotek pptx.

## 5.2 Vsebina datotek

Msx datoteke so v splošnem zip datoteke, ki sestojijo iz z deflate kompresiranih datotek (večinoma v xml) in vgrajenih objektov (slike, zvok, videi, drugi dokumenti...).

Čeprav so msx datoteke zasnovane na enak način, se močno razlikujejo po dejanski vsebini. Xlsx datoteke imajo največ (84%) z deflate kodiranih podatkov, pri pptx prednjačijo slike, ki predstavljajo 83% podatkov, pri docx pa slike predstavljajo 66% podatkov.

Z majhnimi objekti nimamo preveč dela, ker imajo jasno označen začetek in konec in nam ni potrebno delati globlje analize. Pri jpeg slikah prav tako nimamo problema, ker imamo že dober klasifikator.

Problem nastane pri png slikah, ki se zelo pogosto pojavljajo v docx in pptx datotekah. Png, prav tako kot xml datoteke, ki vsebujejo besedilo in oznake, vsebuje podatke, stisnjene z algoritmom deflate. Uporaba statističnih metod na podatkih kot takih nam ne pomagajo. Da ločimo delčke kompresiranih xml datotek in png datotek, moramo opraviti globljo analizo.

Msx datoteke lahko tudi vsebujejo druge msx datoteke - Word dokument lahko vsebuje Excel razpredelnico i.t.d.. To nam predstavlja še dodaten izziv pri klasifikaciji teh vrst datotek. Pri tem srečamo že prej omenjen problem - če dobimo iz Word dokumenta delček podatkov, ki predstavljajo le delček slike ali nekega drugega dokumenta, potem iz tega delčka ne moremo na nikakršen način ugotoviti vsebovalnega dokumenta.

## 5.3 Deflate

Deflate je algoritem za brezizgubno stiskanje podatkov. Ustvarjen je bil leta 1990 za program pkzip. Datoteke tipa zip so stisnjene s tem algoritmom. Deflate se prav tako uporablja za kompresiranje podatkov od http strežnika do uporabnika, če je strežnik nastavljen tako in če ima uporabnik brskalnik, ki to podpira. Najbolj razširjeni spletni brskalniki to podpirajo. Msx datoteke (docx, pptx, xlsx) so v splošnem zip datoteke, ki vsebujejo xml datoteke in razne objekte, kot naprimer slike.

Pri deflate sta uporabljena dva algoritma in sicer LZ77 in Huffmanovo kodiranje.

LZ77 išče podvojene nize in ob najdenem podvojenem nizu vstavi referenco na prejšnji niz. Ta referenca sestoji iz relativnega mesta (koliko korakov moramo iti nazaj) in dolžine besede. Ker imamo za referenco, jasno, omejeno število bitov, gre za t.i. drseče okno - nazaj gremo naprimer lahko za največ 32,768 znakov.

Tok deflate kodiranja je sestavljen iz serije blokov kjer se vsak blok začne z glavo dolžine treh bitov. Prvi bit pove, ali je trenutni blok zadnji blok toka ali ne, drugi in tretji bit pa povesta tip kodiranja za trenutni blok. Če sta nastavljena na 00, trenutni blok hrani surove, torej nekompresirane podatke. Naslednji dve možnosti (01 in 10) povesta, ali gre za statično Huffmanovo kodiranje, ki uporablja po standardu definirano tabelo, ki se od bloka do bloka ne spremeni, ali gre za dinamično Huffmanovo kodiranje, pri katerem se tabela med bloki spreminja. Pri slednji opciji glavi sledi Huffmanova tabela. V večini primerov (v 99,5%) se uporablja možnost 10 - dinamično Huffmanovo kodiranje.

## 5.4 Ključni nizi

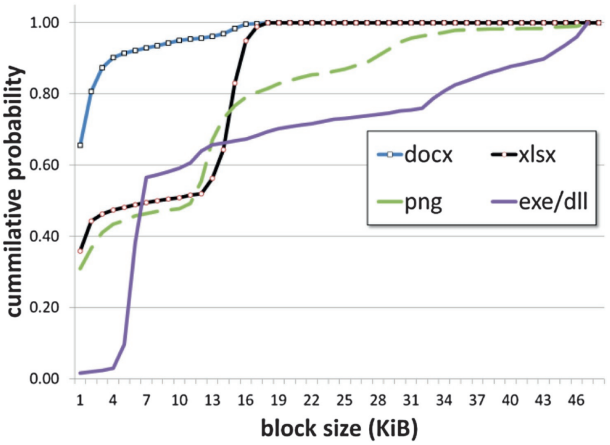
Učinkovita in preprosta metoda za prepoznavanje delčkov je iskanje ključnih nizov (n-gramov), ki so značilni za določen tip. V tem primeru je splošen zapis datoteke zip arhiv, vendar je stuktura datotek in map v njem za različne vrste dokumentov drugačna.

Docx datoteke vsebujejo 23 nizov v obliki (word/*), xlsx datoteke vsebujejo 22 nizov v obliki "xl/*", pptx datoteke pa vsebujejo 21 nizov v obliki (ppt/*).

Ideja je zelo dobra, vendar se izkaže, da imamo probleme pri datotekah, v katerih nastopajo velike komponente. Če dobimo le delček neke datoteke, je, v primeru da vsebuje velike komponente, precej majhna možnost, da bo ravno ta delček vseboval enega od ključnih nizov. Pri pptx, naprimer, je precej večja možnost, da bomo dobili košček kakšne slike.

## 5.5 Porazdelitev kompresiranih blokov

Naslednja stvar, ki nas zanima, je, kakšna je možnost, da bomo v delčku datoteke dobili cel deflate blok. Če ne dobimo celotnega deflate bloka, potem namreč ne moremo ničesar odkodirati. Če vzamemo delček docx dokumenta velikosti 1/4/16 KiB, imamo 65.6/90.2/99.6% možnosti, da bomo dobili celoten deflate blok. Pri xlsx dokumentu so te možnosti 35.9/47.4/94.9%. Če ne dobimo celotnega deflate bloka, potem ne moremo opraviti dekodiranja. Da skoraj zagotovo naletimo na cel deflate blok v delčku msx datoteke, mora biti ta delček velik 18 KiB.

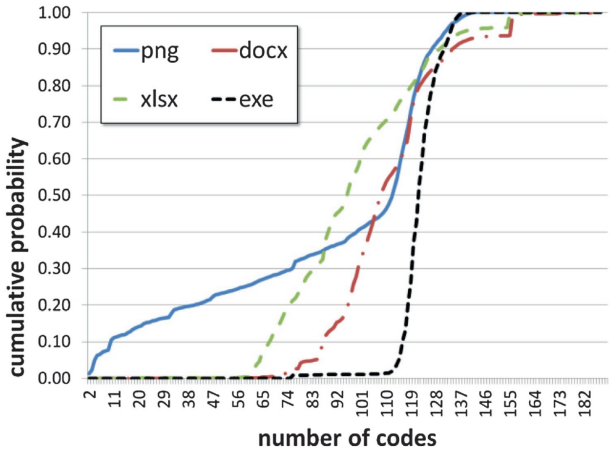



**Slika 2: Vpliv velikosti bloka na verjetnost zajema celotnega DEFLATE bloka.**

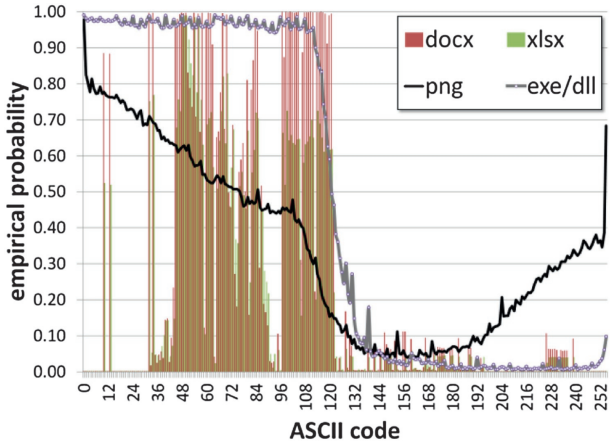
Poleg same velikosti bloka lahko pogledamo tudi njegovo vsebino. Najprej poglejmo kode Huffmanove tabele. Graf na sliki 3 prikazuje, kakšna je verjetnost, da gre za določen tip datoteke, glede na to, koliko kod je definiranih v Huffmanovi tabeli. Graf je kumulativen, torej se verjetnosti prištevajo. Iz grafa je razvidno, da je porazdelitev pri png datotekah precej širša kot pri ostalih. Png datoteke, predvsem tiste z majhno paleto barv, uporabljajo manjši nabor znakov, torej je tudi manj definiranih kod v Huffmanovi tabeli. To število nam torej že nekaj pove, vendar to še ni dovolj.

Na koncu se lahko posvetimo še sami vsebini tabele. Da bi lahko dekodirali vsebino, mora tabela definirati kodo za vsak potreben ASCII znak (0-255) ki nastopa v prvotnem bloku. Slika 4 nam za vsak ASCII znak prikazuje verjetnost, da je definiran v Huffmanovi tabeli. Pri docx/xlsx datotekah so očitni skoki med sosedi, saj te vrste datotek vsebujejo xml vsebino.

# 6. ANALIZA PNG DATOTEK




**Slika 3: Kumulativna verjetnost števila kod v Huffmanovi tabeli.**




**Slika 4: Empirične verjetnosti posameznih ASCII znakov glede na tip datoteke.**

Png je format za shranjevanje slik, ki pri svojem zadnjem koraku uporabi deflate kompresijo. Png pred uporabo te kompresije zakodira sliko tako da jo lahko deflate algoritem bolj učinkovito stisne. Namesto, da za vsako točko shrani barvo, png shranjuje le spremembe barve od točke do točke. Png podatki zaradi tega končnega koraka (deflate) sami po sebi nimajo nobenih posebnih statističnih značilnosti. Porazdelitev blokov je pri png datotekah bolj razprostrena. 81.5% kompresiranih blokov je velikih pod 18KiB, najdaljši bloki pa so dolgi 48 KiB. Delček, velik 48 KiB, bi moral vsekakor zadoščati za prepoznavo png datotek, če bi le imeli način prepoznave. Vsled ločevanja png od kompresiranih xml podatkov gledamo velikosti Huffmanovih tabel. Zanima nas, koliko različnih ASCII znakov je definiranih v Huffmanovi tabeli - torej koliko jih posledično nastopa v dejanskih nekompresiranih podatkih. Razlike so očitne. Če vzamemo v obzir datoteke png, docx, xlsx in kompresirane datoteke exe in najdemo blok, ki definira do 60 znakov, gre skoraj zagotovo za png. Če pa pri tem najdemo delček, ki definira 140 ali več znakov, pa skoraj zagotovo ni png. Pri tem je seveda odvisno, kakšne slike so predstavljene v formatu png. Pri fotografijah smo ponavadi manj občutljivi na malo zabrisane robove kot pri raznih vektorskih slikah in logotipih, ki pogosto nastopajo v dokumentih, zato fotografije shranjujemo v format jpeg, s katerim bolj prihranimo na prostoru. V fotografijah ponavadi nastopa več različnih barv in so bolj raznolike, logotipi in razni grafikoni pa so ponavadi bolj preprosti, z ostrimi robovi in z manj barvami. Posledično se jih da s png učinkovito stisniti. Če imamo raznoliko fotografijo, v kateri nastopa veliko barv, bo png kompresija manj učinkovita, prav tako pa bo v deflate blokih definirano večje število kod. Lahko rečemo, da je entropija fotografij večja kot entropija logotipov in drugih podobnih grafičnih materialov. Brezizgubna kompresija (kot naprimer deflate) najbolj učinkovito kompresira slike z nizko stopnjo entropije.

Če kompresiramo naprimer podatke tipa xml (xml, mathml, html..), pretežno potrebujemo znake, kot so “<”, “>”, “=”, abecedo, i.t.d., če kompresiramo logotip, pa si lahko predstavljamo, da je primer drugačen. Seveda je na tem mestu smiselno ponovno omeniti, da je lahko že samo besedilo kodirano na različne načine, preden je kompresirano z algoritmom deflate. Naslednja stvar, ki jo bomo gledali, je torej dejanska vsebina Huffmanovih tabel. Huffmanova tabela mora vsebovati vse ASCII znake, ki nastopajo v podatkih, ki so kompresirani. Pri png datotekah je ASCII koda 0 definirana v 97,78% primerov, pri docx/xlsx pa je ta koda definirana v 0,05% primerov. ASCII koda 115 (črka s) je v docx definirana v 99,91% primerov, v xlsx v 69% primerov in v png v 25,59% primerov. Z analizo definiranih črk v Huffmanovi tabeli torej lahko do precej dobro sklepamo na tip datoteke.

# 7. ZAKLJUČKI

Statistične metode nad samimi podatki niso dovolj, saj lahko datoteka vsebuje različna kodiranja (npr.: jpeg slika v docx dokumentu) in rekurzivna kodiranja (npr.: jpeg je kodiran z base64).

Velikost delčka lahko močno vpliva na točnost klasifikacije. V primeru, da je delček majhen, se soočamo s problemom, da imamo enostavno premalo podatkov, da bi lahko uspešno ugotovili vrsto vsebine, v primeru prevelikega delčka pa lahko pride do tega, da ta delček vsebuje različna kodiranja.

Če kot delček datoteke dobimo podatke, ki pripadajo sliki jpeg, bo pravilen rezultat klasifikacije jpeg, čeprav je morda ta slika del nekega dokumenta. Če v delčku nimamo dovolj podatkov iz tega dokumenta, potem nimamo nobenega načina, da bi iz teh podatkov ugotovili, da gre za sliko, ki je vsebovana v dokumentu.

Pri nekaterih tipih datotek se pojavljajo tudi večkrat zakodirani podatki - naprimer jpeg datoteka, kodirana z base64, v dokumentu docx. Take stvari so v praksi precej pogoste. Če izdelujemo orodje za prepoznavo delčkov, moramo take scenarije predvideti.

Vsako kodiranje ima svoje posebnosti, kombinacij kodiranj pa je precej. Za klasifikacijo delčkov datotek ni nekega splošnega postopka. Potrebno je uporabiti specializiran pristop in specifično obravnavati več različnih primerov. Pri izdelavi uporabnega orodja za klasifikacijo delčkov je smiselno v obzir vzeti kodiranja oziroma tipe datotek, ki se v svetu uporabljajo najbolj pogosto. Zaradi hitrega napredka v računalništvu se lahko v bližnji prihodnosti tudi na tem področju marsikaj spremeni - če se pojavijo novi zapisi podatkov,

moramo vsled tega posodobiti tudi program za prepoznavo delčkov datotek.

# 8. LITERATURA


[1] Vassil Roussev, Candice Quates. "File fragment encoding classificationd - An empirical approach", DFWRS 2013

[2] Antaeus Feldspar. "An Explanation of the Deflate Algorithm", dostopno na http://zlib.net/feldspar.html

[3] Qiming Li, Alvin Y. Ong, Ponnuthurai N. Suganthan, Vrizlynn L. L. Thing. "A Novel Support Vector Machine Approach to High Entropy Data Fragment Classification"

[4] DEFLATE - Wikipedia `http://en.wikipedia.org/wiki/DEFLATE`

[5] Zip (file format) - Wikipedia `http://en.wikipedia.org/wiki/ZIP_(file_format)`.

[6] JPEG - Wikipedia `http://en.wikipedia.org/wiki/JPEG`.

[7] RFC 1952 - GZIP file format specification version 4.3 `http://tools.ietf.org/html/rfc1952`.

# SSD: Začetek konca trenutne prakse v digitalni forenziki?

[Seminarska naloga pri predmetu
Računalniška forenzika]

Anže Rezelj
Fakulteta za računalništvo in informatiko

Janez Bindas
Fakulteta za računalništvo in informatiko

## ABSTRACT
Članek obravnava možnost začetka konca trenutne prakse v digitalni forenziki SSD diskov.V članku je podroben opis delovanja SSD diskov in obrazloženo zakaj obstoječi postopki ne morejo več veljati. Podane so tudi možne rešitve problema.

## Keywords
SSD, TRIM, forenzika, SLC, MLC, TLC

## 1. UVOD
V zadnjem času prihaja na tržišče vedno več SSD diskov, ki zamenjujejo navadne trdne diske. V tem članku se bomo osredotočili na forenzično preiskavo SSD diskov. Izkaže se, da standardne metode pri HDD diskih niso najboljše v primeru SSD diskov. Največji problem je v kontrolniku SSD diska, ki se na žalost od naprave do naprave spreminja. Posledično pomeni to, da ne moremo zagotoviti pristnosti podatkov na SSD diskih, kajti pri priklopu SSD diska lahko že pride do spreminjanje vrednosti na disku.

## 2. SSD DISK
SSD diski so narejeni iz NAND celic spominskih čipov, ki berejo in pišejo paralelno z več čipi na disk. NAND celice so zgrajene iz kovinskega-oksidnega pol-prevodnega spomina. Po zgradbi je podoben DRAM, EEPROM in NOR celici. SSD disk ne izgubi podatkov, če izgubi električno napajanje (za razliko od DRAM ali SRAM). Zaradi osnovne fizične lastnosti NAND celice, se pojavljaja več problemov za izdelavo SSD diskov. Na začetku je bil problem glede velikosti diska, ki so ga rešili s povečanjem gostote bralno/pisalnih mehanizmov, kar je privedlo do več-ravninske celice (ang. Multi-Level Cell (MLC)). Obstoj podatkov na nosilcu je bil tudi vprašljiv, kajti brisanje in pisanje je bilo omejeno na življenjsko dobo naprave. To težavo so proizvajalci rešili tako, da so izumili postopek pisanja v blok, tako da se je podatek samo enkrat napisal v blok, preden je bilo potrebno spet pisati v ta blok. Temu postopku se reče v angleščini "wear leveling". Ta postopek bi bil na običajnih trdih diskih zelo počasen. Ker SSD disk bere podatke simultano skozi več čipov in ne potrebuje premikanja glav, je ta postopek hiter.

### 2.1 Zgodovina SSD diska
Zgodovina SSD diskov sega v leto 1970, ampak da se je SSD disk razvil do takšne mere, da je bil zanimiv za trg, je trajalo še dosti let. Na začetku je bila tehnologija stroga tajnost, saj so jo uporabljali za vojaške namene in v visoko tehnoloških raziskovalnih laboratorijih. Seveda je bilo tudi veliko poskusov izdelave SSD diskov, ki so na žalost propadli.

V poznih 70. letih, je na trg prišel silicijev nitridni EAROM (električni spremenljiv ROM), ki ga je izdelal General Instrument. Izdelali so električno izbrisljiv blok, ki je potreboval 100 milisekund in 42V električni impulz, da se je izbrisal. Bralni čas naprave je bil 2 mikrosekunde, kar je bilo za tiste čase 4x počasnejše od ostalih naprav. Na žalost se je izkazalo, da življenjska doba ni dosegala niti 10 let. Zato so opustili raziskave na tem področju.

1976 je podjetje Dataram izdelalo SSD, ki se je takrat imenoval BULK CORE. SSD je bil priključen na miniračunalnik, ki je simuliral trdi disk. Naprava je lahko hranila 2 megabajta.

Leta 1978 bi gigabajt SSD-ja stal en milijon ameriških dolarjev. Texas Memory System je v tistem času predstavil 16 kilobajtni SSD disk, ki je temeljil na RAMu, za namene pospeševanja raziskav na področju seizmologije, še posebej za iskanje nafte.

V zgodnjih letih 1980, je Intel predstavil 1M bit tako imenovani mehurčkast spomin (ang. bubble memory), ki je povzročil veliko razburjenje, na področju SSD tehnologije. Intel je na žalost opustil raziskave, ker je ugotovil, da ni profitabilno, In zato je produkt šel v pozabo.

Leta 1982 je podjetje SemiDisk postalo prvo podjetje, ki je prodajalo SSD pospeševalnike za Intelove mikroprocesorje, ki so temeljili na PCih. Prvi emulatorji diskov so bili veliki 512 kilobajtov (stali so 1995$ )

### 2.2 Zgradba SSD diska

Jedro SSD diska predstavlja pomnilna celica, katere naloga je hranjenje vrednosti. Za razliko od trdega diska, je lahko SSD realiziran na veliko različnih načinov. Slednji lahko temeljijo na hitrih DRAM pomnilnikih in se uporabljajo v situacijah, kjer so kritične hitrost dostopa in visoke hitrosti prenosa podatkov. Najpogosteje pa se srečamo s pomnilnimi napravami, ki za hranenje podatkov ne potrebujejo dodatnega napajanja. Za pomnenje se tako uporablja MOSFET tranzistor s plavajočimi vrati, v katerih se shrani ustrezen naboj. Nadaljne ločimo dva (osnovna) načina vezave takih tranzistorjev, in sicer NOR (NOT OR oziroma negiran ali) in NAND (NOT AND oziroma negiran in), razvija pa se tudi že na vezavah, ki bi združevale prednosti obeh. Zaradi razmerja med ceno izdelave in gostote zapisa se je predvsem uveljavila NAND vezava, oziroma, kot NAND pomnilna celica. Raznolikost pa se tu ne ustavi. Zaradi napredka tehnologije sedaj poznamo SLC (single-level cell), ki hrani 1 bit na celico, MLC (multi-level cell), ki hrani 2 bita na celico, in TLC (triple-level cell), ki hrani 3 bite na celico.

## 2.3 Tipi pomnilnih celic

SLC predstavlja osnovo in lahko hrani le 1 bit informacije na celico - stanja 0 in 1. Dopušča tudi najvišje hitrosti pri branju in pisanju ter ima tudi najdaljšo življensko dobo, približno 100000 brisanj.(glej sliko 1)

Iz slike 1je tudi ravidno, da je MLC nadgradnja SLC in lahko hrani 2 bita na celico - stanja 00, 01, 10 in 11. Gledano na fizičnem nivoju se zgradba NAND celice pri MLC ne razlikuje od SLC, razlika je le v debelini izolatorja in načinu odčitavanja vrednosti v celici. SLC pri odčitavanju vrednosti uporablja le dva napetostna nivoja, glede na MLC, ki uporablja 4 napetosnte nivoje. Branje nato poteka tako, da se napetost na liniji postopoma dviguje in preverja izhod. Rezultat je večja kapaciteta vendar ima to svoje posledice. Namreč v primerjavi s SLC je branje približno enkrat daljše, saj mora namesto dveh napetostnih nivoje preveriti 4, programiranje (pisanje) pa je tudi do 3-4 krat počasnejše. Čas brisanja ostaja nespremenjen, zmanjša pa se življenska doba celice in sicer na približno 10000 brisanj.

TLC še dodatno poveča število bitov na celico. Sedaj lahko shranimo 3 bite informacije na celico - stanja 000, 001, 010, 011, 100, 101, 110, 111. Negativni efekti, ki so izraziti pri MLC, se tu še ojačajo. Posledično imamo še daljše čase branja in pisanja ter krajšo življensko dobo. Edina prednost je večja kapaciteta.

V odgovor na TLC pa se že razvija tehnologija, ki dopušča shranjevanje 4 bitov na celico [1] in v prihodnjosti verjetno še več.

## 2.4 Kapaciteta SSD diska

Ne glede na to, ali se uporablja SLC, MLC ali TLC pa je najmanjša enota, ki jo lahko SSD bere/piše, stran (*page*), ki so običajno velikosti 4KB (pri večjih kapacitetah se uporablja strani velikosti 8KB ali 16KB). 128 strani nato predstavlja blok (*block*) velikosti 512KB(glej sliko 2) in predstavlja tudi najmanjšo enoto, ki jo lahko SSD briše. V nadaljevanju nato 1024 blokov predstavlja polje (*plane*) velikosti 512MB. Primer prikazuje slika 2. Več polj skupaj nato predstavlja rezino (*die*), več rezin pa sestavlja čip. Na koncu SSD disk sestavlja eden ali več NAND čipov.

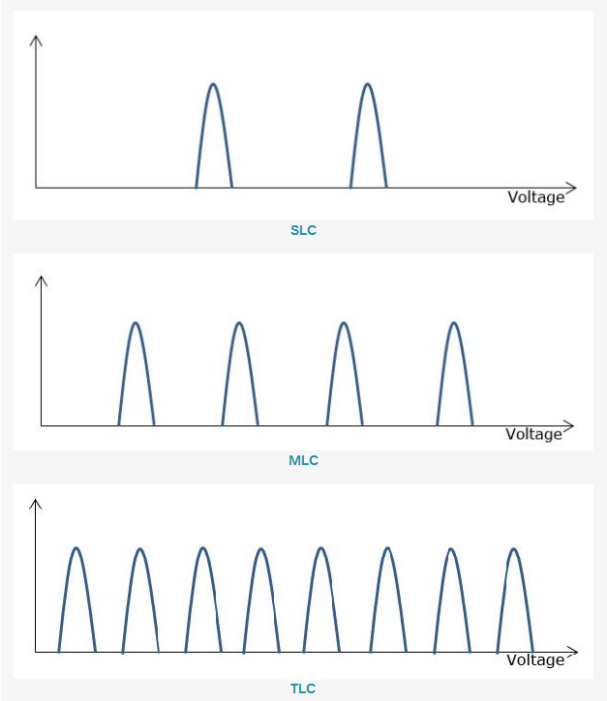



**Figure 1: Primer razporeditve napetostnih nivojev za branja stanje celice in pisanje novega stanja v celico.**

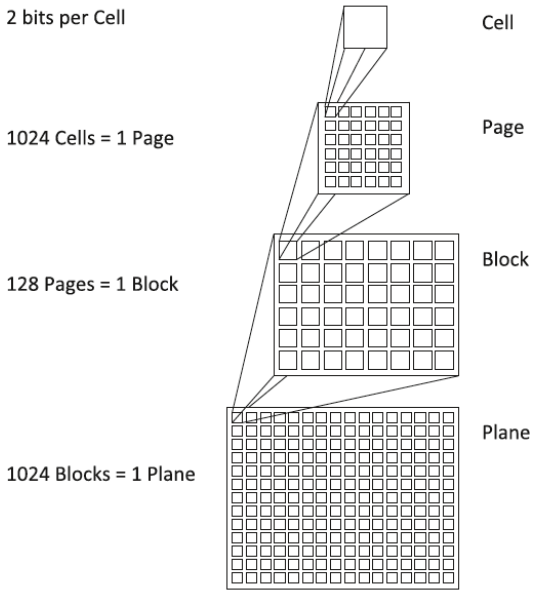



**Figure 2: Več, natančneje 16384, MLC NAND flash pomnilnih celic sestavlja stran (*page*) (tipično) velikosti 4KB, 128 strani nato sestavlja en blok (block) velikosti 512KB ter 1024 blokov naposled sestavlja ravnino (*plane*) velikosti 512MB.**

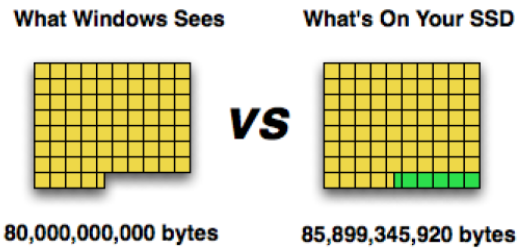



**Figure 3: Operacijski sistem ima na razpolago manjšo velikost, kot je disk dejansko velik.**

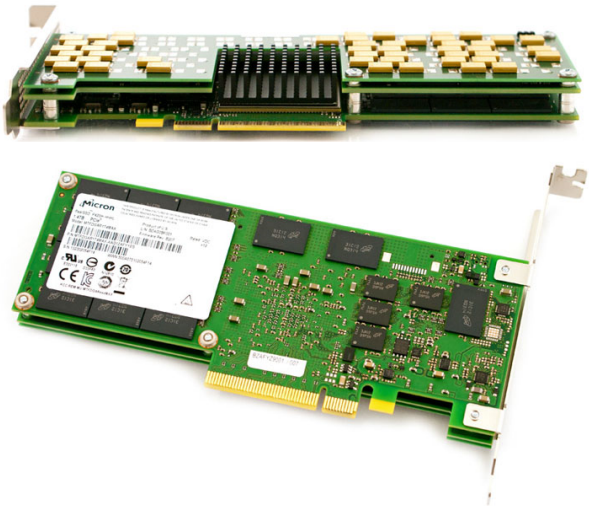


**Figure 4: Micron P420m Enterprise PCIe. Nazivna velikost diska znaša 1.4TB, dejanska velikost pa znaša 2TB (2048GB).**

Za razliko od klasičnih trdih diskov, kjer velja da je 1GB = 1000MB, pri SSD diskih velja, da je 1GB = 1024MB, kar neposredno sledi iz zgradbe. Kljub temu pa se SSD operacijskemu sistemu predstavi, kot da ima manjšo kapaciteto, pri čemer uporablja notacijo klasičnega trdega diska (primer na sliki 3). Razlogov za to je več. Vsaka pomnilna celica ima omejeno življensko dobo in ko postane celica nedelujoča, se cel blok označi kot poškodovan in ga "nadomesti" novi blok. Ravno tako proizvodnja čipov ni brez napak in se lahko že pred prvo uporabo pojavijo nedelujoče celice. S prikrivanjem dejanske velikosti SSD diska proizvajalec doseže nižje prodajne cene, saj čipa s poškodovanim blokom ni potrebno zavreči, poleg tega pa tudi daljšo življensko dobo SSD diska, saj ima disk kar nekaj rezervnih blokov. Recimo, da imamo SSD disk nazivne velikosti 256GB. OS to vidi (približno) kot 256.000.000.000B, v resnici pa je celotna kapaciteta diska 274.877.906.944B. OS tako vidi le približno 93% dejanske kapacitete, procent pa se lahko pri nekaterih proizvajalcih spusti celo do 70%[7] (slika 4). Slednje pa predstavlja oviro pri trenutno uveljavljeni praksi o digitalni forenziki na trdih diskih.

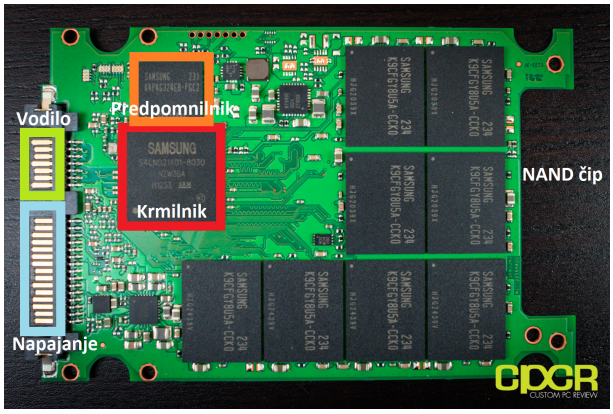



**Figure 5: Drobovje SSD diska podjetje Samsung, modela Samsung 840 250GB. Opremljen je s trojedrnim krmilnikom MDC Cortex-R4 (ARM Cortex R4 frekvence 300MHz) in 512MB LPDDR2 predpomnilnik. Pomnilni čip je tipa TLC NAND izdelan na 21nm tehnologiji.**

### *2.4.1 Krmilnik in prepomnilnik*

V novejših SSD diskih, kot tudi v vseh ostalih pomnilnih napravah temelječih na flash tehnologiji, krmilnik predstavlja pomembno vlogo(slika 5). Od njega je odvisna delovanje in življenska doba diska, saj skrbi za veliko kritičnih operacij, brez katerih SSD diski danes ne bi bili to, kar so. Glavne funkcije so tako:

- Izvaja branje, pisanje in brisanje v pomnilnih celicah,
- Mapiranje LBA (Logical Block Addressing) in PBA (Physical Block Addressing) naslovov,
- Enakomerno izravnavanje obrabe celic (wear levelling),
- Smetar (garbage collector),
- Odpravljanje napak (error correction),
- Enkripcija.

Zaradi implementacije krmilnika računalnik (ali druga naprava) ne dostopa direktno do pomnilnih celic, ampak svoje zahteve sporoča krmilniku, ki nato zahteve ustrezno izvrši. Lahko tudi rečemo, da je krmilnik namenski računalnik, ki opravlja točno določene funkcije, slednje pa, kot bomo videli v nadaljevanju, predstavljajo veliko oviro pri digitalni forenziki.

Krmilnik pa ima za svoje delovanje na voljo lahko tudi predpomnilnik, ki je lahko vgrajen v disk ali pa za predpomnenje koristi sistemski pomnilnik.

## 2.5 Delovanje SSD diska

Kot smo že omenili, da se SSD disku podaljša življenjska doba, ta deluje drugačne kot trdi diski, za kar skrbi programska oprema v vgrajenem krmilniku. Žal ta programska oprema ni prosto dostopna in se tudi od proizvajalca do

**Table 1: Razlike pri operacijah med trdim diskom (HDD) in SSD diskom.**

| Operacija | HDD | SSD |
|---|---|---|
| Branje | Prebere sektor | Prebere stran |
| Pisanje | Zapiše na sektor | Zapiše na stran |
| Prepis | Prepiše sektor | Prostor na voljo: Zapiše na prosto stran<br>Prostor ni na voljo: Kopira blok, spremeni, izbriše, zapiše |
| Izbris | Ne naredi nič | Ne naredi nič (razen če prejme ukaz TRIM, potem krmilni ustrezno ukrepa) |

**Table 2: Hipotetični SSD disk.**

| | Vrednosti |
|---|---|
| **Velikost strani** | 4KB |
| **Velikost bloka** | 5 strani (20KB) |
| **Velikost diska** | 1 blok (20KB) |
| **Hitrost branja** | 2KB/s |
| **Hitrost pisanja** | 1KB/s |

proizvajalca zelo razlikuje (na primer različni algoritmi za razprševanje, smetar in podobno), kljub temu pa vsi sledijo osnovnim principom.

### 2.5.1 Pisanje in branje podatkov

Osnovni operaciji vsakega pomnilnega medija predstavljata cikel pisanja in branja. Obe operaciji pa se pri SSD diskih in ostalih podobnih medijih, izvedeta podobno, kot pri trdem disku, razlika pri delovanju pa je najbolj očitna pri prepisu podatkov, kar prikazuje tabela 1.

Pisanje in branje se pri HDD in SSD ne razlikujeta bistveno, razlika pri delovanju se pojavi v primeru prepisa ali izbrisa podatkov. Pri pisanju podatkov v flash pomnilnik pravimo, da programiramo pomnilnik, za izvedbo pa morajo celice, katere bomo programirali, izpolnjevati določene pogoje. Te celice morajo biti predhodno prazne (vsebovati logično 0), saj pri programiranju v celice vprogramiramo samo logično 1. Če celice predhodno niso postavljene na logično 0, shranjeni podatki ne bodo veljavni, po programiranju novih podatkov. Da je prepis uporabljene strani veljaven, mora krmilnik izvesti nekaj korakov. Najprej prebere celoten blok v predpomnilnik, pobriše blok, v predpomnilniku združi predhodno veljavne podatke iz bloka in nove podatke, ter jih zapiše v predhodno pripravljen blok. Prepis je časovno najpočasnejša operacija, saj je brisanje enega bloka veliko počasnejše od pisanja in branja. Pri dolgi uporabi diska se za pisanje novih podatkov dejansko izvaja prepis obstoječih, zaradi česar hitrost pisanja drastično pade. Poleg tega se tu pojavi tudi težava zaradi obrabe pomnilnih celic, ki imajo omejeno življensko dobo. Za podaljšanje življenske dobe diska krmilnik skrbi tudi za čim bolj enakomerno obrabo vseh celic.

Za podrobnejše razumevanje si poglejmo naslednji primer (povzet po [5]). Imamo hipotetični SSD disk s specifikacijami, prikazan v tabeli 2.

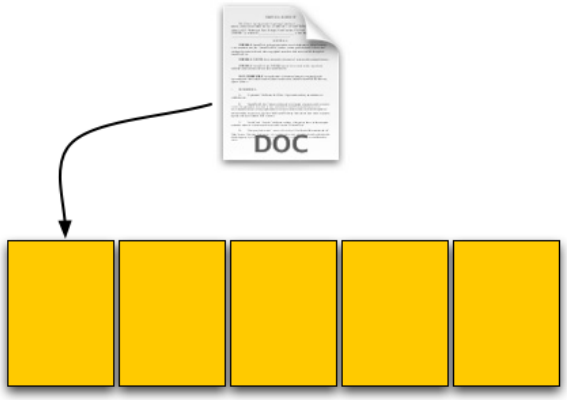



**Figure 6: Hipotetični SSD. Rumene lokacije označujejo prazne strani, ki so priravljene za takojšno uporabo.**

Na začetku je disk prazen, vsebuje le logično 0, in želimo shraniti tekstovni dokument velikosti 4KB. Dokument se shrani v prosto stran (slika 6), ki jo določi preslikovalna tabela v krmilniku.

Nato želimo shraniti sliko velikosti 8KB. Ker je še dovolj praznih strani krmilnik shrani sliko na prosti lokaciji (slika 7).

Naposled bi radi shranili še eno sliko, velikosti 12KB. Ker imamo premalo prostora, se odločimo, da tekstovnega dokumenta ne potrebujemo (glej sliko8) in ga izbrišemo. Operacijski sistem sedaj javi, da imamo na disku 12KB, kar je ravno dovolj za novo sliko. Krmilnik ponovno prejme ukaz in podatke za shranjevanje. Sedaj pa krmilnik ugotovi, da ena fizična lokacija, ki jo je dodelila preslikovalna plast (LBA), ni prazna. Čeprav smo predhodno dokument izbrisali, zaradi delovanja datotečnega sistema krmilnik te informacije ni prejel in se dokument zato še vedno nahaja na disku (slika 8).

Da lahko krmilnik shrani novo sliko (slika 9), mora najprej zbrisati tekstovni dokument (1). To stori tako, da vsebino celotnega bloka prebere in shrani v predpomnilnik (2). V predpomnilniku nato odstrani neveljavne podatke (3) in doda podatke, ki jih želimo shraniti (4). Pred pisanjem podatkov iz predpomnilnika nazaj v pomnilnik, krmilnik predhodno pobriše celoten blok, čemur sledi še pisanje podatkov v blok (5).

V poglavju 2.4 smo omenili, da je dejanska velikost SSD diska večja, kot jo vidi operacijski sistem. Navzven skrito kapaciteto krmilnik uporablja za učinkovitejšo izrabo diska. Na primer, recimo da ima naš hipotetični SSD disk dodatno stran(slika 10), ki pa jo operacijski sistem ne vidi. Namesto da bi opravil časovno potraten prepis, bi kos slike shranil v "skriti" del pomnilnika. Zapisovanje se tako v tem primeru izvede najhitreje.

### 2.5.2 Napredne funkcije krmilnika

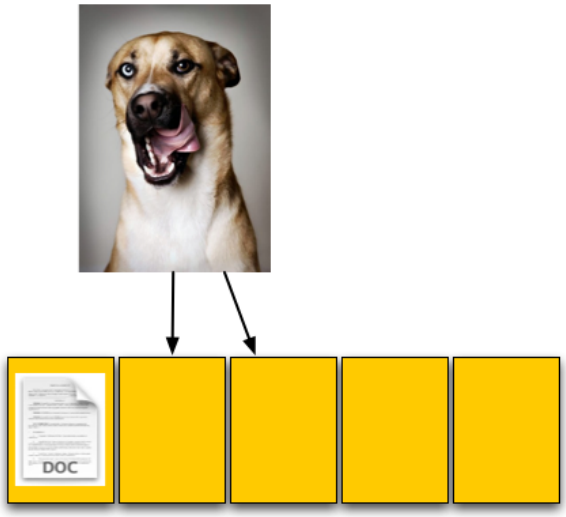



Figure 7: Večje datoteke lahko zasedejo več strani. V primeru datotek manjših od ene strani, datoteka zasede le tolikšen del strani kot je velika, ostalo pa je neizkoriščen prostor.




Figure 8: Hipotetični SSD. Rumene lokacije označujejo prazne strani, ki so pripravljene za takojšnjo uporabo.

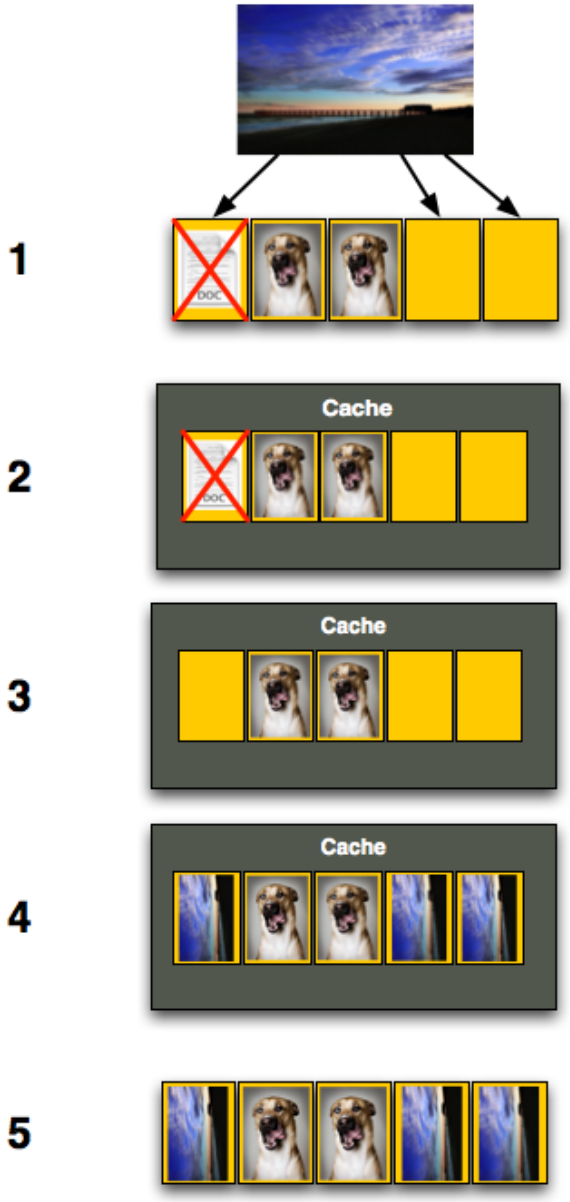



Figure 9: Primer prepisovanja podatkov. Cikelj prepisa podatkov sestavljajo kopiranje, sprememba, izbrisa in pisanje veljavnih podatkov.

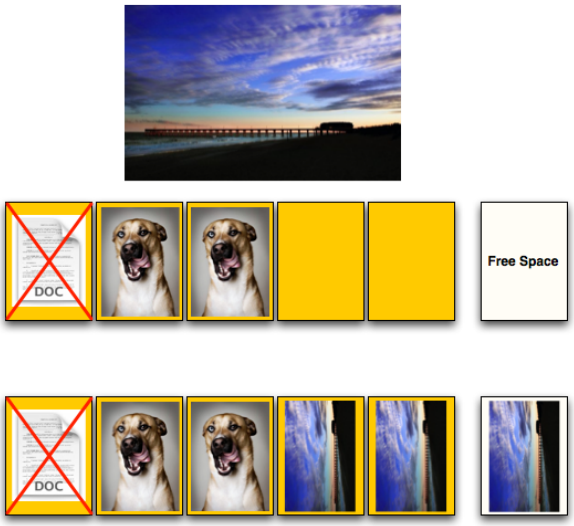



**Figure 10: Krmilnik določen del kapacitete skrije tako, da se ga operacijski sistem ne zaveda. Skriti del pa krmilnik uporablja za ohranjanje performance in daljšo življensko dobo.**

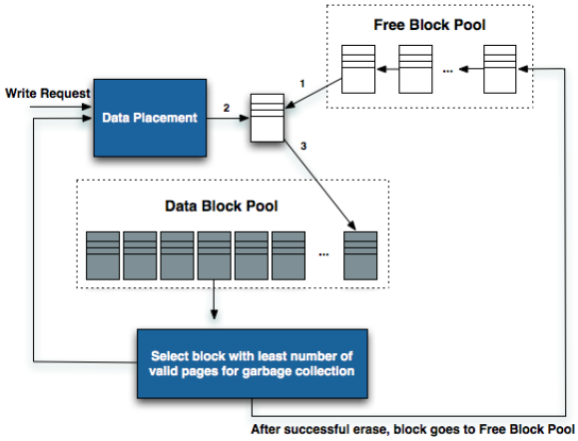



**Figure 11: Primer delovanja smetarja. Smetar išče bloke z najmanj veljavnimi stranmi, podatke ustrezno uredi, veljavne strani združi in zapiše v nov blok, ter nove proste bloke doda v bazen praznih blokov. Pri novem pisanju nato krmilnik iz bazena prostih blokov izbere bloke z najmanj opravljenimi brisanj in zapiše podatke.**

Poleg pisanja in branja sodobni krmilniki za zagotavljanje čim boljših hitrosti operacij in čim daljšo življensko dobo, izvajajo operacije za čim bolj enakomerno obrabo celic. Poleg tega skrbijo, da je vedno na voljo čim več prostih strani pripravljenih za pisanje, kriptiranje podatkov in podobno. Za to krmilniki uporabljajo razne napredne algoritme, ki se lahko od proizvajalca do proizvajalca razlikujejo in so poslovna skrivnost. Kljub temu pa algoritmi sledijo osnovnim načelom. Za čim bolj enakomerno obrabo celic (wear levelling) krmilnik podatke ustrezno razprši po celem pomnilniku. To pomeni, da logični naslovi vedno ne kažejo na isto lokacijo v pomnilniku. V ta namen krmilnik uporablja preslikovalno tabelo, ki logične naslove preslika v fizične (LBA to PBA).Vendar proces čim bolj enakomerne obrabe celic le ni tako enostaven kot se sliši. Na disku se hranijo različni tipi podatkov, od slik, filmov do začasnih datotek. Pomembna razlika med njimi, ki jo občuti SSD disk in ostali flash pomnilniki, je pogostost spreminjanja in ponovnega pisanja. Tako na primer le enkrat shranimo film (cold data), medtem, ko se začasne datoteke sorazmerno pogosto spreminjanjo (hot data) in posledično v istem času tolikokrat prepišejo pomnilnik, ja je kapaciteta ponovnih shranjevanj nekajkrat presegla kapaciteto shranjenega filma. Zato krmilnik poskrbi, da se vsake toliko časa podatki, ki se malo spreminjajo, prepišejo na drugo fizično lokacijo (logično so še vedno na isti lokaciji).

Zaradi razprševanja datotek so deli datotek v straneh, ki niso nujno vse v istem bloku, čeprav je datoteka ravno velikosti enega bloka, pravzaprav je lahko v najslabšem primeru vsaka stran v drugen bloku. Tu se pojavi težava, ko želimo spremeniti vsebino tako razpršene datoteke, saj se morajo posodobiti vse strani v teh blokih [referenca!!] in ne samo en blok. Podobna situacija nastane tudi pri brisanju takšne datoteke. Posledice predstavljajo daljši časi pisanj ter občutno krajša življenska doba. Da se temu čim bolj izognemo, krmilnik uporabi izravnalnik obrabe in smetarja (garbage collector). Naloga smetarja (slika 11) je, da poišče bloke s čim več neveljavnimi stranmi, veljavne strani združi in zapiše v novi blok, nove proste bloke pa postavi v bazen prostih blokov (free block pool) iz katerega nato, glede na obrabo, krmilnik izbira bloke za pisanje.

Ker pa vse napredne funkcije v času izvajanja zmanjšujejo hitrost pisanja in branja, se te izvajajo v ozadju in so skoraj neopazne.

# 3. FORENZIKA DISKOV

V tem poglavju bo govora o forenziki na trdih diskih in SSD diskih.

## 3.1 Forenzika na trdih diskih (HDD)

Zgradba trdega diska je prikazana na sliki 12 Forenzična praksa pri trdih diskih temelji na naslednjih korakih:

- Pridobitev dokazov,
- Preverjanje pristnosti dokazov,
- Analiza.

Pridobitev dokazov v primeru računalniške forenzike temelji na pridobitvi vsebine trdega diska. Analizo vsebine ponavadi ne izvajamo direktno na osumljenem računalniku, ampak se izvaja na kopiji. To storimo zato, da lahko preprečimo prirejanje in spreminjanje podatkov na osumljenčevem disku. Kopije diskov so narejene v tako imenovanem "bit by bit" kopiranjem. Temu postopku rečemo v angleščini tudi "bit-stream backup". Razlog za kopiranje diska na tak način je v tem, da zagotovimo celovito ohranjanje podatkov iz diska. V nasprotnem primeru ne bi kopirali nealociranega dela diska

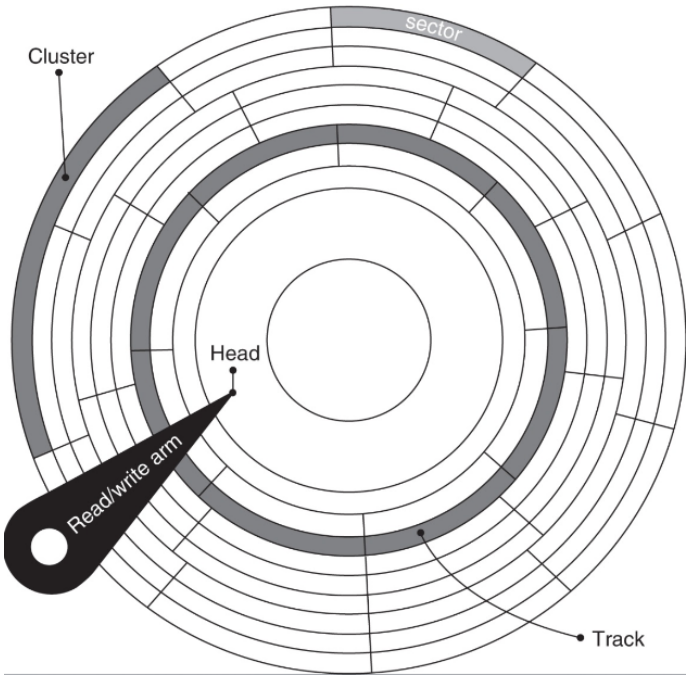



**Figure 12: Sestava trdega diska.**

(kot so izbrisane datotek), swap prostora in slabih sektorjev. V vseh teh delih diska bi se namreč lahko nahajali forenzični dokazi.

Pomemben korak pri forenziki je zagotavljanje pristnosti dokazov, ki zagotavlja, da dokazi niso bili spremenjeni tekom pridobitve podatkov. Kajti spreminjanje dokazov bi bilo nesprejemljivo na sodišču in bi jih lahko sodišče zavrnilo. Preverjanje pristnosti se izvaja tako, da se naredi kontrolna vsota (ang. checksum), ki je na nek način prstni odtis diska. S primerjanjem kontrolne vsote med diskom in kopijo diska, lahko trdimo v primeru, če sta enaki, da je kopija identična disku.

Analiza dokazov je tretji korak v preiskavi, ki je ponavadi najbolj časovno potraten. Pri tem postopku se uporabljajo različni forenzični programi, kot so na primer EnCase, Scalpel in še mnogi drugi programi.

## 3.2 Forenzika na SSD diskih

Tu bomo povzeli rezultate, ki so jih dobili avtorji članka [ 10] iz leta 2011. Pri eksperimentu so se avorji osredotočili na tri operacijske sisteme, Windows 7, XP in Ubuntu 9.04, na dva scenarija uporabe in formatiranje.

### *Scenarij z visoko uporabo (zasedenostjo) diska*

S tem scenarijem so želeli simulirati uporabnika, ki ima disk skoraj v celoti zaseden z datotekami, kot so na primer filmi, glasba in podobno. Operacijski sistem ima tako na voljo zelo malo prostora, ravno tako pa ima ogromno dela tudi krmilnik (predvsem smetar), ki mora neprestano preiskovati celoten pomnilnik in polniti bazen praznih blokov. Disk so popisali z velikim številom binarnih datotek, za čim boljše prilagajanje postavljenemu scenariju.

### *Scenarij z nizko uporabo (zasedenostjo) diska*

Scenarij nizke zasedenosti je ponazoril na novo postavljen sistem, na katerem se nahaja le operacijski sistem in nekaj uporabniških datotek, pri čemer je bil disk minimalno uporabljen. Namen scenarija je preveriti, kako se obnaša smetr v zanj idealni situaciji.

### *Formatacija diska*

Pri tem scenariju so preverjali, koliko podatkov ostane na disku po izvedeni formataciji diska. Predpostavljali so, da povprečni uporabnik uporablja privzete nastavitve za formatiranje, kar za Windows 7 in Ubuntu predstavlja hitra formatacija (quick format), le XP privzeto uporablja polno formatacijo.

Eksperimenti so bili sestavljeni tako, da so testirali naslednje tri hipoteze:

1. Izbrisani podatki pri disku z visoko zasedenostjo bodo imeli manjšo verjetnost, da bodo obnovljeni, saj bo smetar bloke najverjetneje pripravil za nadaljno uporabo.

2. Izbrisani podatki pri disku z nizko zasedenostjo bodo imeli večjo verjetnos, do bodo obnovljeni, saj bo imel smetar veliko manj dela in na voljo veliko več blokov.

3. Obnova podatkov izbrisanih med formatiranjem diska bo zelo odvisna od delovanje programske opreme krmilnika.

Uporabili so 15 različnih SSD diskov in en HDD za referenco. Od vseh 15-ih diskov jih je 6 podpiralo ukaz TRIM, vsi pa so uporabljali MLC pomnilne celice. Uporabili so tri različne operacijske sisteme, Windows 7, XP in Ubuntu 9.04, toda le Windows 7 je uporabljal ukaz TRIM. Teste so izvedli na vsakem disku posebaj, za vsak operacijski sistem in za vsak scenarij. Pred uporabo so vsak disk temeljito pobrisali in prepisali z ničlami ter na to namestili operacijski sistem in dve referenčni datoteki - največja je bila binarna datoteka (650MB), ki je vsebovala neponavljajočo alfanumerično sekvenco in tekstovni dokument velikosti 900KB. Datoteki sta služili kot referenci pri primerjanju z obnovljenimi bloki

### *3.2.1 Rezultati testiran*

Rezultate vseh testov prikazuje slika 13. Hitri pregled tabele pove, da je kar v veliko primerih možno obnoviti vse ali pa vsaj večji del referenčnih datotek. Najmanj možnosti za uspešno obnovitev datotek zasledimo le pri kombinaciji uporabe operacijskega sistema Windows 7 in SSD diska, ki podpira ukaz TRIM, kar nazorneje prikazuje slika 14. Koliko vpliva uporaba ukaza TRIM na obnovo podatkov opazimo, če primerjamo sliki 14 in 15.

### *Preverjanje hipoteze 1*

Pri prvi hipotezi so predvidevali, da bodo izbrisani podatki pri disku z visoko zasedenostjo težko obnovljivi zaradi delovanja smetarja. Hipoteza se izkaže za pravilno le pri uporabi operacijskega sistema in SSD diska, ki uporabljata ukaz TRIM,

| Test | Control | Intel1 | Intel2 | Intel3 | OCZ1 | Corsair1 | Crucial1 |
|---|---|---|---|---|---|---|---|
| Large File, Low Usage | 100.00% | 0.00% | 0.00% | 0.00% | 0.00% | 0.00% | 0.00% |
| Large File, High Usage | 100.00% | 0.00% | 0.00% | 0.00% | 0.00% | 0.00% | 0.00% |
| Large File, Format | 100.00% | 0.00% | 0.00% | 0.00% | 0.00% | 0.00% | 0.00% |
| Small File, Low Usage | 99.98% | 0.00% | 0.00% | 0.00% | 25.53% | 25.53% | 27.54% |
| Small File, High Usage | 99.98% | 0.00% | 0.00% | 0.00% | 25.53% | 27.54% | 26.28% |
| Small File, Format | 0.00% | 0.00% | 0.00% | 0.00% | 0.00% | 24.46% | 0.00% |

This chart shows only those disks that support TRIM (and the control), which is 6 of the 16 tested.

**Figure 14: Procent uspešno obnovljenih izbrisanih podatkov na SSD diskih s podporo ukaza TRIM v naboru z Windows 7.**

| Test | Control | Intel1 | Intel2 | Intel3 | OCZ1 | Corsair1 | Crucial1 |
|---|---|---|---|---|---|---|---|
| Large File, Low Usage | 99.99% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% |
| Large File, High Usage | 99.99% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% |
| Large File, Format | 99.99% | 100.00% | 0.00% | 0.13% | 100.00% | 0.00% | 0.00% |
| Small File, Low Usage | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% |
| Small File, High Usage | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% |
| Small File, Format | 96.57% | 59.84% | 86.20% | 46.76% | 49.03% | 24.69% | 0.00% |

This chart shows only those disks that support TRIM (and the control), which is 6 of the 16 tested.

**Figure 15: Procent uspešno obnovljenih izbrisanih podatkov na SSD diskih s podporo ukaza TRIM v naboru z XP, ki nima podpore za ukaz TRIM.**

pa še to le pri največji datoteki (slika 14). Pri obnovi manjše datoteke jim je pri diskih (proizvajalca) Corsair(1), Crucial(1) in OCZ(1) uspelo obnoviti skoraj 30% manjše datoteke (slika 14). Pri diskih in operacijskih sistemih, ki ne podpirajo ukaza TRIM, pa je bila obnova obeh datotek skoraj vedno 100% (sliki 13 in 15). Rezultati torej zavračajo hipotezo, saj je v večini primerov možno obnoviti skoraj vse izbrisane podatke.

### *Preverjanje hipoteze 2*

Pri preverjanju te hipoteze opazimo, da so rezultati skoraj enaki kot pri prvi hipotezi, le pri nekaterih diskih je opaziti manjši procent uspešnosti. Rezultati potrjujejo postavljeno hipotezo, saj je v večini primerov možno obnoviti skoraj vse podatke, pri diskih Corsair(1), Crucial(1) in OCZ(1) celo slabih 30%, čeprav je bil uporabljen ukaz TRIM.

### *Preverjanje hipoteze 3*

Pri analizi rezultatov obnovljenih datotek po formatiranju opazimo da so od primera do primera zelo razlikujejo. Pri uporabi ukaza TRIM v večini primerov datoteke ni bilo možno obnoviti, pri ostalih primerih, kjer se ukaz TRIM ni uporabljal, pa je bila uspešnost obnove vse od 0% do 100%. Tudi pri uporabi privzetih nastavitev za formatiranje na operacijskem sistemu XP, ki opravi celoten format diska, dobimo procent uspešnosti pri obnovi od 0% do 100%. Podobno velja tudi za Ubuntu, ki privzeto uporablja hitro formatiranje. Oba operacijska sistema uporabljata različni datotečni sistem in ne podpirata ukaza TRIM. Rezultati toraj podpirajo hipotezo, da je pri (privzetih) nastavitvah rezultat formatiranja zelo odvisen od delovanja krmilnika.

## 4. PRIHODNOST FORENZIKE SSD DISKOV

Kot smo videli v poglacju 3.2, v primeru uporabe operacijskega sistema, ki podpira in ima omogočen ukaz TRIM, po opravljeni obnovi izbrisanih datotek zelo težko dobimo kaj uporabnega. Seveda to ni odvisno le od operacijskega sistema ampak tudi uporabljenega datotečnega sistema in kako je napisana programska oprema, ki se izvaja na krmilniku. Na mobilnih napravah so populari datotečni sistemi specifično napisani za tovrstne pomnilne medije, vprašanje je le, ali je smiselno take datotečne sisteme uporabiti tudi pri SSD diskih.

Trenutno pa največjo oviro pri opravljanju digitalne forenzike na SSD diskih predstavlja krmilnik. Namreč, kako naj preiskovalec zagotovi, da se ob priklopu diska vsebina ne bo spremenila, pri čemer želimo, da se ohranijo tudi vse strani z vsebino morebitno izbrisanih datotek. Povedano drugače, kako naj zagotovimo enako stopnjo verodojnosti opravljene analize na SSD diskih kot jo lahko na trdih diskih.

### 4.1 Ukaz TRIM

Ukaz TRIM so uvedli zaradi načina delovanja datotečnih sistemov in drugačnega delovanja SSD diskov od trdih diskov. Pri SSD mora biti prostor, kamor se bo zapisal podatek, pred zapisom prazen, vsebovati mora logične ničle, saj se pri pisanju in programiranju v podano lokacijo zapišejo samo logične enice. Posledično hitrost zapisovanja novih podatkov pada, saj mora krmilnik zbrisati podatke na lokaciji in nato šele zapisati nove podatke, brisanje pa je veliko počasnejše kot pisanje ali branje. Da bi se performance diska ohranile in tudi po dolgi uporabi ostale take, kot na začetku uporabe diska,so naboru ukazov ATA8 [3] dodali še ukaz TRIM [2].

Omenili smo že, da je bil razlog za vpeljavo ukaza tudi delovanje datotečnih sistemov. Najpogosteje uporabljeni datotečni sistemi na računalnikih so bili razviti primarno za gibke in trde diske. Podrobneje, oviro predstavlja, kako datotečni sistem ravna z datotekami, ki jih želimo izbrisati.

| Test | Control | Imation1 | Corsair1 (TRIM) | Crucial1 (TRIM) | PQI1 | RiData1 | OCZ1 (TRIM) | OCZ2 | Patriot1 | Patriot2 | Kingston1 | Intel1 (TRIM) | Intel2 (TRIM) | Intel3 (TRIM) | Intel4 | Transcend1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Large File, Low Usage, Win7 | 100.00% | 100.00% | 0.00% | 0.00% | 71.23% | 100.00% | 0.00% | 100.00% | 100.00% | 100.00% | 100.00% | 0.00% | 0.00% | 0.00% | 100.00% | 70.49% |
| Large File, Low Usage, WinXP | 99.99% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% |
| Large File, Low Usage, Linux | 99.99% | 99.98% | 100.00% | 99.99% | 99.98% | 100.00% | 100.00% | 100.00% | 100.00% | 99.98% | 99.99% | 99.98% | 99.98% | 100.00% | 99.98% | 100.00% |
| Large File, High Usage, Win7 | 100.00% | 100.00% | 0.00% | 0.00% | 88.60% | 100.00% | 0.00% | 100.00% | 100.00% | 100.00% | 100.00% | 0.00% | 0.00% | 0.00% | 100.00% | 100.00% |
| Large File, High Usage, WinXP | 99.99% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% |
| Large File, High Usage, Linux | 100.00% | 99.98% | 100.00% | 100.00% | 99.98% | 99.98% | 100.00% | 100.00% | 99.99% | 34.78% | 100.00% | 99.99% | 0.00% | 99.98% | 99.99% | 100.00% |
| Large File, Format, Win7 | 100.00% | 100.00% | 0.00% | 0.00% | 100.00% | 100.00% | 0.00% | 99.87% | 100.00% | 0.00% | 0.00% | 0.00% | 0.00% | 0.00% | 100.00% | 100.00% |
| Large File, Format, WinXP | 99.99% | 100.00% | 0.00% | 0.00% | 0.00% | 99.67% | 100.00% | 0.00% | 100.00% | 100.00% | 0.00% | 100.00% | 0.00% | 0.13% | 99.40% | 99.67% |
| Large File, Format, Linux | 99.86% | 100.00% | 0.00% | 77.88% | 0.00% | 3.83% | 99.87% | 100.00% | 12.37% | 7.36% | 3.83% | 83.52% | 100.00% | 100.00% | 0.00% | 100.00% |
| Small File, Low Usage, Win7 | 99.98% | 0.00% | 25.53% | 27.54% | 0.00% | 99.98% | 25.53% | 99.98% | 99.98% | 0.00% | 99.98% | 0.00% | 0.00% | 0.00% | 99.98% | 99.98% |
| Small File, Low Usage, WinXP | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% | 0.00% | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% |
| Small File, Low Usage, Linux | 99.98% | 99.98% | 99.98% | 99.98% | 99.98% | 99.98% | 99.98% | 86.20% | 99.98% | 99.98% | 99.98% | 99.98% | 99.98% | 99.98% | 96.97% | 99.98% |
| Small File, High Usage, Win7 | 99.98% | 99.98% | 27.54% | 26.28% | 99.98% | 99.98% | 25.53% | 99.98% | 99.98% | 99.98% | 99.98% | 0.00% | 0.00% | 0.00% | 99.98% | 99.98% |
| Small File, High Usage, WinXP | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% | 96.57% |
| Small File, High Usage, Linux | 99.98% | 99.98% | 99.98% | 99.98% | 99.98% | 99.98% | 99.98% | 99.98% | 99.98% | 98.71% | 99.56% | 99.98% | 99.98% | 99.98% | 98.27% | 99.98% |
| Small File, Format, Win7 | 0.00% | 26.96% | 24.46% | 0.00% | 0.00% | 24.63% | 0.00% | 0.00% | 0.00% | 0.00% | 0.00% | 0.00% | 0.00% | 0.00% | 24.61% | 0.00% |
| Small File, Format, WinXP | 96.57% | 57.21% | 24.69% | 0.00% | 0.00% | 54.27% | 49.03% | 53.05% | 46.84% | 0.00% | 37.73% | 59.84% | 86.20% | 46.76% | 58.45% | 48.20% |
| Small File, Format, Linux | 99.98% | 99.98% | 0.00% | 99.98% | 99.98% | 0.00% | 99.98% | 99.98% | 99.98% | 0.00% | 99.98% | 27.41% | 99.98% | 99.98% | 0.00% | 99.98% |
| Average | 93.86% | 87.62% | 49.74% | 56.93% | 69.60% | 81.97% | 66.28% | 85.12% | 86.23% | 57.63% | 74.12% | 59.10% | 54.40% | 57.77% | 81.71% | 89.52% |

**Figure 13: Rezultati po izvedenih testih. Prikazan je scenarij, operacijski sistem in vrednost, ki predstavlja koliko procentov blokov je bilo obnovljenih.**

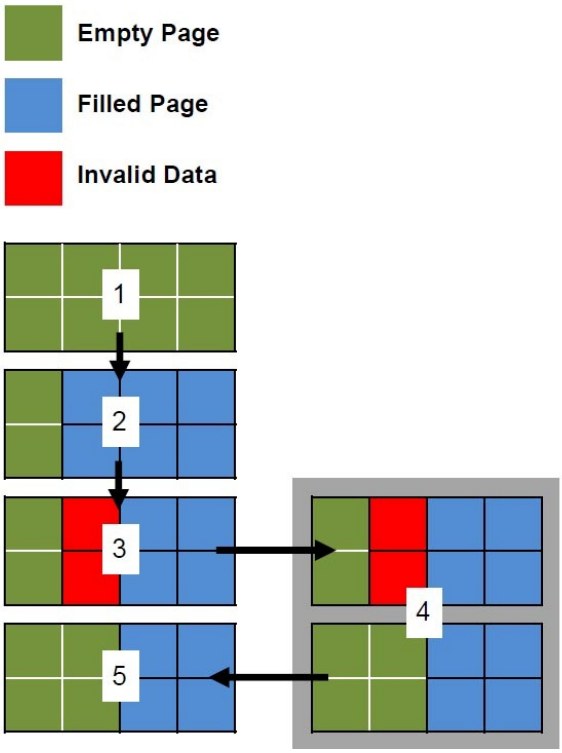



**Figure 16: Vizuelna ponazoritev delovanja ukaza TRIM.**

Ker lahko na gibkih in trdih diskih obstoječe podatke enostavno prepišemo, večina datotečnih sistemov posledično le označi, da je nek prostor prost za nadaljnjo uporabo. Pri SSD diskih pa takšen pristop skozi čas občutno upočasni hitrost zapisovanja novih datotek. Ker se disk ne more zavedati, katere lokacije je operacijski sistem ravnokar sprostil, smetar na disku teh lokacij takoj ne more ustrezno pripraviti. Tu pa nastopi ukaz TRIM. Ko operacijski sistem opravlja brisanje datoteke v danem trenutku, poleg prvotnih operacij disku pošlje ukaz TRIM in mu tako sporoči, katere strani niso več v uporabi. Krmilnik na disku lahko sedaj strani (bloke) pobriše in jih tako pripravi na ponovno zapisovanje. Sedaj ob pisanju novih datotek na disk strani (bloke) ni potrebno brisati kar znatno pohitri zapisovanje. Primer delovanja prikazuje slika 16.

1.) SSD strani ne vsebujejo podatkov.

2.) Uporabnik zapiše podatke na strani.

3.) Uporabnik pobriše nekaj podatkov. Strani OS nato označeni kot "neuporabljene".

4.) TRIM ukaz nato pove SSD krmilniku, da strani vsebujejo neveljavne podatke, ki jih nato krilnik počisti tako, da zbriše vsebino celotnega bloka.

5.) Podatki se sedaj zapišejo nazaj na blok, ostala mesta pa so pripravljena za zapis svežih podatkov.

Žal pa pri uporabi ukaza obstajajo omejitve. Ukaza ne podpirajo vsi operacijski sistemi, predvsem starejši, in trenutno

je podprt le na SATA in eSATA vmesniku. Podrobnejši opis v nadaljavanju:

- **Stari SSD diski:** SSD diski so obstajali že pred upeljavo TRIM ukaza.
- **Stare verzije Windows OS:** Microsoft je ukaz TRIM šele v Windows 7 in novejše. Na starejših je podoben učinek možno doseči z dodatno programsko opremo.
- **Stare verzije MacOS X:** Apple podpira TRIM ukaz le za diske, ki jih dobavljajo sami, ukaz pa je podprt od verzije 10.6.8 naprej.
- **Linux**
- **Windows datotečni sistemi:** Microsoft podpira ukaz TRIM le na datotečnem sisitemu NTFS.
- **Podprti vmesniki:** Ukaz je polno podprt le na vmesnikih SATA, eSATA in SCSI preko UNMAP ukazov. Razširja pa se tudi na druga vodila.

## 4.2 Namenska oprema

Poleg ukaza TRIM oviro predstavlja tudi krmilnik, ki operacijskemu sistemu prikazuje manjšo velikost od dejanske, kljub temu, da (lahko) uporablja ves razpoložljiv pomnilnik. Slednje predstavlja predvsem oviro, ko poženemo orodja za iskanje izbrisanih datotek (recimo, da se pri tem ni uporabljal ukaz TRIM). Če naprimer, da želimo narediti varnostno kopijo zajetega diska, se postopka lotimo tako kot pri trdih diskih. Torej priklopimo disk v bralnem načinu, naredimo podpis diska ter nato eno kopijo, nato pa še, za vsak slučaj, še eno ter za obe kopiji še podpis (na primer s SH1). Ob primerjavi vseh treh podpisov pa naprimer opazimo, da so si med seboj različni. Slednje predstavlja veliko težavo pri zagotavljanju ponovljivosti dokazov in lahko privede do neveljavnosti najdenih dokazov. Poleg tega, čeprav smo prekopirali celoten prostor, ne moremo zagotoviti da smo prekopirali celoten pomnilnik in se lahko v nekopiranem pomnilniku nahajajo dokazi, do katerih pa ne moremo (enostavno) dostopati.

Pri reševanju tega problema bi lahko navedli dva pristopa. S strani zakonodaje bi lahko le-ta uvedla zakon, da je dovoljen uvoz le SSD diskov, pri katerih je proizvajalec vgradil funkcionalnost, ki ustavi vse napredne operacije krmilnika razen preslikovanja (LBA), pri čemer je možen dostop do vseh lokacij na disku. Ta pristop bi bil mogoče še najlažji in najučinkovitejši, saj bi lahko forenzik sledil enakim postopkom kot pri klasičnih trdih diskih.

Drugi pristop je uporaba namenske srojne opreme (sliki 17 in 18) za branje surovih podatkov na čipu. Sledeči postopek je za forenzika veliko težji in zahteva več specialnega znanja. Vprašljivo je tudi odstranjevanje pomnilnih čipov s tiskanega vezja, saj lahko pride pri procesu odstranjevanja do poškodbe čipa ali ostalih komponent, hkrati pa se poveča verjetnost izgube dokaznega gradiva. Pri omenjenem pristopu predstavlja oviro še kriptiranje podatkov, ki ga izvaja vgrajeni krmilnik [4].

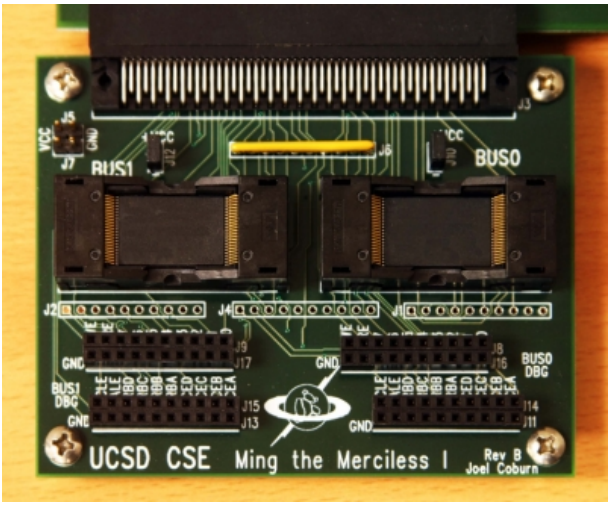



**Figure 17: Primer namensko razvite strojne opreme temelječe na FPGA čipu[8].**



**Figure 18: Primer namenske strojne opreme za branje direktno iz pomnilniškega modula[9].**

## 5. ZAKLJUČEK

V članku je bilo prestavljeno delovanje SSD diskov in problemi pri forenzični preiskavi. Izkaže se, da bo potrebno spremeniti obstoječo prakso preiskovanja SSD diskov. Največji problem je seveda v kontrolnikih diskov, ki so različni od naprave do naprave in programi, ki so na njih prav tako niso dostopni.

## 6. ADDITIONAL AUTHORS

## 7. REFERENCES


[1] First flash memory card with 4bits per cell dostopno na: http://www.sandisk.com/about-sandisk/press-room/press-releases/2009/2009-10-13-sandisk-ships-world%E2%80%99s-first-flash-memory-cards-with-64-gigabit-x4-%284-bits-per-cell%29-nand-flash-technology/

[2] Klasifikacija za ukaz TRIM, dostopno na: http://www.t13.org/Documents/UploadedDocuments/docs2010/e09158r2-Trim_Clarifications.pdf

[3] Primer dokumenta ATA8, dostopno na: http://www.t13.org/Documents/UploadedDocuments/docs2008/D1699r6a-ATA8-ACS.pdf

[4] Podrobnosti Samsung 840 PRO serije, dostopno na: http://www.samsung.com/us/system/consumer/product/mz/7p/d2/mz7pd256bw/samp15693M_8.5x11_SSD_840_PRO_Spec_Sheet.pdf

[5] Članek o SSD diskih z obrazložitvijo delovanja, dostopno na: http://www.anandtech.com/show/2738

[6] Članek o Samsung SSD diskih, dostopno na: http://www.custompcreview.com/reviews/samsung-840-250gb-ssd-review/15955/

[7] Članek o arhitekturi in performancah SSD diska, dostopno na: http://codecapsule.com/2014/02/12/coding-for-ssds-part-2-architecture-of-an-ssd-and-benchmarking/

[8] Primer namenske strojne opreme: https://www.usenix.org/legacy/events/fast11/tech/full_papers/Wei.pdf

[9] Primer namenske strojne opreme proizvajalca ACE: http://www.acelaboratory.com/pc3000flash.php#perryPhoto[Gallary60]/0/

[10] Christoper K., Timothy V.: Empirical analysis of solid state disk data retention when used with contemporary operating system; Dostopno na: http://www.dfrws.org/2011/proceedings/17-349.pdf

[11] Ukaz TRIM na Wikipediji, dostopno na: http://en.wikipedia.org/wiki/TRIM

[12] Flash pomnilnik na Wikipediji, dostopno na: http://en.wikipedia.org/wiki/NAND_Flash#NAND_flash

# Analiza podatkov VoIP klicev

[Članek pri predmetu Digitalna forenzika]

Tomaž Tomažič
Fakulteta za računalništvo in infromatiko
Tržaška 25
Ljubljana, Slovenia
tt3710@student.uni-lj.si

Zupanec Žiga
Fakulteta za računalništvo in infromatiko
Tržaška 25
Ljubljana, Slovenia
zz9698@student.uni-lj.si

## ABSTRACT
Popularnost telefonije preko IP (VoIP) je v zadnjih letih narasla, ker je cenovno ugodnejša in enostavnejša za uporabo. Kakorkoli že, ta tehnologija je tudi atraktivna za kriminal, ker je VoIP globalna telefonska storitev, v kateri je težko identificirati uporabnika. Privlačna je tudi zaradi visoke varnosti, saj veliko implementacij uporablja močno enkripcijo za zaščito, tako zvočnih podatkov kot kontrolo samih sporočil v primerjavi z žično telefonijo kjer je prisluškovanje enostavnejše. Zato so za VoIP klice potrebni drugi načini pridobivanja digitalnih dokazov in informacij. V najini raziskavi smo pogledali kateri dokazi ostanejo po opravljenem VoIP klicu na obeh napravah, trdem disku in bralnopisalnem pomnilniku (RAM).

## Keywords
Digitalna forenzika, dokazi, telefonija preko IP, VoIP, forenzika v pomnilniku

## 1. UVOD
Popularnost telefonije preko IP (VoIP) je v zadnjih letih narasla, ker je cenovno ugodnejša in enostavnejša za uporabo. Kakorkoli že, ta tehnologija je tudi atraktivna za kriminal, ker je VoIP globalna telefonska storitev, v kateri je težko identificirati uporabnika. Privlačna je tudi zaradi visoke varnosti, saj veliko implementacij uporablja močno enkripcijo za zaščito, tako zvočnih podatkov kot kontrolo samih sporočil v primerjavi z žično telefonijo kjer je prisluškovanje enostavnejše. Zato so za VoIP klice potrebni drugi načini pridobivanja digitalnih dokazov in informacij. V najini raziskavi smo pogledali kateri dokazi ostanejo po opravljenem VoIP klicu na obeh napravah, trdem disku in bralnopisalnem pomnilniku (RAM).

Identifikacijske poverilnice so osnovane v informacijah iz glav paketov vsebovanih v VoIP protokolu. Narejeni so bili različni kontrolni testi v katerih je bila forenzična analiza narejena na virtualnih strojih s katerimi so bili opravljeni različni VoIP klici. Eksperimenti so bili ponovljeni na istem protokolu tako na disku kot v ramu.

## 2. PALETA VOIP
Telefonija preko IP protokola je zelo spremenila način kako se podatki telefoniranja prenašajo in tako začela pravo telekomunikacijsko industrijo. Z samo rastjo popularnosti in široko dostopnostjo do interneta, je omogočeno da se klici prenašajo preko internetne infrastrukture kot pa po tradicionalnem javno komutiranem telefonskem omrežju (PSTN). Ta tehnologija, ki se ji reče VoIP, uporablja internetni protokol (IP) za prenašanje paketov ki vsebujejo majhne koške glasovnega pogovora med klicočima.

Popularnost VoIP je narasla zaradi cenejšo in enostavnejše uporabo tako doma kot v podjetjih. Kakorkoli že, ta tehnologija je zelo privlačna za kriminalce, še posebej za VoIP pogovore ki niso spremljani iz strani internetnega ponudnika. To je zaradi tega ker je VoiP globalen telefonski servis v katerem je težko identificirati uporabnikovo identiteto, ker večina implementacij uporablja močno enkripcijo za zavarovanje tako zvočnih podatkov kot kontrolo samih sporočil in nadziranje ali sledenje takšnih VoIP klicev je oteženo zaradi konvencionalnih metod kot so žično prisluškovanje in le teh ni mogoče aplicirati na VoIP klice. Zaradi teh razlogov so za pridobivanje podatkov in informacij iz telefonije preko IP potrebne druge metode. Bistveno za forenzične raziskovalce je, da imajo metodo ki omogoča organom pregona premagati nekatere vidike tega načina telefonije, ki so ugodne za storilce kaznivih dejanj.

Ta uvod podaja pregled transportnih protokolov VoIP in signalne protokole. Ti določajo podatke, iz katerih je mogoče forenzično pridobiti polja iz glave protokola, kot tudi uporabnikovo registracijo za VoIP klic.

VoIP ni en samo protokol ampak kolekcija različnih že obstoječih protokolov, ki so uporabljeni za vzpostavitev, vzdrževanje in končanje klica, protokoli za enkapsulacijo in prenos paketov po internetu. Glavni protokol, ki bdi nad vsemi ostalimi je Internetni protokol (IP). Protokoli IP, UDP in RTP so protokoli ki enkapsulirajo in prenašajo pakete, ki vsebujejo informacije potrebne za identifikacijo izvora in ponora klicateljev in dostavo samih zvočnih podatkov. Session Initiation Protocol (SIP) je signalni protokol uporabjen za vzpostavitev, vzdrževanje in zaključevanje klica.

### 2.1 Internet Protocol

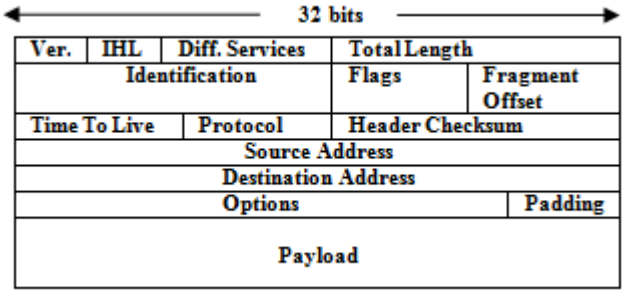



**Figure 1: Format glave paketa IP.**

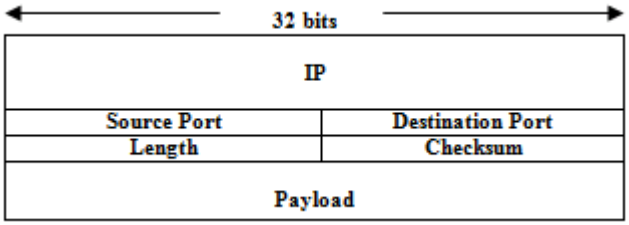



**Figure 2: Glava paketa UDP.**

Protokol IP je odgovoren za hranjenje internetnega naslova v glavi, kar omogoča paketom da so dostavljeni iz svojega izvora do IP naslova. Format glave protokola IP je prikazan na sliki 1.

## 2.2 User Datagram Protocol

Zanesljiv protokol za prenos paketov po internetu je protokol TCP, ki zagotavlja dostavo tudi za zgubljene in pokvarjene pakete s ponovnim pošiljanjem. Vendar za VoIP klice ki potekajo v realnem času ni smisla ponovnega pošiljanja za zgubljene pakete. TCP uporablja nadzorom pretoka, ki začasno zaustavi prenos paketov dokler pokvarjeni paket ni uspešno prenešen.

Namesto protokola TCP je za zahteve VoIP bolj primeren, čeprav je manj zanesljiv pri dostavi paketov, protokol UDP (slika 2). Izbran sistem je tako sklad protokola IP/UDP. IP datagram zagotavlja izvorni in ciljni naslov in izvorna in ciljna vrata. Pogoste aplikacije delujejo na specifičnih vratih. Ker TCP zagotavlja prenos izgubljenih paketov, ni zaželjen v časno občutljivih aplikacijah in je zato za prenos zvoka uporabljen UDP.

Protokol UDP je se uporablja tudi zaradi lažjega prebijanja požarnega zidu pri uporabi načina prevajanje omrežnega naslova (angl. Network Address Translation, krat. NAT). Poznamo več vrst načinov NAT (slika 3) in sicer:

- Direktna preslikava (angl. Full-cone) pri kateri se notranji lokalni IP naslov in vrata preslikajo v zunanji IP naslov z istimi vrati. Zunanji odjemalci lahko s pošiljanjem na začasni par (zunaj vidni IP naslov in vrata) pošiljajo sporočila notranjemu IP naslovu.
- Omejena direktna presilava (angl. Restricted cone): deluje na na istem principu kot direktna preslikava z omejitvijo, da mora prvi paket poslati interni odjemalec.
- Z vrati omejena direktna preslikava: zunanji odjemalec lahko pošlje paket le na vrata, katerim je predhodno pošiljal notranji odjemalec.
- Simetrična preslikava (angl. Symmetric cone): usmerjevalnik vodi preslikovalno tabelo glede na ponorni IP naslov in vrata. Vrata se dodelijo naključno za čas trajanja seje glede na ponorni IP naslov. Le tisti zunaji odjemalec, ki prejme paket paket notranjega odjemalca lahko pošlje paket nazaj (odgovori) notranjemu odjemalcu.

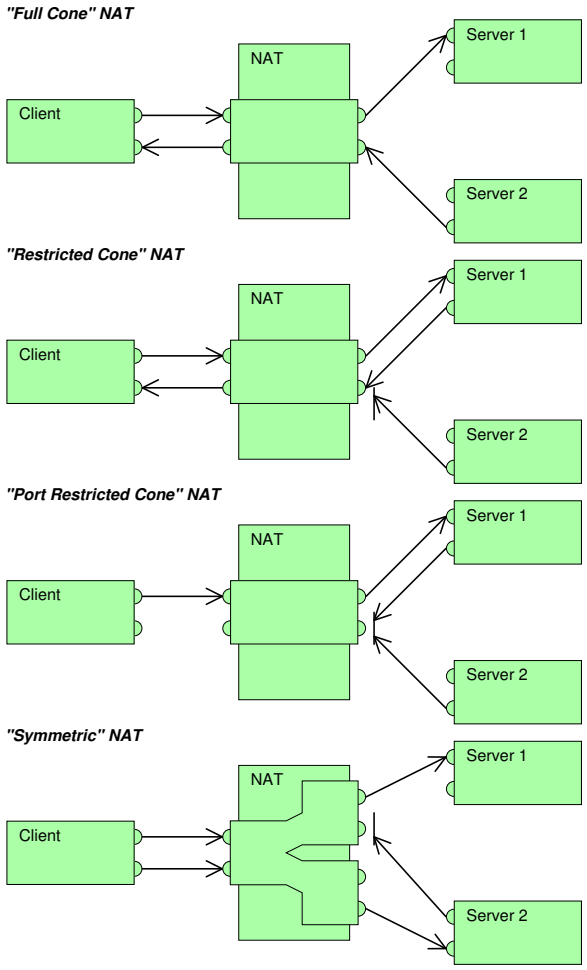



**Figure 3: Primerjava med različnimi metodami NAT.**

NAT prebijamo z orodji za sprehanjanje po takih omrežjih, ki jih imenujemo "Obhod NAT-a", (angl. Session Traversal Utilities for NAT, krat. STUN). Gre za tehniko, ki vzpostavi in vzdržuje IP povezave skozi NAT usmerjevalnike, ki kršijo povezavo iz enega konca na drugi (end-to-end). Uporablja se za peer-to-peer in VoIP povezave. Obstaja veliko metod za rešitev tega problema, vendar ne obstaja ena univerzalna. Veliko tehnik uporablja pomoč serverja. Nekatere pa server

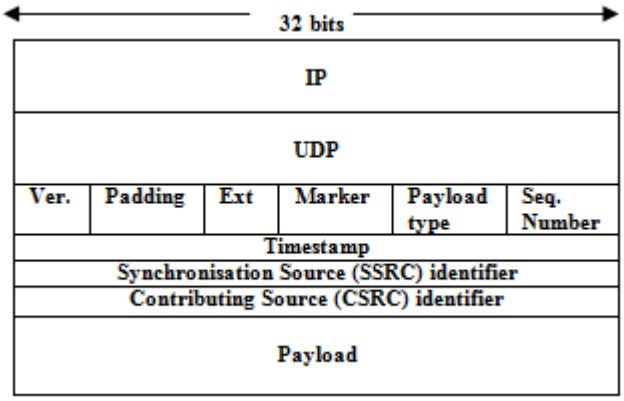



**Figure 4: Sklad UDP/RTP in glava paketa RTP.**

uporabljajo samo za vzpostavitev povezave. Če server uporabljajo za celotno komunikacijo se s tem zniža kvaliteta povezave in poveča latenca. Druga tehnika je, da se dovoli požarnim zidovom in NAT omrežjem da dopuščajo VoIP, skrati pa ohranjajo varnost podjetij. To določajo protokoli so RSIP (realm-specific IP) in MIDCOM (middlebox communications).

## 2.3 Real-Time Transport Protocol

Real-Time Transport Protocol (RTP) , zagotavlja prometno omrežje za aplikacij v realnem času, kot so prenos zvoka preko paketno komutiranega omrežja. Protokol RTP je izbran za prenos govornih podatkov, saj dodaja dodatne informacije, kot je na primer zaporedna številka za vsak paket, da zagotovijo prejemni aplikaciji zvoka možnost razporeditve paketov v pravilnem vrstnem redu in zagotavlja tudi časovno oznako za vsak posamezen paket, ki omogoča predvajanje zvoka v zaporednih časovnih presledkih. Sveženj protokol je postal IP / UDP / RTP dimnika. Za klicanje in sprejemanje VoIP klica ni potrebna uporaba protokola RTP, vendar ta lahko preprosto doda dodatne podatke, ki pripomorejo sprejemanje paketov v vrstnem redu in olajša delo z VoIP konferencami. IP/UDP/RTP sklad in glava paketa RTP je prikazana na sliki 4.

Za vsak paket sta enolično določena časovna značka in sekvenčna številka. Sekvenčna številka se poveča z vsakim poslanim paketom in omogoča ponovno razvrstitev in detekcijo izgube paketov na prejemnikovi strani.

Polje sinhronizacijski izvorni identifikator (SSRC) označuje izvor sinhronizacije, navadno z računalnikovo uro medtem ko polje doprinašani izvor (CSRC) označuje vir posameznih prispevkov, ki sestavljajo en podatkovni tok paketov. Za sodelovanje v VoIP klicu ni nujno uporabljati RTP protokola. Nekatere VoIP aplikacije kot je na primer Skype ne uporabljajo RTP protokola, medtem ko pa ga X-Lite in Ekiga uporabljajo. Pravv zaradi tega smo naš ekspermient naredili na programu Ekiga saj imamo z dodatnim protokolom RTP potem na voljo dodatne polja iz katerih lahko iščemo informacije o klicu.

Zvočni posnetek je razdeljen v paket ki je potem prenesen po omrežju. Če je paket prevelik in ga je potrebno fregmenitrati ali pa izvira iz različnih virov, potem bo imel vsak fragment

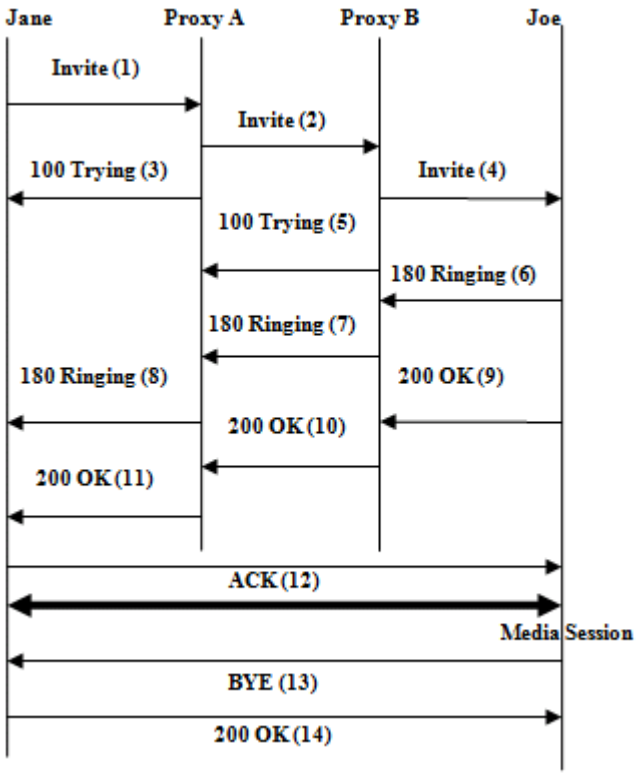



**Figure 5: Primer izmenjave sporočil med sejo SIP.**

enak CSRC za identifikacijo izvora iz istega podatkovnega toka. Različne sekvenčne številke pa omogočajo rekonstrukcijo v pravem zaporedju za vsak CSRC. To je uporabljeno za sinhronizacijo na prejemnikovi strani.

## 2.4 SIP protocol

Protokol SIP omogoča klicanje uporabniških posrednikov (UA) za lokacijo in registracijo z uporabo proxy strežnikov, ki omogočajo drugim uporabniškim posrednikom priključitev v klicno sejo. Vsaka transakcija je sestavljena iz zahteve ki sproži dolečno metodo an vsaj en odgovor, kot je prikazano na sliki 5. Jane uporablja svojo VoIP aplikacijo za pošiljanje zahteve Invite (vabilo) Joetu. Zahteva Invite je SIP metoda, ki določa akcije, ki jih Jane želi od Joeta, to je sprejetje klica. Zahteva Invite gre skozi dva proxy strežnika da doseže Joeta (korak od 1 – 5). Na prejemnikovi strani se začne zvonenje, odgovor pa Joe vrne pošiljatelju ponovno skozi oba proxy strežnika (korak 6 – 8). Joe potrdi prejetje zahtevka Invite z odgovorom OK osebi Jane (9 – 11). Prenos večpredstavnostne vsebine se začne ko Jane potrdi odgovora OK (12). Ta sejo lahko zaključi katerikoli udeleženec v komunikaciji (13 -14).

# 3. ALTERNATIVNI PROTOKOLI VOIP

Vrata, ki se navadno uporabljajo pri paleti protokolov VoIP so povzeta v tabeli 1.

## 3.1 Paleta protokolov H.323

Namesto protokola SIP se lahko za vzpostavitev seje, kontrolo nad sejo, prenos multimedijskih podatkov in kontrolo

**Table 1: Uporaba vrat pri protokolu VoIP**

| Protokol | Vrata | Tip | Razlaga |
|---|---|---|---|
| SUP | 5000 do 5100 | UDP | SIP seja privzeto 5060. Različno pri NAT. |
| STUN | 3478 do 3479 | UDP | |
| RTP | Naključna/razpoložljiva | UDP | Prihajajoč promet na drugi strani. Navadno 5004, 7070, 16382 |
| H.323 | 1720 | TCP | Poslušanje. |
| H.323 | 5000 do 5100 | UDP | Vratarji. |
| H.323 | 30000 do 30010 | TCP | Kanal H.245 za Microsoft Netmeeting. |

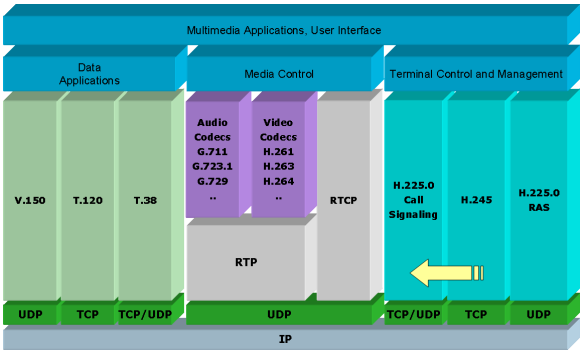



**Figure 6: Celoten sklad protokola H.323.**

pasovne širine uporabi zbirka tehnologij H.323 (slika 6). H.323 lahko po funkcionalnosti grobo primerjamo s protokolom SSL/TLS, a je precej obsežnejši. Protokol povezuje različne naprave (terminale, usmerjevalnike (angl. MCUs), prehode vratarje, in zaključne elemente.)

Terminali so naprave s katere tipično uporablja končni uporabnik (telefoni, programski odjemalci, konferenčni sistemi ...). Usmerjevalniki povezujejo oziroma združujejo terminale ali skupino le-teh med seboj. Prehodi skrbijo za združljivost med različnimi tehnologijami, ki se uporabljajo pri povezovanju naprav (PTSN, ISDN). Vratarji niso obvezni za delovanje sistema skrbijo pa za registracijo končnih naprav, avtentikacijo uporabnikov, preslikavo naslovov, dovoljenja uporabnikov in/ali naprav. Med seboj se vratarji uporabljajo protokol RAS (angl. Remote Access Service). Zaključni elementi prav tako niso potrebni, uporabljajo pa se za nadzor, uporavljanje terminalov in vodenje računov znotraj uporavljalske domene. Po analogiji spominjajo na naprave RAS[1] (angl. Remote Access Server).

### *3.1.1 Primer vzpostavitve klica*

Faze pri vzpostavitvi klica pri protokolu H.323 (slika 7).

1. Setup (Namera o vzpostavitvi povezave z napravo.)
2. Setup acknowledge (Klicana naprava želi komunicirati s klicočo.)
3. Call Porceeding (Potrditev klicane naprave, da ima vse potrebne informacije za usmeritev klica.)
4. Progress (Obvestilo prehoda, da klic v izvajanju.)

[1]Ni enako kot prej omenjeni RAS (Registration, Admission and Status).

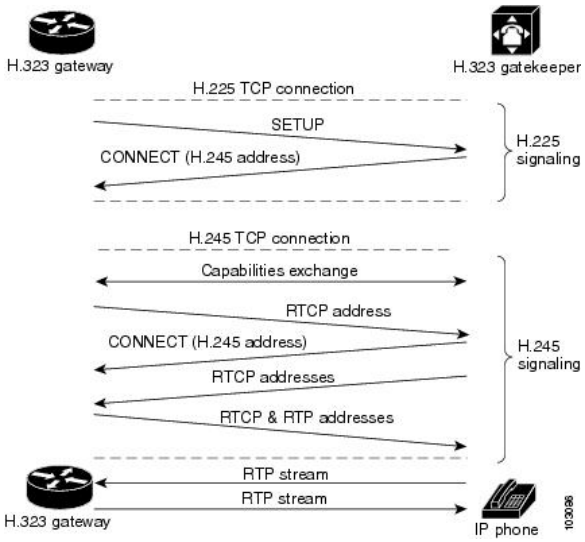



**Figure 7: Primer vzpostavitve klica pri protokolu H.323.**

5. Alerting (Obvestilo klicane naprave, da je pričela z zvonenjem.)
6. Connect (Obvestilo klicane naprave, da je klic sprejet - uporabnik je dvignil slušalko.)
7. User Information (Dodatna, (večinoma) nenujna obvestila o trajajočem klicu.)
8. Release Complete (Obvestila o sprostitvi klica.)
9. Status Inquiry (Poizvedovanje o statusu trenutnega klica.)
10. Status (Sporočilo kot odgovor na poizvedovanje o statusu trenutnega klica.)
11. Information (Dodatne informacije o klicu.)
12. Notify (Sporočila o spremembah, ki so nastale med klicem.)

## 3.2 Skype

VoIP aplikacija Skype je zelo popularna, vendar tudi predmet številnih forenzičnih preiskav zaradi dobrih sposobnosti prečkanja NAT in zaobitja požarnega zida. Katerikoli Skype klient z nameščeno eno Skype aplikacijo na računalniku z namenom opravljanja VoIP klicev, lahko neželeno postane super vozlišče na Skype komunikacijski poti za vzdrževanje Skype omrežja, kar je še posebej riskantno za korporacije. Z

analizo samo Skype prometa v njegovem komunikacijskem omrežju bi bilo mogoče zaznati Skype UDP vtičnike. To bi omogočalo blokiranje Skype prometa.

# 4. TEŽAVE TEHNOLOGIJE VOIP

Poleg že opisanih težav pri povezovanju naprav, ki so v omrežjih varovanimi s požarnim zidom in/ali povezane v medmrežje preko NAT-a.

## 4.1 Quality of service

Komunikacija prek IP omrežja je manj zanesljiva v primerjavi s standardnim PSTN telefonskim omrežjem, ker nima omrežnega mehanizma ki bi zagotavljal da se paketi ne izgubijo, in so dostavljeni v pravilnem vrstnem redu. VoIP telefonija se lahko sreča z raznovrstnimi problemi kot so latenca, izguba paketov in fazno trepetanje (jitter). Usmerjevalniki običajno promet obdelujejo po FIFO vrstnem redu. Lahko se pa zgodi, da v usmerjevalnikih z veliko obremenitvijo, pride do presežka limita latence ki je dovoljena za VoIP. Ni mogoče nadzorovati, da bi se paketi dostavljali s fiksnim zamikom, saj le ti nastajajo z razdaljo, ki jo potrebujejo, da pridejo do cilja. To je še posebej opazno pri satelitskih omrežjih. En od načinov znižanja latence je z označevanjem zvočnih paketov, saj so občutljivi na časovne zamike, z metodo kot je DiffServ. Obstajajo tudi različni protokoli, ki definirajo specifikacije, za dobro kvaliteto VoIP klicev. Ti so RTCP, SIP, H.460.9, H.248.30 in MGCP.

## 4.2 Klic v sili

Standardno PSTN telefonsko omrežje je z žico priključeno v omrežje, ki ga vzdržuje telefonski ponudnik. S tem je tudi določena fizična lokacija. Ko je iz takega omrežja klicana ena izmed številk za klice v sili, je službi za pomoč takoj prikazana fizična lokacija klicatelja, ki jo imajo shranjeno v podatkovni bazi. Pri IP telefoniji, take direktne povezave med fizično lokacijo in komunikacijo ni. Celo ponudnik fizične infrastrukture, kot je na primer ponudnik DSL, lahko ve le približno lokacijo naprave, ki je osnovana glede na IP naslov, saj nekateri ISP ponudniki ne avtomatsko beležijo. komu so bili dodeljeni avtomatski IP naslovi.

## 4.3 Napajanje

Druga težava VoIP je, da zanjo potrebujemo električno napajanje. Tradicionalna telefonija je namreč priključena direktno na telefonskega ponudnika, kateri zagotavlja napajanje za analogno telefonijo. IP telefoni in VoIP telefonski adapterji so priključeni na usmerjevalnike ali na modeme. Za to rešitev obstajajo tudi modemi z vgrajeno baterijo ki zagotavljajo napajanje za nekaj ur. Tipično so to naprave ki lahko uporabljajo tudi analogno tehnologijo ali pa se povežejo na naš mobilni telefon. Ta težava je tudi povezana s prejšnjo saj v nujnih primerih, brez elektrike ne moremo poklicati nikogar, torej tudi telefonskih številk za klice v sili.

## 4.4 Varnost

VoIP je občutljiv na črve, viruse in vdore, vendar je to zelo redko saj se klici kriptirajo. Znani so DOS napadi, snemanje pogovora, kraja seje in nato uporaba plačljivih storitev, itd.

### *4.4.1 Oddaljeni napadi*

Kolumbijski raziskovalci so pokazali, kaj lahko naredijo z vdorom v Cisco VoIP telefon. Hekerji torej lahko oddaljeno spremenijo telefon v navaden mikrofon in prisluškujejo iz kjerkoli. Cisco je globalno najbolj znan VoIP ponudnik, saj ima na milijone VoIP telefonov po svetu, tudi v podatkovno občutljivih podjetjih in vladah. Napravo thingp3wn3r so priključili v serijska vrata in nanj naložili zlonamerno kodo. S posebnim ukazom so dosegli da se je sistem zrušil in izpisal celoten pomnilnik. Z analizo pomnilnika so uspeli spremeniti program na telefonu, kateri je lahko okužil druge računalnike in naprave kot so¸na primer tiskalniki. Lahko so dosegli, da telefon ves čas snema pogovore v prostoru tako, da uporabnik tega sploh ne zazna. Zanimivo je, da ima tak telefon tudi predsednik Obama.

## 4.5 Procesorska moč

Problem z VoIP je tudi, da so ti klici odvisni od posameznih računalnikov. V primeru da odpremo zahteven program med samim klicem se lahko procesor preobremeni in kvaliteta pogovora se zniža. V najslabšem primeru pa se lahko cel sistem sesuje. VoIP klici so torej v odvisnosti vseh napak ki jih ima normalen računalnik.

# 5. VOIP IN FORENZIKA

VoIP predstavlja paketno osnovano internetno tehnologijo za zvočno komunikacijo v primerjavi z tradicionalnim telefonskim klicem. To seveda vpliva na kazenski pregon saj prisluškovanje na žici ni več možen. Potrebna je nova arhitektura za zakonito prestrezanje prometa VoIP, vendar ker paketi potujejo po različnih poteh je njena implementacija težka. Možne subjekte prestrezanja lahko razdelimo na dve področji. Tiste ki pripadajo ponudniku internetnih storitev (ISP), in tiste, ki pripadajo LEA (Law Enforcement Agency), kateri se lahko nahajajo znotraj ISP ali zunaj njega. Subjekti povezani z ISP vključujejo signalizacijo, namestitev, vzdrževanje in zaključevanje IP klicev. Zbiranje podatkov IP zvočnega prometa je lahko realizirano na istem pod omrećju kot je VoIP usmerjevalnik, kateri se poveže do obstoječe telefonske linije. Subjekti v povezavi z LEA lahko obstajajo znotraj ISP arhitekture vendar ostajajo pod nadzorom LEA saj zajeti podatki potrebujejo dovoljenje ISPja za dostop do dekripcijskih ključev.

Bolj kot je omrežje centralizirano, bolj enostavno je zakonito prestrezanje. Na žalost ne obstaja ena sama naprava, ki bi prestrezala ves VoIP promet amapk je celo značilna visoka stopnja decentralizacije, zlasti za mobilne uporabnike, ki se povezujejo na internet za krajša časovna obdobja. To lahko povzroči premik izven dosega vozlišča nadzorovanega pod vodstvom organov pregona kjer je prestrezanje zahtevano od internetnih ponudnikov. Na primer, Skype uporablja dobro enkripcijo glede na objavljene standarde (DES in AES). To od LEA zahteva pridobitev ključev od podjetja Skype v primeru če potrebujejo dekripcijo zajetih VoIP podatkov.

Trenutno obstaja nekaj modelov za preiskavo v digitalni forenziki. Ogrodje za digitalno forenziko DFRWS nam zagotavlja naslednje zaporedne korake za analizo:

- Identifikacija
- Ohranjanje
- Zbiranje

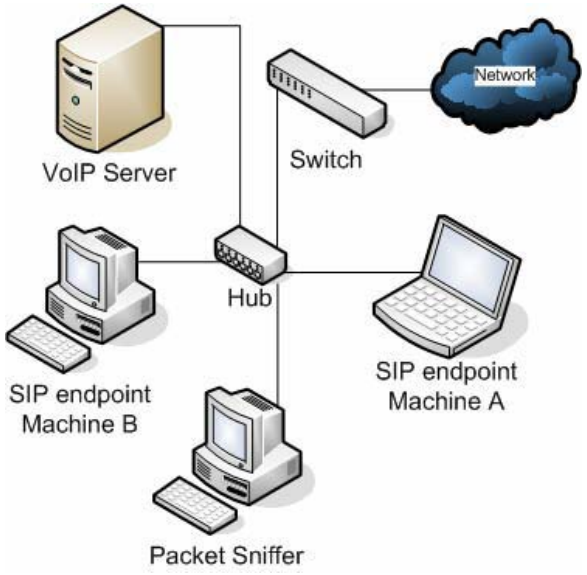



**Figure 8: Postavitev in vloge naprav v eksperimentu.**

```
tcpdump -nnvvXSs 1514 host \
192.168.1.193 -i eth0 -w voip.cap
```

**Figure 9: Zajem prometa za/iz naprave klicoče na usmerjevalniku.**

- pregled
- Analiza
- Predstavitev

Pregled in analiza zajetega pomnilnika se izvajata v načinu samo za branje, kar pomeni da ostanejo prvotne informacije nespremenjene.

# 6. EKSPERIMENTI

Cilj eksperimenta je ugotoviti, ali se kakršnakoli informacija o klicu shrani oziroma ohrani na trdem disku in/ali pomnilniku naprave iz katere je bil narejen klic. Pri ponovitvi eksperimentov smo upoštevali strukturo in vlogo naprav, kot je predstavljeno v članku in povzeto v sliki 8. Predhodno smo se registrirali in pridobili naslov SIP pri ponudniku VoIP storitev Ekiga. Naprava s katero kličemo uporablja 32-bitni operacijski sistem Debian, temelječ na jedru Linux in osnovnimi programi za upravljanje operacijskega sistema GNU. Na napravi uporabljamo VoIP odjemalec Ekiga. Za testni klic smo poklicali kar "Echo test"odzivnik in počakali 30 sekund, nato pa klic prekinili. Promet zajemamo zunaj govorečih naprav, v usmerjevalniku na katerega poganja strojna programska oprema OpenWRT, s programom tcpdump ( 9). Zajet promet analiziramo s programom wireshark. Promet je zaradi poenostavitve in lažje analize nekriptiran.

Prva ugotovitev je, da lahko s prisluškovanjem znotraj lokalnega omrežja brez težav rekonstruiramo celoten klic, vključno z

```
dd if=/dev/sda of=/mnt/storage/debian.img
```

**Figure 12: Zajem slike diska.**

```
dd if=/dev/fmem of=debian_mem.bin bs=1024
```

**Figure 13: Zajem pomnilniške slike.**

avdio in video komunikacijo (sliki 10 in 11). Vendar se tu postavi vprašanje ali je to dovolj dober dokaz na sodišču? Promet na mreži je enostavno ponarediti z npr. napadom replay attack. Večjo stopnjo objektivnosti in verodostojnosti dokaza bi dosegli, če bi ključne informacije klica dejansko našli na fizični napravi, s katere je bil opravljen klic.

Za potrebe seminarske naloge smo napisali skripto v programskem jeziku python, ki v izpisu diska ali pa v pomnilniškem izpisu poišče ključne elemente seje. Za ključne elemente seje smatramo vse tiste elemente s katerimi je mogoče enolično identificirati določen klic. V konkretnem primeru je ta identifikator:
"Call-ID: 381d54a4-ccd2-e311-90e9-00155d014713@debian32".

## 6.1 Eksperiment 1

Po opravljenem klicu zajamemo podatke na disku (ukaz 12).

Na disku nismo našli nobenih sledi klica.

## 6.2 Eksperiment 2

Za zajem pomnilnika smo najprej namestili modul fmem, da lahko zajamemo celotno pomnilniško sliko (ukaz 13).

Z našo skripto smo na različnih lokacijah našli niz "Call-ID", kar je dovolj za identifikacijo klica. Našli smo tudi en paket identičen tistemu, ki je bil zajet na usmerjevalniku.

# 7. ZAKLJUčEK

Za forenziko na področju protokola VoIP velja podobno kot za druga področja digitalne forenzike, da je to še precej neraziskano področje. Da ni (prosto dostopnih) orodji za analizo protokola VoIP in, da se tako stanje ohranja naprej bi deloma pripisali dejstvu, da ta orodja verjetno obstajajo, a so v lasti državnih tajnih služb, ki nimajo interesa, da bi ta orodja postala javna. Deloma tako stanje ustreza tudi ponudnikom ter proizvajalcem strojne in programske opreme, saj smo v članku predstavili kako "ranljiv"je pravzaprav velik del prometa VoIP. Dokler je orodji za analizo prometa VoIP malo, večina proizvajalcev in uporabnikov uživa lažen občutek varnosti, t.i. Šecurity through obscurity".

# 8. LITERATURA

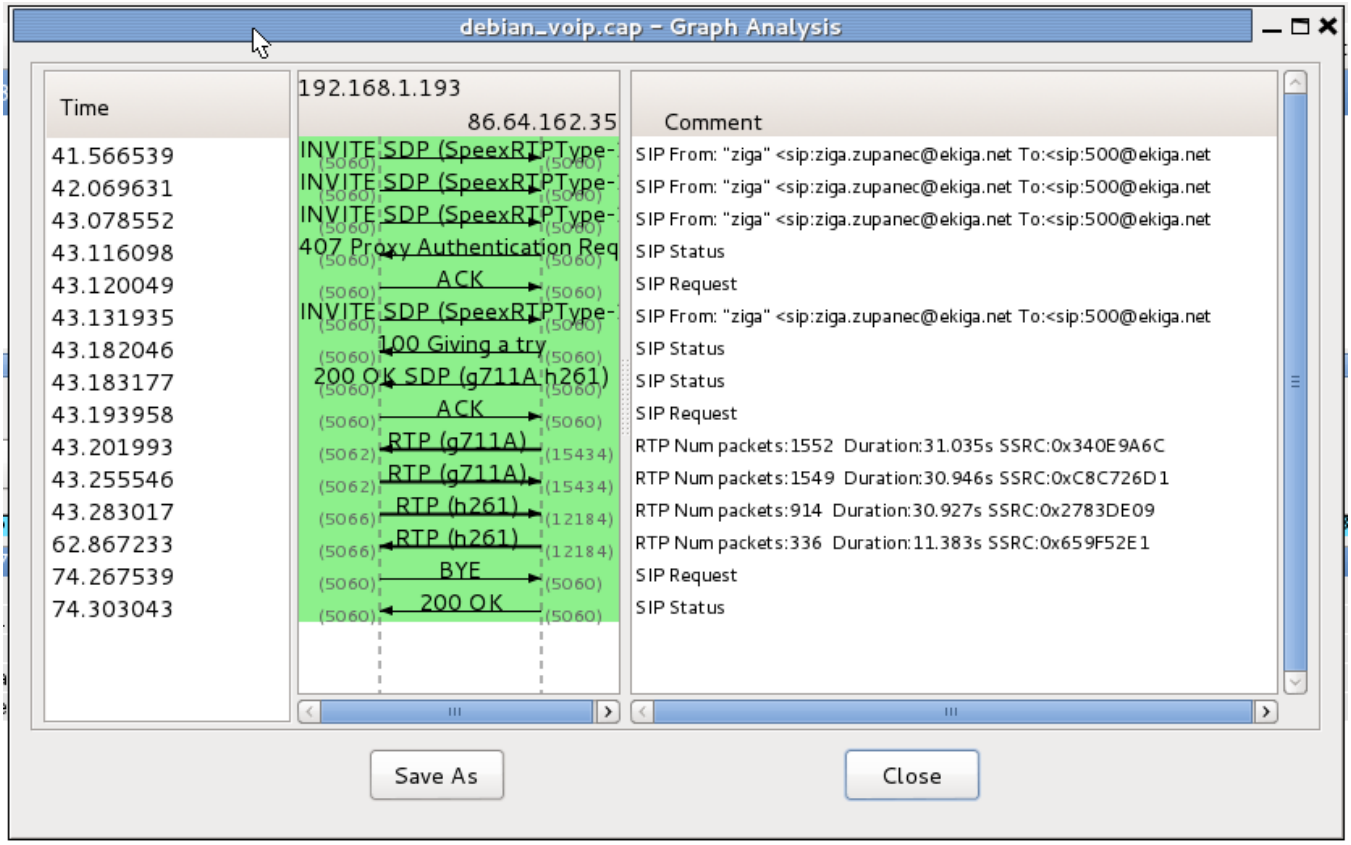



**Figure 10: Vzpostavitev in potek klica.**

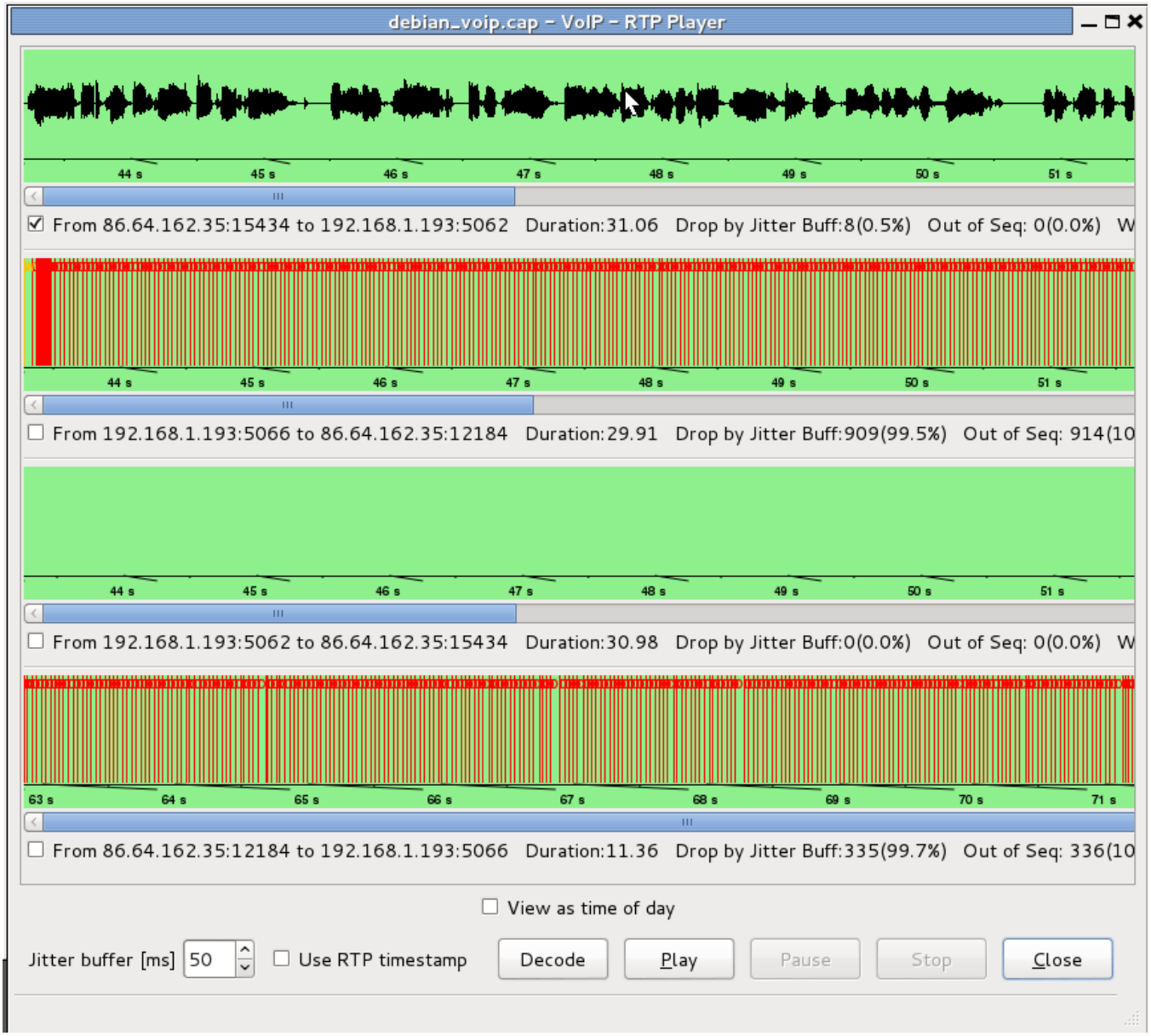



**Figure 11: Rekonstrukcija zvočnega zapisa s programom wireshark.**

# Dropbox analysis: Data remnants on user machines

Article review

Katerina Bashova
University of Ljubljana
Faculty of Computer and Information Science
Ljubljana, Slovenia
kb0548@student.uni-lj.si

Tjaša Saje
University of Ljubljana
Faculty of Computer and Information Science
Ljubljana, Slovenia
ts1846@student.uni-lj.si

Žan Anderle
University of Ljubljana
Faculty of Computer and Information Science
Ljubljana, Slovenia
za8454@student.uni-lj.si

## ABSTRACT

**Purpose** The present paper seeks to determine the value of cloud storage remains on user devices from a forensic standpoint. For this purpose, two main research methods were utilized: (1) a review of relevant literature pertaining to the researched topic and (2) a practical application of researched methods, involving the collection and analysis of cloud data remnants, focusing specifically on Dropbox cloud storage service.

**Design, Methodology & Approach** With the use of experimental research involving several virtual machines set with configurations simulating different cloud usage activities and by analyzing the results with common forensic software, the study attempts to identify data remains within Windows 7 OS and Google Chrome web browser.

**Findings** The key finding emerging from this study is that a variety of locations and files exist that can not only indicate the use of cloud computing but also reveal the contents of a cloud storage account as well as any additional information about the user.

**Practical implications** The paper demonstrates when and how cloud usage can be determined with the presentation of common locations and files the forensic examiner should search for when cloud usage is suspected. Simultaneously, the findings establish a substantial ground for future research.

## Categories and Subject Descriptors

C.2.0 [**General**]: Security and Protection; C.2.4 [**Computer Communication Networks**]: Distributed Systems—*Cloud Computing*; K.6.m [**Management of Computing and Information Systems**]: Miscellaneous

## General Terms

Theory, Practical application

## Keywords

Cloud storage, Cloud forensics, Dropbox, Computer forensics, Digital forensics

## 1. INTRODUCTION

Theoretical concepts of cloud computing date back to 1960s. "Computing may someday be organized as a public utility just as the telephone system is a public utility," said John McCarthy in 1961[6] and the main characteristics of what we today define as cloud computing, were explored in 1966 in *The Challenge of the Computer Utility*, a book by Douglas Parkhill. The term itself was not coined until the 2000s, but the telecommunications offerings of VPN services in late 1990s already closely resembled SaaS (Software as a service) offerings.

Since 2000, we have seen rise of many services, imperative in the cloud computing development, such as AWS (Amazon Web Services), Eucalyptus, OpenStack, and innumerous commercial cloud services. According to a 2012 study, cloud computing will have created nearly 14 million new jobs between 2011 and 2015, and that by 2015, business revenues from IT innovation enabled by the cloud could reach $1.1 trillion a year[4].

Given those numbers it should come as no surprise that cloud storage is a popular choice for personal or business storage and backup solution. For this research Dropbox was chosen as a usecase. Dropbox is only one of the major cloud storage hosting providers, others being Microsoft SkyDrive, Google Drive, iCloud, Ubuntu One, etc. Dropbox was officially launched in 2008 and in 2013 it had 200 million users.[3]

As popularity of the cloud storage increases, so does the danger of cyber criminal activity. It can be used as a storage for criminal data (child pornography) or for communication with different crime groups. With this arises a new issue - data can be spread among different machines and even different locations. Obtaining data of interest therefore presents a technical and also a jurisdictional problem. That by itself can hamper a forensic investigation, but the sheer number of cloud storage providers makes this an even bigger challenge. And that is what this research tries to address: by establishing what data remnants we can find on the user's computer after a cloud storage service has been used, we can help the forensic researchers to more quickly determine whether a cloud storage has been used or not. A forensic

researcher could then use this information to take further steps.

# 2. DROPBOX AS A CASE STUDY

## 2.1 Research focus

In light of recent trends in digital forensic research, Quick & Choo, 2013 [5] utilized Dropbox as a tool to investigate data remnants of cloud storage on user devices, focusing specifically on Windows 7 computer and Apple iPhone 3G. Their research incorporated the examination of remains as an after-effect of various user actions such as storing, uploading, and accessing data in the cloud. The main purpose of their research was to determine which data a forensic investigator should aim to obtain in case of a suspected cloud storage use. Their intent was to identify any usernames, passwords and files associated with the cloud storage account, determine whether the service was used via browser or software, analyze associated date-time stamps and any other remnants that could benefit the progress of the investigation. The paper also explored the correlation between the data remains and the use of different browsers to access cloud storage. To accommodate such analysis, four different browsers were employed in the exploration: Internet Explorer, Mozilla Firefox, Apple Safari and Google Chrome. Furthermore, the study makes a distinction between accessing stored files by means of a web browser and interacting with cloud data through a user installed application or software. Hence, the Dropbox Windows client software version 1.2.52 was installed for the use within Windows 7 OS, whereas on iPhone 3G, an installation of the Dropbox iOS Application was performed. Additionally, the research included the identification of data traces in situations of a user applying anti-forensic tools in an attempt to hide evidence of their cloud storage use. To measure the impact, Eraser 5.8.8 and CCleaner v3.17.1689 were used to simulate aforementioned concealment methods.

## 2.2 Research methodology

In order to collect the necessary information and replicate distinct circumstances of cloud usage, 28 virtual machines were produced, each with a different configuration. The study adopted 7x4 factorial design, with the VMs representing seven forms of communication with the cloud service for four different browsers: control VM, upload VM, uninstall VM, access VM, download VM, Eraser VM, CCleaner VM. The creation of test configurations is displayed on Figure 1 (for more details on the creation of VMs see Quick & Choo, 2013 [5]). Virtual machines were generated using VMware Player 4.0.1 and the analysis was undertaken with professional forensic software, namely Guidance Software En-Case (version 6.19.4) and AccessData FTK (versions 1.81.6 and 4.01).

Beside the activity circumstances included in the examination, the authors noted possible other situa-tions which may produce data that can aid the forensic examiner, including emails from Dropbox per-taining to a Dropbox account creation, emails between Dropbox users, or a shared link to files stored in a Dropbox account.

## 2.3 Research findings

### *2.3.1 Control*

The study observed that the control VM, in which no user actions relating to Dropbox were per-formed, contained references for the term Dropbox in index.dat files, msjint40.dll.mui, page-file.sys, and unallocated clusters. This suggests that the keyword search matches will exist even in the event of no cloud activity. Therefore, their presence in itself does not directly imply the undoubted use of a cloud storage service.

### *2.3.2 Browser access*

The research demonstrated that when a browser is used to access the Dropbox files, enough filename references on the hard drive are preserved to enable successful identification of the filenames the user accessed. Moreover, when files were downloaded, additional references were found in the $MFT and directory listing, link files among those. It must also be noted, that when accessing cloud files via a web browser, Dropbox also offers a feature called Show Deleted Files, which enables users to display deleted files. These files can be either restored or deleted from the browser permanently. Further, Dropbox creates a snapshot of a file every time it is uploaded, deleted or restored. Thus, event records can also be viewed, including the date and time for each event. Of note is also the observation that Dropbox preserves an account of every computer used to access and synchronize with a particular user account, along with the IP number. This feature may prove instrumental in determining whether a particular computer was synchronised to a Dropbox account. The study also revealed that the account username could be found when Dropbox has been accessed with Mozilla Firefox (located in the formhistory.sqlite database) and Google Chrome (in the Autofill Web Data file). Likewise, a Dropbox username can be determined when a user has stored their username with Internet Explorer. In addition, aside from Apple Safari, all browsers stored Dropbox website information in Cookie files, web history, FavIcons, and in the FileSlack of other files. Regardless of the type of browser being used, filenames of downloaded files were also located in the web history for the browsers. However, unlike in other circumstances, no thumbnails for the Dropbox sample pictures or experiment sample pictures were found on the drive when using a browser to access files. This illustrates that certain situations can occur when Dropbox will be used without leaving any trace in thumbnails, for example, accessing files using a browser without downloading them to a computer.

### *2.3.3 Client software*

Regarding the client software, the study observed that the Dropbox client software installs into the Users C:\Users\-[username]\AppData\Roaming\ folder regardless of the browser type. Sample files and folders found at the default Dropbox folder location (C:\Users\[username]\Dropbox) comprised Getting Started.pdf, two How to use text files, and three sample pictures. When compared to the access with a browser, the study found less filename references. However, when the Dropbox client software was used, Dropbox software file references were found not only in registry hives SOFTWARE and SYSTEM but also in UsrClass.dat registry files. No such references were detected when a browser was used to access or download files from the cloud.

### *2.3.4 Uninstallation*

As the examination of the consequences of a user uninstalling Dropbox client software demonstrated, apart from Dropbox.exe file, which was marked as deleted, all other files were retained. Moreover, the Dropbox sync folder, all remains of the client software and all file contents at the location: \[username]\Dropbox\, remained unaffected by the uninstallation. This finding suggests that even after a user uninstalling Dropbox client software, the data will still remain on the hard drive to be re-covered by the examiner.

#### *2.3.5 Eraser and CCleaner*

When all features of Eraser and CCleaner were used, no references to the filenames of the uploaded files were found. However, even after erasing was performed, there was significant information found in prefetch files, such as notepad.exe.pf, wordpad.exe.pf, explorer.exe.pf and dllhost.exe.pf to suggest the presence and path of the uploaded data files and the sample Dropbox files. The erasing also re-moved all references in the Mozilla Firefox and Internet Explorer use situations, however information in the Google Chrome FavIcon history remained, including dates and times for last access. Moreover, NTUSER.dat registry files in Chrome and IE CCleaner-VMs contained deleted data, including filenames and URL references. The same files were not located when Firefox was used. Therefore, using Eraser or CCleaner does not erase all data traces regarding to the manipulation of Dropbox files nei-ther does it completely remove the information relating to the previous presence of the uploaded files or the Dropbox sample files.

### 2.4 Discussion

As illustrated, there is a variety of ways for an investigator to determine whether a Dropbox account has been accessed using client software or a web browser, including directory listings, thumbnails, registry, browser history, link files, prefetch files, and memory captures. This research inquiry established that all the information, inclusive of Dropbox username, URLs in the network traffic, filename data in prefetch and link files, filename references in directory listings, RecentDocs and typed URLs in registry files, browser history, and information recovered from memory files, accurately indicated Dropbox use and could prove invaluable in an investigation relating to cloud storage. However, the scope of this study was limited, since a typical use of cloud storage over an extended time period and numerous user operations may produce a larger volume of data remains. Nonetheless, the paper forms a solid base for future research in this field, since it demonstrates that even minimal cloud storage activity could generate sufficient data remnants for the investigator to examine.

## 3. METHODOLOGY

### 3.1 Research contributions, objectives and strategy

The present paper builds on the research of Quick & Choo, 2013 [5] and offers further support to the notion that cloud storage service, user files and user account information can be determined by examining data remnants on user devices. In the attempt to move beyond mere replication of the methodology and findings, the paper extends the generalizability of results by using most recent versions of forensic tools and Dropbox client software as well as examining an expanded range of file formats. The principal aim of the paper is to determine whether traces can be found of user files, default files from the Dropbox software, browsing history and any associated usernames and passwords. In order to achieve this aim, the objectives of this research were to:

1. Critically examine the findings of prior research relating to the remnants of Dropbox cloud storage use.
2. Critically evaluate methods, concepts and tools in relation to digital forensic appraised as in consonance with the research topic.
3. Produce a critical analysis of the data, collected through a research strategy and with forensic tools assessed as the most expedient for the present research inquiry.
4. Arrive at a conclusion based on the findings and to make recommendations that can be of use for forensic investigators as well as academic researchers interested in further investigations of the cloud storage forensics.

In order to facilitate replication, reliability and external validity, the study adopts a highly structured and systematic research strategy. The investigators performed an experimental research, as it was found to be the most applicable for collection of the required data. Additionally, experimental method facilitates the successful identification of distinct outcomes engendered by different experimental inputs.

### 3.2 Research scope

Due to time and resource limitations, this study aimed to locate and identify data remains using a single browser on a singular operating system. According to recent statistic reports [11], Google Chrome is considered the most widely used web browser. An extensive prevalence of a certain browser engenders a higher possibility of this exact browser being used by the investigated individual. Following this assumption, Google Chrome was employed as a default browser for the present research inquiry. Using analogous reasoning, the analysis was performed on a standard personal computer (PC) with installed Windows 7 OS, as the latter retains its dominance in the world market [12, 8]. Additionally to the examination of the standard Dropbox access via a web browser, the paper also explored the traces left by the Dropbox client software usage. Furthermore, the paper also examined the remnants when a user applied anti-forensics to hide the evidence of cloud use. The inspection of data remains included the analysis of a browsing history, cookies, user's home directory keyword search matches and the registry hives.

### 3.3 Research methodolgy

To gather primary data, the present investigation was performed with the use of virtual machines, since this method proved reliable in prior studies [5]. Also, as previously highlighted by several authors, virtual machines represent a quick and manageable way to install, setup, delete, and duplicate different test configurations. For the purpose of this study and following the example of Quick & Choo, 2013 [5], six virtual machines were generated, each simulating a distinct

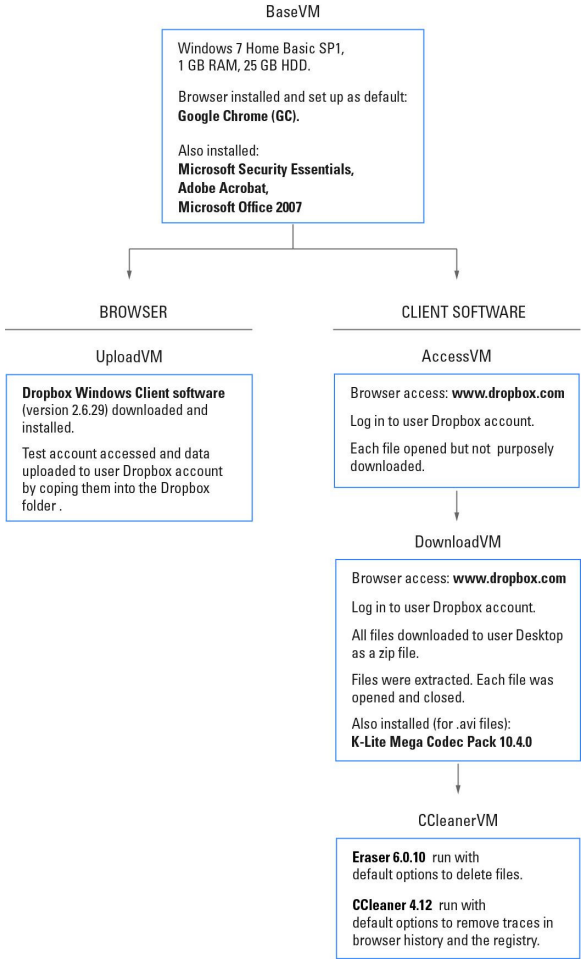



**Figure 1: Created VMs**

cloud use situation. All VMs were created with Oracle VirtualBox 4.3.10 and had Microsoft Windows 7 64bit OS installed with 1GB RAM and 25GB HDD. The VMs were named in accordance with the particular use scenario: BaseVM, UploadVM, AccessVM, DonwloadVM, and CCleanerVM. The experiment was conducted with the use of Google Chrome version 34.0.1847.116.

## 3.4 Selection of test files

To test the differences in remains of various file types, the experiment used 17 distinct test files, including files with the following extensions: .txt, .docx, .pdf, .png., .jpg, .avi, .mp3, and .zip. Similarly to Quick & Choo 2013 [5], the first part of test files was acquired from UC Berkley Enron Emails subset data files [2]. The file 175721.txt was converted to a .docx file and the 175316.txt was saved as .pdf using Adobe PDF Converter. File 175490.txt was converted to a .png file and renamed 228981.png, while 175752.jpg was computer-generated. The .zip file contained a pdf file with complete works of William Shakespeare. These test files were chosen for the specific purpose of this study due to their uncommon content. The uniqueness of the files not only facilitates more accurate keyword searches, it also limits the possibility of finding similar content on an average personal computer. However, it is important to note, that in a typical use situation, recovering data remains may be hindered by the similarities between files. Hence, the remaining .txt file, three .jpg files, as well as the .avi file and the .mp3 file were selected at random.

## 3.5 Research procedure

The experiment involved two stages. In the first stage, a test persona was developed, along with their user email and Dropbox accounts. With the help of an online random name generator [10] the investigated user was named Raymond Beck. With the addition of a random number to avoid collision with other gmail accounts, a gmail account raymond.beck431@gmail.com was produced. Using the email address as a username, an empty Dropbox account was created. The passwords for both the Gmail and Dropbox accounts were generated using the website Norton Password Generator [9]. The second stage included the making of typical Dropbox use situations. The test scenarios simulated various eventualities pertaining to the cloud usage with each virtual machine corresponding to a particular scenario. Prior to the creation of VMs the MD5 sums of all the seventeen files were calculated and later used to search the files in forensic copies. The creating of experiment scenarios was conducted in five steps as displayed in Image 1. The UploadVM and AccessVM were created from BaseVM with an open source program CloneVDI v2.05. The same tool was used to make DownloadVM from AccessVM and a CCleanerVM from DownloadVM. In order to display files in a correct manner, additional software was installed on certain VMs 1. In CCleanerVM, two erasing tools were installed and used, namely Eraser 6.0.10 and CCleaner 4.12. Finally, to keep the originals intact, forensic copies of all VMs were produced to use in the analysis. The MD5 hash value of each VM was calculated to verify the copies.

## 3.6 Data Analysis

The analysis of the VM forensic copies was undertaken with common forensic software, more specifically Autopsy versions 2.24 and 3.0.09, Registry Decoder Version 1.1, SQLite Expert Personal version 3.3.29 and SQLiteman v 1.2.2. The aforesaid tools were evaluated as the most applicable for the present research inquiry primarily due to their free accessibility. Additionally, the selected tools are perceived as highly effective in recovery and verification of data, a view supported by existing academic research and testing reports[1, 7].

# 4. FINDINGS AND DISCUSSION

The main focus of the analysis were the traces found in user's home directory, Google Chrome's browsing history, cookies and the registry hives. For that purpose, the BaseVM, which was treated as a control base, was tested against four experiment VMs: UploadVM, AccessVM, DownloadVM and CCleanerVM. The summary of findings is displayed in Image 2.

## 4.1 Control (BaseVM)

The analysis conducted with Autopsy and Registry Decoder found no traces indicating the access to Dropbox with a web browser or the use of Dropbox client software. Also, no test files or references to Dropbox sample files or Enron test files were discovered, confirming that no data was originally

| Control (BaseVM) | Data artifacts found |
|---|---|
| Username | Nil |
| Password | Nil |
| Software | Nil |
| URL | Nil |
| Enron sample filenames | Nil |
| Enron sample files | Nil |
| KWS terms | Matches found for 'dropbox' |
| **Client Software (UploadVM)** | **Data artifacts found** |
| Username | Nil |
| Password | Nil |
| Software | Installed Dropbox in /Users/Raymond/AppData/Roaming/Dropbox |
| URL | Users/Raymond/AppData/Local/Google/Chrome/User Data/Default<br>'History' database: 'downloads' (installer), 'download_url_chains', 'urls' and 'segments'<br>Several cookies found in (chrome's) 'Cookies' database in table 'cookies'<br>Dropbox icon found in 'Favicons' database in 'favicon_bitmaps' table |
| Enron sample filenames | Multiple matches |
| Enron sample files | All the files found in /Users/Raymond/Dropbox |
| KWS terms | Multiple matches to KWS terms |
| **Browser Access (AccessVM)** | **Data artifacts found** |
| Username | Google Chrome History 'Web Data' database: 'Autof ill' (Raymond.beck431@gmail.com) |
| Password | Nil |
| Software | Nil |
| URL | Cookies database: dropbox links<br>Favicon database: 'icon_mapping', 'favicon_bitmaps'<br>Web Data database: 'autof ill' (Raymond.beck431@gmail.com)<br>History database: 'segments', 'urls' |
| Enron sample filenames | Sufficient references to the filenames in registry and browsing history |
| Enron sample files | Nil |
| KWS terms | Multiple matches to KWS terms |
| **Browser Download (DownloadVM)** | **Data artifacts found** |
| Username | Google Chrome History 'Web Data' database: 'Autof ill' (Raymond.beck431@gmail.com) |
| Password | Nil |
| Software | Nil |
| URL | Cookies database: dropbox links<br>Favicon database: 'icon_mapping', 'favicon_bitmaps'<br>History database: 'segments', 'urls' |
| Enron sample filenames | Sufficient references in registry |
| Enron sample files | In /Users/Raymond/Dropbox, and via uncompressed zip or folder name; 'dropbox.zip' |
| KWS terms | Multiple matches to KWS terms |
| **Anti-forensics (CCleanerVM)** | **Data artifacts found** |
| Username | |
| Password | Nil |
| Software | Nil |
| URL | Google Chome's SQLite databases:<br>"History" database<br>"Favicons" database: Dropbox icon |
| Enron sample filenames | Nil |
| Enron sample files | Nil |

**Figure 2: Summary of the findings**

present relating to Dropbox use. However, when searched for the keyword dropbox links in the form dl.dropbox.com were found. This suggests that there were existing references to the term Dropbox already present on the platform. The fact that these keyword matches occurred in a VM with no previous cloud activity posits that they are not a direct consequence of Dropbox usage. This key finding is concurrent with results of Quick & Choo, 2013 [5] and should also be taken into account when determining cloud usage. As outlined, a forensic examiner may not be able establish the definite use of Dropbox based on keyword matches alone and would do well to examine additional locations of remnants presented in this paper.

## 4.2 Experiment configurations (UploadVM, AccessVM, DownloadVM)

### *4.2.1 Browser analysis*

The web browsing history was analyzed using Autopsy Version 2.24 for Linux OS, Autopsy Version 3.0.9 for Windows OS, SQLite Expert Personal 3.3.29 and SQLiteman v 1.2.2. In all test VMs traces of Dropbox usage were found in Google Chome's default browsing history's databases, located in \Users\Raymond\AppData\Local\Google\Chrome\User Data\-Default. A significant amount of remains was found in UploadVM, AccessVM, and DownloadVM, whereas in CCleanerVM hardly any traces were detected.

In DownloadVM, in Chrome's Login data database, dropbox url was found in the table logins, which suggests a user login to a Dropbox account occurred. Additionally, in AccessVM and DownloadVM the table autofill from the database Web Data contained the user's login address. All of this is in line with previous studies [5] and may clearly indicate the use of a Dropbox account as well as determine which user accessed Dropbox. Following this we may assume that, since the user address was also found in AccessVM, accessing and opening files within Dropbox web service is sufficient to identify the account and thus no downloading of files is required.

In the History database of UpIoadVM, a Dropbox installer in the table downloads was found. This finding may assist the forensic investigator in determining whether Dropbox client software was downloaded on a user's device. Dropbox references in UploadVM and DownloadVM were also found as a part of download urls in table download_url_chains and dropbox urls in tables urls and segments.

Furthermore, in Chrome's History database of DownloadVM, a download reference of the test files (downloaded as dropbox.zip) was found in the table downloads. This indicates that the user performed a login and downloaded the files as a zip file.

In UpIoadVM, AccessVM and DownloadVM several cookies were found in Chrome's Cookies database in table cookies. Additionally, in all of the VMs, the Dropbox icon was found in Favicons database in favicon_bitmaps table.

### *4.2.2 Registry*

The registries were analyzed with Registry Decoder version 1.1., which found Dropbox remnants in UploadVM and references to files in both UploadVM and DownloadVM. In the

**Table 1: The selected files with their MD5 values**

| File | MD5 |
|---|---|
| 100_3731.JPG | a0aef9fa956775f27dcc074f2b0dd55e |
| 100_3732.JPG | be17531c1fd9330711716dacde881c2c |
| 173415.pdf | 416fc0bc70868d3c9173797cb59121d1 |
| 175142.txt | 48216bde4bbde63268df99a3039c1b30 |
| 175221.txt | e177d7739397822dae26d8d1ba61f095 |
| 175316.txt | 43b77967490bcf5b84476b50ce943c0b |
| 100_3733.JPG | 3d78fb186d13c7b6ab6cfd89d034778c |
| 175317.txt | 11f7e681ef92791331321c6b2996c4ca |
| 175449.txt | ad7e1cd6b411e61fc297990915e3517e |
| 175490.txt | 2104f08d05c8b0f1237f404c4b546e1c |
| 175560.txt | 0a1c411a3c8553b97ed76a7c303f0104 |
| 175721.docx | 43ebfab14e3f5290c3f670c35154c102 |
| 175752.png | ae96521f5dac76cbb31739e88793c975 |
| 228981.jpg | 2104f08d05c8b0f1237f404c4b546e1c |
| 52992.mp3 | 47ccbb724e2ab61e19022f0773868541 |
| 54658.rar | f93282057a273366f6e8a4fac8589bdd |
| 176960.avi | 3bb351e344554f26992783cda14bf654 |

UploadVM, the references to files uploaded to the dropbox account were found with a plugin called "Shell Bags" in the registry file group NTUSER. Moreover, as analyzed with the plugin "User Software" the Dropbox software was listed as installed software. It is proposed, that this finding may help the investigators to determine when Dropbox client software was installed and which files were uploaded. Although no references to Dropbox default files or uploaded test files were found in DownloadVM, the analysis uncovered references to files with two types of plugins: Shell Bags and Recent Docs. The first finding suggests the presence of the files in the system and the latter indicates that the files had been accessed by the user.

All this demonstrates that a sufficient amount of Dropbox remains will be present on a user device, regardless of the mean used to access, store and open files. No significant differences exist between the remains of Dropbox client software and the traces of web browser access. Additionally, it appears that the traces are independent of the file-format, since no clear divergence between remnants of different files was detected.

### 4.3 Results of applying anti-forensic techniques (Eraser,CCleaner)

The analysis of the remaining virtual machine, CCleanerVM, exhibited the presence of the tools Eraser and CCleaner. This fact alone should be taken into consideration when searching for cloud data remains, since it indicates the possibility of remains being removed from the device. In contrast to the results of Quick & Choo, 2013 [5], no keyword search matches occurred. Moreover, the analysis of the CCleanerVM registry found no remnants of test files or Dropbox sample files. This indicates that Eraser and CCleaner may successfully remove all traces of cloud files from a user device. However, in line with the study of (Quick & Choo, 2013) [5], remnants were found Google Chrome FavIcon history. Although this suggests that Dropbox was accessed, it fails to provide any additional information, for instance, whether user login occurred. To conclude, the examiner must be aware of any possible anti-forensic methods that may affect the remnants cloud data.

## 5. CONCLUSION

The main objective of this research was to explore the forensic value of cloud data remains and how it can be used to promote the investigation. The results of this study provide a basis for understanding what data traces can be retrieved and where to find them. As a result, the study offers interesting and useful implications for forensic examiners and academics alike. The research outlined how to determine whether cloud storage was used, identify the type of cloud service, and discover stored files and any associated data about the files and the user. As demonstrated, there is a wide range of locations and files that can aid the examiner in determining the use of Dropbox, including directory records, link files, browser history, cookies and the registry. The study met its endeavor demonstrating that the growth of cloud computing not only provides diverse possibilities for the person of interest to distribute, access and hide information, but also acts as a new vehicle for forensic investigators to discover relevant digital evidence.

### 5.1 Limitations and Future Research

While contributing to the understanding of data remains of cloud storage use, the present study has limitations that need to be noted. In terms of research scope, data remnants analyzed in this paper are limited to Windows 7 platform. Hence, a cross-platform analysis including other popular operating systems may provide more insight and extend the practical value of the results. Additionally, future research may build on the experiment to include alternative cloud storage services, such as Google Drive, Microsoft OneDrive, iCloud and UbuntuOne. Furthermore, as the experiment was conducted in only one point in time with the use of current versions of forensic tools, the generalizability of the findings is limited. Not only is Dropbox as a cloud service updated regularly, the study results are tied with the particular version of the forensic software and its features. Replications are needed to establish the predictive validity and generalizability of results as well as develop a consistent cloud forensic methodological procedure, such as the one adopted in this research.

## 6. REFERENCES


[1] P. Bennett D. J., & Stephens. A cognitive walkthrough of autopsy forensic browser. *Information Management & Computer Security*, 17:20–29, 2009.

[2] Berkeley. Uc berkeley enron email analysis, 2012. retrieved May 2014.

[3] J. Constine. Dropbox hits 200m users, unveils new 'for business' client combining work and personal files, November 2013. retrieved May 2014.

[4] J. F. G. Cushing Anderson. Climate change: Cloud's impact on it organizations and staffing. *IDC*, November 2012.

[5] K.-K. R. C. Darren Quick. Dropbox analysis: Data remnants on user machines. *Digital Investigation*, 10:3–18, February 2013.

[6] S. Garfinkel. The cloud imperative, October 2013.

[7] . T. J. Manson D., Kaufman M. Is the open way a better way? digital forensics using open source tools. *HICSS '07 Proceedings of the 40th Annual Hawaii International Conference on System Sciences*, page 266b, 2007.

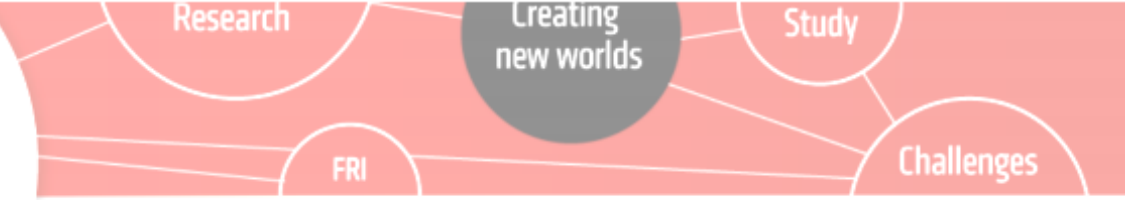
[8] NetMarketShare. Desktop operating system market share, April 2014. retrieved May 2014.

[9] Norton. Password generator, 2014. retrieved May 2014.

[10] B. the Name. Random name generator, 2014. retrieved May 2014.

[11] w3schools. Browser statistics, April 2014. retrieved May 2014.

[12] w3schools. Os platform statistics, April 2014. retrieved May 2014.




# Digitalna forenzika na omrežjih

# Preprečevanje nepooblaščene rabe mobilnega telefona s pomočjo biometričnih podatkov

## Analiza uporabe biometričnih sistemov v mobilnih napravah

Miha Mohorčič
Fakulteta za računalništvo in informatiko
Tržaška cesta 25
Ljubljana, Slovenija
mm8803@student.uni-lj.si

Rok Povšič
Fakulteta za računalništvo in informatiko
Tržaška cesta 25
Ljubljana, Slovenija
rok.povsic@gmail.com

Marko Škrjanec
Fakulteta za računalništvo in informatiko
Tržaška cesta 25
Ljubljana, Slovenija
marko.skrjanec@gmail.com

## POVZETEK

Večina uporabnikov mobilnih telefonov za zaščito pred vdorom uporablja preproste načine varovanja. Večinoma se poslužujejo PIN številk, gesel in vzorcev. Zaradi obilja kraj pa čedalje bolj postaja jasno, da so potrebni boljši in učinkovitejši načini varovanja mobilnih naprav in podatkov na njih. Varovanje prenosnih telefonov in podatkov s pomočjo biometričnih sistemov izgleda kot zelo verjetna in priporočljiva možnost [5, 13, 14, 15, 18]. Glede na popularnost in čedalje bolj množično implementacijo skenerjev prstnih odtisov ter aplikacij za prepoznavo obrazov na telefonih, je razvidno, da gre razvoj v to smer. Poleg teh dveh najbolj pogostih načinov pa obstaja še mnogo drugih biometričnih sistemov za varovanje mobilnih telefonov. Vsi ti biometrični sistemi pa še vedno imajo pomanjkljivosti.

## Ključne besede

Mobilne naprave, mobilna varnost, mobilni telefoni, biometrika, biometrični podatki

## 1. UVOD

V zadnjem desetletju smo bili priča tehnološkemu preskoku, ki omogoča večjo mobilnost elektronskih naprav. V skladu s tem se je močno povečala količina podatkov, ki jih vsebujejo naše prenosne naprave. Pametni telefoni tako lahko hranijo vsebino naših poštnih predalov, upravljajo naše urnike in koledarje, zbirajo podatke o lokacijah, hranijo naše slike in dokumente. Zaradi vsesplošne dostopnosti interneta so postali uporabni tudi za uporabo v bančništvu in spletnih nakupih. Povprečni uporabnik pametnega telefona tako hrani na eni napravi veliko količino osebnih in varnostno občutljivih podatkov. Podatki, kot so zgodovina klicev, zgodovina sporočil SMS in MMS, kontakti, številke, zgodovina lokacij, slike, dnevniki, koledarji, dostopi do omrežij, gesla, certifikati, dostopi do spletnih strani in še mnogi drugi, so vsi hranjeni na mobilnem telefonu.

Vrednosti mobilnih naprav so se zelo povečale in s tem tudi kraja le teh[1]. Vse te naprave pa niso postale zanimive za nepridiprave zgolj zaradi obilja povpraševanja na črnem trgu, pač pa tudi zaradi količine osebnih podatkov prisotnih na mobilnih napravah. Iz ukradenih naprav lahko nepridipravi dostopajo do velikih količin osebnih in drugih občutljivih podatkov. Zaščita mobilnega telefona in samih podatkov na njem je postala še toliko bolj pomembna.

Na te spremembe so se odzvali proizvajalci mobilnih naprav, ki omogočajo varovanje s pomočjo biometričnih podatkov. Dva največja proizvajalca mobilnih naprav (Apple in Samsung) sta v svoja najbolj prodajana modela vključila bralnik prstnih odtisov. Splošna javnost je tako v prepričanju, da je prstni odtis unikaten, verjela obljubam proizvajalcev o popolni varnosti njihovih podatkov, vendar se je kmalu izkazala pomanjkljivost tega načina zaščite. Biometrični bralnik obeh omenjenih modelov je bil zlomljen[2][3] manj kot teden dni po pričetku prodaje naprave.

Kljub temu, da je zaščita relativno lahko zlomljiva, je še vedno potrebno znanje s področja biometrike, da lahko ponaredimo odtis. Ob vseh teh zlorabah so tako forenziki postavljeni pred težko nalogo kako dokazati zlorabo biometričnega sistema in poiskati storilca. Namen zaščite pa je postal predvsem pridobiti čas za izbris osebnih podatkov na daljavo.

Članek je organiziran v 8 poglavij. 2. poglavje opisuje različne biometrične sisteme, ki se uporabljajo ali pa so bili samo razviti. 3. poglavje opisuje varnostne metode na strojni ravni in dva primera takšnih implementacij. 4. poglavje opisuje delovanje Android prepoznave obrazov, ki ga vsebujejo vsi telefoni z novejšim Android operacijskim sistemom. Naslednje 5. poglavje opisuje bralnike prstnih odtisov

[1] http://www.policija.si/index.php/preventiva-/preventiva/423-kraje-mobilnih-telefonov-preventivni-nasveti, maj 2014
[2] http://www.ccc.de/en/updates/2013/ccc-breaks-apple-touchid , april 2014
[3] https://slo-tech.com/novice/t606747 , april 2014

in same prstne odtise. 6. poglavje na kratko opiše uporabo umetne inteligence pri izboljšavi biometričnih sistemov. 7. poglavje opisuje glavne točke pri forenzični preiskavi mobilne naprave. Zadnje poglavje pa vsebuje kratek povzetek članka in zaključek.

# 2. BIOMETRIČNI SISTEMI

Za zaščito pred nepooblaščenim dostopom so bili razviti biometrični sistemi, ki zajemajo različne lastnosti osebka. Sistemi se tako lahko odločajo glede na podlagi slike (računalniškega vida), glasu (proizvedenega zvoka) in določenih akcij posameznika. Namen biometričnih sistemov je predvsem olajšanje avtorizacije in/ali avtentikacije posameznika. Želja je sistem zgraditi tako, da bo za uporabnika naraven in nemoteč, hkrati pa visoko zanesljiv in ponovljiv zgolj s strani pravega uporabnika. Z vidika forenzike je potrebno pogledati, kakšne so možnosti potencialne forenzične obravnave takih sistemov.

## 2.1 Računalniški vid

Vsi novejši mobilni telefoni imajo vgrajene relativno zmogljive kamere. Aparati v zadnjem času omogočajo visokoresolucijske posnetke tudi s sekundarno kamero na sprednji strani telefona. Za ta namen so bili implementirani razni sistemi, ki temeljijo na algoritmih računalniškega vida za prepoznavo obrazov in telesa, ki iz slike zajete s pomočjo kamere avtenticirajo posameznika.

### *2.1.1 Prepoznava obrazov*

To je verjetno najbolj razširjen in poznan tip biometričnih sistemov mobilnih telefonov [8, 9, 13]. Že nekaj generacij telefoni s pomočjo sekundarne kamere ob odklepanju zajamejo sliko obraza osebka in jo s tehnikami prepoznavanja obrazov uporabijo kot ključ za dostop do telefona. Tehnika je za uporabnika relativno preprosta, vendar se je ob uporabi v realnem svetu pokazalo, da ima relativno veliko napak. Zaskrbljujoče je to, da se je v večino sistemov možno prijaviti s sliko posameznika, ki pa jo lahko za večino posameznikov pridobimo kar iz socialnih omrežij [14]. Z napredkom tehnologije se izboljšuje stopnja prepoznave in število nelegalnih dostopov, predvsem ob uporabi novih tehnologij, kot so na primer 3D slike in infrardeči posnetki.

Glede na študijo [9] poznamo dva tipa aplikacij prepoznave obrazov. Prvi tip je identifikacija obraza glede na shranjeni obraz. Drugi tip pa je avtorizacija obraza za dostop. V tem članku je opisan sistem, ki na podlagi obraznih točk zajete slike omogoči ali onemogoči vse funkcije sistema.

Ob primerni implementaciji ima lahko tak način veliko forenzično vrednost. V primeru, da aparat zajete slike shranjuje na neki oddaljeni lokaciji skupaj s časovno kodo in GPS koordinatami, lahko te podatke uporabimo kot forenzično gradivo z relativno veliko zanesljivostjo [8].

### *2.1.2 Prepoznava telesa*

Ob uporabi 3D tehnologije in visokoresolucijskih kamer lahko vzamemo v biometrično obdelavo kar celoten osebek. Implementacijo takega sistema lahko vidimo v obliki Microsoftovega Xbox One, ki s pomočjo 3D kamere in slike prepozna uporabnika in ga samodejno prijavi v sistem. Z miniaturizacijo naprav lahko podobne sisteme pričakujemo tudi v naslednjih generacijah prenosnih naprav.

Uporabnik se lahko avtenticira tudi na podlagi svoje hoje, kot opisano v članku [5]. Hoja se za avtentikacijo uporabi le, če uporabnik dejansko hodi, v primeru pa da uporabnik stoji pri miru pa se uporabi PIN številka. Za uspešnost takšnega načina avtentikacije je potrebna tudi uporaba umetne inteligence.

Podobno kot pri prepoznavi obrazov lahko ob primernem beleženju dogodkov take podatke forenzično obdelujemo. Zaradi narave sistema, ki poleg obraznih potez v obzir zajame še gibanje osebka in obnašanje, je možnost napak manjša, v primeru nepooblaščenega dostopa pa pridobimo še dodatne informacije o osebku (višina, način hoje ipd.).

### *2.1.3 Forenzična vrednost*

Mnenja smo, da je uporaba računalniškega vida za biometrično prepoznavo možna, hkrati pa ima lahko tudi veliko vrednost ob možni forenzični preiskavi. Ob izboljšanju ravni prepoznave in stalni povezavi v omrežje, je tak način tudi varen. V primeru, da se vsak dostop do naprave beleži z ustreznimi metapodatki, zraven pa se doda tudi slika s katere smo določili prepoznavo, smo mnenja, da je tak način uporaben tudi za kasnejšo forenzično obdelavo in kot možen dokaz v postopkih.

## 2.2 Prepoznava glasu

Prepoznava glasu predstavlja manj razširjeno obliko biometričnega preverjanja, vendar je prisotna v nekaterih sistemih [11, 17, 18]. Na podlagi uporabnikovega vzorca in digitalne analize njegovega glasu prepoznavamo objekt s pomočjo izgovorjave določene fraze ali besede. Kljub temu, da je glas s pomočjo digitalne obdelave lahko ločiti med osebki, je metoda lahko zlomljiva ob posnetkih žrtvinega glasu. Tako je možno glas ob prvi uporabi posneti, ob naslednjih pa zgolj predvajati njegov posnetek. Varnost s tega stališča se da povečati s tem, da mora posnetek vsebovati določeno frazo oz. geslo. Vendar metoda prepoznave glasu ni zanesljiva v hrupnem okolju ali ob spremembah glasu objekta (prehlad, adolescenca...).

Uporaba zgolj prepoznave glasu pri sistemu za avtentikacijo tako ni zelo zanesljiva metoda [18], saj sama v večini primerov ne loči med predvajanjem kakovostnega posnetka in dejanskim živim govorcem. Ob hkratni uporabi drugih sistemov (npr. računalniškega vida) pa doda stopnjo varnosti in kredibilnosti pridobljenim podatkom. V članku [18] je opisan primer boljšega sistema, ki uporablja prepoznavo glasu in prstnega odtisa za prijavo.

## 2.3 Prstni odtis

Prepoznava prstnih odtisov je v zadnjem času postala del mnogih mobilnih naprav in aplikacij [12, 13, 14, 16]. Bralci prstnih odtisov se pojavljajo na najbolj priljubljenih modelih telefonov največjih proizvajalcev. Zaradi unikatnosti prstnih odtisov je njihova uporaba relativno varna, saj je za uspešno replikacijo prstnega odtisa potrebno kar nekaj časa, opreme in znanja. Po zajemu prstnega odtisa je potrebna izdelava odlitka, novejši bralci pa testirajo tudi za živo tkivo. Enostavnejši to počnejo z zaznavanjem vlage v prstu, boljši pa s pomočjo električne prevodnosti kože.

Potencialna težava pri uporabi bralnika prstnih odtisov na telefonu se pojavi zaradi dejstva, da se telefoni uporabljajo z rokami in obstaja velika verjetnost, da ob kraji telefona na njem ostane prstni odtis žrtve. Tako lahko usposobljeni tat prstni odtis enostavno zajame.

V študiji [13] so se poizkušali prebiti v sistem, ki za avtentikacijo uporablja prstne odtise. Izkazalo se je, da se je možno z umetnimi kopijami prstnih odtisov prijaviti v sistem v kar 81% primerih. V članku [14] je prikazan preprost postopek pridobivanja prstnega odtisa s površine in izdelava kopije. Avtorji so lahko z namakanjem umetnega odtisa v vodo preslepili naprednejši bralec prstnih odtisov in se uspešno prijavili v sistem. Ob pridobitvi prstnega odtisa in izdelavi kopije lahko usposobljen tat dostopa do vseh osebnih podatkov na telefonu.

Glede na videno smo mnenja, da je uporaba prstnega odtisa primerna za identifikacijo osebka, ne pa tudi za njegovo avtentifikacijo (uporabniško ime, ne geslo). Tehnološki napredek je tako dohitel bralnike, saj je s slike prstnega odtisa, narejene s telefonom, in z uporabo določenih forenzičnih znanj trenutne bralnike prelahko preslepiti. Za samo varnost tako predlagamo hkratno uporabo z drugimi biometričnimi sistemi. Prav tako, je v primeru obravnave takega gradiva v forenzični preiskavi, glede na dokazane ranljivosti, težko ugotoviti zanesljivost zbranih podatkov o identifikaciji.

## 2.4 Kombinirani biometrični sistemi

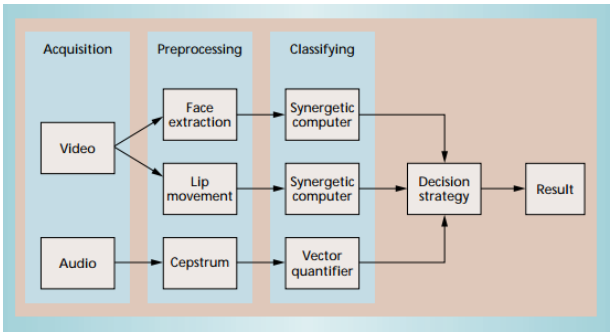
Glede na pomanjkljivosti posameznih biometričnih sistemov lahko za doseganje večje zanesljivosti in varnosti združimo več že opisanih sistemom [5, 6, 18, 20, 21]. Ta združitev pomeni veliko dela za potencialne tatove, kar bi jih mogoče odvrnilo vsaj od zajemanja in izrabe osebnih podatkov.

### *2.4.1 ePet*

Članek [20] opisuje avtentikacijo z vsemi opisanimi metodami, kombiniranimi z umetno inteligenco. Avtorji članka predlagajo združitev vseh teh metod, ker so posamično preprosto preveč ranljive. Rezultat tega dela je predstavitev avtentikacijskega sistema, ki se prilagaja uporabniku kot virtualni hišni ljubljenčki. Sistem za avtentikacijo ePet ne uporablja le obraza, prstnega odtisa in glasu, ampak tudi informacijo o lokaciji in hoji uporabnika. Umetna inteligenca je vključena pri pametnem zajemanju in upoštevanju podatkov ter pri sami avtentikaciji. Uporaba fizičnih in vedenjskih značilnosti uporabnika, vključno z informacijami o okolju, naj bi prinesla zelo učinkovito in težko zlomljivo avtentikacijo.

### *2.4.2 BioID*

Enega prvih kombiniranih biometričnih sistemov je že leta 2000 implementiralo podjetje Dialog Communication Systems in ga poimenovalo BioID [6]. Sistem deluje tako, da vizualno prepozna obraz, premikanje ustnic ter auditorno identificira glas. Na najvišji varnostni stopnji mora sistem uspešno validirati vse tri dimenzije, kar bi bilo primerno npr. za dostop do visoko zaupnih prostorov. Shema delovanja sistema je prikazana na Sliki 1. Sistem omogoča avtentikacijo na podlagi biometričnih podatkov ali pa identifikacijo uporabnika glede na vneseno avtorizacijsko kodo na podlagi biometričnih podatkov.




**Slika 1: Shema delovanja sistema BioID [6]**

Z večjo zaščito pride tudi do več legalnih dostopov, ki so identificirani kot napačni. Zato za boljše prepoznavanje legalnih dostopov priporočajo uporabo dveh dimenzij, ki morajo biti validirane uspešno, se pa pri tem pojavi nekoliko večja možnost dejanskega nelegalnega dostopa. Sistem BioID je primeren za vse aplikacije, pri katerih uporabniki potrebujejo dostop do tehničnega sistema.

# 3. VARNOST NA RAVNI STROJNE OPREME

Za bolj zanesljivo varnost sistema je potrebno uporabiti dva kosa strojne opreme, ki eden brez drugega ne moreta delovati [14]. Podobno kot ločimo ključavnico in ključ. Podoben način se uporablja pri avtentikaciji nekatere programske opreme s pomočjo USB ključev. Takšna zasnova se je že izkazala kot zelo uporabna. V podpoglavjih sta opisana dva primera takšnega sistema.

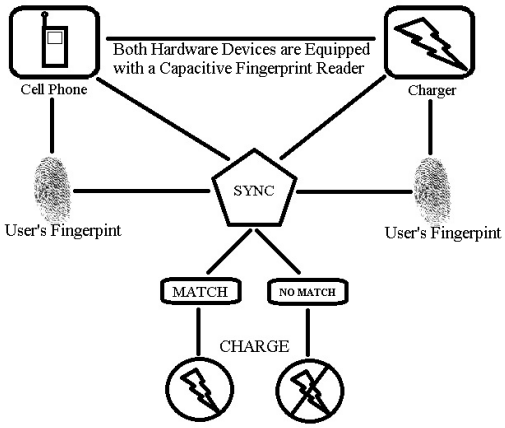
## 3.1 Unikatni polnilci za mobilne naprave

Med uporabniki se pogosto poraja vprašanje [14] zakaj so biometrični sistemi sploh boljši od PIN številk ali gesel? Glede na anketo v članku [20] zgolj 18% anketiranih uporablja PIN številko ali geslo za zaklep mobilnega telefona. Večina uporabnikov sploh ne uporablja zaščite za svoje osebne podatke na mobilni napravi. Že gesla oz. PIN številke so se celo izkazale kot zelo slabe rešitve zaradi uporabnikovega vnosa [19]. Uporabniki večinoma izbirajo zelo preprosta gesla ali pa si jih celo zapisujejo.

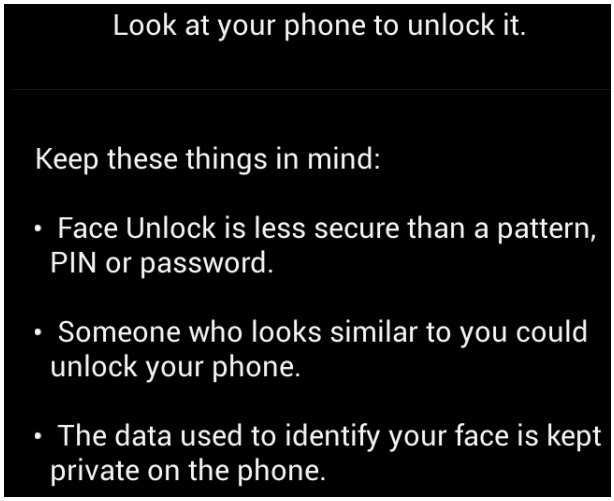
Na sliki 2 je prikazan varnostni sistem, ki bi lahko bil rešitev proti tatvinam mobilnih telefonov [14]. Tukaj polnilec in mobilni telefon vsebujeta čitalnik prstnih odtisov. Za polnjenje mobitela je pri tem sistemu treba imeti unikaten polnilec in pravi prstni odtis, ki se sprogramira pri nakupu telefona. Na novo lahko polnilec sprogramira le proizvajalec. Uporabnikov prstni odtis je torej ključ, ki omogoča delovanje celotnega sistema (v tem primeru polnjenja telefona).

Sistem vsebuje še dodatne zahteve za boljšo varnost [14]. Kot prvo bi mobilni telefon moral imeti vgrajeno baterijo, ki je ne bi bilo mogoče odstraniti. Ob odtujitvi telefona od lastnika oz. pripadajočega polnilca le ta postane neuporaben. Poleg tega bi bilo potrebno kriptirati datoteko s prstnim odtisom in onemogočiti dostop do nje, vsaj dokler telefon ni odklenjen.

V članku [14] je poleg zgoraj naštetega omenjena še ena ključna izboljšava. Avtorji predlagajo spremembo funkcije




**Slika 2: Arhitektura biometričnega telefona in polnilca opisanega v članku [14]**




**Slika 3: Vmesnik prepoznave obraza**

gumba za izklop. Gumb za izklop naj ne bi izklopil naprave brez avtorizacije. Na tak način tatovi ob kraji telefona ne bi mogli izklopiti mobilnega telefona, lastnik telefona pa bi lahko na daljavo izsledil bi lahko na daljavo s pomočjo aplikacije (npr. Lookout) izbrisal osebne podatke še preden bi lahko tatovi prišli do njih.

Očitno je nemogoče zagotoviti 100% varnost osebnih podatkov na mobilnih napravah, zato lahko s pomočjo biometričnih sistemov zgolj pridobimo čas za izbris podatkov na napravi na daljavo. S pomočjo unikatnih polnilcev pa lahko preprečimo uporabo ukradenih naprav, kar bi posledično odvračalo od samih kraj.

## 3.2 Uporaba tehnologije RFID

Ena od potencialno uporabnih tehnologij za znižanje števila kraj mobilnih telefonov je tudi RFID (Radio Frequency Identification) [7, 19]. To je v zadnjem času zloglasna tehnologija, ki se uporablja od identifikacij oseb s pomočjo kartic v stavbah in avtomobilih do sledenja izdelkov velikih trgovskih verig in sledenja živalim [7].

Vsak čip RFID vsebuje vsaj 96 bitov dolgo število, ki je unikatno na globalnem nivoju. Tako lahko za vsak čip točno definiramo na katerem izdelku se nahaja.

Eden od vrst tipov čipa RFID je nizkofrekvenčni, ki deluje na elektromagnetnem spektru od 125 do 134,2 KHz. Ta vrsta čipa deluje na razdalji nekaj centimetrov. V mobilni industriji bi bila uporaba čipa smiselna npr. pri unikatnih polnilcih. Mobilni telefon bi imel vgrajen bralec čipov RFID in bi se lahko polnil le na določenih polnilcih, ki bi imeli vgrajen čip RFID. Tako bi, poleg prstnih odtisov, uporaba te tehnologije omogočala večjo varnost s stališča tega, da se ukraden telefon ne bi več mogel napolniti.

Še ena vrsta čipa RFID deluje v frekvencah ranga več sto MHz. Te delujejo na razdalji 10m in več praznega prostora in nekoliko manj, ko je čip pritrjen na vsakdanje objekte. Ena od uporab bi lahko bila, da je mobilni telefon odklenjen le, ko je v okolici takega čipa. Vendar se lahko pojavi težava pri tem, kje se ta čip nahaja. Če je čip stacionaren, npr. v domu lastnika, lahko telefon uporablja le tam, kar se ne sklada z definicijo mobilnega telefona. Če je čip pritrjen na drug objekt, ki ga lastnik vedno nosi seboj, npr. ključe, pa se lahko pojavi težava v primeru izgube tega objekta. Še ena možnost bi bila, da je čip vgrajen neposredno v človekovo tkivo, vendar mislimo, da ta način morda ne bi bil moralno in etično primeren.

Pri tem načinu uporabe čipa RFID, kjer ga ima oseba vse čas na sebi, bi se prav tako pojavilo vprašanje zasebnosti. Razlog, da so čipi RFID v medijih tako zloglasni, je omogočanje sledenja osebam in s tem velik vdor v zasebnost. Zaradi tega razloga bi priporočali le uporabo nizkofrekvenčnega čipa RFID za identifikacijo polnilcev, podobno kot v prejšnjem primeru.

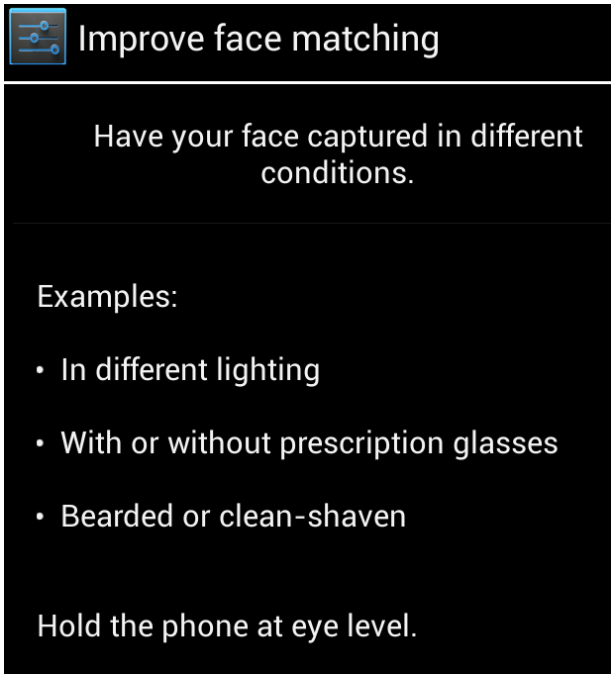
# 4. ANDROID PREPOZNAVA OBRAZOV

V sklopu tega dela smo preizkušali Android OS prepoznavo obrazov, ki omogoča odklepanje telefona brez uporabe PIN številke, gesla ali vzorca. Uporabili smo Samsung Galaxy Nexus z Android OS verzijo 4.2.2[4]. Uporabnik mora najprej določiti vzorec, geslo ali PIN številko, ki se uporablja v primerih, ko nas naprava ne prepozna (ta je sicer po navedbah razvijalcev tudi bolj varna kot prepoznava obrazov). Postopek je prikazan na Sliki 3.

Po začetnih nastavitvah smo v sistem nato vstopali s pomočjo prepoznavanja obrazov. Sistem deluje relativno zanesljivo in je uporaben v omejenih okoliščinah. Prepoznava je odvisna predvsem od zajetja obraza, pozicioniranja telefona in osvetlitve. V primeru slabih zaznav sistem omogoča nadaljnje učenje, kjer se sistem priuči na različna ozadja in svetlobne pogoje ter izgleda uporabnika. Izboljšanje prepoznavanja je prikazano na Sliki 4.

V drugem delu pa smo v sistem poizkušali vdreti kot neavtorizirane osebe in pod različnimi svetlobnimi pogoji. Sistem je dobro deloval v normalnih pogojih pri poizkusu vstopa

[4]Točna verzija: aokp_maguro_jb-mr1_milestone-1



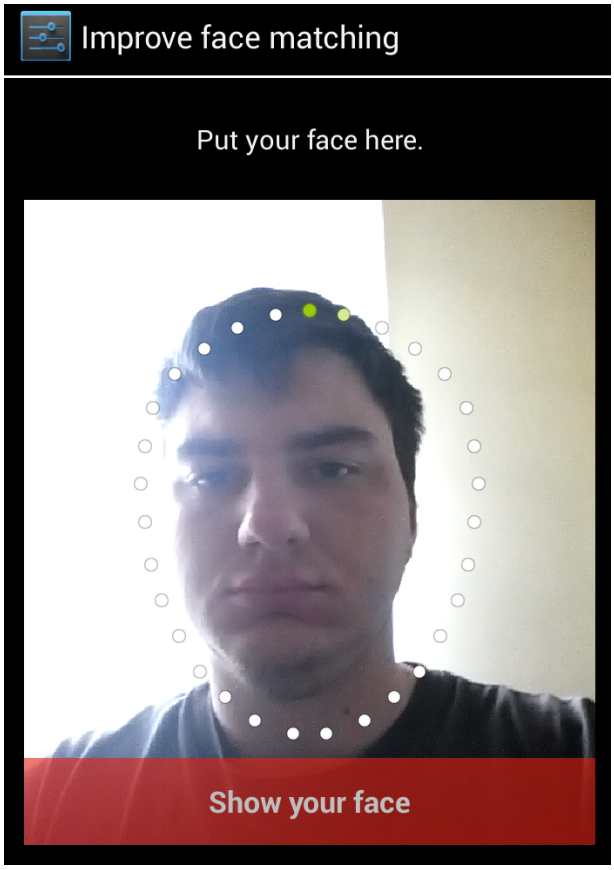

**Slika 4: Izboljšava prepoznave obraza**

druge osebe. Zahtevki druge osebe za prijavo v sistem so bili zavrnjeni. Sistem je dovolil vstop drugi osebi šele po tem, ko smo uporabili ekstremno močno osvetlitev, ki je močno omehčala poteze na sliki. V tem primeru je lahko druga oseba prišla do osebnih podatkov na mobilnem telefonu.

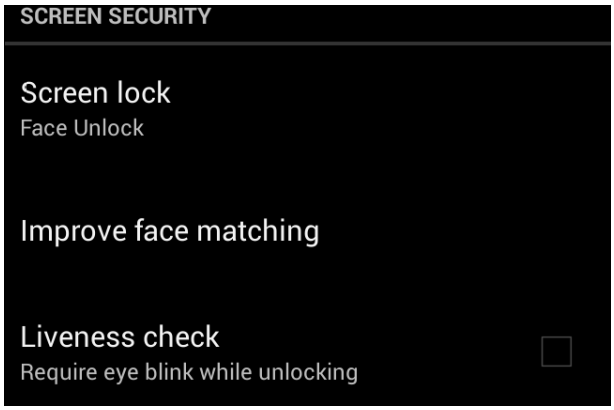
V sistem smo se uspeli prijaviti tudi z sliko uporabnika, ki jo lahko brez večjih težav pridobimo iz socialnih omrežij. Ta problem smo opisali že v enem izmed prejšnjih poglavij. Očitno pa se ta problem pojavlja tudi pri Anroid OS prepoznavi obrazov. Natisnili smo več slik uporabnika in se nato z njimi prijaviti v sistem. Biometrični sistem nas je v avtoriziral z praktično enako zanesljivostjo kot dejanskega uporabnika. Ob poznavanju osnov prepoznav obrazov (iskanje robov) smo se odločili uporabiti tudi črno-belo sliko. Tudi s tako sliko, je sistem prepoznal uporabnika in ga prijavil.

Sistem prepoznave obrazov omogoča, da uporabnik med prepoznavo mežika, kar je prikazano na Sliki 6. Z vklopljenim dodatnim varovanjem nam s pomočjo slike ni več uspelo vstopiti v sistem. Tak način prepoznave je veliko bolj varen, saj ne omogoča preproste neavtorizirane prijave s sliko. Po drugi strani pa je tak sistem za prijavo za uporabnika zaradi več napačno negativnih odločitev manj prijazen.

Mnenja smo, da je tak sistem lahko alternativa identifikaciji ali pa dopolnilo kakšni drugi obliki avtentikacije, saj so trenutne implementacije še vedno nezanesljive. Potrebno pa se je pa zavedati, da je takšna oblika avtentikacije za uporabnika veliko zahtevnejša in zamudna. Naši predlogi za izboljšave za potrebe forenzične analize so shranjevanje slike ob poizkusu odklepanja in shranjevanje metapodatkov. V primeru shranjevanja teh podatkov na neki oddaljeni lokaciji (npr. na strežnik) so možnosti odkritja zlorabe sistema




**Slika 5: Ekstremni svetlobni pogoji**




**Slika 6: Možnosti prepoznave**

dosti večje oz. je izpeljava forenzične analize veliko hitrejša.

# 5. BRALNIKI PRSTNIH ODTISOV

Uporaba bralnikov prstnih odtisov in možna uporaba v mobilnih napravah je podrobno opisana v članku [14]. Raziskovalci so predstavili alternativno možnost avtentikacije uporabnika in testirali varnost bralnikov prstnih odtisov. Avtorji so v prispevku predlagali arhitekturo, ki bi preprečevala kraje mobilnih naprav s pomočjo bralnikov prstnih odtisov. Predlagana rešitev vsebuje vgrajen bralnik na mobilni napravi in polnilcu baterij, med katerima se ob priklopu preveri identifikacija in avtentikacija uporabnika. S tem nepooblaščeni uporabniki izgubijo večji del motivacije za krajo. Prenosni telefon zaradi povezanosti z enim uporabnikom postane neuporaben, zaradi nezmožnosti polnjenja pa se skrajša tudi časovni okvir v katerem lahko uspešno vdrejo v sistem in pridobijo podatke.

## 5.1 Unikatnost prstnih oddtisov

Prstni odtisi so unikatni identifikator vsakega posameznika. Njihov namen je, da lažje primemo predmete zaradi trenja, ki se ustvari med prsti in predmetom. Prstni odtisi se razvijejo v maternici, na njihovo formacijo pa vplivajo naključni dotiki prstov maternične stene med nosečnostjo. Zaradi tega razloga prstni odtisi postanejo zelo različni drug od drugega.

Prstni odtisi ostanejo nespremenjeni od rojstva do smrti ter predstavljajo skoraj idealen način identifikacije oseb. Spremenijo se lahko le v primeru poškodbe, bolezni ali dekompozicije. Prepoznava se deli na tri nivoje podrobnosti. Na prvem nivoju poteka klasifikacija robov, na drugem nivoju sekanje robov in pik, na tretjem pa se pregleduje obliko in širino robov ter por. Za prepoznavo je treba preiskati minimalno drugi nivo, saj prvi ne nudi dovolj podatkov za enolično identifikacijo oseb [14].

Moderni bralniki so večinoma dveh vrst: optični in kapacitivni, ki z različnimi načini določata poteze odtisa. Optični so praviloma manj varni, saj jih je veliko lažje zaobiti s pomočjo detaljne fotografije. Kapacitivni pa zavzamejo 3D obliko odtisa, ki je težja za ponarejanje.

## 5.2 Ponarejanje prstnih odtisov

V večini znanstvenih študij, kjer so uspešno prešli zaščito prstnih odtisov, so uporabili živi prst, da so naredili 3D model prsta. V realnem življenju je to precej težko in bi se zgodilo le pri redkih žrtvah. Vseeno pa so v [14] raziskovalci ponaredbe prstnega odtisa odkrivali skozi dve tematiki: z voljo in brez volje žrtve. Prstni odtis posameznika je možno brez volje posameznika pridobiti iz mobilnega telefona ali katerega drugega predmeta (v članku [14] kozarca).

Z voljo žrtve so raziskovalci v [14] naredili naslednje eksperimente. Žrtev je enega od prstov pritisnila na blazinico s črnilom in ga nato odtisnila v rokavico. Rokavico so nato obrnili naokoli in jo uporabili na čitalniku prstnih odtisov. Poskus nelegalnega dostopa ni uspel. Nato so prstni odtis, ki je bil pritisnjen na steklo, prekrili z magnetičnim pudrom za prstne odtise in s pomočjo snovi Mikrosil pridobili gumijasti odtis prstnega odtisa. Preslepitev čitalnika prstnih odtisov na ta način ni uspela. Nato poskušali replicirati metodo iz ene od prejšnjih študij in so na prstni odtis nanesli zmes, sestavljeno iz polovice želatina in polovice vroče vode. Ko se je zmes strdila, so odtis preizkusili na bralcu. Ugotovili so, da je zaradi vlage bralec zaznal, da je odtis "živ", ni pa zaznal 3D grebenov prstnega odtisa. Zaradi tega so ponovno uporabili prejšnji model iz Mikrosila in ga navlažili z plastjo vode. Po tem je bralec prstnih odtisov odtis prepoznal skoraj v celoti. Toda ker je bil odtis na Mikrosilu ravno invertiran, ga bralec ni prepoznal kot pozitivnega. Zato so pri naslednji metodi prstni odtis, ki je bil skeniran na računalnik, natisnili na list prosojnega papirja. Nato so list postavili na platirano bakreno ploščo in nanjo prenesli sliko prstnega odtisa s pomočjo električnega železa. Po nekaj minutah so ploščo postavili na plastični pladenj z jedko raztopino s sliko odtisa obrnjeno navzdol. Ko se je baker odstranil, so večkrat poskusili narediti 3D prstni odtis. Rezultati so bili zaradi slabe zmesi bakra nedoločeni.

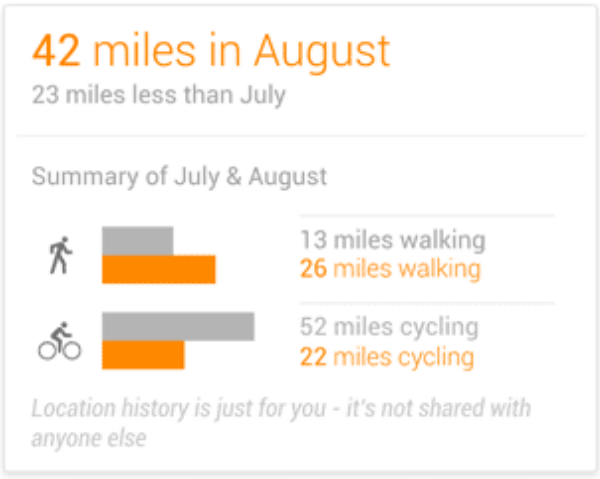
# 6. IDENTIFIKACIJA S POMOČJO UMETNE INTELIGENCE

Z uporabo mobilne naprave in integracijo večih sistemov je možno učinkovito podajanje informacij uporabniku in z določenimi značilkami tudi opisati posameznega uporabnika. V zadnjem času vse bolj popularni Google Now[5] nam brez našega eksplicitnega povpraševanja podaja relevantne informacije o našem okolju. Rezultati so vzpodbudni, saj mobilna naprava brez našega posega glede na statistiko zgodovine in predvidevanja prikazuje relevantne kraje, novice in osebna obvestila. Glede na način uporabe in naših navad je tako možno z relativno primitivnimi senzorji zbrati veliko količino relevantnih podatkov.

Metode, ki bi uporabnika identificirale na tak način, so ob idealni implementaciji za uporabnika povsem nemoteče, zaradi količine identificiranih podatkov pa postanejo izredno varne [20]. Poleg fizičnih vsak posameznik poseduje veliko količino psiholoških lastnosti, ki so težje ponovljive. Mobilna naprava lahko v statistiko vzame tudi pogostost pogleda na zaslon ali majhne, a značilne premike rok ob določenih akcijah. V članku [4] opisana metoda opisovanja hoje uporabnika, ki je povsem neopazna, zagotavlja pa stopnjo zanesljivosti, ki je primerljiva z drugimi biometričnimi sistemi. Na Sliki 7 je vidno spremljanje uporabnikovega premikanja, ki ga že izvaja Google Now. Predvidevamo, da je s pomočjo nizko energijsko potratnih senzorjev v novejših telefonih nadaljnja identifikacija uporabnika smiselna in enostavna za implementacijo.

Mnenja smo, da je identifikacija s pomočjo umetne inteligence izredno uporabna kot možen način avtentikacije uporabnika. Zaradi specifičnosti področja, količine informacij in njihovega hranjenja je potrebno veliko strokovnega znanja. Večje korporacije, ki zbirajo take podatke, so zaradi varovanja osebnih podatkov in poslovnih skrivnosti težko dostopna širši javnosti. Forenzična vrednost teh podatkov je velika, saj novejše mobilne naprave vsebujejo veliko senzorjev, ki lahko pripomorejo k identifikaciji uporabnika (GPS, kamere, giroskopi, senzor srčnega utripa, bralnik prstnih odtisov, ...). Za uspešno dedukcijo dokazov iz podatkov je potrebno sodelovanje z hraniteljem podatkov ali pa vzpostavitev standardnega načina hranjenja takih podatkov.

[5]https://www.google.com/landing/now/ - april 2014




**Slika 7: Google Now spremljanje aktivnosti [1]**

# 7. FORENZIČNA PREISKAVA MOBILNE NAPRAVE

Kljub porastu prodaje mobilnih naprav, katerih se trenutno proda petkrat toliko kot računalnikov, je forenzična znanost še vedno v zaostanku [2]. V nadaljevanju je predstavljeno pravilno ravnanje s telefonom kot forenzičnim dokazom, povzeto po [2].

## 7.1 Primerjava z računalniško forenziko

V tem poglavju so predstavljeni naslednji pogledi reprodukcija dokazov, povezovalne zmožnosti in vpliv na analizo, operacijski in datotečni sistem, strojna oprema in uporabljena forenzična orodja.

### *7.1.1 Reprodukcija dokazov*

Naprave so ob forenzični analizi lahko izklopljene ali vklopljene. V primeru izklopljenega računalnika se ob forenzični analizi naredi bitna kopija diska, s katero v nadaljevanju izvajamo preiskavo. Ta slika se čez čas ne spreminja, njeno integriteto pa preverjamo z različnimi zgoščevalnimi funkcijami. Mobilni telefoni zaradi različne tehnologije ob urinem taktu zapisujejo podatke na telefon. Bitna kopija pomnilnika telefona tako ob različnih časih ne poda iste zgoščene vrednosti in slike pomnilnika [10].

### *7.1.2 Povezovalne zmožnosti in vpliv na analizo*

V primerjavi z računalniki mobilne naprave vsebujejo veliko več različnih načinov komunikacije (Bluetooth, GSM, IR, WI-FI, ...). V primeru prižganega mobilnega telefona je forenzična preiskava močno otežena, saj ne obstaja standardiziran postopek ohranjanja informacij ob vseh različnih konfiguracijah mobilnih naprav, ki bi zagotavljal ohranjanje podatkov in ne bi sprožil varnostnega zaklepa naprave [3].

### *7.1.3 Operacijski in datotečni sistem*

V računalnikih poznamo in uporabljamo majhno število operacijskih in datotečnih sistemov. Zaradi naglega razvoja in različnih tehnologij pa se v mobilnih napravah pojavlja več različic te programske opreme. Redki operacijski sistemi so odprti (npr. Maemo), drugi so polodprti (npr. Android), tretji pa so popolnoma zaprti (npr. iOS). Proizvajalci naprav zaradi strahu pred odhajanjem informacij ne izdajajo dokumentacij svojih sistemov, s tem pa močno otežijo forenzično preiskavo.

Ob hitrem razvoju mobilnih sistemov forenzična orodja le redko dohajajo razvoj. Nove verzije operacijskih sistemov so predstavljene letno, forenzična orodja pa v tem času še ne dozorijo za uporabo [2].

### *7.1.4 Strojna oprema*

Podobno kot programsko opremo proizvajalci močno ščitijo tudi njihovo strojno opremo. Njihova zaščitena strojna oprema ni izdelana z mislijo na forenzično preiskavo. Velik spekter naprav nam praktično onemogoči komunikacijo z napravo na najnižjem nivoju (različni priključki na PCB-jih), zato se zanašamo na povezavo preko USB vmesnika. Moderni mobilni telefoni nam ponujajo široko možnost uporabe fizične povezave (npr. terminal, izmenjava zaslonov, ...).

### *7.1.5 Uporabljena forenzična orodja*

V preteklosti so se forenziki zaradi omejenih zmožnosti odkrivanja podatkov na mobilnih napravah zanašali predvsem na ponudnike storitev. Z napredkom se meja med računalniško in mobilno forenziko zmanjšuje. Mobilne naprave uporabljajo splošne datotečne sisteme (npr. Android - EXT, FAT..), temeljijo na poznanih tehnologijah (npr. jedro Linux in knjižnice). Razširjene zmožnosti naprav pa nam omogočajo praktično enako globino vpogleda v napravo, kot je to možno na računalniških sistemih.

## 7.2 Mobilna naprava kot dokaz

Digitalni dokazi na mobilnih napravah so še posebej občutljivi, zaradi možnosti oddaljenega dostopa ali brisanja podatkov na napravi. Pri obravnavi se je potrebno držati naslednjih načel [2]:

- Naše akcije ne smejo spremeniti podatkov, na katere se sklicujemo v obravnavi
- V primeru neizogibnega dostopa do originalnih podatkov to počne usposobljena oseba z izkušnjami
- Ohraniti je potrebno verigo zaupanja
- Vodja preiskave osebno odgovarja za potek preiskave

Posebno pozornost je treba nameniti zajemu in hranjenju podatkov naprave, zaradi že omenjenih omejitev pri preiskavi mobilnih naprav.

- Vklopljene naprave ne izklapljamo in poskrbimo za ustrezno polnjenje (naprava se lahko zaklene)
- Zapišemo podatke na zaslonu (še boljše slikamo ali posnamemo)
- Če je naprava izklopljena, je ne vklapljamo
- Ob zaplembi naprave kontaktiramo upravitelja omrežja, napravo pa odstranimo iz vseh možnih komunikacijskih omrežij (možnost oddaljenega dostopa)

- Pred preiskavo poiščemo dokumentacijo o napravi

V varnem okolju, ki je izolirano od zunanjih signalov, glede na zmožnost upravljanja s telefonom izvajamo nadaljnje preiskovanje. Med preiskavo pomnilnika preiščemo celoten datotečni sistem. Mobilne naprave hranijo veliko količino zgodovinskih podatkov (klici, SMS-i, zgodovina brskanja, WI-FI dostopi, ...). Ti podatki so lahko tudi v neznanih formatih ali podatkovnih bazah. V primeru, da ne moramo dostopati do pomnilnika telefona in nam ostane le ročni pregled naprave, vsak korak beležimo in obrazložimo namen za njegovo izvedbo.

# 8. ZAKLJUČEK

Tehnološki napredek zadnjih nekaj let je prinesel večjo mobilnost elektronskih naprav. Uporabniki imajo sedaj ogromno osebnih podatkov shranjenih kar na svojih mobilnih napravah, sama vrednost le teh pa se je tudi zelo močno povečala. To sta glavna razloga za veliko število tatvin mobilnih naprav in izrabo podatkov na njih. Vzpostavitev biometričnega sistema na mobilni napravi, ki ne uporablja zgolj PIN številke, gesla ali pa vzorca, je nujna za preprečevanje izrab osebnih podatkov, če že ne odvračanja od kraj.

Iz dosedanjih raziskav je razvidno, da so biometrični sistemi zelo popularna in uporabna rešitev za avtentikacijo uporabnika. Po drugi strani pa so razvidne izjemne pomanjkljivosti posameznih biometričnih sistemov, ki pa se še odpravljajo na tem zelo razvijajočem se področju. Repliciranje prstnih odtisov, glasu in obraza je postalo z novejšimi sistemi težavnejše, vendar ne nemogoče. Kljub konstantnemu izboljševanju posameznih biometričnih sistemov se pojavljajo vedno novi načini za nedovoljene dostope in uporabo.

Rešitev na pomanjkljivosti biometričnih sistemov pa leži v kombiniranju le teh v en zanesljiv sistem vključno z strojnimi rešitvami in uporabo umetne inteligence za pametno izrabo biometričnih in ostalih podatkov iz okolice. Pridobitev večih biometričnih podatkov s strani potencialnega tatu je zelo težavno. Poleg težavnosti pridobitve pa lastniku naprave močno podaljša čas za oddaljeni dostop do mobilne naprave in izbris osebnih podatkov z nje. Na drugi strani pa strojne rešitve fizično onemogočajo celostno uporabo mobilnega telefona brez vseh ujemajočih delov sistema. Zanesljiva in učinkovita biometrična zaščita je možna še z veliko drugimi izboljšavami, vendar pa je vsem skupno, da od uporabnika terjajo malce več napora kot pa ostale preprostejše in lahko zlomljive metode.

# 9. LITERATURA


[1] Google now - summary of activities. Dostopno na: https://www.google.com/landing/now/images/gnow-card-activity-summary.png. Accessed: 2014-05-03.

[2] R. Ahmed and R. V. Dharaskar. Mobile forensics: an overview, tools, future trends and challenges from law enforcement perspective. In *6th International Conference on E-Governance, ICEG, Emerging Technologies in E-Government, M-Government*, pages 312–23, 2008.

[3] B. D. Carrier. Risks of live digital forensic analysis. *Communications of the ACM*, 49(2):56–61, 2006.

[4] J.-J. L. Choon-Young Lee. Estimation of walking behavior using accelerometers in gait rehabilitation. 2002.

[5] M. O. Derawi, C. Nickel, P. Bours, and C. Busch. Unobtrusive user-authentication on mobile phones using biometric gait recognition. In *Proceedings of the 2010 Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, IIH-MSP '10, pages 306–311, Washington, DC, USA, 2010. IEEE Computer Society.

[6] R. W. Frischholz and U. Dieckmann. Bioid: A multimodal biometric identification system. *Computer*, 33(2):64–68, Feb. 2000.

[7] S. L. Garfinkel, A. Juels, and R. Pappu. Rfid privacy: An overview of problems and proposed solutions. *IEEE Security and Privacy*, 3(3):34–43, May 2005.

[8] R. Hulsebosch and P. Ebben. Enhancing face recognition with location information. *Electronics and Communication Engineering Journal*, pages 397–403, 2013.

[9] Y. Ijiri, M. Sakuragi, and S. Lao. Security management for mobile devices by face recognition. page 49. IEEE Computer Society, 2006.

[10] W. Jansen and R. Ayers. Guidelines on cell phone forensics. *NIST Special Publication*, 800:101, 2007.

[11] S. Kopsidas, D. Zisiadis, , and L. Tassiulas. Voice interactive personalized security (voipsec) protocol: Fortify internet telephony by providing end-to-end security through inbound key exchange and biometric verification. *Electronics and Communication Engineering Journal*, pages 1–10, Nov. 2006.

[12] S. Liu. Anti-counterfeit system based on mobile phone qr code and fingerprint. In *Proceedings of the 2010 Second International Conference on Intelligent Human-Machine Systems and Cybernetics - Volume 02*, IHMSC '10, pages 236–240, Washington, DC, USA, 2010. IEEE Computer Society.

[13] H. Manabe, Y. Yamakawa, T. Sasamoto, and R. Sasaki. Security evaluation of biometrics authentications for cellular phones. In *Proceedings of the 2009 Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, IIH-MSP '09, pages 34–39, Washington, DC, USA, 2009. IEEE Computer Society.

[14] D. J. Ohana, L. Phillips, and L. Chen. Preventing cell phone intrusion and theft using biometrics. *2013 IEEE Symposium on Security and Privacy Workshops*, 0:173–180, 2013.

[15] A. Pocovnicu. Biometric security for cell phones. *Informatica Economică*, 13, 2009.

[16] R. Ribalda, G. Glez de Rivera, A. de Castro, and J. Garrido. A mobile biometric system-on-token system for signing digital transactions. *IEEE Security and Privacy*, 8(2):13–19, Mar. 2010.

[17] H. Shabeer and P. Suganthi. Mobile phones security using biometrics. In *Conference on Computational Intelligence and Multimedia Applications, 2007. International Conference on*, volume 4, pages 270–274, Dec 2007.

[18] H. A. Shabeer and P. Suganthi. Mobile phones security using biometrics. In *Proceedings of the International Conference on Computational*

*Intelligence and Multimedia Applications (ICCIMA 2007) - Volume 04*, ICCIMA '07, pages 270–274, Washington, DC, USA, 2007. IEEE Computer Society.

[19] E. Syta, S. Kurkovsky, and B. Casano. Rfid-based authentication middleware for mobile devices. In *Proceedings of the 2010 43rd Hawaii International Conference on System Sciences*, HICSS '10, pages 1–10, Washington, DC, USA, 2010. IEEE Computer Society.

[20] M. Tamviruzzaman, S. I. Ahamed, C. S. Hasan, and C. O'brien. epet: When cellular phone learns to recognize its owner. In *Proceedings of the 2Nd ACM Workshop on Assurable and Usable Security Configuration*, SafeConfig '09, pages 13–18, New York, NY, USA, 2009. ACM.

[21] P. Tresadern, C. McCool, N. Poh, P. Matejka, A. Hadid, C. Levy, T. Cootes, and S. Marcel. Mobile biometrics (mobio): Joint face and voice verification for a mobile platform. *IEEE Pervasive Computing*, 99, 2012.

# Ponarejanje sporočil SMS

Saša Makorič
Fakulteta za računalništvo in informatiko
Univerza v Ljubljani
saso.makoric@gmail.com

Sandi Šemrov
Fakulteta za računalništvo in informatiko
Univerza v Ljubljani
semrola07@gmail.com

## POVZETEK

TODO:

- neki whitespaci so k se jih ne da odstranit
- še 2 strani :p

Preiskovalcem, ki so zadolženi za pregled mobilnih naprav, se pogosto postavlja vprašanje, če je možno na kakšen način ugotoviti, ali so sporočila SMS na sami napravi ponarejena. Narejena je bila raziskava, če je možno ponarediti sporočilo SMS brez dostopa do privilegirane programske oz. strojne opreme. Ugotovljeno je bilo, da je z uporabo prosto dostopnih orodij možno ponarediti vsebino in pošiljatelja sporočila SMS. Raziskani so bili tudi pristopi, kako ugotoviti, da gre za ponarejeno sporočilo.

## Ključne besede

preiskava mobilnih naprav, ponarejanje sporočil, sporočila SMS, Android, Nokia 6021

## 1. UVOD

Preiskovalci oz. forenziki so pogostokrat soočeni z različnimi vprašanji, če je možno ponarediti (spremeniti) sporočilo SMS. Prvo možno vprašanje je, če je programska oprema, ki je pregledala mobilno napravo, generirala pomanjkljivo poročilo in so tako bila pridobljena sporočila drugačna. Drugo vprašanje je, če se je poročilo spreminjalo po sami preiskavi in pregledu. Tretje vprašanje pa, če je možna ponaredba (sprememba) sporočila SMS na sami mobilni napravi.

Za odgovor na tretje vprašanje se je v zadevo potrebno poglobiti. Pri kopiranju pomnilnika mobilnih naprav na disk (ang. HEX dump) z različnimi orodji, je bilo posumljeno, da se ta orodja lahko izkoristijo za ponarejanje sporočil SMS, ki so shranjena na mobilni napravi. Izvedena je bila raziskava, v kateri je bilo potrebno najti orodja, ki bodo zmožna ponarediti sporočilo SMS in koliko znanja bo taka operacija zahtevala.

## 2. TERMINOLOGIJA

- ang. Commercial Off-The-Shelf (COTS) - Strojna oz. programska oprema, ki je na voljo širši množici ljudi in ni spremenjena, da bi delovala zunaj svojih standardnih parametrov.
- tabele PM - Logična struktura, ki se uporablja za shranjevanje uporabniških in sistemskih podatkov v mnogih napravah znamke Nokia. Tabela sestoji iz ključev in podključev. Ključi opredeljujejo kategorijo (kot so sporočila SMS, podatki kartice SIM ...), podključi pa hranijo konkretne podatke, ki spadajo v to kategorijo (npr. posamezna sporočila SMS, številka IMSI zadnje uporabljene kartice SIM ...).
- Testiranje znotraj mej - Testi, ki se uporabljajo za evaluacijo odziva sistema, kateremu podamo podatke, ki so (za sistem) v mejah pričakovanega (npr. za vnos časa vnašamo take podatke, kot jih pričakuje sistem).
- Testiranje zunaj mej - Testi, ki se uporabljajo za evaluacijo odziva sistema, kateremu podamo podatke, ki so za sistem nepričakovani (npr. za vnos časa uporabimo sistemu nepričakovano število sekund in opazujemo odziv sistema).
- ang. HEX Dump - Način preiskave mobilne naprave, s katerim lahko pridobimo izbrisane oz. skrite podatke.
- ukazi AT - (ang. AT Commands) množica ukazov, ki skrbi za upravljanje s kompatibilnimi napravami

## 3. PRISTOP IN OPREMA

Vsi eksperimenti so bili narejeni s programsko in strojno opremo, ki je na voljo vsem in za njo ne potrebujemo (veliko) denarja (ang. COTS – Commercial Off-The-Shelf hardware/software). Za vse eksperimente je bila uporabljena mobilna naprava znamke Nokia, model 6021. Prvi in glavni razlog za to je, da mobilni telefoni znamke Nokia uporabljajo standardiziran in dokumentiran način shranjevanja podatkov - tabele PM. Tabele PM se uporabljajo za shranjevanje uporabniških in sistemskih podatkov. Tabela je sestavljena iz ključev in podključev. Ključ predstavlja kategorijo podatkov (npr. sporočila SMS), podključ pa relevantne podatke (npr. eno sporočilo SMS). Drugi razlog je, da še vedno veliko mobilnih naprav znamke Nokia uporablja tabele PM (razen novejših pametnih telefonov, ki uporabljajo operacijski sistem Windows Phone). Popularen je tudi operacijski sistem Android, ki ga uporablja veliko pametnih telefonov, zato

sva preverila še, kakšne aplikacije obstajajo za ponarejanje sporočil SMS za pametne telefone z Androidom.

Uporabljena strojna oprema za preiskovanje mobilne naprave Nokia 6021:

- Advance Turbo Flasher by AdvanceTeam [2]



**Slika 1: Pripomoček Advance Turbo Flasher.**

- CYCLONE-Box [3]



**Slika 2: Pripomoček Cyclone Box.**

Uporabljena programska oprema:

- PSPad urejevalnik besedil
- UFSxHWK

# 4. STRUKTURA IN HRANJENJE SPOROČIL SMS NA MOBILNI NAPRAVI NOKIA 6021

Nokia 6021 shranjuje sporočila SMS pod ključem številka 140 (v tabelah PM). Potrebno je bilo pregledati veliko sporočil, da smo ugotovili, kje se kaj nahaja (npr. kje se nahaja številka pošiljatelja, datum ipd.). Mobilna naprava Nokia 6021 ni shranjevala sporočil SMS po standardu GSM 3.40 [5] ampak so bili nekateri elementi sporočila zamenjani, nekateri pa dodani.

Enolični identifikator sporočila SMS se nahaja na šestem mestu in je dolg 2 bajta (Slika 3). Obstaja možnost, da identifikator zavzema še 2 zloga prej in je tako omogočeno imeti na mobilni napravi več kot 255 sporočil SMS (prej $2^8 = 255$, potem pa $2^{16} = 65536$). Struktura sporočila SMS vsebuje 2 polji za dolžino. Prvi je dolžina celotnega zapisa SMS (na mestu 14, morda tudi 2 bajta prej). Vrednost tega parametra je bila enaka številu bajtov od 12. bajta do konca zapisa. Drugi parameter je dolžina zakodiranega sporočila in se nahaja na mestu -6 - 6 bajtov pred koncem vsebine sporočila SMS. Vrednost tega parametra je bila enaka številu bajtov od mesta -8 do konca zapisa.

Datumi in časovni žigi so shranjeni v obratnem vrstnem redu. Prvi v mesecu ni shranjen kot 01, ampak kot 10. Format datuma je isti kot v GSM 3.40 specifikaciji [5] - YYMMDDhhmmss, kjer je Y leto (ang. year), M mesec (ang. month), D dan (ang. day), h ura (ang. hour), m minuta (ang. minute), s sekunda (ang. second). Številka pošiljatelja sporočila SMS in številka klicnega centra sta zapisani podobno kot v specifikaciji GSM 3.40 [5], kjer dolžini naslova sledi tip naslova (npr. International, National, Network ...), ki ga predstavlja določena številka in na koncu še številka sama.

Vsebina sporočila SMS je shranjena v formatu PDU. Pred vsebino se nahajajo še dolžina vsebine sporočila (User Data Length), dolžina zakodirane vsebine PDU in dolžina strukture sporočila SMS. Če je sporočilo SMS krajše od večkratnika štirih bajtov, je potrebno preostali prostor dopolniti z znaki 0x55, da dolžina postane večkratnik števila 4 (dolžina po modulu s 4 mora biti enaka 0). Na primer, beseda “Abcdef” - zakodirano v PDU je “41F1985C3603” - je predstavljena kot “41F1985C36035555”. Tako postane beseda dolga 8 bajtov, kar je dvakratnik štirih bajtov.

Sporočila SMS, ki so sestavljena iz več delov, so bila shranjena v več različnih podključih. Datumi in časovni žigi, ki so prikazani na mobilni napravi se navezujejo na zadnji del sporočila SMS (sestavljenega iz več delov).

# 5. TESTIRANJE IN REZULTATI

## 5.1 Nokia 6021

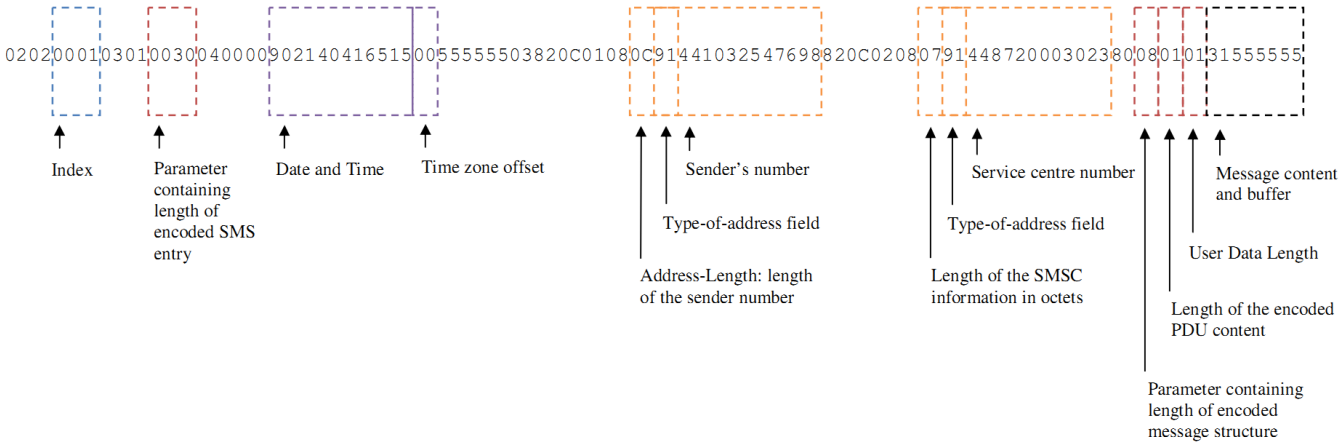
Kreiranje sporočila SMS s pomočjo COTS je rezultiralo v neodzivanju mobilne naprave. Problem je bil v enoličnih identifikatorjih, ki so shranjeni pod ključem 140 (in v podključih). Potrebna je bila nova namestitev stojne programske opreme (ang. firmware) različice v5.22 in vsi testi so se izvajali na tej različici opreme. Vsi nadaljnji eksperimenti so se izvajali samo na urejanju sporočil SMS. Za urejanje tabel PM smo uporabili PSPad različice 4.5.3, druga orodja pa so še UFS/HWK flasher/service box, ki uporablja DCTxBB5 version V2.0.8.0.

### *5.1.1 Testiranje znotraj mej*

Opazovali smo mobilno napravo, kako bo reagirala na spremembe podatkov, ki so veljavni. Spreminjali smo številko centra za sporočila SMS (ang. Short Message Service Center (SMSC) number), številko pošiljatelja (ang. sender's number) sporočila SMS, datume in časovne žige ter vsebino sporočila SMS.

Številko centra in številko pošiljatelja smo lahko zamenjali vedar je potrebno paziti, če se nova številka razlikuje od stare po dolžini znakov. V tem primeru moramo spremeniti tudi vrednost polja, ki vsebuje dolžino številke.

Datume in časovne žige smo tudi lahko urejali in vnašali katerekoli veljavne vrednosti. Zamenjali smo lahko tudi del časovnega žiga, ki predstavlja časovni pas. Toda ta sprememba je bila vidna le znotraj zakodirane različice PDU, ki jo lahko




**Slika 3: Struktura sporočila SMS.**

pridobimo s pomočjo ukazov AT (ang. AT Commands), kot tudi z orodjem MicroSystemation XRY, ki ustvari kopijo sporočila SMS.

Vsebino sporočila SMS smo lahko zamenjali, vendar je kot pri številkah centra in pošiljatelja, potrebno paziti na vrednost polj, ki vsebujejo dolžine sporočila SMS. Če se dolžina nove vsebine razlikuje od dolžine stare vsebine je potrebno popraviti tudi vrednost polj, ki vsebujejo dolžine. Prva dolžina je dolžina sporočila SMS, ki se nahaja v polju User Data Length. Drugi dve dolžini sta še dolžina zakodirane vsebine PDU in dolžina celotne strukture SMS (Slika 3). Potrebno je omeniti, da se v dolžino vključuje tudi tisti "prazen" prostor, ki smo ga morebiti napolnili z vrednostjo 0x55.

Vse spremembe podatkov, razen spremembe časovnega pasu, so bile vidne na mobilni napravi (pod prikazom podrobnosti sporočila SMS (izbira v podmeniju mobilne naprave)). Nič ni nakazovalo na to, da so bili podatki sporočila SMS spremenjeni. Tudi s pomočjo orodij za kopiranje sporočil (MicroSystemation XRY in Envisage Systems Phonebase), ni bilo videti, da so bila polja sporočila SMS spremenjena. Orodje XRY je prikazalo tudi spremenjen časovni pas, kot odmik od časovnega pasu GMT (Greenwich Mean Time).

### *5.1.2 Testiranje izven mej*

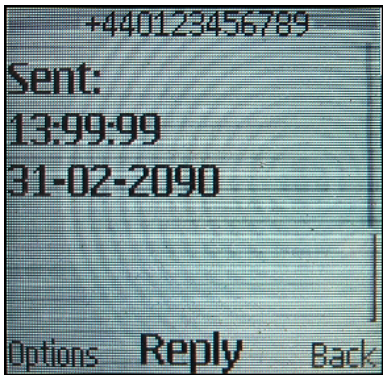
Napravo Nokia 6021 smo preizkusili tudi tako, da smo spremenili in vnesli podatke, ki niso bili veljavni. Vnešeni so bili neveljavni datumi in časovni žigi. S tem je bilo testirano, če naprava prepozna neveljavne podatke in kako na njih reagira.

Datumi in časovni žigi so v formatu, kot že omenjeno, po specifikaciji GSM 3.40 [5]. Ta format je oblike YYMMDDhhmmss. Vidimo, da leto zavzema 2 mesti. Števila, ki smo jih lahko vnesli na to mesto so od 00 do 99, kjer je naprava za vnos 00 pokazala leto 2000, za vnos 99 pa leto 1999. Pri kateri številki se stoletje zamenja, nismo testirali.

Polje za mesec je bilo omejeno na številke od 01 do 12, polje za dan na vrednosti od 01 do 31 in polje za uro na vrednosti od 00 do 23. Če so bile vrednosti nastavljene izven teh meja, mobilna naprava ni prikazala podrobnosti časovnih žigov in datuma sporočila SMS. Vseeno pa smo lahko dostopali do vsebine sporočila in drugih elementov, ki so v podmeniju sporočila.

Za minute in sekunde smo lahko vnesli veljavne (00-59) in tudi neveljavne (60-99) vrednosti. Te vrednosti je prikazala tudi mobilna naprava pod prikazom podrobnosti sporočila SMS. Tako je naprava lahko prikazala uro 23:99:99.

Za podatke, ki so bili vneseni izven mej, so različna orodja za kopiranje podatkov prikazala različne podatke, kot so bili prikazani na mobilni napravi. Sporočilo SMS, ki je imelo spremenjen datum na 31.02.90 13:99:99, se je pod podrobnostmi prikazal časovni žig 31.02.2090 13:99:99 (Slika 4). Z orodjem XRY smo dobili datum 31.02.1990 (UTC) brez časa. Orodje Phonebase je prikazalo časovni žig 03.03.1990 14:40:39. Orodje AT, ki se uporablja za pregled sporočila SMS v formatu PDU, je dekodiralo orginalen datum v isti obliki 31.02.90 13:99:99. Različna orodja različno interpretirajo neveljavne podatke in zato prihaja do razlik s tisto interpretacijo podatka, ki je v sami mobilni napravi [7].




**Slika 4: Zajet zaslon Nokie 6021 s podrobnostmi ponarejenega sporočila SMS z neveljavnimi podatki.**

# 6. PONAREJANJE SMS SPOROČIL NA NAPRAVAH Z OPERACIJSKIM SISTEMOM ANDROID

Ker članek pokriva ponarejanje SMS sporočil le na napravah znamke Nokia, natančneje le na napravah z operacijskih sistemom Symbian OS, sva se odločila na enak način kot v prvotnem članku [7] testirati ponarejevanje na napravah z operacijskim sistemom Android. Za slednjega sva se odločila, ker trenutno pokriva največji del tržišča - kar 80 % celotnega svetovnega trga [6].
Testiranje se je izvedlo na mobilnem telefonu Samsung Galaxy Ace 2, na katerem je nameščen operacijski sistem Android verzije 2.3.6 (Gingerbread). V naslednjem delu bodo pokrita naslednja poglavja:

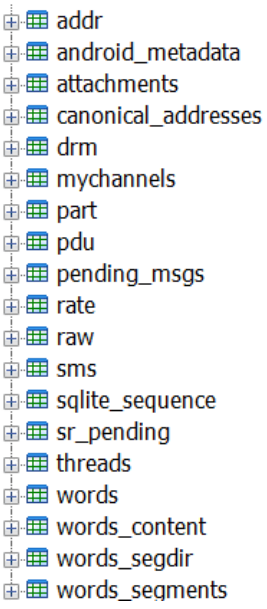
- Pregled strukture shranjevanja sporočil SMS.
- Ročno ponarejevanje sporočil SMS.
- Ponarejevanje z uporabo obstoječih aplikacij.
- Pregled ponarejenih sporočil SMS.

## 6.1 Struktura shranjevanja SMS sporočil

Sporočila se shranjujejo v podatkovni bazi *SQLite*. Sama podatkovna baza je shranjena na standardni lokaciji na napravi - */data/data/com.android.providers.telephony/mmssms.db*. V veliki večini operacijskih sistemov Android se lokacija ne spreminja.

Potrebno je omeniti, da sam dostop do baze ni samoumeven. Potreben je superuporabniški dostop na telefonu, da imamo pravico dostopati do sicer nedostopnih podatkov. Mobilno napravo je bilo zato potrebno odkleniti (ang. rooting). Ob slednjem se podatki na telefonu ne spremenijo.

Na spodnji sliki (Slika 5) si lahko ogledamo vsebino podatkovne baze s pripadajočimi tabelami:




**Slika 5: Tabele v podatkovni bazi.**

Na žalost ni bilo mogoče najti uradne dokumentacije slednje baze, katera bi nam lahko detajlno razložila uporabo vseh tabel v bazi.
Po ročnem pregledu lahko opazimo, da najbolj pomembne tabele so:

- *sms* - vsa sporočila shranjena na mobilni napravi
- *thread* - sporočilne niti (niti predstavljajo sporočila med dvema sporočevalca)
- *part* - podatki za prikaz sporočil MMS (v obliki XML (Extensible Markup Language))
- *canonical-address* - naslovi pošiljateljev sporočil

Nam najbolj pomembna tabela v bazi je tabela *sms*. Vsa testiranja in analize se bodo izvajala na tej tabeli (shema tabele je prikazana na sliki 6).

| sms | table | |
|---|---|---|
| _id | field | INTEGER PRIMARY KEY |
| thread_id | field | INTEGER |
| address | field | TEXT |
| person | field | INTEGER |
| date | field | INTEGER |
| protocol | field | INTEGER |
| read | field | INTEGER |
| status | field | INTEGER |
| type | field | INTEGER |
| reply_path_present | field | INTEGER |
| subject | field | TEXT |
| body | field | TEXT |
| service_center | field | TEXT |
| locked | field | INTEGER |
| error_code | field | INTEGER |
| seen | field | INTEGER |
| deletable | field | INTEGER |

**Slika 6: Shema tabele sms.**

Med različnimi verzijami Android OS se imena in število polj v tabeli *sms* le malo spreminja.
Za boljše razumevanje bomo opisali bolj pomembna polja:

- *thread* - kateri niti pripada sporočilo;
- *date* - datum pošiljanja/prejemanja sporočila (shranjeno v milisekundah);
- *body* - vsebina našega sporočila;
- *read* - ali je bilo sporočilo že prebrano (vrednost 1) ali ne (vrednost 0);
- *type* - ali je bilo sporočilo prejeto (vrednost 1) ali poslano (vrednost 2).

## 6.2 Ročno ponarejevanje SMS sporočil

Na začetku sva skušala ponarejanje sporočil SMS izvršiti ročno. Podatkovna baza s sporočili SMS je bila prenešena na računalnik, kjer smo podatke spreminjali z uporabo *SQlite* urejevalnika. Baza je bila nato nazaj prenesena na napravo, kjer smo naleteli na težavo - spremembe v sporočilih niso bile vidne.

Android OS namreč podatkovno bazo v celoti shrani v predpomnilnik in vse spremembe (večinoma dodajanje novih sporočil) izvrši na bazi v predpomnilniku, katero nato shrani na pomnilnik mobilne naprave ob izklopu le-te.

To izvede zaradi varnosti. Ob ponovnem zagonu naprave, aplikacija, ki je namenjena za prikaz sporočil ni več delovala, ter ni bilo mogoče sprejeti novih sporočil. Spremenjena podatkovna baza namreč ni bila več enaka kot ob vklopu naprave, ko se je prenesla v predpomnilnik.

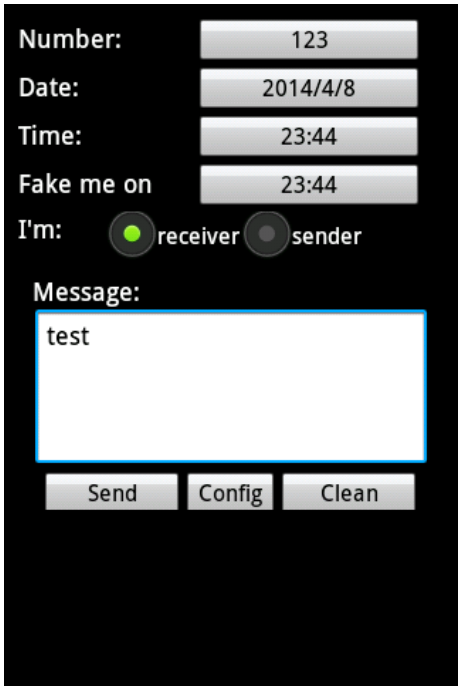
## 6.3 Ponarejanje z uporabo obstoječih orodij

Zaradi težave iz prejšnjega poglavja smo poiskali ali že obstajajo aplikacije, katere nam bi lahko bile v pomoč. Po krajšem iskanju smo se odločili za dve: *Fake SMS sender* [1] in *SMS Editor* [4].

### 6.3.1 Opis orodij

Opisani orodji sta prosto dostopni na Android Marketplace in zelo enostavni za uporabo.

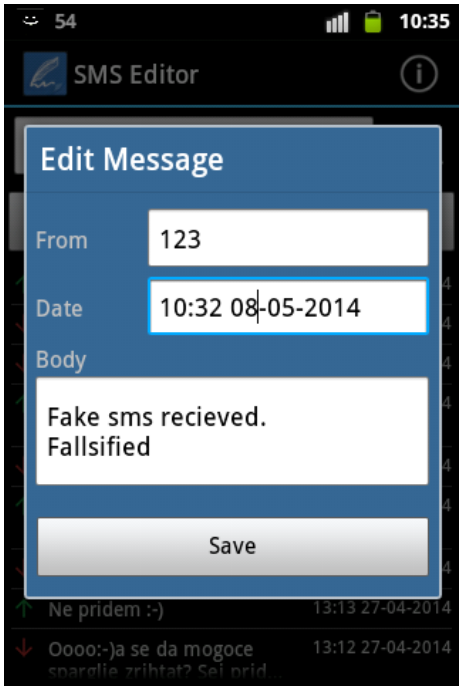
- *Fake SMS Sender* - omogoča kreiranje novih ponarejenih SMS sporočil. Nova sporočila so nato "poslana" na napravo, kjer je aplikacija zagnana. Slika 7 prikazuje zajet zaslon mobilne naprave z zagnano aplikacijo *Fake SMS Sender*.




**Slika 7: Orodje Fake SMS Sender.**

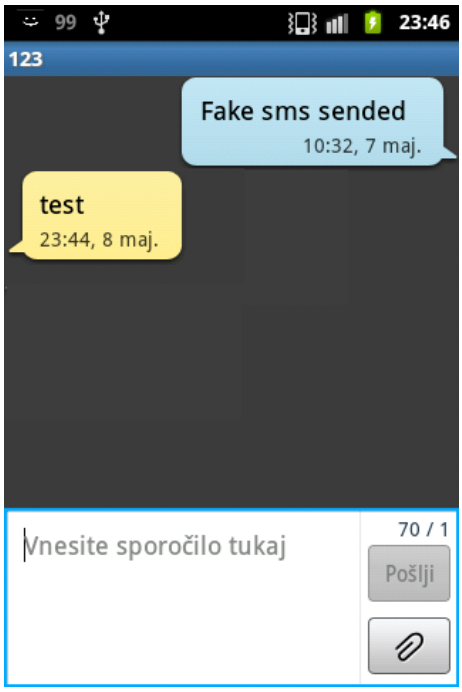
- *SMS Editor* - omogoča ponarejanje obstoječih sporočil SMS. Za posamezno sporočilo lahko spremenimo:
  - številko pošiljatelja,
  - datum pošiljanja,
  - vsebino sporočila.

  S prvo aplikacijo sva lahko ustvarjala nova sporočila, ki niso bila ne poslana, ne prejeta, ampak shranjena direktno v bazo, trenutno v predpomnilniku. Slika 8 prikazuje zajet zaslon mobilne naprave z zagnano aplikacijo *SMS Editor*.




**Slika 8: Orodje SMS Editor.**

### 6.3.2 *Testiranje znotraj mej*




**Slika 9: Sporočila ustvarjena s pomočjo aplikacije Fake SMS Sender.**

Kot je razvidno s slike 9, vsa sporočila na prvi pogled izgledajo legitimna. Lahko tudi izberemo poljubno telefonsko številko iz katere ali v katero pošljemo/prejemamo sporočila.

Z aplikacijo SMS Editor smo lahko ponarejevala tako prava sporočila SMS, kot tudi tista ustvarjena s prej omenjeno aplikacijo.

Sporočilu, ki je bilo sprejeto 7. maja ob 10:32 sva spremenila datum (na 8. maj) in vsebino sporočila. Iz zaslonske slike (Slika 10) lahko vidimo, da se podatki pravilno izpisujejo.




**Slika 10: Ponarejena sporočila.**

### 6.3.3 *Testiranje izven mej*

Kot v članku [7] je bilo tudi za operacijski sistem Android testiranje spreminjanja podatkov izven dovoljenih mej. Na žalost smo lahko izven meja spreminjali le datume (zaradi same strukture uporabljenih aplikacij).

Kot že rečeno se datumi shranjujejo v milisekundah, katere se pretvarjajo v *java.util.Date* format.
Zaradi take pretvorbe se datum izven meja še zmeraj pretvori v milisekunde ter shrani v bazo, le da pretvorba izgubi neveljaven datum ter ga pretvori v drugega, kateri se pravilno prikaže.
Za lažje razumevanje si poglejmo naslednji zgled:

- Imamo začetni datum 8. 5. 2014, 10:32, kar je v milisekundah predstavljeno s številom 1,407,249,120,000.
- Nato vnesemo neveljaven datum, 38. 5. 2014, 11:78.
- Datum se pretvori v število 1,486,270,800,000, ki se nato prikaže v sami mobilni napravi kot 2. 5. 2017 12:00.

## 6.4 Pregled ponarejenih sporočil

Po opravljenih ponarejanjih sporočil sva začela s pregledom le teh. Sama sporočila sva neposredno pregledala v bazi za morebitne nekonsistentnosti z obstoječimi (neponarejenimi) sporočili.

Sporočila ustvarjena s posamezno aplikacijo smo obravnavali posamezno. Vsa sporočila so bila primerjana z enim resničnim sporočilom (s poljem *id=8991*), ki je že pred ponarejanjem bilo prisotno v podatkovni bazi.

### 6.4.1 *Sporočila SMS ustvarjena z aplikacijo Fake SMS Sender*

Sporočilo ima dodeljen pravilen *id*. Nekoherentnost se izkaže pri poljih *protocol* ter *reply path present*. Obe polji sta vedno prazni, pri čemer se razlikuje z resničnim sporočilom, ki ima v teh polji vrednost 0.

Po slednjem lahko med forenzično analizo sklepamo, da sporočilo ni bilo prejeto preko omrežja, ampak ročno ustvarjeno na sami mobilni napravi.

Prazna polja lahko vidimo na spodnji sliki (označena z rdečim kvadratom, Slika 11).

| _id | thread_id | address | person | date |
|---|---|---|---|---|
| 8991 | 12 | +386409... | | 1397457... |

| protocol | read | status | type | reply_pat.. |
|---|---|---|---|---|
| 0 | 1 | -1 | 1 | 0 |

| _id | thread_id | address | person | date |
|---|---|---|---|---|
| 9398 | 313 | 123 | | 1399451... |

| protocol | read | status | type | reply_pat. |
|---|---|---|---|---|
| | 1 | -1 | 2 | |

**Slika 11: Pregled sporočil ustvarjenih s Fake SMS Sender.**

### 6.4.2 *Sporočila SMS ponarejena z uporabo aplikacije SMS Editor.*

Za razliko pri sporočilih ustvarjenih s Fake SMS Sender, imajo sporočila ponarejena z SMS Edtitor vsa polja vpisana. Do nekonsistentnosti pride v dodeljenem *id* in datumom sporočila, kar lahko forenzik kar hitro analizira. V drugih poljih se sporočilo ne razlikuje od pravega.

Zgleda da samo sporočilo aplikacija najprej kopira v predpomnilnik, nato spremeni njene podatke in nato ponovno vpiše v podatkovno bazo (shranjeno v predpomnilniku). V našem primeru ima sporočilo s starejšim datumom, višji dodeljen identifikator. Kar lahko vidimo na spodnji sliki (Slika 12).

| _id | thread_id | address | person | date |
|---|---|---|---|---|
| 9399 | 313 | 123 | | 1402129920000 |

| _id | thread_id | address | person | date |
|---|---|---|---|---|
| 9433 | 313 | 123 | | 1399537920000 |

**Slika 12: SMS ponarejena z uporabo SMS Editor.**

## 6.5 Zaključek

Z uporabo že obstoječih aplikacij lahko opazimo, da se ponarejena sporočila le shranijo v bazo, ter se pravilno prikazujejo tudi na sami mobilni napravi.

Po podrobnem pregledu podatkovnih baz, forenzik lahko relativno enostavno preišče za morebitne nekonsistentnosti med podatki opisanimi v tem članku. Po čemer lahko sklepa, da so sporočila nedvomno ponarejena.

# 7. NAČINI ZA PREPOZNAVANJE PONAREJENEGA SPOROČILA SMS

Forenzik sam, na prvi pogled, ne more ugotoviti ali gre za ponarejeno sporočilo SMS. Ob možnem zunanjem pozivu se lahko forenzik v primer poglobi in razišče, če je prišlo do modifikacij vrednosti.

Za ugotovitev, če je prišlo do ponarejanja sporočil SMS, lahko preverimo izpisek s strani ponudnika mobilnih storitev. Na ta način lahko zagotovimo, da so datumi pravilni, ne pa tudi vsebina. Če se vsi elementi časovnega žiga ujemajo, potem lahko sklepamo, da do sprememb datumov ni prišlo. Ker pa zapiski ne vsebujejo vsebine sporočila SMS, ne moremo zagotoviti kateri vsebini pripada določen časovni žig. Obstaja tudi možnost, da datum in časovni žig poslanega sporočila SMS ni točen, saj se časovni žig ustvari na mobilni napravi.

Dolžina spremenjenega sporočila SMS se je najverjetneje tudi spremenila, zato je dobro preveriti tudi število delov, ki sestavljajo sporočilo SMS in to število primerjati s številom sporočil SMS pri ponudniku storitev. Če se ti števili ne ujemata, je možno sklepati, da je bila vsebina sporočila SMS ponarejena oz. spremenjena.

Za konec bi lahko v poštev pri preiskavi, če posumimo da gre za ponarejeno sporočilo, prišel tudi osumljenčev računalnik, saj obstaja možnost, da bi se tam nahajale beležke uporabljenih orodij za spreminjanje sporočil SMS ali pa varnostne kopije tabel PM. Nadaljnja preiskava teh datotek, bi lahko potrdila oz. ovrgla sum ponarejanja sporočil SMS.

# 8. SKLEP

Vsi testi in eksperimenti so bili izvedeni zato, da lahko potrdimo, da se kratka sporočila SMS da ponarediti. Na mobilni napravi Nokia nam programsko ni uspelo ustvariti novega sporočila ampak le urediti že obstoječe, z ustrezno COTS strojno oz. programsko opremo. Medtem ko pri napravah z operacijskim sistemom Android lahko (ponarejena) sporočila SMS tudi ustvarjamo. Vse kar je uporabniku vidno, je mogoče spremeniti tako, da je to forenziku oz. preiskovalcu skrito. Z minimalnim finančnim vložkom lahko tako vsak uporabnik, z malo tehničnega znanja o strukturi sporočil SMS in njihovih kodiranjih, ponaredi podrobnosti in vsebino sporočila SMS, še lažje je to narediti na operacijskem sistemu Android.

# 9. REFERENCES


[1]

[2] Advance Turbo Flasher by AdvanceTeam. http://www.advance-box.com/. Dostopano: maj 2014.

[3] Cyclone-BOX. http://www.cyclonekey.com/index.html. Dostopano: maj 2014.

[4] SMS Edit. https://play.google.com/store/apps/details?id=com.rayy.android.ed Dostopano: maj 2014.

[5] European Telecommunications Standards Institute. Technical realization of the short message service (sms).

http://www.etsi.org/deliver/etsi_gts/03/0340/05.03.00_60/gsmts_0340v050300p.pdf, 1996.

[6] Natasha Lomas. Android took 79% global share of smartphones in 2013. http://techcrunch.com/2014/01/29/android-79-ios-16-wp-4/. Dostopano: maj 2014.

[7] Thomas Marryat and John Corcoran. Falsifying sms messages.

# Analiza sistemov za izsledite IP naslova

Grega Gašperšič
Fakulteta za Računalništvo in Informatiko,
Fakulteta za Matematiko in Fiziko
Univerza v Ljubljani
Ljubljana, Slovenija
gaspersic.grega@gmail.com

Karmen Bezlaj
Fakulteta za Računalništvo in Informatiko,
Fakulteta za Matematiko in Fiziko
Univerza v Ljubljani
Ljubljana, Slovenija
bezlajk@gmail.com

## ABSTRACT
V nalogi predststavimo metode za učinkovito izsledenje IP naslova vse do izvora v primeru DoS ali DDoS napada. Predstavimo trenutne metode in vidike razvrščanja pristopov za izsleditev IP naslova. Na podlagi razvrščevalnih kriterijev uvrstimo metodi DPM in DFM v posamezne kategorije. Nadaljujemo z predstavitvijo ključnih komponent v posamezni metodi, njihovo analizo in primerjavo med njima. Opisane metode in pristopi omogočajo žrtvi napada identifikacijo poti v omrežju, ki jo naredi napadalni omrežni promet in ne zahteva nobene interakcije ponudnika internetnih storitev (ISP). Poleg tega se te metode lahko izvedejo v "post mortem" - po tem ko je bil napad končan.

## Keywords
IP, izsleditev, Deniel of Service (DoS), Destributed Deniel of Service (DDoS), IP spoofing, DPM (Deterministic Packet Marking), DFM (Deterministic Flow Marking), Authenticated Flow Marking

## 1. UVOD
Denial of Service (DoS) in Destributed Deniel of Service (DDoS) napadi porabijo sredstva oddaljenega gostitelja ali omrežja, s čimer povzročijo preobremenjenost naprav in onesposobijo ali otežujejo omrežne storitve uporabnikom. Celo visoko zmogljivi strežniki so lahko enostavno preobremenjeni s temi vrsti napadov. Napadalec pošlje žrtvi več tisoč paketov v kratkem časovnem obdobju, s čemur povzroči preobremenjenost naprave na strani žrtve in ovira komunikacijski medij med predvidenimi uporabniki in žrtvijo. Prav tako so takšni napadi eni izmed najtežjih varnostnih problemov, ker jih je zelo enostavni izvesti, težko preprečiti in zelo težko izsledit.

V zadnjih letih so postali DoS napadi vse pogostejši. Prav tako so posamezni napadalci med seboj začeli komunicirati, se usklajujevati in poskušali izvesti DoS napade na večja omrežja ali ločena mesta na omrežju (imenovano tudi DDoS napad). Razlogi kot so kompleksnost interneta, to internet nima stanja oziroma validnih prehodov stanj ("state-machine"), zaupljivost v IP protokol (prvotno ni vključeval avtentikacijo paketov), enostavnost in razširjenost orodij s katerimi lahko napadalci izvršijo napad, so povzročili da se napadalci ne bojijo da ga bodo ujeli. To je tudi glavni razlogi da so se te napadi tako zelo razširili.

Pojavitev Denial of Service (DoS) in Destributed Deniel of Service (DDoS) napadov je hitro spremenilo varnost omrežji. Do nekaj let nazaj so metode za ravnanje s takšnimi vrstami napadov zelo zaostajale v hitrosti razvoj, ampak samo število in kompleksnost teh napadov je povzročila tudi hitrejši razvoj orodij, ki jim bo nasprotovala.

Do sedaj je bilo predlaganih več pristopov za boj proti DoS in DDoS napadi. Te pristope lahko razdelimo v naslednje 4 skupine: preprečevanje vdora, zaznavanje vdora, blaženje vdora in odziv na sam vdor. Namesto preprečevanja, zaznavanja in blaženja vdora so bomo osredotočili na samo izsleditev IP naslova napadalca, ko je napad v teku oziroma se je že zgodil. Izsleditev IP naslova spada v skupino za odziv na vdor in je namenjena nadzoru in zadrževanju samega napada. V razdelku 2 opišemo osnovne značilnost izsleditev IP naslova in razdelimo metode izsleditve na različne vidike in kategorije. Slednje je pomembno za razumevanje zgradbe metod in postopka izsleditve.

Nadaljujemo z opisom DPM metode (razdelek 3) in njeno analizo. Tesno se navezujemo na prejšnje poglavje - začnemo s samo kvalifikacijo DPM metode v vidike in kategorije naštete v prejšnjem poglavju. Tekom razdelka opišemo osnovno metodo in izboljšave.

Sledi opis DFM metode (razdelek 4), njena analiza ter primerjava DFM metode z DPM. Tudi v temu poglavju se tesno navezujemo na poglavje 2. Delo povzamemo v zaključku.

## 2. IZSLEDITEV IP NASLOVA
Namen izsleditev IP naslova je identifikacija izvora katerega koli paketa in ni omejena samo na DoS in DDoS napade. Trenutne metode in metode opisane v temu delu so bile razvite z namenom, da postanejo neobčutljive tudi na napade v katerih se IP naslova v vsakem paketu zamenjava z novim lažnim IP naslovom (IP spoofing).

Doslej je bilo predlaganih veliko število pristopov izsleditev IP naslova, ki jih lahko razdelimo v naslednje tri vidike:

- Osnovni principi (Basic principles),
- Metode procesiranja (Processing methods),
- Lokacija (pošiljanja podatkov o izsleditvi IP-ja) (Location).

Posamezne vidike pa obstoječe metode razvrščajo v več kategorij, ki so napisane v nadaljevanju.

## 2.1 Osnovna načela

Po klasifikaciji osnovnih principov lahko obstoječe metode razdelimo na dve kategorije, Logging (metode logiranja) in Marking (metode označevanja).

Pri metodi logiranja vsak usmerjevalnik ohrani specifično informacijo (verzija paketa, dolžina glave, enolična identifikacija paketa, identifikacija protokol pošiljanja... [2]) za potujoče pakete. Nato lahko s pomočjo data mininga določimo pot, ki so jo prečkali paket. Kot primer lahko vzamemo implementacijo opisano v [3]. V njej predlagajo generiranje digitalnih odtisov iz nespremenljivih delov paketov (izvorni naslov, ciljni naslov, ...) in prvih 8 bajtov tovora paketa. Poleg generiranja digitalnih odtisko v opisanemu postopku predlagajo da se med samim procesom izsleditve usmerjevalniki preveri ali je bil sumljiv paket posredovan ali ne.

Nadaljne izboljšave te metode so vidne v logiranju majhnega dela vsakega paketa na prehodnih usmerjevalnikih [8]. Ta metoda ima uporabno lastnost, da lahko izsledi napad dolgo po temu, ko je bil že končan [7]. Glavni problemi te metode so velika obremenjenost procesorja in poraba spomina na usmerjevalnikih, ki so na poti napada, saj morajo predelati veliko količino podatkov. Prav tako je sama učinkovitost logiranih podatkov omejena na velikost prostora na usmerjevalnikih in informacije, ki jih shranjujemo.

V metodi označevanja nekateri ali vsi usmerjevalniki v poti napada pošiljanjo določene informacije skupaj z paketom. Sama informacij je lahko vsebovana v glavi IP paketa ali pa poslana z ustvarjanjem novih paketov, ki porabijo dodatno pasovno širino. Informacijo lahko nato uporabimo za izsledite IP naslova, tudi če je bil spremenjenv v IP paketih.

Savage v [7] opisuje metodo označevanja paketov, ki omogoča sledenje napadom, ki poplavljajo omrežje z anonimnimi paketi vse do izvora. Ta metoda omogoča rekonstrukcijo celotne poti z združevanjem izjemno majhnega števila paketo in jo je možno izvršiti šele, ko je bil napad odkrit. Sama metoda temelji na temu, da usmerjevalniki na naključne pakete zapišejo specifične informacije.

Poleg zgornje metode je Savage v [7] predstavil dve novi tehniki označevanja: Advanced Marking Scheme in Authenticated Marking Scheme, ki dopoljujejo in izboljšajo obstoječe metode. Pomembnejša izmed obeh je Authenticated Marking Scheme. Ta omogoča avtentikacijo označevanja na usmerjevalnikih. S tem lahko preprečimo, da ogroženi usmerjevalnik (usmerjevalnik, ki je podlegel napadalcu) ne pošilja paketov z oznakami neogoroženih usmerjevalnikov, s čemur bi lahko svoj IP-naslov prekril.

## 2.2 Metode procesiranja

Tako kot pri klasifikaciji osnovnih načel lahko obstoječe metode vidik predelave podatkov razdelimo na dve kategoriji, Deterministic (deterministične metode) in Probabilistic (verjetnostne metode).

V determinističnih metodah, ne glede na izbrano kategorijo iz vidika osnovnih načel (metode logiranja ali označevanja), moramo vsak paket obdelati na izvori in cilju. V primerjavi z verjetnostnimi metodami, te zahtevajo več procesiranja in daljši čas obdelave paketov, ampak omogočajo večjo natančnost podatkov. Kot primer lahko vzamemo implementacijo Belenky-ja v [5], ki v razdrobljenem ID (fragment ID) delu paketa z verjetnostjo 0.5 shrani zgornjo ali spodnjo polovico IP naslova vdirnega vmesnika. Poleg vnosa IP naslova se določi rezervni bit, ki označuje kater del IP naslova je shranjen na razdrobljenem ID del paketa.

Kljub večji natančnosti večina implementacij izsledite IP naslova uporablja verjetnostne metode in ne deterministične. V primerjavi z deterministčnimi metodami je čas obdelave paketov hitrejši pri verjetnostnih metodah in potrebujejo manj pasovne širine, ampak imajo pa bolj zapleteno obdelavo podatkov na ciljni strani. Kot primer lahko vzamemo verjetnostno metodo PPM (Probabilistic Packet Marking)[7], kjer usmerjevalnik vsak paket označi z IP naslovom usmerjevalnika ali z IP naslovom od koder je paket prišel do usmerjevalnika. Song and Perrig sta v [4] kot izboljšavo predlagala, da namesto zapisovanja IP naslova, kateremu je dodana hash vrednost, zakodiramo IP naslov v 11 bitni hash in vdržujemo 5 bitno številnko hop shranjeno v 16-bitnem fragment ID delu paketa. Nadaljne izboljšava je uporaba dveh različnih hash funkcij, s čemur se loči vrstni red usmerjevalnikov. Z uporabo hash funkcij si zagotovimo integriteto in validnost podatkov.

## 2.3 Lokacija (pošiljanja podatkov o izsleditvi IP naslova)

Z vidika lokacije podatkov podatkov o izsleditvi IP naslova lahko obstoječe metode razdelimo na dve kategoriji, Source group metoda in Network group metoda.

Pri Source group metodi so podatki o izsleditvi IP naslova poslani robnemu usmerjevalniku, ki je najbližji izvoru. V primerjavi z Source group metodo, katerega namen je identifikacija izvora napada, je namen Network group metode identifikacija celotne ali vsaj dela poti napada. Podatki o izsleditvi IP naslova so poslani preko vseh ali samo nekaterih usmerjavalniko na poti napada. Slabost te metode je velika poraba virov in potrebna je sinhronizacija med večimi usmerjevalnik.

Nadaljne izboljšave Network group metode sta Song and Perrig predlagala v [4]. Ta razširi možnost sledenja napadalca ne samo do robnega usmerjevalnika vira napada ampak tudi v avtonomne sisteme. To lahko dosežemo z vpeljavo metod označevanja, ki svoje oznake enkriptirajo in avtenticirajo na usmerjevalnikih. Zaradi samega avtenticiranja so te metode varne pred usmerjevalniki, ki so bili ogroženi s strani napadalca in poskušajo spremeniti svoje oznake v oznake neogorženih usmerjevalnikov.

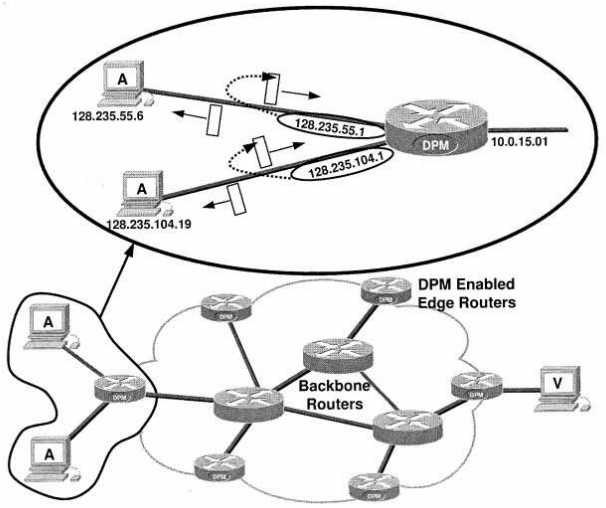



**Figure 1: DPM (Deterministic Packet Managment)**

*Marking procedure at router R, edge interface I:*
**for each** incoming packet $w$
let $x$ be a random number from [0, 1)
**if** $x < 0.5$ **then**
write $I_{0-15}$ into $w$.ID_field
write '0' into $w$.flags[0]
**else**
write $I_{16-31}$ into $w$.ID_field
write '1' into $w$.flags[0]

*Ingress address reconstruction procedure at victim V:*
**for each** packet $w$ from source $S_x$
**if** $IngressTbl[S_x] == NIL$ **then**
create $IngressTbl[S_x]$
**if** $w$.flags[0] == '0' **then**
$IngressTbl[S_x]_{0-15}$ := $w$.ID_field
**else**
$IngressTbl[S_x]_{16-31}$ := $w$.ID_field

**Figure 2: DPM (Deterministic Packet Managment)**

# 3. DPM (DETERMINISTIC PACKET MARKING)

DPM je znana metoda za izsleditev IP naslova in ima številne privlačne lastnosti, kot enostavna implementacija, nizka procesorska in spominska poraba na žrtvinem računalniku in usmerjevalnikih, ki sodelujejo v sistemu. Izmed vseh prej naštetih del le DPM spada v isto kategorijo kot DFM. DFM lahko zvrstimo na naslednje vidike:

- Osnovni principi: Marking
- Metode procesiranja: Deterministic
- Lokacija (pošiljanja podatkov o izsleditvi IP-ja): Near the Source

## 3.1 DMP metoda

Osnovno metodo DPM sta Belensky in Ansari v [5] , ki sta jo nato tudi izboljšala v [6]. DPM kot vidik osnovnega načela uporablja metodo označevanja, za katero je značilno, da označujejo samo vmesniki na robnem usmerjevalniku kateri so bili napadeni. DPM svoje označevalne informacije shranjuje z 17 biti v glavi IP naslova. Prvih 16 bitov informacije predstavlja identifikacijo paketa, 1 bit je pa rezerviran za označevalno informacijo za vsak paket posebaj.

32 bitni IP naslov napadenega vmesnika je razdeljen na dva segmenta po 16 bitov (segment 0 - biti od 0 do 15 in segment 1 - biti od 16 do 31). Ko paket pride skozi robni vmesnik je eden od segmentov izbran z enako verjetnostjo $p$. 1 bit, ki je bil rezerviran za označevalno informacijo nam pove izbiro segmenta IP naslova. Segment izbran z verjetnostjo $p$ se nato vstavi v polje za identifikajo. Žrtev napada vzdržuje tabelo ujemanj izvornih naslovov in naslovov vdorov. Ko žrtev dobi oba segmenta robnega usmerjevalnika, lahko rekonstruira celotni IP naslov vstopnega vmesnik usmerjevalnika.

Pri tem je treba opozoriti da so označeni samo dohodni in ne odhodni paketi. To nam zagotavlja da izstopni usmerjevalnik ne bo prepisal oznake v paketu, ki ga imajo vstopni usmerjevalniki.

Poleg zgorja že opisane lastnosti ima DPM ima naslednje ključne značilnosti:

- DPM označuje samo najbližji robni usmerjevalnik blizu napadalca, v katerega je napadalec vdrl.
- DPM označuje vse pakete na vdornem vmesniku robnih usmerjevalnikov.

Osnovna različica DPM je zelo učinkovita pri DoS napadih, ampak ima pa visoko stopnjo lažnih pozitivov pri DDoS napadih. Razlog za to je, da žrtev povezuje segmente naslovov vdora z izvornim naslovom napadalca. V DDoS napadu so pa IP naslovi velikokrat spremenjeni ali drugačni od izvornega naslova napadalca. V primeru takšnih napadov obstajata naslednje dva primera, ko bo rekonstrukcija IP naslova neuspešna:

- dva ali več gostiteljev, ki napadajo žrtev imajo isti IP naslov,
- DDoS napadalci spremenijo izvorno polje v vsakem poslanem paketu.

Da bi odpravili to napako so osnovno različico DPM-ja nadgradili s pomočjo uvedbe hash funkcije. Z njeno pomočjo so lahko izračunali hash vrednost vstopnega naslova, ki je bil napaden in podatek shranili v označevalno informacijo. Vsi paketi, ki pripadajo vdornem vmesniku robnega usmerjevalnika imajo enako hash vrednost. Z uporabo hash vrednosti pridobljene iz označevalnih informacij lahko žrtev primerja pravilne označevalne podatke s svojo tabelo in podatke uspešno pretvori v veljavni IP naslov. Pri sami izbiri hash funkcije moramo paziti, da ne pride do konfliktov pri izračunu hash vrednosti.

Z uvedbo hash funkcije je bila označevalna informacija razdeljena na naslednje segmente:

- segment vstopnega naslova - a,
- indeks segmenta - d,
- razgrajen vstopni naslov - k

Najboljša konfiguracija zgornje naštetih parametrov glede na velikost paketa je naslednja: $a = 4$, $d = 3$ in $k = 10$, vse skupaj 17 bitov.

## 3.2 Analiza DMP metode

Pri analizi DPM metode smo upoštevali naslednje glavne točke:

- **Obremenjenost procesorja:** Pri DPM metodi je objemenjenost procesorja v primerjavi z ostalimi metodami izsleditve IP-ja majhna. Kot primerjavo lahko vzamemo PPM, kjer kjer usmerjevalnik vsak paket označi z IP naslovom usmerjevalnika ali z IP naslovom od koder je paket prišel do usmerjevalnika. Pri DPM pa označuje samo robni usmerjevalnik, ki je najbližje napadalcu. Poleg tega pa pri DPM ni tako kompleksne logike odločanja, kater paket naj bo označen in kater ne, kot pri PPM.

  Vendar pa pri metodi DPM obstajajo druge zahtevnejše operacije, kot so priprava označevalne informacije in nadgrandnje označevalnih polj. Pri tem moramo pa vedeti, da s pomočjo prej naštetih operacij postane proces rekonstrukcije poti napada preprostejši kot pri PPM in potrebuje veliko manj sredstev. Poleg tega če uporabljamo nadgrajeno različico DPM metode, ki IP naslove pretvarja s pomočjo hash funkcije, se izognemo kombinatoričnemu problemu PPM metode.

- **Poraba spomina:** Poraba spomina na vsakemu usmerjevalniku je pri DPM metodi zanemarljiva. To je zato, ker DPM potrebuje samo $32/a$ paketov za rekonstrukcijo IP naslova.

- **Napadalec poskuša spremeniti oznako paketa:** Pri DPM metodi je vsak paket označen ko vstopi v omrežje. Če bo napadalec poskušal spremeniti oznako, bo oznako popravil robni usmerjevalnik, ko bo prišlo v omrežje.

- **Stopnja lažnih pozitivov (false positive rate):** Osnovna različica DPM metode ima veliko stopnjo lažnih pozitivov. To se da odpraviti z uvedbno hash funkcije, kot napisano v prejšnjem razdelku. Problem ostane samo pri DDoS napadih, kjer mora biti število istočasnih napadalcev omejeno na določeno število N, da ostanje stopnja lažnih pozitivov manj kot 1%. Z enačbo o skupnemu številu lažnih pozitivov $[2^a - 2^a(1-\frac{1}{2^a})^{\frac{N}{2^d}}]^k - N$ lahko poračunamo maksimalno število N:

$$[2^a - 2^a(1 - \frac{1}{2^a})^{\frac{N}{2^d}}]^k - N \leq 0.01N$$

- **Spreminjanje oznake z napadenimi usmerjevalniki:** DPM metoda predpostavlja, da oznaka na paketu ostane nespremenjena, dokler paket potuje skozi omrežje. Ker sama metoda ne vsebuje avtentikacije paketov na usmerjevalnikih, lahko pride do spremembe oznake na usmerjevalniku, ki je bil napaden. Če je ogrožen usmerjevalnik na poti napada in usmerjevalnik spreminja oznake paketov ne bomo morali ugotoviti IP naslova izvora napada. To problem bi lahko odpravili tako, da bi DPM razširili z Authenticated Marking Scheme.

- **Število potrebnih paketov za izsleditev IP naslova:** Število paketov potrebnih za izsleditev IP naslova je enako $32/a$ (32 bitni IP naslov). Pri predpostavki, da je $a = 4$ [6]. DPM potrebuje 8 paketov za izsleditev IP naslova vmesnika robnega usmerjevalnik najbližjega napadalcu.

- **Vloga ponudnika storitev (ISP):** Vloga ponudnika storitev je pri DPM metodi zelo majhna. Ponudnik storitev bi samo moral nadgraditi robne usmerjevalnike z podporo determinitičnega označevanja paketov. Za primerjavo lahko vzamemo metodo PPM v katei vsak usmerjevalnik od ponudnika velika igra pomembno vlogo pri označevanju paketov. Pri DPM pa usmerjevalniki od ponudnik, ki niso na poti napada niso odgovoren za za nobeno funkcionalnost DPM metode.

- **Razdrobljenost:** Metoda DPM pri označevanju paketov shranjuje označevalne informacije v del glave IP naslova, ki je namenjen nadzoru razdrobljenosti datagramov paketa. Sama posledica te implementacije je, da če bo samo en paket razdrobljenega datagrama označen, bo na strani prejemnika nemogoče sestaviti celotni paket.

Iz zgornje analize lahko zasledimo naslednje pozitivne lastnosti DPM metode:

- enostavna implementacija,
- nizka dodatna obremenitev na pasovno širino,
- skoraj nobene dodatne kompleksnosti procesiranja,
- primeren za različne napade (ne samo DoS in DDoS),
- združljivost z starejšo strojno in programsko opremo,
- ne podeduje pomanjlkjivosti ostalih varnostnih sistemov,
- ne razkrije topologije omrežja,
- prilagodljiv.

Kljub temu da DPM metoda vsebuje zgoraj naštete lastnosti ima tudi nekatere negativne:

- Če želimo, da je število negativnih pozitivov (false positive rate) manjše kot 1%, moramo poskrbeti, da se metoda DPM ne prilagodi na večja omrežja. Boljše je če je DPM metoda uporabljen na več manjših podomrežjih.
- DPM metoda lahko izsledi IP naslov do robnega usmerjevalnika blizu napadalca in ne natačnega vozlišča napadalca.

- DPM metoda je deterministična in ima višjo natančnost kot verjetnoste metode, ampak ta natančnost je dosežena z označevanjem vseh paketov v omrežju.
- DPM metoda predpostavlja, da informacija o označevanju ostane nespremenjena medtem ko paket potuje skozi omrežje.

Zaradi zgoraj naštetih negativnih lastnosti je prišlo do razvoja metode DFM opisane v naslednjem razdelku.

# 4. DFM(DETERMINISTIC FLOW MARKING)

## 4.1 DFM metoda

DFM je obetaven metoda za izsleditev IP naslovov. DFM ima nekoliko drugače pristop označevanja in sicer namesto, da bi označil vsak paket, kot do dela DPM, označi znotraj vsakega toka samo nekaj paketov. DFM z označevanjem paketov ohranja natančnost, ki jo ima DPM z označevanjem le nekaj paketov namesto vseh pa pridobi še lastnosti PPM (probability packet marking) metod. PPM so metode, ki paket označi z neko verjetnostjo. Glavna prednost DFM metode je poleg natančnosti tudi to, da lahko izsledimo napadalčev IP na vozlišču znotraj LAN naslova za zadnjim usmerjevalnikom. Za izsleditev IP naslova, DFM uporablja tri identifikatorje in sicer: (i) IP naslov izstopnega vmesnika zadnjega usmerjevalnika; (ii) NI-ID, kar je identifikator dodeljen vsakemu vmesniku sestavljen ali iz MAC naslova omrežnega vmesnika zadnjega usmerjevalnika ali iz VLAN ID virtualnega vmesnika, če robni usmerjevalnik uporablja VLAN vmesnik; in (iii) Node-ID, kar je identifikator dodeljen vsakemu izvornemu MAC naslov, ki se pojavi znotraj vhodnih podatkov v lokalnem omrežju.

V tej metodi pakete označuje le napadalčev robni usmerjevalnik. DFM označi vsak tok z 60 biti identifikacijskih podatkov, od tega ja 32 bitov namenjeno IP naslova napadalčevega vmesnika, 12 bitov za NI-ID in 16 bitov za Node-ID, s čimer je moč ločiti med prometom iz različnih . Teh 60 bitov identifikacijskih podatkov je razdeljeno na $K$ kosov; torej je oznaka sestavljena iz $M = 60/K$ bitov identifikacijskih podatkov in za identifikacijo posameznega kosa potrebujemo $S = log_2(K)$ bitov. DFM uporablja en bit za zastavico, s katro loči med označenimi in neoznačenimi paketi znotraj toka. V vsakem toku ima prvih $K$ paketov označen del, ki vsebuje tudi $M$ bitov namenjenih identifikacij kosa podatkov, $S$ bitov odmika, ki predstavljajo eno od $2^s$ možnih delov in en bit namenjen zastavici $F$, le ta je nastavljena na "1", če je označen in na "0" sicer.

Vsak ciljni naslov vsebuje tabelo, ki se ujema z ID toka in vsem $K$ možnim kosom. Tokovni ID je v primeru TCP ali UDP protokol sestavljen iz peterice: izvorni IP naslov, ciljni IP naslov, L4 tip protokola (TCP/UDP), izvorna vrata in ciljna vrata za TCP in UDP tok. Če pa uporabljamo ICMP pa je sestavljena iz šesterice: izvorni IP naslov, ciljni IP naslov, L4 tip protokola (ICMP), tip ICMP, kodo ICMP in ICMP ID. Ko prispe paket, ki je del neznanega toke v ciljni usmerjevalnik, usmerjevalnik poišče znotraj oznake bit, ki je namenjen zastavici. Ko na ciljni naslov prispejo vsi kosi, ki pripadajo nekemu toku, ciljni usmerjevalnik pridobimo informacijo o izvornem usmerjevalniku. Z uporabo DFM lahko ciljna lokacija loči med tokovi iz različnih usmerjevalnikov. Kar nam omogoča, da ob pritoku nenavadnih podatkov obdelujemo vsak usmerjevalnik individualno.

Zaradi optimalnega označevanja toka nam DMF zagotavlja, da označevalne informacije niso bile pokvarjene ali zmanipulirane na poti med izvornim in ciljnim usmerjevalnikom. Tega pri na primer DPM ne moremo doseči. DFM uporablja ECDSA ("Elliptic Curve Digital Signature Algorithm") algoritem [9] in s tem zagotavlja prej omenjeno lastnost. Primer delovanja je sledeč. Robni usmerjevalni ustvari 42 bajtov podpis, ki je sestavljen iz ECDSA podpisa, ki znaša 160 bitov (eliptnični kluč), identifikacijskih podatkov iz 60 bitov in 13 bajtov za ID kodo. DFM teh 42 bajtov vključi na konec $K$-tega paketa vsakega toka. Recimo, da končni usmerjevalni želi poslati podpisano označbo na ciljni naslov, to stori s svojim privatnim ključem. Za vsak tok ustvari 42 bajtni podpis in ga pošlje z $K$-tim paketom toka. Ko ciljni naslov prejme podpis pošiljatelja, ga primerja z njegovim javnim ključem. Tok je sprejet, če se ključa ujemata, kar pa prejemniku zagotavlja, da je oznaka pravilna. Ta postopek imenujemo avtentikacija.

## 4.2 Rezultati praktične uporabe

Rezultate dobljene z DFM metodo primerjamo z rezultati DPM. Oba algoritma sta bila preizkušena na CAIDA "anonymized Internet traces October 2012" podatkovni bazi [10]. Ta baza vsebuje anonimne pasivne sledi prometa iz CAIDA" Equinix-Sanjose monitorja" na visoko hitrostni Internetni hrbtenici povezav. Z drugimi besedami CAIDA bazo uporabimo kot standardno tcpdump datoteko. Ta baza je bila izbrana, ker je javno dostopna in omogoča drugim raziskovalcem primerjanje njihovih metod z dobljenimi rezultati na isti bazi.

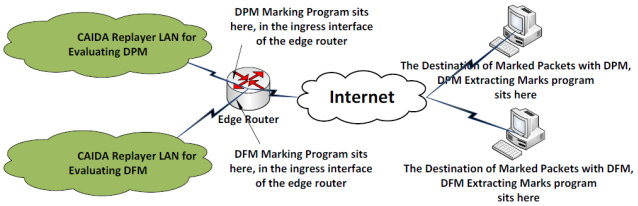



**Figure 3: Mrežna baza za analito DFM in DPM metod za izsletidtev IP naslova**

Za analizo in oceno metod DPM in DFM, je bilo impelmetirano omrežje, kot ga vidimo na sliki 3 . V ta namen sta bila postavljeni dve različni LAN mreži in sicer ena na kateri je bila uporabljena DPM in ena na kateri je bila uporabljena DFM metoda. Tcpreplay in Tcprewrite prosto dostopni aplikaciji sta bili uporabljeni, za večkratno uporabo CAIDA podatkovne baze. Dodatno sta bili implemetirana dva programa, za vsako od metod, ki uporabljata winpcap knjižnico iz C++. En od programov nam pove kateri tok je trenutno v obdelavi znotraj LAN vmesnika na robnem usmerjevalniku, drugi pa za izsleditev paketa na končni strani. Označevalni program deluje na robnem usmerjevalniku in označuje le tiste tokove, ki potujejo iz notranjega v zunanje omrežje. Iztočno, izsledili program deluje na ciljnem vozlišču in poskuša ugotoviti vir označenega prometa.

Kot že omenjeno, je del oznake, torej del identifikacijski po-

datkov, velikosti $M = 60/K$ bitov v vsakem paketu. Od tega je $S = log_2(K)$ bitov namenjenih identifikaciji kosa in en bit za zastavico. V [1], najdemo analizo, ki nam zagotavlja, da je najboljša izbira parametrov ali $M = 30, S = 1$ in $F = 1$, kar znaša 32 bitov, ki vsebuje identifikacijsko polje, zastavico in odmik za del namenjen IP glavi, ali $M = 12, S = 3$ in $F = 1$, kar je 16 bitov, ki nosijo le informacijo identifikaciji in IP glavi. Za izračun uspešnosti, se uporabi

$$TR = \frac{tevilouspenoizsledenihpaketov}{tevilovsehpaketov},$$

in

$$MR = \frac{oznaenipaketi}{vsipaketi}.$$

TABLE I. EVALUATING BOTH DPM AND DFM ON THE CAIDA DATASET, USING THE SAME METRICS

| Comparison Metrics | DPM | DFM with $K$=2 | DFM with $K$=5 |
|---|---|---|---|
| Number of Marked Packets | 241,589,706 | 24,059,752 | 32,470,758 |
| $MR$ | 100% | 9.96% | 13.44% |
| Traced traffic in term of Number of Packets | 23,948,7875 | 22,445,5742 | 219,912,254 |
| TR in term of Number of Packets | 99.13% | 92.91% | 91.03% |
| Traced traffic in term of Size (byte) | 72,019,480,296 | 70,178,321,305 | 69,615,277,632 |
| TR in term of traffic Size | 99.26% | 96.72% | 95.95% |

**Figure 4: Ocena DPM in DFM metod na CAIDA bazi.**

V tabeli 4 je prikazana uspešnost metod DPM in DFM na bazi CAIDA merjena z $TR$ in $MR$. Poudariti je potrebno, da DPM metoda pri $TR$ ne doseže 100%, kar izvira iz dejstva, da DPM nima mehanizma za delo z pokvarjenimi in spremenjenimi oznakami znotraj poti med izvornim in ciljnim usmerjevalnikom.

Rezultati kažejo, da če označimo v vsakem toku prva dva paketa, kot ga označuje DFM je mogoče pravilno najti izvor približno 93% paketov($TR$) in približno 97% prometa v odvisnosti od bajtov, čeprav smo označili le približno 10% vseh paketov ($MR$).Še več DFM pravilo najde izvor od 91.03% vseh paketov ($DR$) ker je 95.05% vsega prometa, gledano iz vidika velikosti, pri označevanju le 13.44% vseh paketov ($MR$), če označimo le prvih pet paketov vsakega toka. Čeprav je zmožnost odkrivanja DFM metode (, ki znaša 92.91%) manjša od zmožnosti odkrivanja DPM metode (, ki znaša 99.13%), je uporaba DFM še vedno bolj priporočljiva, saj zmanjša število paketov, ki so bili prepoznani kot označeni, čeprav niso bili (lažni pozitivi, "false pozitive"), ki znaša pri $K = 2$ 90.04%, pri $K = 5$ pa 86.56%. Poglejmo si nekaj pomembnih lastnosti DFM.

1. **Poraba spomina DMF metode v robnem usmerjevalniku:** Potreben prostor, za uporabo DFM na robnem usmerjevalniku znaša velikost treh tabel, to so: tabela toka, NI-ID tabela in Node-ID tabela [1]. V praktični analizi, cel zahtevan prostor skupaj z CAIDA bazno zahteval manj kot 26 KB.

TABLE II. THE SIZE OF TRANSMITTED TRAFFIC WITH AND WITHOUT THE SOURCE EDGE ROUTER AUTHENTICATION IN BYTES

| $K$ | Marking without Authentication | Marking With Authentication | Increment |
|---|---|---|---|
| 2 | 72,556,397,639 | 72,701,839,187 | 0.2% |
| 5 | 72,556,397,639 | 72,618,776,795 | 0.08% |

**Figure 5: Velikost posredovanega prometa z in brez avtentikacije robnega usmirjavalnika.**

- **NI-ID tabela:** Za vsak vmesnik robnega usmerjevalnika in VLANa, če obstaja, imamo za vsak VLAN 9 bajtno tabelo, ki vsebuje 12 bitov za NI-ID, 48 bitov za MAC naslov in 12 bitov za VLAN ID. Ker je imelo implementirano omrežje dodeljeno en vmesnik za ocenjevanje DFM metode (3), je tabla pri praktični analizi v robnem usmerjevalniku znašala 9 bajtov.
- **Node-ID tabela:** Za vsak zapis v NI-ID tabelo, si DFM zapiše Node-ID v svojo tabelo. Za vsak na nov izvorni MAC naslov, se zapiše 60 bitov, ki vsebujejo 12 bitov z NI-ID naslovom in 48 bitov za MAC naslov. Velikost zaradi tega ni fiksna in je odvisna od števila izvornih MAC naslovov. V praktičnem primeru je implementirana DFM metoda, ki uporablja algoritem za nadzor spomina, ki če nekaj časa nek izvirni MAC ni bil aktiven izriše iz tabele Node-ID. V tem primeru je ta tabela zavzela 890 bajtov.
- **Flow table:** V tej tabeli je vsaka vrstica pripada že opazovanemu toku. DFM shranjuje 180 bitov za vsak tok. Ta zapis je sestavljen iz:
  - ID toka iz 13 bajtov: sestavljen iz peterice, če je tok tipa TCP ali UDP (4 bajti izvorni IP, 2 bajta število izvornih vrat, 4 bajti ciljnega IP naslova, 2 bajta število ciljnih vrat in 1 bajt za protokol), ali iz šesterice če je tok tipa ICMP (4 bajti za ciljni IP, 1 bajt za protokol, 1 bajt za ICMP kodo in 2 bajta za ICMP ID).
  - Oznaka toka iz 60 bitov (kot zgoraj opisano).
  - Števila paketa velikosti 2 bajta: Robni usmerjevalnik poveča število za ena v ustreznem zapisku toka, za vsak poslan paket. Torej nam pove število paketov v toku. DFM uporablja to število, za prvih $K$ paketov vsakega toka.

Ker DMF po koncu toka ne potrebuje več informacij o toku, ga izbriše. Torej je velikost potrebna za tabele odvisna od števila istočasnih tokov. V praktične primeru je bila uporabljena baza CAIDA, ki istočasno izvaja 1131 tokov, kar nanese v maksimalnem primeru približno 25 KB prostora.

2. **Poraba spomina DMF metode na strani žrtve:** Žrtev hrani tabelo za rekonstrukcijo, ki se ujema z tokom ID in $K$ možnimi označbami kosi. Za vsak opazovan tok si zabeležimo 13 bajtov velik ID toka, 60 bitov identifikacijskih podatkov. Tudi tu DFM, ne potrebuje več podatkov, ko je enkrat tok končan, zato

jih zavržemo in prostor je zopet odvisen od sočasnih tokov. V primeru CAIDA je le teh 1131, kar je v maksimalnem primeru naneslo 23 KB.

3. **Poraba spomina DPM metode v robnem usmerjevalniku:** Tabela za rekonstrukcijo zahteva $f$ poti in vsaka od teh poti zahteva $2^{17}$ struktur ($2^d$ območje, $k$ segmentov v vsakem območju in $2^a$ bitov v vsakem območje) [6]. $f$ je število hash funkcij. V praktičnem primeru je bil implementirana DPM metoda z 4 hash funkcijami [6]. Od tod sledi, da je rekonstrukcijska tabela velika 64 Kb.

4. **Pasovna širina Autentikacijske DFM (ADFM) metode:** Z autentikacijo se poveča pasovno širino, to dosežemo z dodatkom 42 bajtov podpisa na konec $K$-tega paketa vsakega toka. Količina teh podatkov dobimo z primerjanjem količine posredovanega prometa in z in brez podpisa 5. V tej tabeli vidimo, da če uporabimo autentikacijo pridobimo 0.2% pasovne širine, ko je $K = 2$ in 0.08%, ko je pasovna širina $K = 5$.

5. **Cena preračuna z ADFM metoda:** Ocena je bila pridobljena z primerjanjem podpisovanjem in oviranjem 164 bitov informacij sestavljenih iz 60 bitov identifikacijskih podatkov in 13 bajtov ID toka. Ocene je bila pridobljena z primerjavo delovanja algoritma na PC-ju z 3.4 GHZ procesorjem z operacijski sistemom Ubuntu 10.04. Eksperiment je pokazal, da podpis potrebuje man, kot pol milisekunde. Še več uporabimo autentikacijo je čas overjanja podpisa manj kot eno milisekundo.

6. **Uporaba spomina ADFM:** Ker je podpis ustvarjen in dodan toku ko le tega pošiljamo, podatki podpisa niso shranjeni v končnem usmerjevalniku. Torej ne potrebujemo nobenega dodatnega prostora.

## 4.3 Analiza DFM metode

- **Obremenjenost procesorja:** Kot DPM je obremenjen le tisti usmerjevalnik, ki je najbližje napadalcu. Leta je torej obremenjen z izdelavo in posodobitvijo oznake potreben za identifikacijo. Za razliko od DPM pa DFM ne potrebuje izračuna hash vrednosti napadalčevega naslova. Da žrtev pridobi informacijo o napadalčevem IP naslovu uporabi ID toka. Dodatna ralika med DPM in DFM je, da DFM označuje le prvih $K$ paketov v toku iz katerih pridobi informacije o toku. Od tod vidimo, da ima DFM manjšo obremenitev procesorja. Metoda potrebuje za overjanje in izgradnjo manj kot eno milisekundo časa.

- **Poraba spomina:** Kot DPM je poraba spomina na usmerjevalniku zanemarljiva (okoli 25 Kb), na strani žrtve pa DFM zahteva prostor za rekonstrukcijsko tabelo, ki je ravno tako zanemarljive velikosti (23Kb). Kar je manj kot 64 Kb, ki jih potrebuje DPM.

- **Stopnja lažnih pozitivov (false positive rate):** DFM za razliko od DPM, se z problemom lažnih pozitivov ne sreča, saj za izsledite IP naslova uporablja ID toka. Pri DDoS napadu DPM odpove, DFM pa deluje brez težav.

- **Napadalec poskuša spremeniti oznako paketa:** DFM metoda označi vsak tok, ko vstopi v robni usmerjevalnik. V tem primeru, tudi, če je napadalec želi spremeniti oznako, bo le ta prepisana, z pravilno oznako, ko bo tok vstopil v robni usmerjevalnik. Zato, spreminjanje oznake ni problem.

- **Spreminjanje oznake z napadenimi usmerjevalniki:** Za razliko od DPM, ki nimam nobenega mehanizma, ki bi odpravila napake nastale zaradi okvare usmerjevalnikov na poti napada ima DFM avtentično oznako , ki zagotavlja da informacije o oznaki na poti ni bila spremenjena.

- **Število potrebnih paketov za izsledite IP naslova:** Če uporabimo $NB = 32$, DFM zahteva 2 paketa, če pa imamo $NB = 16$, DFM potrebuje 5 paketov za izsledite vozlišča napadalca [1]. Kar pa je manj od 8 paketov, ki jih potrebuje DPM za izsledite napadalčevega usmerjevalnika.

- **Vloga ponudnika storitev (ISP):** Kot DPM, je vpletenost ISP omejena. Samo robni usmerjevalnik mora biti posodobljen, tako da podpira funkcijo za deterministično označevanje paketov, tako za DPM kot za DFM.

- **Razdrobljenost:** Podobno kot DPM tudi DFM uporablja ID polje v IP glavi paketa, ki je uporabljena za deljenje podatkov. Torej, če je samo en paket označen, potem rekonstrukcija ne bo mogoča.

K preostalim prednostim DFM, ki smo jih že opisali dodajmo še eno unikatno lastnost, ki je ne najdemo v nobeni že znani metodi. Kar je omogočanje žrtvi, da najde vir napada, vse do izvirnega vozlišča znotraj LAN naslova in ne le do zadnjega usmerjevalnika, kot nam to omogoča večina metod za izsledite IP naslova. DFM predpostavlja, da vsako vozlišče v lokalnem omrežju lahko spremeni svoj IP naslov in da je možno filtrirati MAC naslov v robnem usmerjevalniku. Še več DFM predpostavlja, da napadalec lahko spremeni MAC naslov in v primeru, ko se to zgodi, DFM pride do treh nivojev nad vozlišče napadalca. Le v primeru, ko napadalec spremeni svoj MAC naslov z več obstoječimi naslovi z bele liste, DFM pride le do dva nivoja nad vmesnik izvornega omrežja v robnem usmerjevalniku.

Torej, kot opisano zgoraj, je uporaba autentične metode za označevanje v DFM metodi stvar izbire, izsledite je dobra tudi brez tega dodatka. Razlika nastane, ko je žrtev napadena in zahteva, verifikacijo ECDSA podpisa, kar pa obremeni napadalčev CPU in spomin.

# 5. ZAKLJUčEK

V tem delu sta podrobneje opisana dve metodi za izsledite IP naslova in sicer dobro znan "Deterministic Packet Marking" DPM metoda ter nova shema za izsledite IP naslova, predlagana od avtorejv [1] "Deterministic Flow Marking" DFM metoda, z vidika praktične in izvedljivosti. Uporabljena sta bila na bazi CAIDA Internet traces October 2012 dataset. Merjeno je bilo več parametrov, ki so pomembni pri uporabi posamezne metode. Merjena je bila obremenitev procesorja, obremenitev spomina, pasovna širina, stopnja izsledilјivosti

TABLE III. COMPARISON OF DPM AND DFM

| Comparison Metrics | DPM | DFM |
|---|---|---|
| Percentage of marked packets | 100% | If $K = 2$: 9.96%<br>If $K = 5$: 13.44% |
| Mark Spoofing by subverted routers | Yes | No |
| Maximum traceback ability | Up to the ingress interface of the edge router | Up to the attacker node |
| Mark Spoofing by Attacker | No | No |
| Computational Overhead on routers | Low | Fair |
| Computational Overhead on victim | Low | Fair |
| Memory Overhead on routers | Low | Low |
| Memory Overhead on victim | Low | Low |
| Bandwidth Overhead | None | Low |
| Tracdeback Rate | Good | Fair |
| False Positive Rate | Low, but the number of concurrent attackers is limited | Low |
| Number of required packets for traceback | 8 | 2 or 5 |
| ISP Involvement | Low | Low |
| Ability to handle Fragmentation | No | No |
| Ability to handle major DDoS attacks | Fair,<br>The Maximum Number of Concurrent Attackers is limited | Good |
| Number of Marking bits | 17 | If $K = 2$: 16<br>If $K = 5$: 32 |

**Figure 6: Primerjava DPM in DFM.**

, stopnja lažnih pozitivov, odziv na spreminjanje oznake s strani napadalca ali sprememba oznake z napadom usmerjevalnika na potki napada, število potrebnih paketov za izsleditev IP naslova napadalca, procent označenih paketov, ISP vpletenost v postopek, sposobnost odziva na razdrobljenost, sposobnost odziva na veliki DDoS napad in maksimalna sposobnost izsleditve. Tabela 6 je pregled vseh ocen obeh algoritmov za izsleditev IP naslova. Rezultati kažejo, da DFM zmanjšuje število potrebnih paketov pri natančnosti izsleditve 90% v povprečnem primeru brez lažnih pozitivov. Še več DFM onemogoči spreminjanje oznake in omogoča dostop do vozlišča izvora napada tudi, znotraj NAT, za obrambnim zidom ali proksi strežnikom.

Nadaljnjo delo bo raziskalo, kako najbolje vključiti DFM metoda za izsleditev IP naslovov v že obstoječe sisteme za obrambo pred vdorom ter požarne zidove.

## 6. REFERENCES


[1] Vahid Aghaei-Foroushani and A. Nur Zincir-Heywood, "On Evaluating IP Traceback Schemes: A Practical Perspective", 2013 IEEE Security and Privacy Workshops http://www.ieee-security.org/TC/SPW2013/papers/data/5017a127.pdf.

[2] S. Matsuda et al., "Design and implementation of unauthorized access tracing system," SAINT 2002, pp. 74–81, January/February 2002.

[3] A.C. Snoeren et al., "Single-packet IP traceback," IEEE/ACM Transactions on Networking, vol. 10 , No 6, pp. 721-734, Dec 2002.

[4] D.X. Song and A. Perrig, "Advanced and authenticated marking schemes for IP traceback"INFOCOM 2001,Vol.2,pp.878-886, Apr 2001.

[5] A. Belenky and N. Ansari, "IP Traceback with Deterministic Packet Marking," IEEE Commun. Lett., vol. 7, no. 4, pp. 162–64, April 2003.

[6] A. Belenky and N. Ansari, "On deterministic packet marking," Computer Networks: The International Journal of Computer and Telecommunications Networking,Vol.51. No.10, pp.2677-2700,Jul 2007

[7] S. Savage, D. Wetherall, A. Karlin and T. Anderson, "Network support for IP traceback," IEEE/ACM Transactions on Networking, Vol. 9, no. 3, pp. 226-237, June 2001.

[8] J. Li et al., "Large-scale IP traceback in highspeed Internet: practical techniques and theoretical foundation," IEEE/ACM Transactions on Networking, Vol. 16, no. 6, pp. 1253-1266, December 2008

[9] D. Johnson, A. Menezes and S. Vanstone, "The Elliptic Curve Digital Signature Algorithm (ECDSA)", Certicom Corporation, Dept. of Combinatorics Optimization, University of Waterloo, Canada, http://cs.ucsb.edu/ koc/ccs130h/notes/ecdsa-cert.pdf.

[10] The CAIDA Anonymized Internet Traces 2012 Dataset, http://www.caida.org/data/passive/passive_2012_dataset.xml.

# Vpliv vzorčenja na algoritme za detekcijo anomalij v omrežnem prometu

## [Predstavitev področja]

Tadej Jagodnik
Fakulteta za računalništvo in informatiko
Tržaška 25
1000, Ljubljana
tadej.jagodnik@gmail.com

Jan Češnjevar
Fakulteta za računalništvo in informatiko
Tržaška 25
1000, Ljubljana
jc8389@student.uni-lj.si

## POVZETEK

Vzorčenje se je pojavilo z namenom zmanjšanja zahtev po procesorski moči in spominu na usmerjevalnikih. Raziskave so pokazale, da vzorčenje popači omrežne statistike. Trenutno aktualne raziskave poskušajo ugotoviti, kakšen je vpliv vzorčenja na algoritme za detekcijo anomalij. V tem članku so podrobno opisane metode vzorčenja omrežnega prometa in algoritmi za detekcijo anomalij. Opisali smo njihovo delovanje in ocenili kako posamezne tehnike vplivajo na detekcijske algoritme. Dodatno smo opozorili na metrike, ki se jih raziskovalci morajo zavedati, da bodo tehnike med seboj pravično ocenjene. Raziskave so pokazale, da vzorčenje zelo poslabša performanse detekcijskih algoritmov. Ugotovili smo, da TAPS bolj uspešno odkriva scannerje kot TRW in da se za najboljšo izkaže tehnika vzorčenje z izbiranjem. Na koncu smo opisali tehniko Online Selective Sampling, ki je nastala zaradi dobrih rezultatov vzorčenja z izbiranjem.

## Termini

Algoritmi, Performanse

## Ključne besede

vzočenje, TAPS, TRW, DWT

## 1. UVOD

Nadzor in analiza omrežnega prometa sta dandanes bistvena za učinkovito upravljanje omrežij in omrežne varnosti [5]. Razumevanje dogajanja na omrežju je vse bolj zahtevno zaradi nenehnih novih aplikacij in računalniških napadov [7]. Za zmanjšanje zahtev po procesorski moči in spominu na usmerjevalikih ter omilitve ogromne porabe pasovne širine na omrežju za prenos podatkov je bilo uvedeno vzorčenje pri merjenju omrežnega prometa [4].

Znano je, da vzorčenje paketov popači omrežne statistike. Toda nejasno je, kako vzorčenje vpliva na detekcijo anomalij. Anomalije lahko razvrstimo v dve skupini. V prvo skupino spadajo anomalije, ki se odražajo kot nenadne spremembe v količini paketov ali tokov (angl. volume based), kot npr. Denial-of-Service (DOS) [4] napad. V drugo skupino uvrščamo njihovo nasprotje, skeniranje (tipično predhodnik virusov ali črvov).

Za detekcijo teh groženj so bili razviti algoritmi in sistemi za detekcijo in preprečevanje napadov (intrusion detection / prevention systems), kot sta Bro IDS in Snort. Namesto omenjenih sistemov pa se članek osredotoča na algoritme Threshold Random Walk (TRW), Time Access Pattern Scheme (TAPS) in Entropy-Based Scan Detection, ki se uporabljajo za detekcijo napadov s skeniranjem, in Discrete Wavelet transform (DTW) za drug tip anomalij. Nadalje bomo pogledali, kako različne tehnike vzorčenja omrežnega prometa (tehnika vzorčenja paketov, vzorčenja tokov in druge), vplivajo na performanse algoritmov za detekcijo napadov (TRW, TAPS, Entropy-Based, DTW).

Glavno vprašanje, ki se poraja, je [4]: ali vzorčeni podatki zajamejo dovolj informacij za učinkovito detekcijo anomalij?

Zgradba članka je sledeča. Drugo poglavje prikazuje sorodna dela. Tretje poglavje podrobno opiše tehnike vzorčenja in algoritme za detekcijo anomalij ter opredeli testno množico in metrike ocenjevanja. Ocena vpliva vzorčenja na algoritme za detekcijo anomalij je prikazana v četrtem poglavju. Peto poglavje vsebuje predstavitev tehnike Online Selective Sampling.

## 2. SORODNA DELA

Na tem področju je napisanih veliko člankov ([1], [2], [4], [5], [6], [7], [8]). Vsi članki raziskujejo vpliv vzorčenja na algoritme za detekcijo anomalij.

Članek [4] natančno obrazloži naslednje tehnike: vzorčenje paketov, vzorčenje tokov, pametno vzorčenje in vzorči in zadrži. Nadalje opisuje obe skupini anomalij, omenjenih v uvodu, ter algoritme DWT, TAPS in TRW. V zaključku članka so predstavljeni rezultati algoritmov pod vzorčenjem, pri čemer je uporabljena metrika *enak delež paketov*, ki je razložena v podpoglavju *Doseganje pravične delitve paketov.*

Članek [5] poleg algoritmov TAPS in TRW opisuje in testira vpliv vzorčenja na algoritem za detekcijo anomalij na

podlagi entropije.

Članka [6],[7] predstavljata osnovno izhodišče, iz katerega izhaja ta članek.

# 3. OZADJE IN TEHNIKE

Napadi s skeniranjem (angl. scanning attack, portscan) so običajno predhodniki računalniških zločinov (črvi) in predstavljajo več kot 80% računalniških napadov na internetu [7]. Za identifikacijo scannerjev so pomembne informacije, kot so vzorci dostopa, statusi povezav, obnašanje izvora v malih tokovih in podobno [4].

## 3.1 Tehnike vzorčenja

Obstaja več tehnik vzorčenja omrežnega prometa. Izbrani članki [4],[7] obravnavajo tehnike vzorčenje paketov (angl. packet sampling - PS), vzorčenje tokov (angl. flow sampling - FS), pametno vzorčenje (angl. smart sampling - SS), vzorči in zadrži (angl. sample-and-hold - S&H) in vzorčenje z izbiranjem (selective sampling - SES). Najslabše rezultate dosegata SS in S&H. Slednja tehnika ni testirana v tem članku, pregled njene uspešnosti je opisan v [4]. Zaradi najboljših rezultatov tehnike SES je v članku [7] predlagana nova tehnika Online Selective Sampling (OSES).

**Vzorčenje paketov (packet sampling)** vzorči vsak paket z majhno verjetnostjo $r < 1$. Nato vzorčen promet združi v tokove glede na izvorni IP naslov, ponorni IP naslov, vhodna vrata, ponorna vrata in protokol. Tokove opustimo po preteku 1 minute ali pa glede na semantiko TCP protokola [4].

**Vzorčenje tokov (flow sampling)** najprej razvrsti pakete po pravilih konstrukcije tokov, ki so omenjeni zgoraj. Nato vzorči vsak tok z neko verjetnostjo $p < 1$ [4].

**Pametno vzorčenje (smart sampling)** vzorči tok, dolžine $x$ bajtov z verjetnostjo $p_z(x) = min\{1, x/z\}$, kjer je $z$ prag vzorčenja. Tokovi, katerih velikost je večja od $z$ so vedno izbrani, manjši tokovi pa so izbrani z verjetnostjo, ki je v sorazmerju z njihovo velikostjo [1]. Ta pristop vedno izbira velike tokove. Je nekakšen kompromis med vzorčenjem paketov in vzorčenjem tokov.

**Vzorči in zadrži (sample-and-hold)** [2] za vsak paket pogleda: ali v tabeli obstaja pripadajoč zapis toka, kateremu paket pripada. Če ga najde, se zapis posodobi (poveča velikost števca v zgoščevalni tabeli). V nasprotnem primeru je paket naključno vzorčen z verjetnostjo $h_s$ in zapis dodan v tabelo. Naj bo $h$ verjetnost, s katero vzorčimo bajt. Torej je verjetnost vzorčenja paketa velikosti $s$ enaka $h_s = 1 - 1(1-h)^s$, kar je približno $h \cdot s$. Izbira velikega $h$ zagotavlja, da bodo tokovi nad pragom zelo verjetno detektirani. Zelo razumljiv opis z grafom se nahaja v članku [2].

**Vzorčenje z izbiranjem (selective sampling)** [8] vzorči tok po sledeči formuli:

$$p(x) = \begin{cases} c & x \leq z \\ z/n \cdot x & x > z \end{cases}$$

Spremenljivka $x$ predstavlja velikost toka (v paketih), $0 < c \leq 1$ predstavlja verjetnost vzorčenja malega toka (v paketih), $n \leq 1$ je parameter za regulacijo deleža vzetih velikih paketov in $z$ je prag (velikost malega toka v številu paketov). Tokovi, ki so manjši kot $z$ so vzorčeni z verjetnostjo $c$, večji tokovi pa z verjetnostjo, ki je inverzno proporcialna njihovi velikosti. Tehnika se osredotoča na majhne tokove.

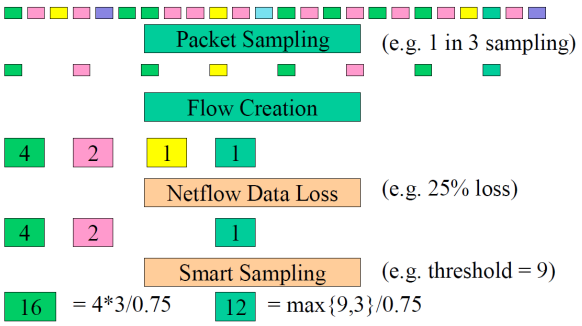



**Figure 1: Potek vzorčenja paketov.**

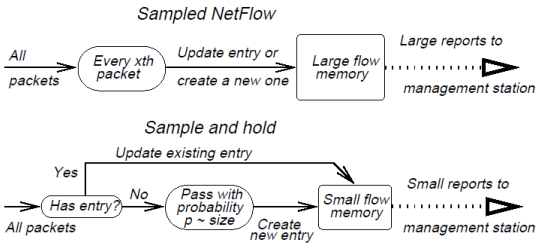



**Figure 2: Razlika med vzorčenjem paketov in vzorči in zadrži.**

## 3.2 Algoritmi za detekcijo anomalij

### 3.2.1 *Za detekcijo scannerjev*

Predlaganih je bilo veliko metod za odkrivanje anomalij. Najbolj direkten pristop identificira scanner, ko se poveže z več kot določenim številom ponorov v nekem točno določenem časovnem intervalu [7]. Ta način uporabljata Bro in Snort. Obstajajo še drugi algoritmi, ki so bolj učinkoviti.

**Threshold Random Walk (TRW) in TRWSYN** deluje na podlagi hipotez. Predpostavlja, da nenevaren gostitelj vzpostavi več uspešnih povezav kot scanner, ki naključno raziskuje naslovni prostor. Torej $H_0$ predstavlja nenevarnega gostitelja in $H_1$ predstavlja scanner. Naj bo $Y = \{Y_1, Y_2, ..., Y_i\}$ naključni vektor povezav, ki jih opazujemo iz izvora. $Y_i = 0$, če je i-ta povezava uspešno vzpostavljena in $Y_i = 1$ v nasprotnem primeru. Za vsako vrednost $Y_i$ se razmerje

$$\Lambda(Y) = \prod_{i=1}^{n} \frac{P_r[Y_i|H_1]}{P_r[Y_i|H_0]}$$

posodobi. Ko $\Lambda(Y)$ preide enega izmed pragov, je izbrana ustrezna hipoteza [4]. Ker je združen promet običajno enosmeren je težko ugotavljati, kdaj je povezava uspela in kdaj

ni. Zato se je pojavila prilagoditev TRWSYN, ki predpostavi, da je bila povezava uspešna, če je v toku več kot en SYN paket.

**Time Access Pattern Scheme (TAPS)** [6],[4] opazuje vzorce dostopa. Sklepa, da scanner običajno vzpostavlja povezave z večjo množico ponornih IP naslovov ali vrat. Torej je razmerje $y$ med ponornimi IP naslovi in številkami portov, ali obratno, večje za scanner kot za nenevarnega gostitelja. Ko to razmerje preseže določen prag, se razmerje izvornega IP posodobi. Ko združena vsota doseže določeno mejo, izvor predpostavi kot scanner. Algoritem v vsakem časovnem oknu za vsak izvor izračuna razmerje $y$ in ga primerja s pragom $k$. Časovna spremenljivka $Y_i$ je v tem časovnem oknu postavljena na 0 ali 1 glede na to, ali $y$ leži nad ali pod pragom. Nato se razmerje posodobi kot pri TRW. Spremenljivka $k$ vpliva na natančnost detekcije, časovno okno pa nadzoruje točnost pri odločevanju.

**Entropy-based Scan Detection** [5] temelji na ugotovitvi, da večina anomalij in ostalih zanimivih dogodkov povzroči spremembo vzorca v distribuciji prometa, kar lahko zazna vzorčna entropija (angl. sample entropy). Vzorčna entropija je mera regularnosti. Za dano dimenzijo $m$, toleranco $r$ in število podatkovnih točk $N$ je entropija negativni logaritem verjetnosti, da če imata dve zaporedni množici podatkov dolžine $m$ razdaljo $< r$ imata tudi dve zaporedni množici dolžine $m+1$ razdaljo $< r$. Naj bo $X$ naključna spremenljivka, ki jo opazujemo $N$ krat in vzame vrednosti iz množice $\{x_i\}$ kardinalnosti $M$. Spremenljivka $n_i$ predstavlja število pojavitev vrednosti $x_i$ znotraj opazovalnega okna. Vzorčno entropijo izračunamo po formuli

$$H(X) = -\sum_{i=1}^{M} p(x_i) \log_2 p(x_i),$$

kjer je $N = \sum_{i=1}^{M} n_i$ (vsota števil pojavitev posameznih $x_i$) in $p(x_i) = n_i/N$ (verjetnost vzorčenja) za $X = x_i$. Vrednost $H(x)$ je v meji $0 \leq H(x) \leq \log_2 M$. Če $X$ vedno zavzame isto vrednost je $H(X) = 0$ in če je $X$ uniformno porazdeljen $(p(x_i) = 1/M, i = 1, ..., M)$ je vrednost $H(X) = \log_2 M$. $X$ predstavlja število tokov z določeno lastnostjo ali dimenzijo. Tipične dimenzije so število tokov za dan izvorni IP, ponorni IP, vhodna vrata, ponorna vrata. Članka [9], [3] uporabljata isto definicijo vzorčne entropije, a uporabljata različna pristopa obdelave omrežnih podatkov.

V članku [3] so avtorji sestavili entropijske časovne vrste po štirih dimenzijah (izvorni IP naslov, ponorni IP naslov, vhodna vrata, ponorna vrata) na širokem omrežnem prometu. Nato so s pomočjo analize PCA (angl. Principal Component Analysis) razkrinkali nenavadne vzorce v obnašanju prometa. Medtem so avtorji v članku [9] analizirali komunikacijske vzorce za pomembne gruče (angl. significant clusters). Te gruče so definirane kot entitete, ki ogromno prispevajo pri porazdelitvi vzorcev po vsaki dimenziji.

Opisali bomo metodo članka [9] in smernice za uporabo entropije pri detekciji napadov s skeniranjem. Za izločanje pomembnih gruč naprej množico $\{x_i, 1 \leq i \leq M\}$ razvrstimo v padajočem redu po $\{x_i\}$ glede na verjetnost distribucije $p(x_i)$. Relativna negotovost (RU) definiramo kot:

$$RU(X) = \frac{H(X)}{H_{max}(X)} = \frac{-\sum_{i=1}^{M} p(x_i) \cdot \log p(x_i)}{\log_2 |M|}.$$

RU je dejansko relativna entropija in normalizira stopnjo uniformnosti porazdelitve. To pomeni $0 \leq RU(X) \leq 1$ z $RU(X) = 0$, če $X$ zavzame eno samo vrednost in $RU(X) = 1$, če ima $X$ uniformno porazdelitev čez celoten vzorčni prostor. Vrhnji $\{x_i\}$ so izločeni eden za drugim in označeni kot pomembne gruče dokler ima preostanek množice $X_r$ $RU(X_r)$ blizu 1. To pomeni, da je uniformno porazdeljena in tako nepomembna.

Gruče so nato razporejene glede na podobnosti in različnosti komunikacijskih vzorcev v razrede obnašanja (angl. behavior classes). Za vsako pomembno gručo, po dimenziji izvorni IP naslov, izračunamo relativno negotovnost izvornih vrat, relativno negotovnost ponornih vrat, relativno negotovnost ponornih IP in pripnemo vrednosti na tri stopnje: 0 (nizko), 1 (srednje) in 3 (visoko) označenih z $L_{RU}$. Te gruče lahko razporedimo v različne razrede obnašanja, predstavljene z vektorjem

$$[L_{RU(izvornaVrata)}, L_{RU(ponornaVrata)}, L_{RU(ponorniIP)}].$$

Primer: izvori, ki so podobni razredom obnašanja oblike $[*, 2, 0]$ so skenerji, ki pregledujejo veliko število naključnih portov na malo IP naslovih. Izvori, ki so podobni razredom obnašanja oblike $[*, 0, 2]$ so skenerji, ki pregledujejo malo število naključnih portov na veliko različnih IP naslovov.

Dodatne informacije, kako je entropija uporabljena pri vzorčenju, opisuje članek [5].

### 3.2.2 *Za detekcijo anomalij z nenadnimi spremembami v količini paketov*

Nenadne spremembe paketov so najbolj pogost pokazatelj različnih anomalij omrežnega prometa in povzročajo tako imenovane "volume" anomalije. Za njihovo zaznavanje oziroma detekcijo se uporablja postopek, ki temelji na diskretni valovni transformaciji (angl. discrete wavelet transformation - DWT).

**Discrete wavelet transform (DWT)** je postopek transformacije, kjer so valovi vzorčeni diskretno. Prednost diskretne vzorčne transformacije je predvsem zajem informacije o frekvenci (angl. frequency) in času. Torej gre za transformacijo diskretnega signala iz časovne domene v domeno frekvence in časa (angl. time-frequency). Rezultat takšne transformacije so koefincienti, ki omogočajo spektralno analizo signala in časa. Slednje je mogoče doseči s postopkom dekompozicije (angl. decomposition), ki razdeli vhodni signal na dve komponenti. Tipično se v postopku dekompozicije uporabljata nizkopasovni filter (angl. low pass filter) $g$ in visokopasovni filter (angl. high pass filter) $h$. Signal filtriran s pomočjo nizkopasovnega filtra vsebuje informacije o počasnih spremembah signala in je pod izvornim signalom, le da je dvakrat krajši v smislu števila vzorcev. Signal filtriran z visokopasovnim filtrom pa vsebuje informacije o hitrih spremembah signala. Kot je bilo že rečeno, so rezultat takšne transformacije koeficenti. Z uporabo nizkopasovnega filtera tipično pridobimo približne (angl. approximation)

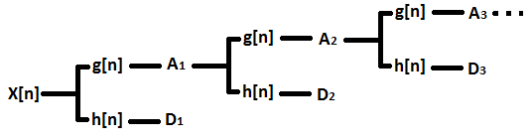



**Figure 3: Postopek dekompozicije (prikazano do tretjega nivoja).**

koeficente, z uporabo vioskopasovnega filtra pa podrobne (angl. detail) koeficente.

Algoritem za detekcijo uporablja DWT v povezavi s paketi in časovnimi vrstami hitrosti toka (angl. flow) na različnih časovnih intervalih. Razdeljen je na tri dele: dekompozicije časovnih vrst, ponovne sinteze (angl. resynthesis) in zaznave oziroma detekcije.

V fazi dekompozicije se vhodni signal razstavi tako, da je mogoče zaznati spremembe na različnih časovnih intervalih. Originalni vhodni signal $X[n]$ dolžine $N$, gre skozi visokopasovni realni valovni filter (angl. high pass real-valued wavelet filter) $h_l$ lihe dolžine $L$. Izhod slednjega filtra je množica valovnih koeficientov $W_{1,t}$ na originalnem časovnem intervalu dolžine $\frac{N}{2}$. Na podoben način pridobimo množico valovnih filtrov $V_{1,t}$, dolžine $\frac{N}{2}$, ki so izhod nizkopasovnega filtra $g_l = (-1)^{l+1} + h_{L-1-l}$. Izvedba inverzne diskretne vzorčne transformacije neodvisno nad množico koeficentov $W_{1,t}$ in $V_{1,t}$ omogoča izločitev približkov prvega nivoja $A_1$ in podrobnosti prvega nivoja $D_1$. Postopek dekompozcije se ponavlja tako, da se vzame $A_1$ kot vhodni signal, kar vrne približke drugega nivoja $A_2$ in podrobnosti drugega nivoja $D_2$, itd (glej sliko Figure 3). Maksimalen nivo DWT je enak $j \leq \log_2 N$, kjer $j$ na vsakem nivoju predstavlja moč frekvence signala – večja kot je vrednost $j$, manjša je frekvenca signala.

Po končani dekompoziciji nastopi faza ponovne sinteze. Različne frekvence $j$ iz različnih nivojev dekompozicije se združijo v nizke, srednje in visoke pasove. Nizek pas je sestavljen iz vsote višjih nivojev dekompozicije:

$$X_{low} = \sum_{i=l}^{j} D_i + A_j,$$

visok pas je sestavljen iz vsote nižjih nivojev dekompozicije:

$$X_{high} = \sum_{i=1}^{h} D_i,$$

srednji pas pa je sestavljen iz vsote preostalih nivojev dekompozicije. Zaznava oziroma detekcija anomalij je tretja in zadnja faza algoritma. Izračuna se lokalna varianca ali odmik visoko in srednje pasovnih signalov na časovnem intervalu, ki je določen z drsečim oknom (angl. slide window). Nato sledi izračun odklona (angl. deviation score), ki predstavlja razmerje med lokalno varianco v oknu in globalno varianco. Okno, ki ima višji odklon kot vnaprej določen prag (angl. threshold) vsebuje anomalije z nenadnimi spremembami v količini paketov oziroma "volume" anomalija.

## 3.3 Splošna resnica

| Trace | Start Time | Duration | Flows | Packets | Bytes |
|---|---|---|---|---|---|
| *dataset-1* | 03:00 | 60 min. | 5.5M | 59.5M | 39.6G |
| *dataset-2* | 09:00 | 60 min. | 7.9M | 128.3M | 95.1G |
| *dataset-3* | 15:00 | 60 min. | 8.6M | 136.2M | 101.5G |
| *dataset-4* | 21:00 | 60 min. | 6.5M | 94.3M | 67.4G |

**Figure 4: Zajet promet iz članka [7].**

Da lahko analiziramo posledice vzorčenja na algoritme za detekcijo anomalij, je potrebno pridobiti podatke, s katerimi bomo lahko primerjali dobljene rezultate. Vzeli bomo neko testno množico. Na tej testni množici bomo pognali algoritme za detekcijo anomalij z optimalno nastavljenimi parametri. TRW in TAPS bomo nastavili, da bo verjetnost napačne klasifikacije (nescanner opredeljen kot scanner, čeprav to ni - false positive) 0.01, verjetnost detekcije 0.99, verjetnost uspešne povezave za scanner 0.2 in verjetnost uspešne povezave za nescanner 0.8. Parametra $k$ in $t$ smo za TAPS prilagajali za različne frekvence vzorčenja, tako da dobimo optimalne rezultate [7]. Rezultat izvedbe algoritma nad testno množico je široka množica scannerjev. Z uporabo orodja FIM (frequent item-set mining) za podatkovno rudarjenje se preveri, ali so dobljeni scannerji zares zlonamerni računalniški napadi.

FIM deluje na podlagi iskanja pogostih korelacij med elementi. Uporaben je pri detekciji anomalij, saj se v primeru napada veliko lastnosti prometa z istimi vrednostmi pojavi skupaj v veliki količini prometnih tokov. Npr. pri vertikalnem skeniranju ima ogromno prometa isti izvorni in ponorni IP naslov, kar FIM zlahka odkrije [7]. Končna množica nam predstavlja **splošno resnico (ground truth)**.

Na koncu se preverijo scannerji iz splošne resnice in scannerji, dobljeni iz vzorčenja. Uspešnost opredelimo z dvema metrikama: **uspešnost (angl. success ratio - SR)** in **razmerje napačno klasificiranih (angl. false positive - FP)**.

$$SR = \frac{resnicni\ scannerji\ pod\ vzorcenjem}{vsi\ scannerji\ iz\ splosne\ resnice}$$

$$FP = \frac{detektirani\ scannerji,\ ki\ to\ niso}{vsi\ scannerji\ iz\ splosne\ resnice}$$

## 3.4 Doseganje pravične delitve paketov

Frekvenca vzorčenja ima drug pomen pri vzorčenju paketov kot pri vzorčenju tokov. Pri vzorčenju paketov pomeni delež vzorčenih paketov, pri slednjem pa delež vzorčenih tokov.

Npr. če vzorčimo 10% tokov iz množice 3 (dataset-3), PS dobi 3.01% paketov, FS 9.2%, SMS 85.35% in SES 0.66%. Ko vzorčimo 10% paketov, PS dobi 24.21% tokov, FS 11.08%, SMS 0.0062% in SES 78.15% [7].

Razlog, da isti delež tokov ustreza tako malemu številu paketov za PS, je v tem, da je verjetnost vzorčenja paketa iz istega toka zelo majhna, ker so tokovi majhni. Da lahko primerjamo metode, je torej potrebno uporabiti dve metriki[6]:

**Enak delež paketov**. Prvotna izračunana frekvenca vzorčenja za PS je bila $1/N$. Za ohranitev frekvence vzorčenja

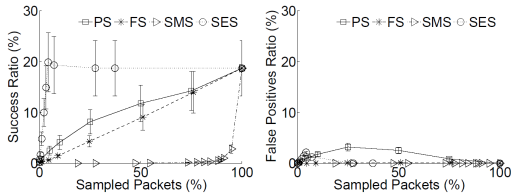



**Figure 5: Vpliv vzorčenja na algoritem TRW [7].**

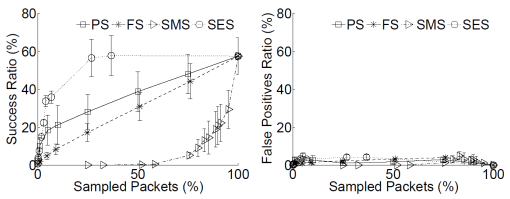



**Figure 6: Vpliv vzorčenja na algoritem TAPS [7].**

je potrebno opraviti številne teste, da se odkrijejo primerne frekvence vzorčenja za ostale tehnike vzorčenja, tako da bo delež izbranih paketov enak.

**Enak delež tokov**. Prvotna izračunana frekvenca vzorčenja tokov je $1/N$. Za dan delež tokov, ki jih je potrebno sprejeti, je potrebno opraviti številne teste, da dobimo ustrezno frekvenco vzorčenja, da bomo vzorčili enak delež tokov.

Tipa analize (enak delež paketov, enak delež tokov) sta si komplementarna in nam pomagata razumeti obnašanje algoritmov za detekcijo anomalij pod vzorčenjem.

# 4. OCENJEVANJE PERFORMANS

## 4.1 Vpliv vzorčenja na algoritem TRW

TRW brez vpliva vzorčenja zna zaznati le okoli 20% vseh scannerjev. Vzorčenje pa še dodatno poslabša njegove performanse.

**Uspešnost** vseh tehnik vzorčenja, z izjemo vzorčenja z izbiranjem (SES), se izjemno poslabša. Za izjemno slabo se izkaže tehnika SMS, pri kateri uspešnost (SR) doseže 0% že ob 75% vzorčenih tokov. Druga najslabša tehnika je vzorčenje tokov (FS), nato sledi vzorčenje paketov (PS).

Da PS dosega boljše rezultate od FS je v nasprotju z ugotovitvijo članka [4]. Razlika je v izbiri metrike. Analiza v tem članku je opravljena nad enakim deležem paketov, v članku [4], pa nad enakim deležem tokov. Torej, da je PS boljši od FS ali obratno, ne moremo trditi. Vse je odvisno od izbire metrike. Skupno obema tehnikama je linearno padanje performans s frekvenco vzorčenja.

Edina tehnika, ki ne kaže bistvenega poslabšanja, je vzorčenje z izbiranjem (SES). SES dobro deluje pod TRW, ker se osredotoča na majhne tokove, ki so ravno tisti, ki jih TRW išče [7]. SES je uspešnejši, kot nevzorčen primer, ker zavrača velike tokove, in vodi TRW v scenarij, kjer večina gostiteljev generira tok z enim samim SYN paketom [7].

**Stopnja napačno klasificiranih**[7] scannerjev je pri vseh tehnikah zelo nizka. Izjema je vzorčenje paketov (PS), zaradi efekta $flow - shortening$, ki je opisan v [5]. Pri SES se lahko zgodi nerealističen scenarij, kjer večina gostiteljev pošilja zgolj tokove z enim SYN paketom (TRW ne zazna nobenega). Kljub temu SES ohranja stopnjo napačno klasificiranih scannerjev pod 2% za vse frekvence vzorčenja. FS in SMS ne proizvedeta skoraj nič napačno klasificiranih scannerjev. Prvi zaradi ohranjanja porazdelitve velikosti toka, drugi pa zaradi sistematičnega upada malih tokov.

## 4.2 Vpliv vzorčenja na algoritem TAPS

Rezultati pri uporabi algoritma TAPS so veliko boljši kot pri TRW. TAPS zna brez vzorčenja odkriti 57.47% scannerjev iz splošne resnice. Podobno kot pri TRW se performanse močno poslabšajo, ko je uvedeno vzorčenje. Za najboljšo tehniko se znova izkaže SES, sledijo ji PS, FS in SMS. Iz obeh algoritmov je razvidno, da se tehnike vzorčenja, ki se osredotočajo na majhne tokove, odrežejo bolje kot tiste, ki se osredotočajo na velike tokove. Zopet je PS uspešnejši od FS, v nasprotju z ugotovitvami članka [4]. Vse tehnike proizvedejo majhno število napačno klasificiranih scannerjev.

## 4.3 Vpliv vzorčenja na algoritem za detekcijo skeniranj s pomočjo entropije

Rezultati iz članka [5] nakazujejo, da je zanesljivost detekcije skeniranj s pomočjo entropije močno poslabšana pri vzorčenju. V članku [5] so analizirali omrežne tokove preko štirih dimenzij: izvorni IP naslov, ponorni IP naslov, izvorna vrata, ponorna vrata. Odkrili so, da je vrednost entropije in relativne negotovosti pod vzorčenjem višja, kot pri originalnem toku. Izkazalo se je, da vrednost relativne negotovosti monotono narašča z vzorčnim intervalom. Npr. pri vzorčenju enega paketa izmed petstotih je relativna negotovost 0.96, kar je 20% zvišanje glede na izvorno vrednost. Hkrati pa naraščanje relativne negotovosti pomembno vpliva na identifikacijo pomembnih gruč.

Vemo, da vzorčenje paketov preferira velike tokove. Entropija, ki opisuje porazdelitev tokov skozi dimenzije (IP, vrata) se bo spremenila, saj vzorčenje paketov popači porazdelitev. Vsak tok večji od $1/p$ paketov, t.j. 10 paketov za $p = 0.1$, bo imel vzorčen vsaj en paket in verjetnost zajetja toka bo velika. Nasprotno bodo mali tokovi vzorčeni z verjetnostjo, ki bo proporcionalna njihovi velikosti. Entropija porazdelitve je manjša od 1, ker je večina tokov manjših in samo majhen delež je večjih. Z zvišanjem vzočnega intervala je porazdelitev vedno bolj odvisna velikih tokov.

## 4.4 TAPS proti TRW

Ključna ugotovitev je, da TAPS zazna več scannerjev kot TRW in da je bolj robusten na vzorčenje. Glavna razlika je v tem, da je TRW zelo odvisen od specifičnega paketa (SYN paket), medtem ko algoritma TAPS ne zanima, kateri paket je vzet iz toka, temveč kakšen je vzorec dostopanja do vsakega gostitelja. Zaradi teh razlogov je TRW zelo občutljiv, če je določen paket zavržen. Druga prednost TAPS je, da zazna UDP in TCP skeniranja, medtem ko TRW zazna samo TCP [7].

## 4.5 Vpliv vzorčenja na algoritme za detekcijo anomalij z nenadnimi spremembami v količini paketov

| Sampling interval | 10 | 100 | 1000 |
|---|---|---|---|
| Percentage of flows (%) | 36.7 | 8.03 | 1.47 |
| Random packet sampling | 19 | 6 | 1 |
| Random flow sampling | 21 | 18 | 13 |
| Smart sampling | 18 | 1 | 1 |
| Sample-and-hold | 18 | 2 | 1 |

**Figure 7: Število detektiranih anomalij, za različne tehnike vzorčenja, na različnih intervalih.**

Podatki, uporabljeni za merjenje detekcij anomalij z nenadnimi spremembami v količini paketov so originalne časovne vrste in vzorčene časovne vrste. Vzorčna analiza je bila opravljena nad detekcijo anomalij na orginalnih časovnih vrstah. Njen rezultat služi za primerjavo z detekcijo anomalij na vzorčenih časovnih vrstah. Opravljen je bil eksperiment na 17-urnem intervalu paketov. Dekompozicija signala je potekala s pomočjo 12 nivosjke diskretne valovne transformacije DWT. V fazi ponovne sinteze je bil $h$ nastavljen na 5 in $l$ na 10 zato, da visokofrekvenčni signali nosijo informacije o kratkočasnih spremembah, srednjefrekvenčni signali nosijo informacije o spremembah na intervalu 1 minute in nizkofrekvenčni signali nosijo informacije o spremembah na intervalu 15 minut. V zadnji fazi oziroma fazi detekcije je bila velikost okna za izračun lokalne variance nastavljena na 1 minuto (interval). Prag je bil nastavljen na 4.5 zato, da je bila zagotovljena tudi detekcija na originalnih časovnih vrstah.

Za primerjavo je uporabljen podatek števila detektiranih anomalij na originalnih časovnih vrstah, ki je 21. Detekcija anomalij je bila izvedena s pomočjo različnih tehnik vzorčenja. V tabeli Figure 7 je prikazano število detektiranih anomalij pod različnimi tehnikami vzorčenja. Analizirani so bili naključni paketi na interavlih 10, 100 in 1000 za vsako posamezno tehniko vzorčenja posebej. Rezultati kažejo, da se število detekcij zmanjšuje z večanjem intervala, kar je tudi pričakovano. Najboljše rezultate detekcij je dosegla tehnika vzorčenja tokov, in sicer na vseh treh intervalih. Pri preostalih tehnikah vzorčenja, ki so se odrezale slabše, opazimo občutno zmanjšanje števila detekcij z večanjem intervala.

# 5. ONLINE SELECTIVE SAMPLING - OSES

Problem vzorčenja z izbiranjem (SES) je v tem, da je potrebno pred samim vzorčenjem celotnega toka najprej zajeti vse pakete. Kar pomeni, da morajo biti ti tokovi shranjeni v pomnilniku dokler ne potečejo. Takšno delovanje ni v skladu z glavnim ciljem vzorčenja, ki je zmanjševanje porabe virov (angl. resources). Predlagan pristop OSES se osredotoča na isti tip prometa vendar ne zajame celotnega toka, ker sprejema odločitve na nivoju posameznega paketa. Posledica je predvsem manjša poraba virov in zmožnost delovanja na spletu (angl. online).

OSES temelji na vzdrževanju toka oziroma vzorčenju vseh njegovih paketov, dokler je še majhen. Če je majhen tok vsebuje $x$ paketov, kjer je $x \geq 1$, potem bo OSES vzorčil vse tokove, ki imajo največ $x$ paketov z visoko verjetnostjo. Vse pakete, ki imajo več paketov kot $x$, pa bo vzorčil z manjšo verjetnostjo (več paketov, manjša verjetnost vzorčenja).

Najbolj enostavna rešitev je hranjenje vseh tokov v zgoščevalni tabeli (angl. hash table). Ko tokovi potečejo, se odstranijo iz zgoščevalne tabele z določeno verjetnostjo. Ta rešitev porabi veliko virov (predvsem pomnilnika) in ni dovolj hitra pri povezavah visokih hitrosti. Boljša je uporaba Bloomovih filtrov, predvsem zaradi njihove izvedljivosti. Prednost Bloomovih filtrov je predvsem hiter čas izvedbe iskanja in zelo majhna poraba pomnilnika. V tej rešitvi se tokovi tudi hranijo v zgoščevalni tabeli, vendar se v njej hranijo le dokler jih OSES ne zavrže, ali dokler ne potečejo.

Cilj OSES je isti kot pri SES, vendar brez zajemanja celotnega toka. Želja je, da je verjetnost vzorčenja toka po posameznih paketih ista kot pri vzorčenju celotnega toka. Zato se mora verjetnost $p$ prilagoditi na vsakem koraku tako, da bo skupna verjetnost točno enaka kot, če bi vzorčili celoten tok z SES. OSES deluje tako, da za vsak dohodni paket z uporabo Bloomovih filtrov preveri ali je bil njegov tok že zavržen. Če ugotovi, da je bil njegov tok zavržen, potem zavrže tudi paket. V obratnem primeru najprej preveri verjetnost $p$. Če je vrednost $p$ manjša od določene naključne verjetnosti potem paket zavrže, njegov tok se odstrani iz zgoščevalne tabele in v Bloomove filtre se doda 5-terica (angl. tuple) toka. Če pa je vrednost $p$ večja, se paket vzorči.

Postopek izračuna verjetnosti $p$ je naslednji. Če sta $p_x^{SES}$ in $p_x^{OSES}$ verjetnosti za vzorčenje toka, ki vsebuje $x$ paketov, potem je cilj:

$$p_x^{SES} = p_x^{OSES}$$

.

Verjetnost SES je odvisna od ene same naključne odločitve, verjetnost OSES pa od akumulacije $x$ naključnih neodvisnih odločitev (ena odločitev za vsak paket v toku):

$$p_x^{OSES} = \prod_{i=1}^{x} r_i^{OSES}$$

Iz predhodnih dveh enačb je razvidno, da je verjetnost vzorčenja $x$ paketov v toku:

$$p_x^{SES} = p_x^{OSES} = \prod_{i=1}^{x} r_i^{OSES} \rightarrow \frac{p_x^{SES}}{\prod_{i=1}^{x-1} r_i^{OSES}} = \frac{p_x^{SES}}{p_{x-1}^{OSES}}$$

Verjetnostni funkciji za SES in OSES sta:

$$p_x^{SES} = \begin{cases} c & x \leq z \\ z/n \cdot x & x > z \end{cases}$$

$$p_{x-1}^{OSES} = \begin{cases} c & x \leq z+1 \\ z/n \cdot (x-1) & x > z+2 \end{cases}$$

Verjetnost vzorčenja i-tega paketa za OSES($r_i^{oses}$) izračunamo kot:

$$r_i^{OSES} = \begin{cases} c & i = 1 \\ 1 & 1 < i \leq z \\ \frac{z}{c \cdot n_{\cdot i}} & i = z+1 \\ \frac{i-1}{i} & i > z+1 \end{cases}$$

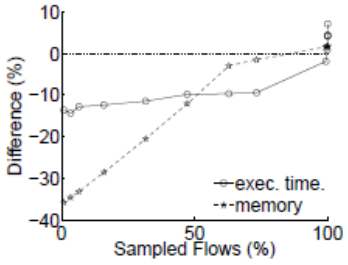



**Figure 8: Razlika v izvajalnem času in porabi pomnilnika med OSES in SES pristopoma.**

Promet vzorčen s strani OSES pristopa je ekvivalenten SES pristopu. Ocenjevanje uspešnosti je temeljilo na primerjavi odstotka vzorčenih tokov in paketov, povprečni velikosti toka in distribuciji velikosti toka. Rezultati OSES pri $c = 0.9$, $z = 2$ in $n = 1$ so bili: 76.53% vzorčenih tokov, 11.99% vzorčenih paketov in povprečna velikost toka 1.68 paketov. SES je dosegel naslednje rezultate: 76.54% vzorčenih tokov, 11.99% vzorčenih paketov in povprečna velikost toka 1.68 paketov. OSES je dosegel skoraj identične rezultate verjetnosti, kot SES. Razlika med SES in OSES pristopom je zanemarljiva. Minimalne razlike se pojavijo zgolj zaradi naključnih odločitev. Ugotovljeno je bilo, da je bil OSES pristop ekvivalenten SES, brez zahteve po zajemu celotnega toka. OSES je dosegal boljše rezultate od SES glede na čas izvajanja in glede na manjšo porabo virov, predvsem pomnilnika (glej Figure 8). Razlika med SES in OSES v izvajalnem času je izračunana na naslednji način:

$$\frac{t_{OSES} - t_{SES}}{t_{SES}},$$

kjer $t$ predstavlja izvajalni čas. Podobno je izračunana tudi razlika v porabi pomnilnika:

$$\frac{m_{OSES} - m_{SES}}{m_{SES}},$$

kjer $m$ predstavlja maksimalno porabo pomnilnika. Pri obeh izračunih je pozitivna vrednost pomenila, da se je OSES odrezal slabše kot SES, pri negativnih pa ravno obratno. OSES se je odrezal veliko boljše kot SES pri izvajalnem času in prav tako pri porabi pomnilnika. Pri majhnih vzorčnih stopnjah je OSES porabil kar 40% manj pomnilnika kot SES, predvsem zaradi spuščanja velikega števila paketov s pomočjo bloomovega filtra. OSES pristop je bil tudi hitrejši od SES pristopa in sicer za 15%. Pri velikih vzorčnih stopnjah se je razlika med pristopoma zmanjšala, vendar je OSES pristop še vedno dosegel boljše rezultate.

# 6. ZAKLJUČEK

Algoritmi za detekcijo anomalij se pod vzorčenjem v veliki večini primerov slabo odrežejo. Področje se aktualno raziskuje, kar smo prikazali skozi navedene članke znotraj tega članka. Izkazuje se, da detekcijski algoritmi že brez vzorčenja slabo odkrijejo anomalije, nakar vzorčenje še dodatno poslabša uspešnost. Ugotovljeni rezultati kažejo, da je prostora za izboljšave še veliko, ali v smislu razvoja novih algoritmov za detekcijo anomalij ali pa novih tehnik vzorčenja, na kar uspešno kaže tehnika OSES.

# 7. REFERENCES

[1] N. Duffield and C. Lund. Predicting resource usage and estimation accuracy in an ip flow measurement collection infrastructure. In *Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement*, IMC '03, pages 179–191, New York, NY, USA, 2003. ACM.

[2] C. Estan and G. Varghese. New directions in traffic measurement and accounting. *SIGCOMM Comput. Commun. Rev.*, 32(4):323–336, Aug. 2002.

[3] A. Lakhina, M. Crovella, and C. Diot. Mining anomalies using traffic feature distributions. In *Proceedings of the 2005 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '05, pages 217–228, New York, NY, USA, 2005. ACM.

[4] J. Mai, C.-N. Chuah, A. Sridharan, T. Ye, and H. Zang. Is sampled data sufficient for anomaly detection? In *Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement*, IMC '06, pages 165–176, New York, NY, USA, 2006. ACM.

[5] J. Mai, A. Sridharan, C.-N. Chuah, H. Zang, and T. Ye. Impact of packet sampling on portscan detection. *Selected Areas in Communications, IEEE Journal on*, 24(12):2285–2298, Dec 2006.

[6] I. Paredes-Oliva, P. Barlet-Ros, and J. Solé-Pareta. Portscan detection with sampled netflow. In *Proceedings of the First International Workshop on Traffic Monitoring and Analysis*, TMA '09, pages 26–33, Berlin, Heidelberg, 2009. Springer-Verlag.

[7] I. Paredes-Oliva, P. Barlet-Ros, and J. Sole-Pareta. Scan detection under sampling: a new perspective. *Computer*, 46(4):38–44, 2013.

[8] V. R. P. Vaddella and S. Rachakulla. Two-stage opportunistic sampling for network anomaly detection.

[9] K. Xu, Z.-L. Zhang, and S. Bhattacharyya. Profiling internet backbone traffic: Behavior models and applications. *SIGCOMM Comput. Commun. Rev.*, 35(4):169–180, Aug. 2005.

# Zaščita pred botneti

## [Samoozdravljivi sistemi]

Leon Noe Jovan
Univerza v Ljubljani, Fakulteta za računalništvo in informatiko
Tržaška 25
Ljubljana, Slovenija
leon.jovan@gmail.com

David Novak
Univerza v Ljubljani, Fakulteta za računalništvo in informatiko
Tržaška 25
Ljubljana, Slovenija
novakdavid.dn@gmail.com

Dejan Petrović
Univerza v Ljubljani, Fakulteta za računalništvo in informatiko
Tržaška 25
Ljubljana, Slovenija
petrodej@gmail.com

## POVZETEK

Ta dokument opisuje nastanek, delovanje in vlogo botnetov v digitalnem svetu danes. Povzame njihovo arhitekturo in uporabo, kot orodje v rokah spletnih zločincev. Spletne goljufije in ostale vrste zločinov, postajajo vedno bolj priljubljena oblika kriminala, zaradi relativno lahko dostopne opreme, ki se v te namene uporablja in količine zaslužka, ki ga tovrstna dejavnost prinaša. Veliko botnetov omogoča anonimnost njihovega uporabnika, kar pomeni, da se zločincem ni potrebno pretirano izpostavljati, pri kršenju zakona. Mreža računalnikov, ki sestavljajo botnet, prav tako lahko deluje samostojno, brez dodatnega upravljanja. Te lastnosti odlikujejo enega največjih orodij spletnega kriminala danes.

## Ključne besede

Računalniška varnost, botneti, spletni kriminal, samoozdravljivi sistemi

## ABSTRACT

This document encompasses the formation, operation and function of botnets in the digital world of today. Summed in this paper are the botnet's architecture and use as a tool in the hands of cybercriminals. As of late, internet frauds and similar forms of cybercrime have gained significant popularity amongst cybercriminals on account of the relatively easy accessible equipment and the amounts of profit this activity yields. Most botnets ensure their user's anonimity, which in turn results in less risk of exposure for the criminal. Also, a botnet network is fully capable of operating of it's own accord, not requiring any further management. All these attributes define one of the world's greatest tools in cybercrime today.

## Keywords

Computer ecurity, botnets, cybercrime, self-healing systems

## 1. UVOD

Spletni kriminal je postal donosna dejavnost, ki podjetja, države in potrošnike stane več sto miljard evrov letno. Nekateri ocenjujejo, da je spletni kriminal postal bolj donosen kot trg marihuane, kokaina in heroina skupaj [3]. Posledično je obramba pred spletnim kriminalom postala poglavitna naloga mnogih držav. Evropska unija je ravno pred kratkim ustanovila EC3 (European Cybercrime Centre), katerega glavna naloga je pregon spletnega kriminala storjenega s strani večjih organiziranih kriminalnih skupin, otroške pornografije in napadov na infrastrukturo in informacijske sisteme Evropske unije [3]. Eno glavnih orodij spletnih kriminalcev, ki jim omogočajo dostop do milijonov računalnikov po celem svetu, so tako imenovani Botnet-i.

## 2. KAJ JE BOTNET?

Izraz botnet se nanaša na gručo računalnikov, okuženih z isto zlonamerno programsko opremo. Vsak računalnik v tem omrežju služi kot programski robot in lahko avtomatično opravlja določena dejanja ali ukaze. Boti tudi pogosto oponašajo različna človeška dejanja, kot je naprimer prijava v sistem.

Čeprav so boti nastali, kot uporabni programi, ki izvajajo dolgotrajne in ponavljajoče postopke, so se kmalu začeli uporabljati tudi v zlonamerne namene [5]. Boti lahko oblikujejo mrežo okuženih računalnikov ali zombijev, katere nadzira in upravlja centralni nadzorni računalnik (C&C - command & control). Stroj večinoma postane ogrožen, ko uporabnik odpre ali prenese zlonamerno programsko opremo.

### 2.1 Življenjski cikel botneta

Rojstvo in rast botneta omogoča širjenje zlonamerne programske opreme. Napadalec mora najprej distribuirati svoje slabogramje med ostale računalnike, preden ti postanejo del mreže, ki lahko učinkovito izvaja napadalčeve ukaze. Več kot je že okuženih računalnikov, hitreje bo botnet rastel in izvajal kriminalno dejavnost. Kje v hierarhiji botneta je prvi računalnik s katerim je mreža osnovana, je odvisno od arhitekture, ki jo računalniki vzpostavijo. Ko napadalec okuži računalnik, ga najprej pregleda za informacije s katerimi, bi se lahko okoristil. Pogosto so to gesla ali številke kreditnih kartic. Naloga novega člena botneta je nato okužba novih računalnikov preko spam elektronske pošte ali pa goljufij s klikanjem ("click fraud"). Poleg širjenja botneta, ti računalniki izvajajo DDoS napade, pošiljajo spam nezaželeno elektronsko pošto ali gostujejo strani za spletno ribarjenje.

Širjenje botneta je tako eksponentno, saj se z vsakim okuženim računalnikom število nadaljnjih okužb še poveča.

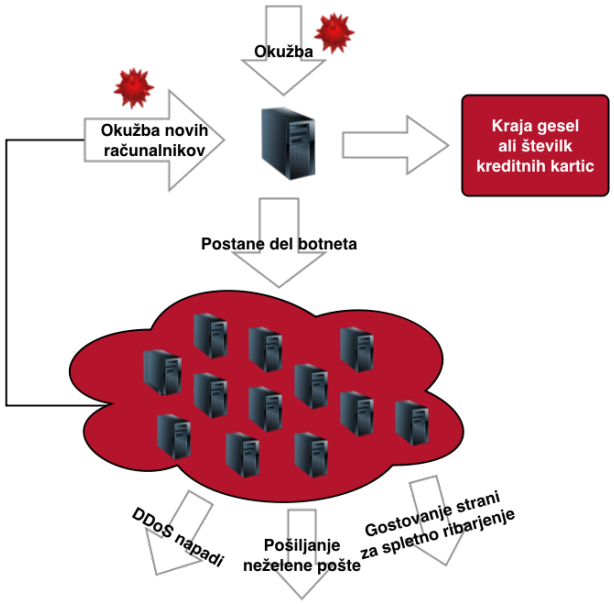



**Figure 1: Življenjski cikel botneta**

## 2.2 Uporaba botnetov

### *2.2.1 Računalniške goljufije in prevare*

Prvotni namen internetnih zločincev je zaslužiti na račun njihove zlonamerne opreme ali natančneje nelegalnih operacij, ki jih izvršujejo. Če posplošimo, gre na najnižjem nivoju za krajo poverilnic, ki se jih uporabniki poslužujejo pri komunikacijah z banko in pri transakcijah na svojem bančnim računom. Pri kraji plačilnih informacij še najbolj izstopata Zeus in ZeroAccess. ZeroAccess je botnet, ki je bil prvič zaznan v 2012, danes pa je najhitreje rastoč tovrstna arhitektura na svetu. Okužil je na milijone računalnikov, ki so potem izvajali goljufije s klikanjem ("click fraud") ali pa so rudarili za Bitconi. Oba botneta se osredotočata s krajo denarja na omrežnih bančnih računih, vendar pa je zaradi hitre rasti v zadnjem času pozornost zločincev pritegnil tudi Bitcoin. Novo slabogramje se osredotoča predvsem na zasedanje procesorske moči okuženih osebnih računalnikov, ki jih botnet nato izkorišča za rudarjenje kriptovalute.

### *2.2.2 Kibernetični napadi in hekerstvo kot storitev*

Botneti se pogosto uporabljajo tudi za kibernetične napade proti različnim tarčam. Družbe jih uporabljajo za oškodovanje konkurence, ali za širjenje političnih mnenj. Vedno več je organizacij, ki ponujajo zmožnosti botnetov kot storitve. Ponujajo vse komponente, ki so potrebne za izgradnjo botneta in izvršitev kibernetičnega napada. Zločince, ki najemajo arhitekture ali prodajajo zlonamerne storitve kot so DDoS napadi je relativno preprosto odkriti. Poslovni model, ki ga uporabljajo je učinkovit, cene pa se razlikujejo. Model se oglašuje pod imenom slabogramje kot storitev ("malware as service"), privablja širok spekter zločincev, med drugimi tudi take, ki niso nujno usposobljeni za to vrsto kriminala.

### *2.2.3 Kibernetično vojskovanje in vohunjenje*

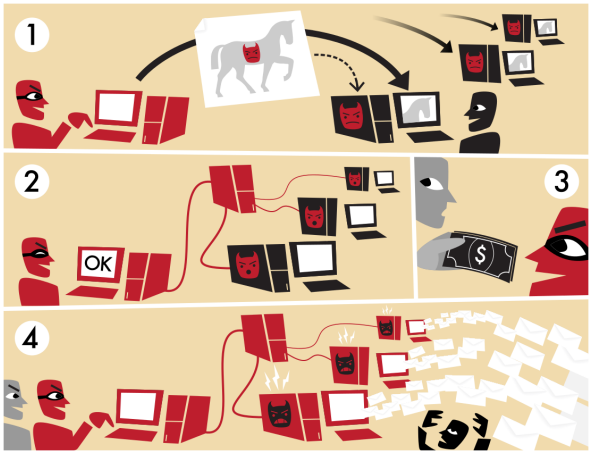



**Figure 2: Hekerstvo kot storitev**

V zadnjem času se je izkazalo, da je prilaščanje tujih informacij in zbiranje različnih statistik tudi interes številnih vlad in vladnih organizacij. Velikokrat je vodstvo države glavni pokrovitelj kibernetičnega vohunjena, ki ga izvajajo zlonamerne arhitekture. Znane države, ki se poslužujejo teh sredstev so Kitajska, Rusija in Koreja, tako ravnanje pa se nedvomno smatra kot škodovanje tujim državam. S stališča uporabnikov botnetov gre za idealen način prikritega vojskovanja. S pomočjo interneta ima država lahko na milijone računalnikov znotraj izbranih držav, in tako prikrito zbira podatke. Razlika med običajnimi in vladnimi botneti je nedvomno v tem, da slednji omogočajo napadalčevo anonimnost. Njihova prednost je tudi v pogostejšem izvajanju zero-day napadov, ki zahtevajo konkretno financiranje in večjo količini raziskav, vloženih v njihov razvoj.

### *2.2.4 Pošiljanje neželene pošte*

Botneti pošiljajo neželeno pošto v namen okužbe drugih računalnikov ali pa v namen oglaševanja in s tem zaslužka. Elektronske naslove botnet ponavadni pridobi z iskanjem po spletnih straneh in podatkovnih bazah, včasih pa jih preprosto uganejo s kombiniranjem pogostih uporabniških imen in domen. Eden bolj znanih botnetov za pošiljanje neželene pošte je bil Waledac, ki je bil dejaven med letoma 2008 in 2010 in je okužil približno 80.000 računalnikov preko neželene pošte. Botnet je bil zmožen poslati preko 1.5 milijard neželenih sporočil dnevno, kar je 1% celotne neželene pošte.

### *2.2.5 Gostovanje strani za spletno rudarjenje*

Spletno ribarjenje ali "phising" je vrsta spletne goljufije, ki se uporablja za pridobivanje osebnih podatkov kot so poverilnice, gesla in uporabniška imena. Napadalec žrtvam preko različnih medijev posreduje vsebine, ki na prvi pogled sicer delujejo poznane in zaupanja vredne, v resnici pa so škodljive oz. so ponaredki vsebin, ki jim uporabnik sicer zaupa. Dober primer teh vsebin so elektronske pošte, ki uporabnika pozivajo k obisku določenih spletnih strani, ali pa kar same spletne strani, ki simulirajo vnosne forme iz katerih napadalec lahko izlušči občutljive podatke. Bistvo goljufije je, da vsi elementi delujejo kar najbolj pristno. Čeprav se žrtvi zdi, da ve kaj je destinacija povezave na katero klika, to skoraj gotovo ni tako. Od leta 1987, ko je bil izraz spletno ribarje-

nje prvič uporabljen, postaja ta vrsta spletne goljufije vedno večji problem. Velja pa si zapomniti, da se še tako dobre ponaredke lahko odkrije, saj napadalci pogosto ne upoštevajo vseh podrobnosti. Pozorni moramo biti na slovnične napake in spletne povezave, ki računajo na to, da ne bomo opazili, da je spletni naslov narobe črkovan (myspacee.com).

Botneti za izvajanje phishing napadov uporabljajo tehniko imenovano fast flux, ki omogoča skrivanje strani za spletno ribarjenje za konstantno spreminjajočim omrežjem okuženih računalnikov, ki nastopajo kot proxy strežniki. Osnovna ideja je, da imamo eno domeno povezano z velikim številom IP naslovov, ki se s spreminjanjem DNS zapisov menjajo z visoko frekvenco, naprimer vsake tri minute. Tako so takšne strani bolj odporne na odkritje in protinapade [1].

## 2.3 Topologije botnetov

Botneti se lahko oblikujejo v mreže različnih velikosti in oblik. Najbolj pogoste oblike so:

- Zvezda
- Obroč
- Hierarhična topologija
- Naključna topologija

### *2.3.1 Zvezda*

Pri topoligiji zvezda so vsi okuženi računalniki povezani s centralnim nadzornim računalnikom. Zvezda sicer omogoča hitro in direktno komunikacijo z okuženimi računalniki, vendar se lahko centralni nadzorni računalnik hitro odkrije in blokira ali onesposobi. S tem onesposobimo celoten botnet.

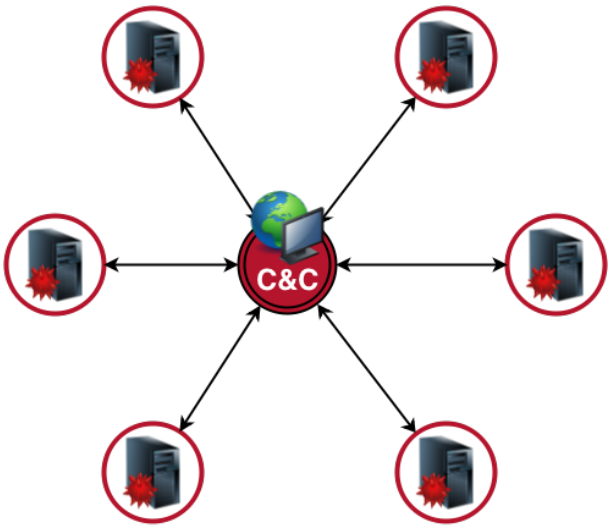



**Figure 3: Topologija zvezda**

### *2.3.2 Obroč*

Topologija obroča ali večstrežniška topologija je precej bolj odporna, saj vsebuje več nadzornih računalnikov, ki so povezani med seboj. Tako lahko ob izpadu enega od nadzornih računalnikov botnet še vedno funkcionira.

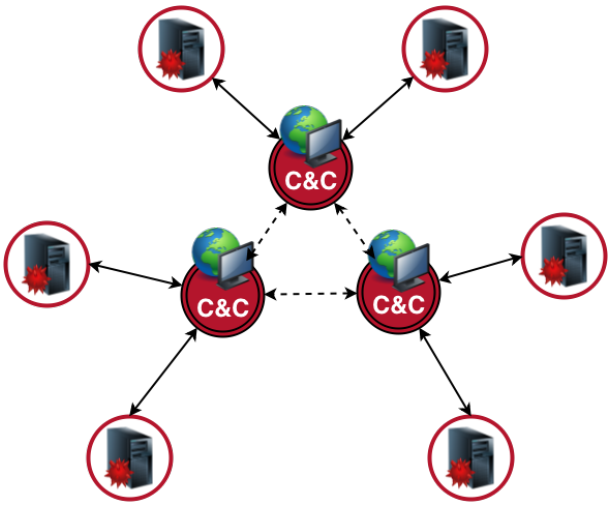



**Figure 4: Topologija obroča**

### *2.3.3 Hierarhična topologija*

Hierarhična topologija omogoča, da je ostane centralni računalnik čimbolj skrit. Vmesni boti delujejo kot proxy strežnik, ki pošilja zahteve nadzornega računalnika svojim potomcem. Napadalci so s takšnimi botneti bolj varni in težje izsledljivi, vendar pa je pošiljanje ukazov počasnejše, zato težko služijo za upravljanje v realnem času.

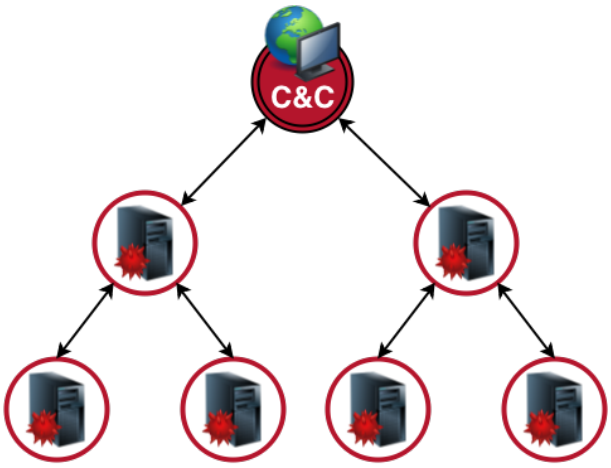



**Figure 5: Hierarhična topologija**

### *2.3.4 Naključna topologija*

Botneti z naključno topologijo nimajo centralizirane nadzorne infrastrukture. Boti so men seboj povezani naključno. Ukazi so lahko poslani iz kateregakoli bota. Ko drugi bot dobi te ukaze, jih tudi on pošlje naprej ostalim povezanim botom. Takšni netboti so izjemno odporni [7].

## 2.4 Znani botneti

V preteklosti se je pojavilo število botnetov, ki so postavljali nove mejnike, tako z vidika ustvarjanja novih, pametnejših botnetov, kot tudi z vidika obrambe pred njimi. V nadaljevanju so opisani najbolj znani, in tudi največji botneti od

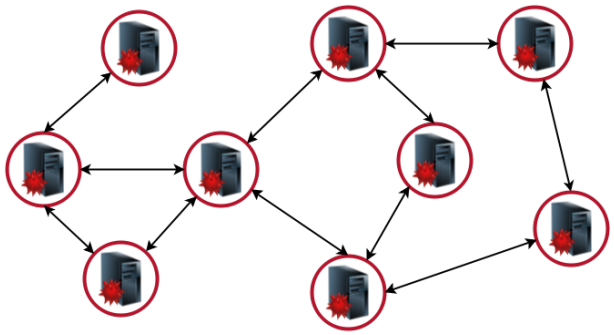


**Figure 6: Naključna topologija**

leta 2005 do danes.

### 2.4.1 Torpig

Prvič je bil odkrit leta 2005, širil se je vse do leta 2009, ko ga je takrat RSA security opisala kot eno najbolj naprednih kriminalnih programov do tedaj. Ta botnet je bil oblikovan z namenom kraje podatkov o bančnih računih in kreditnih karticah. Raziskovalcem z Univerze v Kaliforniji je uspelo dobiti nadzor nad botnetom za deset dni leta 2009, v tem času so našteli več kot 1.2 milijona različnih IP naslovov, od koder so se poskušali povezati na nadzorni računalnik.

### 2.4.2 Virut

Odgovoren je za več kot pol milijona infekcij po vsem svetu, sposoben pa je narediti pravzaprav vse, kar kriminalni botneti lahko počnejo: DDoS napadi, pošiljanje neželjene pošte, kraja podatkov, finančne prevare (phishing)... Pojavil se je leta 2006, deloval pa je vse do pred kratkim, ko so onesposobili večino botov, s skupnim delom večih organizacij.

### 2.4.3 Zeus

Eden najbolj finančno donosnih botnetov v zgodovini, prvič je bil odkrit leta 2007. Razvit je tako, da neopazno spremlja žrtvin računalnik in nadzornemu računalniku pošilja bančne podatke. FBI je ocenil, da je Zeus povzročil za več 100 milijonov dolarjev izgube. Leta 2010 je razvijalec Zeusa zaustavil njegovo delovanje, izvorno kodo pa izročil ustvarjalcu SpyEye botneta. Leto kasneje se je izvorna koda Zeusa pojavila na spletu, kar je povzročilo velik narast novih različic botnetov.

### 2.4.4 Storm

Ni znano točno število računalnikov, ki jih je okužil Storm, a ocena se giblje pri okoli 50 milijonih okužb. Znan je predvsem po tem, da je zelo odporen na "reverse engineering" in analizo. Njegovi avtorji so še vedno neznani, na prostosti.

### 2.4.5 Grum

Grum botnet je bil v svojem času največji botnet na spletu, v svojem vrhuncu je bil odgovoren za več 100 milijard poslane neželjene pošte ("spam") v obliki farmacevtskih oglasov. Onesposobljen je bil julija 2012.

### 2.4.6 Lethic

V svoji mreži je imel več kot 300.000 računalnikov, ki so bili sposobni pošiljati več 10 milijard elektronskih sporočil dnevno, kar je bilo takrat 10% vse neželjene pošte globalno. Januarja 2010 je podjetje Neustar prevzelo nadzor in zaustavilo večino botov v omrežju, a ustvarjalci Lethica so ponovno prevzeli nadzor dva meseca kasneje. Lethic je še danes aktiven, ocenjeno je, da dnevno pošlje okoli 2 milijardi elektronskih sporočil.

### 2.4.7 Mariposa

Primarno uporabljen za DDoS napade in internetne prevare. Leta 2009 so španske oblasti zaustavile delovanje botneta, ki je bil takrat odgovoren za več kot 13 milijonov okužb in je ustvarjalcem prinesel 250.000 evrov mesečno. Nadzorni računalnik Maripose je bil sicer onesposobljen, a zaradi pojavitve izvorne kode na spletu in velikega števila novih različic se še danes pogosto pojavlja na naših računalnikih.

### 2.4.8 SpyEye

Razvit leta 2009 kot neposredna konkurenca botnetu Zeus, njegov namen je bil enak kot Zeusov: finančna kraja. Poleg tega je SpyEye dovolj inteligenten, da odkrije okužbo Zeusa na računalniku in jo tudi odstrani. Kasneje je avtor zeusa izročil svojo izvorno kodo botnetu SpyEye. Od takrat SpyEye ostaja velik del škodljive programske opreme na internetu.

### 2.4.9 Waledac

Znan je predvsem po zmožnostih pošiljanja neželenih sporočil, pod svojim nadzorom ima okoli 100.000 okuženih računalnikov, ki dnevno pošljejo 1.5 milijard neželenih elektronskih sporočil. Leta 2010 je Microsoft prevzel nadzorni računalnik in močno zmanjšal število okužb. Dve leti kasneje se je botnet znova množično pojavil.

### 2.4.10 ZeroAccess

Okužil je med enim in dvema milijonoma računalnikov. Ocenjeno je, da omrežje zombijev še danes generira milijon dolarjev letno v obliki rudarjenja bitcoinov in spletnih prevar.

### 2.4.11 Jeff

Začel se je kot virus v obliki parazitne datoteke v letu 2007. Od leta 2013 je opaziti velik porast v okužbah in sicer odkar se je Jeef razvil v P2P botnet in se začel širiti veliko hitreje.

## 3. ZAŠČITA PRED BOTNETI

Obramba pred botneti se deli na preprečevanje, odkrivanje in odstranjevanje. Preventivno se botnetov ubranimo predvsem s posodabljanjem operacijskega sistema in programov, povečanjem stopnje varnosti v brskalniku in ignoriranjem priponk v elektronskih sporočilih pri katerih ne poznamo izvora [6].

Če računalnik postane okužen, moramo to čimpreje opaziti. Običajno se to opazi po upočasnjenem delovanju računalnika ali pa povečanem omrežnem prometu. To pogosto opazi administrator omrežja. Takrat se okužen računalnik odklopi od omrežja, odstrani škodljivo programsko kodo ter zamenja gesla, ki so se uporabljala na računalniku. Nato se računalnik ponovno priklopi na omrežje in preveri, če smo škodljivi program uspešno odstranili.

### 3.1 Zaščita na nivoju računalnika

Računalnik najbolje zaščitimo s posodabljanjem programske opreme, predvsem operacijskega sistema, protivirusne zaščite in brskalnika. Priporočena je nastavitev brskalnika na najbolj varen način. Poleg tega je odsvetovano odpiranje priponk v elektronskih sporočilih, pri katerih ne poznamo izvora, ter obiskovanje sumljivih spletnih strani.

## 3.2 Zaščita na nivoju omrežja

Če želimo ustrezno zaščititi računalnike v omrežju moramo za to imeti ustrezno tehnologijo na prehodu:

- ustrezno protivirusno zaščito (anti-virus)
- zaščita proti neželeni pošti (anti-spam)
- požarni zid (firewall)
- sistem za preprečevanje in detekcijo vdorov (IPS/IDS)
- sistem za celostno varnostno zaščito (UTM)

Potrebno je opazovanje prometa v omrežju, saj lahko nepričakovan porast omrežnega prometa nakazuje na napade DDoS. Pomembno je tudi, da imamo ustrezno zaščito za takšne napade. Okužene računalnike moramo prepoznati čimprej in jih odklopiti od omrežja, saj tako lahko preprečimo okužbo tudi drugih računalnikov. Če nam uspe najti škodljivi program, ki ga uporablja botnet, ga posredujemo ponudnikom protivirusnih programov. Z analizo zabeležk požarnega ziduali UTM lahko identificiramo nadzorni strežnik botneta, ter ga poskušamo onesposobiti. To lahko predstavlja zelo težko nalogo za posamezno podjetje, zato se tu lahko obrnemo na zunanjo pomoč, kot so razni eksperti in državni varnostni organi [2].

Za zaščito omrežij se pogosto uporablja tudi Honeypot. Honeypot je računalniška vaba, nastavljena za odvračanje napadov na resnično pomembne dele sistema, za detekcijo poskusov nepooblaščene uporabe informacijskih sistemov, ter za opazovanje in analiziranje botneta. Ponavadi gre za računalnik, ki izgleda kot del omrežja, vendar je izoliran in nadzorovan. Pogosto izgleda, kot da vsebuje pomembne informacije in podatke, da bi tako privabil napadalce. [wiki]

Zaščita, detekcija in odprava škode netbotov pogosto zahteva človekov poseg v sistem, s tem pa zakasnjen odziv na napad in daljše čase izpadov sistemov. Avtorji članka [4] predlagajo samoozravljivi sistem, ki omogoča omrežnim sistemom nenehno iskati anomalije v obnašanju sistema, ter se na njih ustrezno odzovejo in jih odpravijo.

# 4. SAMOOZDRAVLJIVI SISTEMI

Predlagana arhitektura sistema omogoča omrežnim sistemom, da se ozdravijo sami, tako da neprestano iščejo spremembe v običajnem delovanju in nato uporabijo ustrezne popravne ukrepe, da se povrne prvotno stanje sistema. Koncept samoozdravljivih sistemov prihaja iz narave, in sicer, kako se organizmi prilagodijo na njihovo okolje, ko razvijejo odpornost proti škodljivim virusom, bakterijam in strupom. Tak način delovanja je ponekod že uporabljen, na primer pri nekaterih operacijskih sistemih so razvijalci vgradili zanimive samoozdravljive mehanizme, kot je ponovno nalaganje programske kode, izolacija posameznih komponent in avtomatski ponovni zagoni. Pri vgrajenih sistemih je primer takega načina delovanja, da teče več instanc opravila istočasno z nekaj mirujočimi instancami. V primeru, da pride do napake, sistem prenese opravila na mirujoče instance, ki nemoteno nadaljujeo z delovanjem.

Samoozdravljiv sistem lahko prepozna spremenjeno delovanje, ko ta ne deluje pravilno in z le malo posredovanja uporabnika ali celo brez ali obnovi sistem na prvotno stanje ali pa vzdržuje zdravje sistema dokler administrator ne ukrepa. Za razliko od sistemov, ki tolerirajo napake, samoozdravljivi sistemi običajno delujejo tako, da izolirajo okuženo ali nedelujočo komponento, jo odklopijo od omrežja in jo nato popravijo ali pa zamenjajo z delujočo komponento. Cilj vsakega takega sistema je vzdrževanje dosegljivosti, zdravja in nadaljnjega delovanja. Za dosego teh ciljev ima omenjen sistem mehanizme za detektiranje anomalij v funkcionalnosti sistema, med drugim tudi prepoznavanje prisotnosti tujih in potencialno nevarnih agentov, diagnosticiranje težav in obnavljanje sistema.

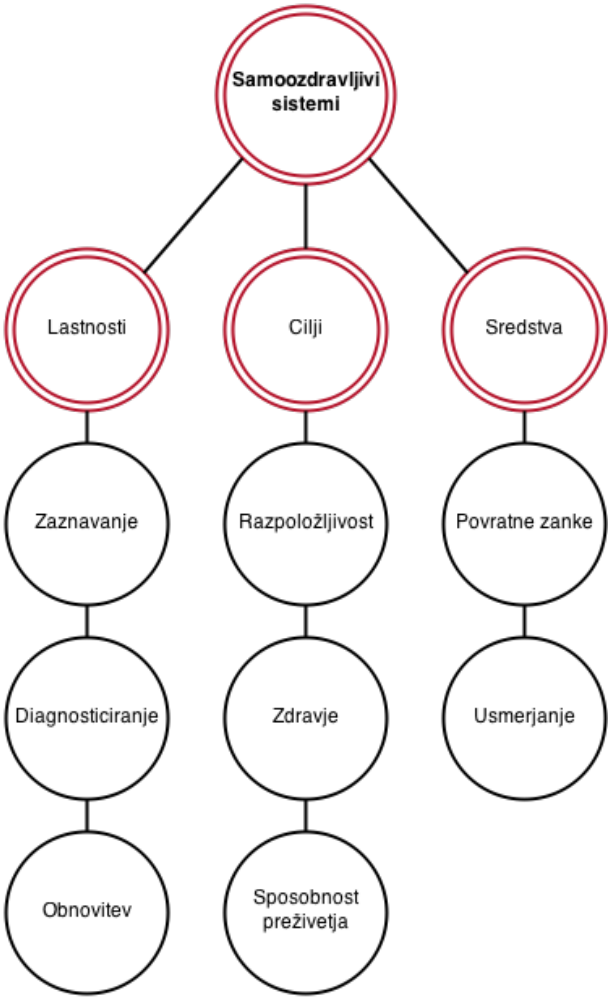



**Figure 7: Cilji, lastnosti in sredstva samoozdravljivih sistemov**

## 4.1 Arhitektura sistema

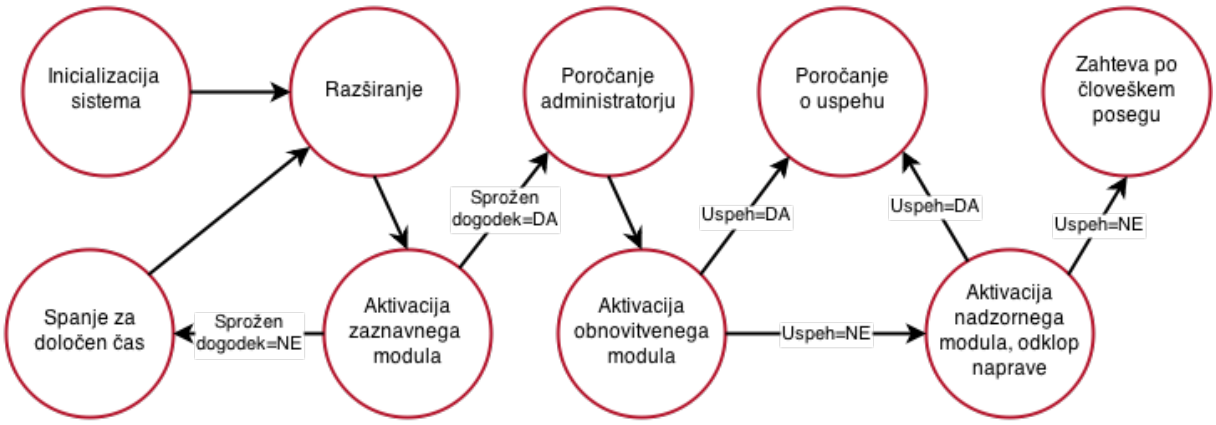



**Figure 8: Delovanje samoozdravljivega sistema**

Razvoj samoozdravljivih sistemov temelji na raziskavah delovanja več botnetov. V ta namen sta bila preučena dva na HTTP-ju temelječa botneta Zeus in Black-Energy in dva P2P botneta Waledac in Storm. Samoozdravljivi sistemi so tako sposobni ublažiti okužbe, ki so sicer sposobne obiti običajne varnostne sisteme. Sama arhitektura teh sistemov je prilagojena delu z domenami, ki delujejo preko krmilnega strežnika, ki skrbi za varnost in avtentikacijo vseh naprav v omrežju. Take strukture so zelo pogoste v središču vladnih agencij, bolnic, univerz ipd.

Arhitekturo lahko razdelimo na pet modulov:

- Komunikacijski modul
- Dnevniški modul
- Zaznavni modul
- Obnovitveni modul
- Nadzorni modul

### *4.1.1 Komunikacijski modul*

Komunikacijski modul upravlja z vsemi komunikacijami sistema, tako vhodnimi kot izhodnimi

### *4.1.2 Dnevniški modul*

Dnevniški modul dokumentira vso dejavnost sistema in obvešča administratorja o morebitnih nepravilnostih v delovanju komponent. Generira dnevna, tedenska in mesečna poročila in jih arhivira v podatkovni bazi.

### *4.1.3 Zaznavni modul*

Zaznavni modul je preventivni modul, ki na omrežju išče znake zlonamerne dejavnosti. V ta namen nadzoruje vso dejavnost na vhodu in pregleduje datoteke DNS gostiteljev. Načeloma mora nadzor nad posameznimi porti in dodajanje izjem določiti administrator, kljub temu pa je modul sposoben seznam le-teh dopolniti tudi sam. Določena programska oprema uporablja unikatne porte, vendar to še ni znak da je zlonamerna. V takih primerih se modul posvetuje z administratorjem, ki odloči ali se vrata lahko doda na seznam izjem ali ne. Nekateri botneti so sposobni ustvariti nove porte in jih nato ustrezno prirejati. Število vrat in njihova splošna stanja opredeljuje datoteka tcp.sys. Zato zaznavni modul redno pregleduje integriteto te datoteke z računanjem MD5 podpisa.

### *4.1.4 Obnovitveni modul*

Obnovitveni modul je posebej opremljen za primere, ko okužbe ni bilo mogoče preprečiti. Modul to opravlja v sodelovanju s ti. agenti, ki skrbijo za vsebovanje okužbe in preprečevanje njenega širjenja. Ko sistem zazna prisotnost bota, modul agentom posreduje ukaze za zaključevanje storitev oz. zapiranje ogroženih vrat kar omeji botove zmogljivosti. Okužbe v sistemu se odražajo na več načinov, kot so porast prometa na določenih vratih. Preden boti lahko pričnejo učinkovito delovati, se morajo najprej registrirati oz. v primeru HTTP botov, pridobiti ukaze za delovanje z botnet nadzornega centra. Tovrstne DNS poizvedbe, modul lahko prestreže preko glavnega DNS strežnika, kar odpravi dolgotrajno dodajanje individualnih gostiteljev, v primeru da bi zlonamerna stran spremenila svojo domeno.

### *4.1.5 Nadzorni modul*

Če zdravljenje ni bilo uspešno, je naloga nadzornega modula, da okuženo vozlišče odstrani. To prepreči nadaljnje širjenje okužbe po omrežju. Poizvedba po okuženih vozliščih se izvede s pomočjo ICMP echo zahtevka. V primeru, da modul ne prejme nobenega odziva, se vozlišče smatra kot ogroženo. Seznam tovrstnih vozlišč se nato posreduje administratorju, ki posreduje če modul ni sposoben sam odstraniti oporečnega vozlišča.

## 4.2 Delovanje sistema – interakcije med moduli

Ko je sistem vzpostavljen, zaznavni modul najprej pregleda celoten sistem za obstoječe okužbe. Pregledovanje je periodično, ponavlja se v določenih časovnih intervalih. Če pregled ne sproži odziva se iskanje prestavi, v nasprotnem primeru pa modul obvesti administratorja, ki požene zdravilni modul. Če zdravljenje ni uspešno, nadzor prevzame kontrolni modul, ki poskuša ogroženo vozlišče onesposobiti.

Skupaj s tehnologijo se razvijajo tudi kibernetični napadi, ki za strokovnjake predstavljajo vedno večjo grožnjo. Ki-

berkriminalci se poslužujejo botnetov za krajo informacij iz tako manjših/osebnih, kot tudi večjih sistemov vladnih organizacij, bank in podobnih institucij. Cilj samoozdravljivih sistemov je preprečevanje takšnih zlorab in splošno povečanje odpornosti sistemov na napade, z minimalnim oviranjem omrežnih storitev.

# 5. ZAKLJUČEK

V prihodnosti bodo botneti vse bolj množični in vse bolj napredni. Postajajo vedno večja grožnja tako za osebne računalnike, kot tudi velike institucije. Obramba pred njimi poteka v večih nivojih. Na nivoju posameznika: vsak uporabnik skrbi za varnost svojega računalnika. Sem spadajo posodobitve operacijskega sistema in druge programske opreme, antivirus, požarni zid in pazljivo ravnanje z nepreverjenimi oz. vprašljivimi vsebinami. Naslednji nivo obrambe je na omrežju: zaščita na usmerjevalniku, neprekinjen nadzor nad vhodnimi vrati v sistem, uporaba sistemov, ki so sposobni sami prevzeti iniciativo pri preprečevanju delovanja botnetov in nenazadnje tudi izolacija, pregled in posodabljanje entitet(računalnikov), ki so bili na katerikoli stopnji izpostavljeni kar pomeni, da so delovali zunaj zaščitenega sistema. Očtino je, da bo botnet napadov v prihodnosti le še več. Že v prvi polovici leta 2005 je vsak dan postalo aktivnih dodatnih 10.352 botov, kar je takrat predstavljalo 140-odstotno porast števila zabeleženega v prejšnjem polletju. Danes botom povzročanje škode ni več samo sebi namen, saj so prerasli v organizirane arhitekture s točno določenimi cilji.

# 6. REFERENCES


[1] How fast-flux service networks work. http://www.honeynet.org/node/132. Accessed: 2014-05-16.

[2] What are botnets, what can they do & how to protect against them. http://www.excitingip.com/2755/botnets-what-are-botnets-what-can-they-do-how-to-protect-against-them/. Accessed: 2014-05-16.

[3] Tackling crime in our digital age: Establishing a european cybercrime centre. Marec 2012. Dostopno na: http://ec.europa.eu/home-affairs/doc_centre/crime/docs/Communication%20-%20European%20Cybercrime%20Centre.pdf.

[4] A. Alhomoud, I. Awan, J. F. P. Disso, and M. Younas. A next-generation approach to combating botnets. *Computer*, 64(4):62–66, April 2013.

[5] M. T. Banday, J. A. Qadri, and N. A. Shah. Study of botnets and their threats to internet security. *Sprouts: Working Papers on Information Systems*, 9(24), 2009. Dostopno na: http://sprouts.aisnet.org/594/1/Botnet_Sprotus.pdf.

[6] R. Challoo and R. Kotapalli. Detection of botnets using honeypots and p2p botnets. *Sprouts: Working Papers on Information Systems*, 5(5):496–502, 2011. Dostopno na: http://www.cscjournals.org/csc/manuscript/Journals/IJCSS/volume5/Issue5/IJCSS-584.pdf.

[7] G. Ollmann. Botnet communication topologies: Understanding the intricacies of botnet command-and-control. 2009. Dostopno na: https://www.damballa.com/downloads/r_pubs/WP_Botnet_Communications_Primer.pdf.

# Detecting Influential Spreaders in Complex, Dynamic Networks

[Seminarska naloga]

Rok Gomišček
Fakulteta za računalništvo in informatiko
Tržaška cesta 25
Ljubljana, Slovenija
rok.gomiscek@gmail.com

Igor Lalić
Fakulteta za računalništvo in informatiko
Tržaška cesta 25
Ljubljana, Slovenija
igypigi@gmail.com

Jon Premik
Fakulteta za računalništvo in informatiko
Tržaška cesta 25
Ljubljana, Slovenija
jon.premik@gmail.com

## ABSTRACT

## Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

## General Terms

Theory

## Keywords

ACM proceedings, LaTeX, text tagging

## 1. UVOD

V zadnjih letih smo priča vse večjemu številu omrežij, katerim pripadamo. Na eni strani se večajo družbena omrežja, kjer se povezujemo s prijatelji, sodelavci in ljudmi s podobnimi interesi. Na taka omrežja lahko gledamo kot na graf, kjer vozlišča predstavljamo uporabniki. Twitter je v sedmih letih po nastanku pritegnil 200 milijonov uporabnikov, ki prispevajo 400 milijonov tweetov vsak dan [4]. Facebook je že več kot leto nazaj presegel miljardo aktivnih uporabnikov, večina katerih ima čez 200 prijateljev [3]. Konec koncev lahko tudi na elektronsko pošto gledamo kot na omrežje, kjer naš imenik predstavlja povezave med vozlišči. Poznamo tudi omrežja, ki so namenjena poslovnemu povezovanju, kot na primer LinkedIn, ki je v enajstih letih pridobil čez 300 milijonov članov in se širi s hitrostjo dva nova člana na sekundo [1].

Zaradi svoje velikosti so ta omrežja zelo kompleksna, zaradi pogostih dodajanj tako vozlišč kot povezav med njimi pa tudi dinamična. Na raznih družbenih omrežjih uporabniki delijo razne podatke o sebi, kar je seveda zanimivo za raziskovalce in je botrovalo razvoju računalniških družbenih ved, ki vnašajo računalniški pristop k družbenim vedam. Njihov namen je predvsem zbirati podatke in jih analizirati. S tem poskušajo odkriti skrite vzorce v mrežah ter ugotoviti, kdo so vplivna vozlišča. To so taka vozlišča, ki lahko hitro in uspešno razširijo informacijo na druga vozlišča. Zaradi njihove vplivnosti in učinkovitosti, so taka vozlišča lahko tarča nepridipravov in je zato še posebej pomembno, da jih zavarujemo pred napadi.

V nadaljevanju bomo pogledali, na kakšne načine je možno napasti omrežja s pomočjo vplivnih vozlišč ter nekaj klasičnih načinov, kako ta vozlišča odkriti. Po tem si bomo pregledali še novejši način za odkrivanje vplivnih vozlišč in ga primerjali z ostalimi. Na kocnu bomo vse načine še preiskusili na praktičnih primerih.

## 2. NEVARNOSTI

Identifikacija vplivnih vozlišč ima veliko praktično vrednost v kompleksnih socialnih omrežjih. Ljudem lahko pomaga kontrolirati širjenje okuženosti, govoric, računalniških virusov in popularnosti produktov in idej.

A ta omrežja se žal ne uporabljajo samo za širjenje zaželenih informacij. Omrežja se lahko uporabi tudi za širjenje takih in drugačnih zlonamernih informacij, na primer širjenje računalniških virusov in uporaba omrežij v terorizmu.

Tako se lahko neresnična govorica, ki se je začela kot šala ali pa napaka, preko vplivnih vozlišč zelo hitro razširi po omrežju. To lahko resno prizadene osebo, povzroči propad podjetja ali pa sproži uničenje javne ali zasebne lastnine.

V Sloveniji je bil zelo zanimiv primer ene izmed večjih strani z novicami, kjer so zgolj za kratek čas objavili, da se je v Prlekiji zgodil pravi torando. Slika je bila seveda obdelana in tako neresnična, kar pa novinar sprva ni vedel in je zaradi pritiska konkurence članek takoj objavil. V zelo kratkem času se je novico delilo po vseh socialnih omrežjih in celo na nekaterih drugih straneh z novicami. Kljub temu, da so originalni članek hitro zbrisali, je bilo že prepozno. Na srečo pa pri temu ni prišlo do nobene škode in se je hitro razvedelo, da je slika neresnična. Pa vendar bi ob bolj občutljivi informaciji, hitro bil nekdo oškodovan.

Ker imajo vlivna vozlišča veliko moč za hitro širjenje komu-

nikacij, bi jih bilo potrebno zatreti že na začetku. Različne vlade bi rade svoje prebivalce zaščitile na ta način, kar pa je seveda spet poseg v svobodo interneta.

Kot primer vzemimo mobilne naprave. Če je naprava, ki je predstavlja vplivno vozlišče okužena (je dobila okužbo preko SMS sporočila), je zelo velika verjetnost hitrega širjenja okužbe tudi na druge naprave. Tukaj bi ob znani listi vplivnih vozlišč, administrator takoj vedel katere naprave je potrebno odklopiti od omrežja najprej in čimbolj omejiti hitro širjenje okužbe.

# 3. SITUACIJA NA PODROČJU

## 3.1 K-Shell dekompozicija

Danes se za iskanje vplivnih vozlišč uporablja predvsem tako imenovana K-Shell dekompozicija, ki se izvaja iterativno in je sestavljeno iz n enostavnih korakov.

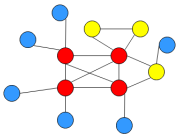


**Figure 1: Probabilities of 1-mers.**

V prvem koraku se odstrani vsa 1-stopnjska vozlišča (imajo 1 povezavo ali manj, modra vozlišča) in njihove povezave. V drugem koraku algoritem se odstrani vsa vozlišča z dvema povezava ali manj ter njihove povezave. Tako nadaljujemo tudi v i-tem koraku, kjer i predstavlja število povezav, ki jih vozlišče ima (rumena vozlišča). Tako nam na koncu ostanejo tako imenovana jedrna vozlišča, ki predstavljajo vplivna vozlišča (rdeča vozlišča).

K-shell metoda, pa ima dve resni pomankljivosti pri iskanju vplivnih vozlišč.

Prvo, kljub temu, da je metoda algoritmično zelo enostavna, določanje indeksa k-lupine (k-shell) zahteva tako globalno znanje topologije omrežja in številne iteracije. Ker je algoritem iterativne narave in je izvajalni čas dolg, ta metoda ni uporabna za aplikacije, ki delujejo realnem času. To so na primer varnostne aplikacije.

Druga slabost pa je ta, da je k-shell dekompozicija pogosto neuspešna pri vzpostavljanju monotonega razmerja med k-jem in skupnim okuženim območjem, kar ima lahko hude posledice. Administator, ki ima omejen čas in sredstva bi rad v svojem času izklopil tista vozlišča, ki imajo največjo zmožnost širitve okužbe. Pri k-shell dekompoziciji, pa se pogosto zgodi, da vozlišča z največjim indeksom k-lupine, niso tudi vozlišča z največjo možnostjo širitve (tega k-shell metoda ne zagotavlja). To težavo bi lahko rešili z večjim številom administratorjev, ki bi v istem času lahko zaustavili več naprav oziroma vozlišč ter tako preprečili hitro širjenje.

## 3.2 Centralnost

Koncepte centralnosti se uporablja pri identificiranju centralnih vozlišč v kompleksnih omrežjih. Za to se po navadi uporablja stopnja vozlišča, število najkrajših poti med vozlišči, ki gredo skozi dano vozlišče, ali razdalja do drugih vozlišč. Z iskanjem centralnih vozlišč naj bi poiskali taka vozlišča, ki imajo mnogo povezav do drugih vozlišč in s tem velik vpliv na povezana vozlišča. Čeprav je intuitivno enostavno razumeti, kaj je centralnost, se je algoritmična definicija izkazala za težak problem. Ko so se spoprijemali z njim, so prišli do več možnih rešitev. To so stopnja vozlišča, je na sredini med čim več vozlišči in minimalna razdalja do vseh vozlišč. Na drugi strani so vozlišča z nizko stopnjo, ki jih zaznavamo kot obrobna.

Stopnja vozlišča je najpreprostejši način za ocenjevanje centralnosti vozlišča. Stopnja vozlišča je enaka številu vozlišč, ki so v neposredni povezavi z danim vozliščem. V omrežju je maksimalna stopnja vozlišča torej enaka številu vozlišč manj ena. V kontekstu komuniciranja predstavlja visoka stopnja vozlišča tako vozlišča, ki izmenjuje sporočila z velikim številom drugih vozlišč in je pomemben pri prenosu informacij iz enega področja omrežja na drugo, tudi med vozlišči, s katerimi ni v neposredni povezavi, temveč samo leži na poti med njimi.

Drug način za ocenjevanje centrolnosti je merjenje, koliko krat se vozlišče nahaja na najkrajši poti med dvema vozliščema. Vozlišče na poti med drugimi vozlišče je pomembno, ker lahko tako vozlišče zadrži al spremeni informacijo. S tem so taka vozlišča pomembna za vzdrževanje komunikacije znotraj omrežja. Seveda ima vozlišče na poti med vozlišči tudi nadzor nad prenosom podatkov in ravno ta nadzor kaže na njegovo centralnost.

Tretji intuitivni način določanja centralnosti temelji na bližini danega vozlišča do vseh ostalih vozlišč. V tem primeru centralnost razumemo kot to, da je vozlišče pod čim manjšim vplivom, oziroma nadzoru, drugih vozlišč. V tem primeru so centralnost enačili z neodvisnostjo. Neodvisnost vozlišča določa bližina drugih vozlišč v omrežju. Bližje, kot je vozlišče ostalim, manj se mora zanašati na druga vozlišča, da se bo njegovo sporočilo pravilno preneslo, oziroma na manjše število vozlišč se mora zanašati.

# 4. NOVA MERA

Basaras, Katsaros in Tassilulas [2] so ustvarili novo mero, $\mu$-PCI ($\mu$-power community index), ki je hibrid med stopnjo vozlišča in k-shell indexom. Za svojo mero trdijo, da je bolj učinkovita od posamezne tradicionalne mere. Uspeli so kvantificirati možnost širjenja na lokalnem nivoju, kar pomeni, da je $\mu$-PCI uporaben za vse vrste omrežij, ne glede na velikost in dinamičnost.

Ta mera uravnoveša centralnost (vozlišča na mnogih poteh med pari vozlišč) in gostoto (ki jo dobimo iz jedrosti). Cilj je, da najdemo vozlišča v gostih predelih omrežja in s tem vozlišča z večjo verjetnostjo, da bodo širila informacije. Mero izračunamo po naslednji formuli: $\mu$-PCI za vozlišče $v$ je enako $k$, kjer je do $\mu * k$ vozlišč v $\mu$-soseski (preko koliko povezav gremo) $v$ s stopnjo $\leq k$.

S preizkusi so testirali svojo novo mero in jo primerjali s k-shell dekompozicijo in stopnjo vozlišča na dveh omrežjih. Uporabili so model dovzeten-nalezen-ozdravljen za simulacijo širjenja okužbe tako od enega vozlišča, kot od več vozlišč naenkrat. Ugotovili so, da najboljše rezultate dobijo pri $\mu \leq 2$, pri večjih vrednostih $\mu$ se pa rezultati pokvarijo. Nadalje

so ugotovili, da $\mu$-PCI, za razliko od k-shell dekompozicije, monotono napoveduje možnost širjenja in vozlišča z visoko vrednostjo maksimizira širjenje vpliva.

# 5. PRIMERJAVA

Tukaj bi lahko naredili tak eksperiment, kot so ga v originalnem članku.

Da so preverili pravilnost tehnike $\mu$-PCI, so jo primerjali s k-shell dekompozicijo in izhodiščno meritvijo, ki temeli zgolj na stopnji vozlišča, na številnih kompleksnih omrežjih.

Tukaj so predstavljene najpomembnejše ugotovitve dveh dobro poznanih omrežij ca-CondMat in ca-AstroPh, ki sta kolaborativni omrežji iz e-print arXiv in pokrivata fiziko trdne snovi in astrofiziko. Podatki so preneseni s strani Stanford Network Analzsis Platform (http://snap.stanford.edu/data). Tabela 1 predstavlja glavne karakteristike navedenih omrežij.

| Omrežje | Tip | Število vozlišč | Število povezav | Verjetnost okužbe (%) |
|---|---|---|---|---|
| ca-CondMat | Sparse | 23.133 | 186.936 | 8 |
| ca-AstroPh | Dense | 18.772 | 396.160 | 4 |

Tabela 1

Uporabili so model dovzeten-okužen-obnovljen za infekcije z enim izvorom in infekcije z več izvori, da bi raziskali proces širjenja, kot je opisano v poglavju Izvor okužb. Ta model ima tri možna stanja:

- stanje dovzetnosti, v katerem so vozlišča dovzetna za ranljivost
- stanje infekcije, v katerem okužena vozlišča poizkušajo okužiti dovzetne sosede in uspejo z verjetnostjo $\lambda$
- stanje obnovitve, v katerem so se vozlišča obnovila in ne morejo biti ponovno okužena.

Uporabljene so bile relativno majhne vrednosti za $\lambda$, da je poudarjena vloga vplivnih razširjevalcev.

Primerjali so $\mu$-PCI, k-shell dekompozicijo in metodo stopnje vozlišča. Za $\mu$-PCI so predstavljeni le rezultati za $\mu = 1$. Za $\mu = 2$ so rezultati primerljivi, za $\mu > 2$, se pa hitrost metode znatno poslabša. Za predstavitev 1-PCI je uporabljena oznaka km, ks za k-shell in k za vrednost stopnje vozlišča.

Podobno kot pri Kitsak et. al., je za merjenje učinkovitosti uporabljena povprečna velikost okuženega področja omrežja.

Za količinski vpliv enojnega razširjevalnika inf(s), je izračunana povprečna velikost omrežja, ki je okužena z (km, k) parom vrednosti. Povprečen obseg okuženega omrežja je izračunan tako: $INF_{km,k} = \sum_{S \in P_{km,k}} \frac{inf(S)}{N_{km,k}}$, kjer je $P_{km,k}$ množica vseh $N_{km,k}$ razširjevalnikov z enakim (km, k). Isti proces je ponovljen za k-shell dekompozicijo.

Za pridobitev statistično nepristranskih rezultatov, je bilo računanje za vsako točko grafa za en in več izvorov, ponovljeno 1000-krat.

Ugotovljeno je bilo, da 1-PCI kaže stabilno in zanesljivo obnašanje in odpravlja slabosti visokostopenjskih razširjevalnikov in k-shell dekompozicije.

Izbira visokih 1-PCI vozlišč maksimira širjenje vpliva, medtem ko je posledica izbire visokostopenjskega vozlišča ali naključnega vozlišča iz osrednjega jedra slabo širjenje vpliva ali ne maksimira vpliva.

Slika 2 prikazuje sposobnost širjenja vozlišč v ca-CondMat omrežju z enim jedrom okužbe. Obstajajo vozlišča z visokimi indeksi k-shell, od katerih nekatera okužijo velik del omrežja, kakor tudi vozlišča z istim indeksom k-shell, ki okužijo bistveno manjši del omrežja. Pri 1-PCI pa tako obnašanje izkazujejo le vezlišča z zelo majhnim indeksom.

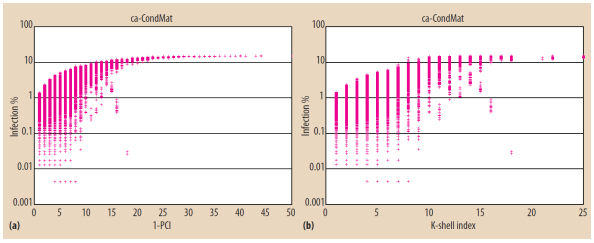



**Figure 2: Primerjava metod 1-PCI in k-shell.**

## 5.1 Izvor okužb

Naša ocena uspešnosti upošteva izvor iz enega jedra okužbe ali pa iz mnogih.

Eno jedro okužbe

Vsa vozlišča so sprva dovzeta za okuženost (vendar ne okužena), razen enega, ki je že v okuženem stanju. To okuženo vozlišče skuša okužiti vse svoje dovzete sosede z verjetnostjo uspešnosti $\lambda$ in svoje stanje nato spremeni v obnovljeno. Vsa vozlišča, ki so okužena, tako enako skušajo okužiti še vse svoje dovzete sosede. Ta proces se nato ponavlja, dokler niso vsa vozlišča v obnovljenem stanju.

Več jeder okužbe

Število vozlišč, ki so na začetku okužena je med 0,5 in 4 odstotki velikosti celotnega omrežja.

# 6. ZAKLJUČEK

Ker se ljudje vse bolj zanašajo na podatke, ki se jih dobi na internetu, je tudi moč vplivnih vozlišč vsak dan večja. Vsaka stvar na internetu, pa ima vedno dve plasti, pozitivno in negativno.

Širjenja uporabnih in pozitivnih informacij je vedno veliko, vendar pa nimajo vsi ljudje istih ciljev. Imamo politike in podjetja, ki ta omrežja že zdaj izkoriščajo za svojo korist in tako blatijo konkurenco. Mnogokrat že sami predstavljajo vplivna vozlišča in tako zelo hitro in učinkovito širijo laži. Koga bodo v tem primeru administratorji odklopili? Tis-

tega, ki več plača ali tistega, ki se bori za pravice ljudi in skuša prikazati korupcijo v vladi?

V prihodnosti lahko zato pričakujemo porast novih in boljših metod za iskanje vplivnih vozlišč. S tem se sicer odpirajo moralne dileme, saj bo s tem dana velika moč administratorjem omrežja. S tem pa bo svoboda velikega interneta še bolj omejena.

## 7. REFERENCES


[1] About linkedin, 2014.

[2] P. Basaras, D. Katsaros, and L. Tassiulas. Detecting influential spreaders in complex, dynamic networks. *IEEE Computer*, 46(4):24–29, 2013.

[3] A. Sedghi. Facebook: 10 years of social networking, in numbers, 2014.

[4] K. Wickre. Celebrating #Twitter7, 2013.

# Network intrusion investigation – Preparation and Challenges

(Digital Forensic)

Branislav Todorov

Jovan Buragev

# Contents


1 Introduction ..... 3

2 Network Intrusion, protection and detection ..... 4

2.1 Common types of network intrusions ..... 4

2.1.1 Spoofing (Identity spoofing or IP Address Spoofing) ..... 4

2.1.2 Sniffing ..... 5

2.1.3 Mapping (Eavesdropping) ..... 5

2.1.4 Hijacking (man-in-the-middle attack) ..... 5

2.1.5 Trojans ..... 5

2.1.6 Denial-of-Service attack (DoS) and Distributed-Denial-of-Service (DDoS) ..... 5

2.1.7 Social Engineering ..... 6

2.2 Prevention of the system ..... 7

2.2.1 Intrusion detection and prevention systems ..... 7

2.2.2 Preventing system from intrusions ..... 8

3 Intrusion Investigation ..... 9

3.1 Identifying the Intrusion (Intrusion Detection Methods) ..... 10

3.1.1 Manuel Intrusion Detection Methods ..... 10

3.1.2 Intrusion Detection Systems (IDS) ..... 10

3.2 Collecting and Analysis the Evidence ..... 11

3.2.1 System Log Analysis ..... 11

3.2.2 File Integrity Assessment (FIA) ..... 11

3.2.3 Intruder Artifacts ..... 12

3.2.4 Historical Analysis ..... 12

# 1 Introduction

A Network attack or security or security incident is defined as a threat, intrusion, denial of service or other attack on a network infrastructure that will analyze your network and gain information to eventually cause your network to crash or to become corrupted. In many cases, the attacker might not only be interested in exploiting software applications, but also try to obtain unauthorized access to network devices. Unmonitored network devices are the main source of information leakage in organizations. In most organizations, every email message, every web page request, every user logon, and every transmittable file is handled by a network device. Under some setups, telephone service and voice messaging are also handled by network devices. If the attacker is able to "own" your network devices, then they "own" your entire network. Network attacks cut across all categories of software and platform type.

# 2 Network Intrusion, protection and detection

There are many ways attacks and intrusion into computer systems that can be performed by the intruder. When we say intrusion, we think about unauthorized access of the intruder into the computer system no matter if the intrude is for good or bad reason. All intrusions are treated like breaking the human rights.

Every average user of computer or computer system knows the term malicious code (malware), which is name for all computer programs that can be used for attacking.

Serious and extensive intrusions, most of the times, are faced by the big companies where the motive is financial, competition or political, while with little and less serious intrusions and attacks users of personal computers are facing every day.

Among the computer systems attacks and intrusions we didn't recognize only attacks with malicious code, but also attacks or intrusions into computer systems trough the network. Most known intrusions can be divided into two groups:

- Active attacks – viruses, worms, Trojan horses and root kits
- Passive attacks/intrusions – wiretapping, fake identity, intrusion with password, denial of service attacks, man in the middle attacks and attacks on the application layer

In active attacks the attacker directly changing or deleting the data on computer system and with that he is making damage. Usually the attacker setting some malicious code which is doing some activity on target system. In passive attacks or intrusion the intruder first observe the target system and then use the knowledge to do the attack. It's not needed to carry any malicious code, often this kind of intrusions is used for getting or changing confidential data. Above we will discuss more about passive type of intrusions in computer systems.

## 2.1 Common types of network intrusions

1) Spoofing.
2) Sniffing.
3) Mapping.
4) Hijacking.
5) Trojans.
6) DDoS.
7) Social engineering.

### 2.1.1 Spoofing (Identity spoofing or IP Address Spoofing)

If the attacker obtains control over the software running on a network device, they can then easily modify the device's protocols to place an arbitrary IP address into the data packet's source address field. This is known as IP spoofing, which makes any payload appear to come from any source. With a spoofed source IP address on a datagram, it is difficult to find the host that actually sent the datagram.

### 2.1.2 Sniffing

Packet sniffing is the interception of data packets traversing a network. A sniffer program works at the Ethernet layer in combination with network interface cards (NIC) to capture all traffic traveling to and from internet host site. Further, if any of the Ethernet NIC cards are in promiscuous mode, the sniffer program will pick up all communication packets floating by anywhere near the internet host site.

The key to detecting packet sniffing is to detect network interfaces that are running in promiscuous mode. Sniffing can be detected two ways:

- Host-based: Software commands exist that can be run on individual host machines to tell if the NIC is running in promiscuous mode.
- Network-based: Solutions tend to check for the presence of running processes and log files, which sniffer programs consume a lot of. However, sophisticated intruders almost always hide their tracks by disguising the process and cleaning up the log files.

The best countermeasure against sniffing is end-to-end or user-to-user encryption.

### 2.1.3 Mapping (Eavesdropping)

Before attacking a network, attackers would like to know the IP address of machines on the network, the operating systems they use, and the services that they offer. With this information, their attacks can be more focused and are less likely to cause alarm. The process of gathering this information is known as mapping.

### 2.1.4 Hijacking (man-in-the-middle attack)

This is a technique that takes advantage of a weakness in the TCP/IP protocol stack, and the way headers are constructed. Hijacking occurs when someone between you and the person with whom you are communicating is actively monitoring, capturing, and controlling your communication transparently. Man-in-middle attacks are like someone assuming your identity in order to read your message. The person on the other end might believe it is you, because the attacker might be actively replying as you, to keep the exchange going and gain more information.

### 2.1.5 Trojans

These are programs that look like ordinary software, but actually perform unintended or malicious actions behind the scenes when launched. Most remote control spyware programs are of this type. The number of trojan techniques are only limited by the attacker's imagination. A torjanizes file will look, operate, and appear to be the same size as the compromised system file.

The only protection is early use of a cryptographic checksum or binary digital signature procedure.

### 2.1.6 Denial-of-Service attack (DoS) and Distributed-Denial-of-Service (DDoS)

A denial of service attack is a special kind of Internet attack aimed at large websites. It is a type of attack on a network that is designed to bring the network to its knees by flooding it with useless traffic. Denial of Service can result when a system, such as a Web server, has been flooded with illegitimate requests, thus making it impossible to respond to real requests or task.

Common forms of denial of service attacks are,

**1) Buffer Overflow Attacks**

The most common kind of DoS attack is simply to send more traffic to a network address than the programmer's expectation on size of buffers.

**2) Smurf Attack**

In this attack, the perpetrator sends an IP ping request to a receiving site. The ping packet specifies that, it is broadcast to a number of hosts within the receiving site's local network. The packet also indicates that the request is from another site, which is the target site that is to receive the denial of service attack. The result will be lots of ping replies flooding back to the innocent, spoofed host. If the flood is great enough, the spoofed host will no longer be able to receive or distinguish real traffic.

**3) SYN floods**

Here is used TCP/SY N packets like tool for attack. The address of the client is often forged so that when the server sends a TCP/SYN-ACK packet back to the client, the message is never received from client because the client either doesn't exist or wasn't expecting the packet and subsequently ignores it. This leaves the server with a dead connection, reserved for a client that will never respond. Usually this is done to one server many times in order to reserve all the connections for unresolved clients, which keeps legitimate clients from making connections.

**Distributed Denial-of-Service attacks (DDoS)**

A distributed denial of service attack (DDoS) occurs when multiple compromised systems or multiple attackers flood the bandwidth or resources of a targeted system with useless traffic. These systems are compromised by attackers using a variety of methods.

In DDoS attacks, the attacker first gains access to user accounts on numerous hosts across the Internet. The attacker then installs and runs a slave program at each compromised site that quietly waits for commands from a master programs running, the master program then contacts the slave programs, instructing each of them to launch a denial-of-service attack directed at the same target host. The resulting coordinated attack is particularly devastating, since it comes from so many attacking hosts at the same time.

Here also ingress filtering only can control DoS attack and that too to a small extent.

### 2.1.7 Social Engineering

Social engineering is the use of persuasion or deception to gain access to information systems. The medium is usually a telephone or e-mail message. The attacker usually pretends to be a director or manager in the company traveling on business with a deadline to get some important data left on their network drive. They pressure the help desk to give them the toll-free number of the RAS server to dial and sometimes get their password reset. The main purpose behind social engineering is to place the human element in the network-breaching loop and use it as a weapon. The human element has been referred to as the weakest link in network security.

Examples of social engineering.

1) **Faked Email**
2) **Fictitious Competition:** Fake competitions on internet
3) **The Helpful Help Desk:** The help desk gets a call from the social engineer impersonating a user reporting a forgotten password.

## 2.2 Prevention of the system

The protection of computer systems in network is divided into several layers. The highest is the personal safety to which each individual must take care of himself. So the individual must ensure security to his hardware in different ways. The next layer is security software system that can provide good protection of attacks and viruses (that kind of software is antivirus software and Intrusion detection system software). In addition, we need to protect data in all sections in the company. It is necessary to make backups and copies of data in order to protect it and store the copies of the data on different places in case of some catastrophe.

### 2.2.1 Intrusion detection and prevention systems

Intrusion detection system (IDS) is software that reviews the system and its vulnerabilities. For this propose IDS requires information of the environment on the basis of which reveals potential attacks against the system, analysis of the system behavior and looking for intrusion attempts. The IDS keep an eye on suspicious events in the system (for example inserting wrong password too many times), keeps log of activities and on defined period scans the entire system to find if there is some potential security vulnerabilities or potential threats. This system scan is performed usually when the system is less loaded.

The Network Intrusion Detection systems are very variable on the market of security products considering the race of products from different companies for network security which protects and support the internet connectivity and commerce. Some of the good known commercial IDS includes RealSecure16 from Internet Security Systems, Sessionwall-317 from Platinum Systems, Netranger18 from Cisco Systems and Cybercop Monitor19 from Network Associates.

There is also many free IDS available on Internet for non-commercial use. These systems includes SHADOW20, Network Flight Recorder21 (NFR) and also SNORT which is one of the most used open source network intrusion prevention and detection system. These freely available systems are not as user friendly as the commercial systems but they are much more customizable. Automated IDS are useful for perimeter defense, however they cannot keep up with the fact that security threats are constantly changing and may differ from enterprise to enterprise. Network based IDS do not address the issue of internal threats to the network and, like firewalls, automated IDS need to be properly monitored and maintained.

### 2.2.2 Preventing system from intrusions

Given the degree of correlation of the system, amount of security of the system increasing in the following order:

1) Computer systems connected on wireless network
2) Computer systems connected on wired network
3) Computer systems temporary connected on wired network
4) Computer systems that are not connected in network

There are several actions that need to be done to successfully reduce the damage which can be obtained from some intrusion, as well the possibility that the attack did occur. These actions include:

- Isolation the data from applications
- Keeping the data on database server
- Daily coping and data backup onto different locations
- "Exit in case of emergency" strategy for resolving the operating system, programs and data
- Using antivirus software
- Updating operating systems and software that is used on computer systems

In case of successful intrusion we must first of all isolate the computer system from all outside communications and try to locate the malicious code and his source. The antivirus programs try to act and prevent an intruder. If we don't success with that procedure, we need to reinstall the operating system and restore the data from backup. Good principle is always to keep to the following ten laws of the security:

1) If the intruder successfully run his program on your system, this system is no longer yours
2) If the intruder change your operating system on your computer, this computer is no longer yours
3) If the intruder have unlimited physical access to your computer system, this computer system is no longer yours
4) If you allow the intruder to upload data to your website, this website is no longer yours
5) Weak password require strong security
6) Computer system is secured as far as his administrator reliability
7) The data is secured as far as the security of the key for decryption
8) No updated virus protection is important little more than no virus protection at all
9) Absolute anonymity doesn't exist in real life, also doesn't exist on internet
10) Technology is not a cure-all

# 3 Intrusion Investigation

There are four main types of investigation performed by digital forensics specialists. The first three are broadly similar in the activities the involve, but differ in terms of the legal restrictions and guidelines imposed as well as the type of digital evidence and form of report.

- **Criminal forensics**
  The largest form of digital forensics and falling under the remit of law enforcement (or private contractors working for them). Criminal forensics is usually part of a wider investigation conducted by law enforcement and other specialists with reports being intended to facilitate that investigation and, ultimately, to be entered as expert evidence before the court. Focus is on forensically sound data extraction and producing report/evidence in simple terms that a lay man will understand.
- **Intelligence gathering**
  This type of investigation is often associated with crime, but in relation to providing intelligence to help track, stop or identify criminal activity. Unless the evidence is later to be used in court forensic soundness is less of a concern in this form of investigation, instead speed can be a common requirement.
- **Electronic discovery (eDiscovery)**
  Similar to "criminal forensics" but in relation to civil law. Although functionally identical to its criminal counter part, eDiscovery has specific legal limitations and restrictions, usually in relation to the scope of any investigation. Privacy laws (for example, the right of employees not to have personal conversation intercepted) and human rights legislation often affect electronic discovery.
- **Intrusion investigation**
  The final form of investigation is different from the previous three. Intrusion investigation is instigated as a response to a network intrusion, for example a hacker trying to steal corporate secrets. The investigation focuses on identifying the entry point for such attacks, the scope of access and mitigating the hackers activities. Intrusion investigation often occurs "live" (i.e. in real time) and leans heavily on the discipline of network forensics.

Evidence for intrusion investigation from where they are funded are separated on 3 broad categories: host based evidence, network based evidence and the first person evidence of witnesses. Collection of data generated by system, network, application and activities by the user is primary for detecting signs of intrusion, conducting real-time investigations and preserving evidence so that it is admissible in court.
Of critical importance is maintaining appropriate records of what has been observed or discovered. A written chronological event log of the intruders suspected activities and site responses should be maintained. The log should detail what is suspected, what actions were taken, who was contacted as a result and what evidence of the intruders activities was located.
Resource utilization including man-hours and equipment used to re-establish the system and locate the perpetrator should also be detailed to assist in later developing a victim impact statement.

## 3.1 Identifying the Intrusion (Intrusion Detection Methods)

Intrusion detection falls into two categories, manual intrusion detection, such as log analysis and manual correlation, and the use of automated Intrusion Detection Systems (IDS). The optimal solution is normally a combination of both mechanisms as IDS do not and will not detect all possible attacks.

### 3.1.1 Manuel Intrusion Detection Methods

Knowledge of, and the ability to competently apply, manual intrusion detection techniques is essential to comprehensive network security. IDS, which rely on various techniques including attack signature recognition and system/network anomaly detection, can lull management and administrators into a false sense of security. Firewalls may engender a similar false sense of security, particularly if they are not managed properly.
Manual intrusion detection relies on the extensive knowledge an administrator has about his or her network. Applying manual intrusion detection methods in a time critical environment such as a post-intrusion scenario requires the sysadm/ISSO to be familiar and competent with the tools they will utilize to conduct their examinations and have a consistent set of procedures for conducting their analyses.
Manual intrusion detection is conducted by:

- identification of anomalous entries in system logs,
- verification of the integrity of critical system files,
- identification (and preservation) of suspicious users, processes, files and other intruder remnants,
- analysis of network transaction records, and
- correlation of all relevant data to develop a picture of what has occurred.

Not all attacks or suspicious activities need to be investigated in the same way, so tools utilized to assist in an investigation must allow the user to access various levels of detail according to their needs. This requires the sysadm/ISSO to maintain a comprehensive suite of audit, data recovery and analysis tools to provide the flexibility necessary to deal with the activities of an unknown intruder.

### 3.1.2 Intrusion Detection Systems (IDS)

The IDS market place is very volatile given the movement of commerce to the Internet and the race to market network security products to protect and support Internet connectivity and commerce. Some of the better known commercial IDS include RealSecure16 from Internet

Security Systems, Sessionwall-317 from Platinum Systems, Netranger18 from Cisco Systems and Cybercop Monitor19 from Network Associates.

There are also a number of free IDS available on the Internet for non-commercial use. These systems include SHADOW20 and Network Flight Recorder21 (NFR). These freely available systems are not as user friendly as the commercial systems but they are much more customizable.

Automated IDS are useful for perimeter defense, however they cannot keep up with the fact that security threats are constantly changing and may differ from enterprise to enterprise.

Network based IDS do not address the issue of internal threats to the network and, like firewalls, automated IDS need to be properly monitored and maintained.

## 3.2 Collecting and Analysis the Evidence

There is often pressure on a digital investigator to resolve the problem quickly. Relevant log files and state tables might be erased at any moment and the system owners/users want to gain access to the information on the system. It is often necessary to interpret digital evidence instantaneously to determine where additional evidence might be found. In addition, there is often pressure from the victim to get answers quickly.

### 3.2.1 System Log Analysis

Log files contain information about what activities have occurred over time. They are often the only record of suspicious behavior making them probably the most crucial form of evidence in an intrusion investigation. There must be recognition at a high level within an organization that system log files are in fact crucial records for an organization and they must be treated as such.

Different systems provide different types of logging information and some systems do not provide adequate logging in their default configuration. Some “trusted” operating systems however, such as those accredited by national security agencies for classified use, produce a much larger quantity of logs with a consequently higher degree of detail.

Sysadms/ISSOs should identify, prior to any intrusion, the types of logs and logging mechanisms available to each system (file access logs, process logs, network logs, application specific logs, etc.) and identify the data recorded within each log. These logging mechanisms should then be enabled to the maximum extant possible. Failure to enable these mechanisms may seriously impact on a site's ability to determine whether or not an intrusion has been attempted or in fact succeeded.

Simply enabling logging is not enough however. A site must have the necessary procedures and tools available to process and analyze the products of logging. There are many tools that can assist in this process.

### 3.2.2 File Integrity Assessment (FIA)

Intruders, as part of their normal Modus Operandi, will alter the configuration of a system to allowed continued access, conceal their presence and carry out their goals on the system. If no record of the system's baseline configuration is maintained determination of what modifications have been made by an intruder to the system will be difficult. File Integrity Assessment (FIA), through the use of cryptographic checksums, creates a baseline database of a system, and then allows the administrator to monitor files for unauthorized changes. System files should not change, except when updated or patched, and log files should only grow in size.

Forensic application of FIA allows administrators to build a complete list of what has been altered on a system. The FIA of the altered system will form an essential part of the evidence supporting prosecution, particularly where legislation imposes greater penalties for the insertion, alteration or deletion of data in a system as is the case with Australian Commonwealth law.

After a compromised system is backed up using binary imaging as described later, a FIA should be run to provide a baseline for the evidence to be handed to LE. This guarantees that no matter how many tests or analyses are done on restored image of the system subsequently, the integrity of the altered system is maintained. This way it can be demonstrated that the file system or backup tape is unaltered at the point in time in the future when the case is ultimately prosecuted.

### 3.2.3 Intruder Artifacts

Intruders often leave all sorts of files on the systems that they compromise. These can range from sniffer log files, stolen password files, exploit scripts, and source code to various programs. Smarter and more experienced intruders will often only leave processes running in memory. Generically, programs that have been left behind by intruders are called remnant files.

Potentially malicious scripts, source code and programs are referred to as artifacts. Some of these files may not, in fact, be malicious however they must always be treated as such until they have been comprehensively analyzed.

Intruders normally replace system files with other files that differ in operation from the original program, but have the same name. These trojan programs are popular among intruders as they offer a method of concealment for their activities. The trojan programs themselves may also provide valuable information or functionality to support further intrusions.

Many trojan programs are identical to their original namesake with respect to all file system attributes except content, meaning that only a proper cryptographic checksum analysis can detect a difference between files. Keeping off-line, read-only lists of checksums of system files and important programs is therefore essential. Cryptographic checksums of the artifact/s on the original system are also essential when giving copies to LE in order to validate the integrity of the copy of the artifact provided. If artifacts have been recovered from a compromised system, LE may sometimes request a site assist them by analyzing the artifact to determine its function. This may prove particularly important if a search warrant is later executed on a suspect and uncompiled Trojan code recovered from their computer system. In some cases the LE agency may in fact provide the victim site with copies of the seized code and request they compare it with the artifacts recovered from the site.

Artifacts should not be analyzed on compromised system/s. Care should be taken to make a copy of the artifact/s plus surroundings that, where possible, exactly mirror the original environment. Ideally artifacts should be analyzed on an isolated system.

### 3.2.4 Historical Analysis

There are a number of benefits to maintaining historical network logs including:

- verifying network security policies are effective,
- detecting attempts to break through firewalls and host-based security mechanisms,
- analyzing network service utilization,
- developing models of network behavior so that deviations can be detected, and
- troubleshooting transient network problems including Denial of Service attacks.

Logs from network infrastructure devices such as routers and switches may also provide an appropriate source of data for network transaction analysis. The issue with these types of logs however comes down to the quantity. If an appropriate parsing mechanism is in place to reduce the quantity to a manageable level then these logs may provide a valuable source of evidence.

Tools like Argus and Tcptrace support analysis of network traffic in both textual and graphical formats. Graphical analysis particularly can make identification of anomalous network conditions, such as those that may be encountered during intrusion incidents, easily discernible.

**References:**


1. INTRUSION INVESTIGATION AND POST-INTRUSION COMPUTER FORENSIC ANALYSIS - *Federal Agent Byron S. Collie (Australian Federal Police) Directorate of Information Warfare Headquarters Air Command, Royal Australian Air Force*
2. Intrusion Detection Systems as Evidence - *Peter Sommer Computer Security Research Centre, London School of Economics & Political Science*
3. Different Types of Network Attacks And Security Threats and Counter Measures - http://ayurveda.hubpages.com/hub/Types-of-Network-Attacks
4. Intrusion detection system - http://en.wikipedia.org/wiki/Intrusion_Detection_System

# Postopek forenzične analize omrežja

## Digitalna forenzika 2013/14

Tomaž Borštnik
Fakulteta za računalništvo in informatiko
Tržaška cest 25
Ljubljana, Slovenija
63090025
tomaz.borstnik@gmail.com

Darko Božidar
Fakulteta za računalništvo in informatiko
Tržaška cest 25
Ljubljana, Slovenija
63090023
darko.bozidar@gmail.com

Gregor Čepin
Fakulteta za računalništvo in informatiko
Tržaška cest 25
Ljubljana, Slovenija
63090008
gregor.cepin@gmail.com

## POVZETEK

V članku opišemo postopek forenzične analize produkcijkega omrežja kampusa univerze v Zurichu, katero so izvedli strokovnjaki s področja računalniške varnosti. Za analizo prometa v omrežju je bil uporabljen odprtokoden program Snort, ki je namenjen spremljanju prometa v realnem času. Za oceno ranljivosti naprav v omrežju sta bila uporabljena plačljiv program Nessus in odprtokodni program OpenVas. Poleg naštetih programov sta bila za analizo omrežnega prometa uporabljena še dva postopka. Ta postopka oziroma vira podatkov sta preverjanje s črnimi listami in preverjanje z iskalnikom Google. Analizirana je bila uspešnost posameznih virov podatkov in njihovih kombinacij pri zaznavanju zlonamerne programske opreme. Opisana je zgradba signatur Snort in njihov pomen pri odkrivanju zlonamerne dejavnosti v omrežju.

Raziskovalci izdelajo tudi korelator, katerega primarni namen je grupiranje sorodnih opozoril. Z njegovo uporabo so zmanjšali število lažnih opozoril, zaradi česar so lažje analizirali rezultate. Iz zbranih podatkov je bilo ugotovljeno, kakšne lastnosti imajo dobre signature Snort. Na podlagi tega so bila podana priporočila za pisanje učinkovitih signatur za zaznavanje zlonamerne dejavnosti.

Poleg tega predstavimo tudi najpogostejše tipe zlonamerne programske opreme, načine njihovega razširjanja in posledice njihovega delovanja.

## Ključne besede

forenzična preiskava, zbiranje dokazov, omrežje, IDS, napadi

## 1. UVOD

Preiskava varnostnih incidentov običajno zajema več vidikov: 1) potrebno je pridobiti in združiti dokaze iz razpoložljivih varnostnih virov, 2) potrebno je razumevanje, kako sumljiva zlonamerna programska oprema deluje in 3) potrebno je razumevanje, kako lahko zlonamerna programska oprema izkoristi informacije o infrastrukturi in konfiguraciji prizadetega omrežja.

Za boljše razumevanje procesa forenzične analize in preučevanja varnostnih incidentov smo preučili članek *Understanding Network Forensics Analysis in an Operational Environment*. Raziskovalci so v sodelovanju z oddelkom IT njihove fakultete izvedli eksperiment, v katerem so štiri tedne opazovali in analizirali produkcijsko omrežje njihove univerze. Pri tem so ročno (brez avtomatizacije) preiskali 200 varnostnih incidentov. Preiskava je zajemala podatke iz štirih virov: 1) opozorila sistema za zaznavanje vdorov Snort, 2) poročila štirih orodij za skeniranje in ocenjevanje ranljivosti, 3) pet neodvisnih črnih list in 4) iskalnik Google.

Pri preiskavi pridejo do več ugotovitev oziroma naredijo več prispevkov. Najprej podajo postopek, kako uporabiti prej omenjene štiri vire podatkov za oddaljeno diagnosticiranje okužb v realnem času. Za boljše razumevanje postopka ročnega preiskovanja nato ovrednotijo vsakega izmed štirih virov podatkov. Namreč podajo rezultate, v koliko procentih primerov je posamezen vir podatkov vseboval dokaz, na podlagi katerega so lahko sklepali, ali je sumljiv gostitelj okužen. Za najbolj uporabna sta se izkazala Snort in iskalnik Google. Poročila ranljivosti in pregledovanja so se izkazala za uporabna predvsem pri diagnosticiranju kompleksnejše zlonamerne programske opreme. Za najmanj uporabne so se izkazale črne liste. Podali so tudi poročilo, katere kombinacije prej omenjenih virov podatkov so bile uporabljene pri diagnosticiranju različnih tipov zlonamerne programske opreme.

Poleg opisanega postopka diagnosticiranja in vrednotenja podajo tudi 138 signatur programa Snort, katere so zaznale zlonamerno programsko opremo brez proženja lažnih opozoril. Analizirajo tudi razliko med dobrimi in slabimi signaturami programa Snort. Ugotovijo, da dobre signature v povprečju pregledajo 23.5 Bajtov podatkov, kateri zajemajo 2.8 različnih polj paketa, med tem ko navadne signature pregledajo le 11 Bajtov, kateri zajemajo le 1.2 polja. Opazili so tudi, da dobre signature uporabljajo zamik bajtov, regularne izraze, pregledujejo pakete natanko določene velikosti in specifična ciljna vrata veliko bolj pogosto kot navadne signature. Na podlagi dobljenih rezultatov podajo karak-

teristike dobrih signatur. Te so koristne pri konfiguriranju programa Snort in kot smernice pri pisanju signatur [6].

# 2. ZLONAMERNA PROGRAMSKA OPREMA

Zlonamerna programska oprema (ang. *malware*, *malicious software*) je programska oprema namenjena oviranju delovanja sistemov, zbiranju zasebnih podatkov ali pridobivanju dostopa do zasebnih sistemov. Pred podrobnejšim pregledom analize omrežja unverze poglejmo nekaj največjih skupin zlonamerne programske opreme [10].

## 2.1 Računalniški virusi

Računalniški virus je zlonamerna programska oprema, katera ima sposobnost repliciranja. To tipično stori s pomočjo vstavljanja kopij samega sebe v obstoječe računalniške programe, datoteke ali zagonske sektorje trdih diskov. Ko je računalnik okužen, virusi izvajajo različne zlonamerne aktivnosti kot so odvzemanje prostora na disku, odvzemanje procesorskega časa, dostopanje do zasebnih informacij, uničevanje ali spreminjanje podatkov, itd. Širijo se preko interneta ali preko prenosnih medijev.

## 2.2 Računalniški črvi

Podobno kot računalniški virusi imajo tudi računalniški črvi (ang. *computer worm*) sposobnost repliciranja samega sebe. Za razliko od virusov se ne pripenjajo oziroma vrivajo v obstoječe programe, ampak tečejo kot samostojni procesi. Poleg tega tipično povzročajo škodo na omrežju (npr. odvzemanje pasovne širine) in ne spreminjajo gostiteljevega sistema, med tem ko virusi tipično uničujejo oziroma spreminjajo datoteke na okuženem sistemu. Črvi lahko izvajajo tudi druge škodljive aktivnosti kot so brisanje datotek, šifriranje datotek (poglavje 2.4), ali pošiljaje nezaželene pošte. Najpogosteje se širijo preko interneta.

## 2.3 Trojanski konji

Trojanski konj je zlonamerna programska oprema, katera se za razliko od računalniških virusov in črvov ne more replicirati. Tipično vsebujejo zlonamerno kodo za krajo zasebnih podatkov ali uničevanje oziroma spreminjanje sistema. Ime so dobili iz grške mitologije (zgodba o skrivanju vojakov znotraj lesenega konja ob napadu Troje), saj se navadno predstavljajo kot legitimni programi. Tako uporabnike prepričajo, da jih ti prostovoljno namestijo na svoje sisteme.

## 2.4 Progami za odkupnino

Program za odkupnino (ang. *ransomware*) je zlonamerna programska oprema, katera omeji lastniku dostop do okuženega sistema. Za povrnitev dostopa zahteva odkupnino, katera mora biti nakazana na transakcijski račun ustvarjalca programa. Omejitev dostopa tipično izvede s šifriranjem uporabnikovih podatkov. V nekaterih primerih tudi zaklene sistem. Šifriranje se navadno izvede z napadalčevim javnim ključem, tako da samo napadalec pozna pripadajoči privatni ključ, kateri lahko dešifrira podatke.

## 2.5 Programi za korenski dostop

Programi za korenski dostop (ang. *rootkit*) so programi, kateri prikrivajo procese in programe pred programi za detekcijo in jim omogočajo korenski oziroma administrativni dostop do sistema. Za namestitev tipično zahtevajo korenski dostop, ki ga lahko napadalci dobijo z izkoriščanjem ranljivosti v sistemu, ukradenim geslom, ipd. Odstranjevanje programov za korenski dostop je zelo zapleten postopek. V nekaterih primerih zahteva celo ponovno namestitev operacijskega sistema.

## 2.6 Vohunska programska oprema

Vohunski programi (ang. *spyware*) so namenjeni zbiranju zasebnih informacij uporabnikov okuženih sistemov brez njihove vednosti. Pridobljene zasebne podatke (npr. navade brskanja po internetu, gesla, informacije o bančnih računih, ipd.) lahko tipično bodisi pošljejo drugim entitetam ali jih uporabijo za pridobitev nadzora nad sistemom brez uporabnikove vednosti. Nekateri vohunski programi lahko tudi spremenijo nastavitve sistema.

## 2.7 Oglaševalska programska oprema

Oglaševalska programska oprema (ang. *adware*) je namenjena samodejnemu prikazovanju oglasnih sporočil z namenom ustvarjanja dobička njihovih avtorjev. Oglasna sporočila se lahko prikazujejo na uporabniškem vmesniku ali v namestitvenih čarovnikih. Oglaševalska programska oprema lahko tudi analizira uporabnikove vzorce brskanja na spletu. Na podlagi pridobljenih informacij lahko prikazuje primernejša oglasna sporočila.

## 2.8 Omrežje računalniških robotov

Omrežje računalniških robotov (ang. *botnet*) je skupina programov, kateri medsebojno komunicirajo in pri tem izvajajo neko skupno aktivnost oziroma nalogo (npr. pošiljanje nezaželene elektronske pošte, napadi DDOS, ipd.). Omrežje računalniških robotov se tipično koordinira s pomočjo ti. *command-and-control* (*C&C*) strežnikov.

## 2.9 Izogibanje avtentikacije

Za razliko od programov opisanih v tem poglavju izogibanje avtentikacije (ang. *backdoor*) ni zlonamerna programska oprema. Gre namreč za (kot že ime pove) metodo izogibanja avtentikacije, s čimer si napadalec zagotovi npr. oddaljeni dostop do računalnika, dostop do datotek, ipd. Pri tem napadalec ostane prikrit. Najpogosteje je implementirana v obliki programa, lahko pa se tudi izvede v obliki programov za korenski dostop (poglavje 2.5).

# 3. PRIDOBIVANJE PODATKOV

Raziskovalci so spremljali infrastrukturo omrežja *Swiss Federal Institute of Technology at Zurich (ETH Zurich)*. Sistem je zelo raznolik, saj vključuje strežnike, prenosnike, namizne računalnike in ostale laboratorijske in pisarniške naprave. Promet, katerega so nadzorovali, ni imel omejitev. Namreč uporabniki so si lahko namestili katerokoli programsko opremo na svoje naprave. Preiskovalci so omrežje opazovali štiri tedne in sicer med 1. in 28. aprilom 2011. V tem času so opazovali aktivnosti *28.665* uporabnikov.

Izbrali so štiri vire podatkov, kateri podajajo vpogled nad varnostjo gostiteljev iz različnih zornih kotov (promet, storitve in ranljivosti). Najprej so pridobili podatke sistema IDS (sistem za zaznavanje vdorov, ang. *intrusion detection system*). IDS so postavili med primarnim usmerjevalnikom in požarnim zidom sistema. Ta je nadziral podatke v obe smeri

(prihodni in odhodni promet). Na podlagi prometa sistema je nato generiral opozorila o zlonamernih signaturah.

## 3.1 Uporabljena programska oprema

Za pridobivanje podatkov o storitvah in aplikacijah nameščenih na gostiteljih znotraj omrežja, so raziskovalci izvedli pregled omrežja z uporabo programov *whois* in *nmap*. Ko so določili dostopne storitve gostitelja, so preiskali ranljivosti s pomočjo orodij *Nessus* (poglavje 3.1.2) in *OpenVas* (poglavje 3.1.3). Tako so pridobili bolj natančen vpogled gostiteljev v omrežju. S pomočjo črnih list in iskalnika *Google* so pridobivali informacije o zunanjih oziroma oddaljenih gostiteljih. Uporabili so črne liste petih ponudnikov, ki pokrivajo različne tipe zlonamernih gostiteljev (aktivni napadalci, pošiljatelji nezaželene/vsiljene pošte, ...).

### 3.1.1 Snort

Snort je odprtokodni IDS z velikim številom registriranih uporabnikov in največjo aktivno skupnostjo. Izvaja analizo prometa v realnem času in beleženje paketov v omrežjih IP. Analizira lahko uporabljene protokole, išče po vsebini in primerja znane vzorce z vzorci v vsebini. Zazna lahko različne napade, kot so pomilniški preliv (ang. *buffer overflow*), prikrito skeniranje vrat, napade CGI, preverjanje šibkosti datotečnega sistema, preverjanje operacijskega sistema, itd.

Možni so trije načini delovanja:

1. **sniffer** - prestreže omrežne pakete in jih izpiše v konzolo,
2. **beleženje** - prestrežene omrežne pakete zapiše na disk,
3. **NIDS** - zaznava napade, preverja omrežni promet in ga analizara s pomočjo vnaprej predpisanih pravil. Ta določajo, kaj naj se preveri pri omrežnih paketih in pri katerih vrednostih v paketu se sproži alarm.

V načinu delovanja NIDS se prejeti paketi primerjajo s pravili Snort. V pravilu Snort je zapisano, kakšna mora biti vsebina paketa, da ga Snort označi kot škodljivega. Zapisana je tudi akcija, ki se izvede v tem primeru.

Običajna polja Snort pravila so: *akcija protokol izovrni_IP izvorna_vrata ciljni_IP ciljna_vrata (možnosti)*. Ko prispe nov paket, se najprej primerjata izvorni in ciljni IP ter izvorna in ciljna vrata z vsemi pravili v množici pravil. Če pride do ujemanja v prejetem paketu in pravilu, se preverijo še ostale možnosti. Če se tudi te ujemajo, se izvede akcija zapisana v pravilu. Polje protokol določa protokol, za katerega velja to pravilo. Podprti protokoli so $IP$, $ICMP$, $TCP$, in $UDP$. Namesto enega naslova IP lahko v pravilu navedemo tudi seznam naslovov. Podobno velja za vrata.

Obstajajo 4 večje kategorije pravil:

**General** - Vsebujejo informacije o pravilu, a ne vplivajo na njegovo delovanje pri preverjanju omrežnih paketov.

**Payload** - Preverjajo podatke znotraj paketa, ki dejansko prenašajo vsebino.

**Non-Payload** - Preverjajo kontrolne podatke v glavi paketa, med drugim tudi izvorni in ciljni naslov IP.

**Post-detection** - Določajo, kaj naj se zgodi, ko je pravilo izvedlo akcijo.

Zbirko pravil Snort, ki zajema veliko večino napadov, lahko brezplačno prenesemo z njihove spletne strani. Ta ne vsebujejo pravil za preverjanje najnovejših napadov. Za tako zbirko pravil moramo plačevati letno naročnino. V tem primeru naročniki dobijo pravila takoj ob njihovem izidu, kar je 30 dni pred uporabniki brezplačne verzije. Pravila lahko napišemo tudi sami [4] [7].

### 3.1.2 Nessus

Nessus je varnostno orodje, ki je brezplačno za osebno in nekomercialno uporabo. Uporablja se za:

- penetracijske teste,
- preiskovanje ranljivosti,
- pomaga pri zbiranju popravkov,
- popravljanje napačnih varnostnih nastavitev, ki bi lahko privedle do zlorabe sistema,
- odkrivanje omrežij računalniških robotov (poglavje 2.8) in druge zlonamerne programske opreme,
- iskanje občutljivih podatkov, ki bi jih lahko odkrili zlonamerni programi,
- preverjanje delovanja spletnih aplikacij,
- preverjanje uporabe privzetih, praznih ali šibkih gesel,
- preverjanje možnosti napada DoS z neveljavnimi paketi.

Testi ranljivosti so napisani z namenskim skriptnim jezikom imenovanim *NASL* (*Nessus Attack Scripting Language*). Tako lahko napišemo test za posamezno ranljivost in ga uporabimo na različnih napravah, ne glede na operacijski sistem. Obenem zagotavlja, da skripta ne bo povzročila škode na ciljnem sistemu. V primerjavi s skriptami napisanimi v razširjenih programskih jezikih, skripte NASL porabijo manj sistemskih virov [8] [11].

### 3.1.3 OpenVas

*OpenVas* (*Open Vulnerability Assesment System*) je brezplačno orodje zasnovano na orodju *Nessus*, ki je bilo do leta 2005 odprtokodno. Iz takratne različice se je nadaljeval razvoj orodja *OpenVas*. Tudi *OpenVas* uporablja Nessusov skriptni jezik *NASL*. Testi uporabljeni v orodju *OpenVas* so imenovani *NVTs* (*Network Vulnerability Tests*). Zbirko testov lahko dobimo brezplačno. Novi testi se dodajajo dnevno in se brezplačno prenesejo preko spleta [1] [3].

## 3.2 Opozorila IDS

Raziskovalci so konfigurirali IDS s signaturami dveh javno najbolj uporabljenih množic pravil - Vulnerability Research Team (VRT) in Emerging Threat (ET). Aprila 2011 sta množici skupaj vsebovali 37 388 različnih signatur.

Najprej so nad dobljenimi opozorili Snorta uporabili korelator (poglavje 3.2.1). Z njegovo pomočjo so zmanjšali oziroma grupirali opozorila IDS. Nato so s forenzično analizo ročno preučili opozorila in pripadajoče gostitelje. Pregledali so tipe in stopnjo opozoril ter čas incidentov. Nazadnje so iz ostalih podatkovnih virov pridobili še dodatne informacije.

Raziskovalci so morali analizirati veliko količino podatkov. V času analize je sistem zbral 37 milijonov opozoril. Večina jih ni bila neposredno povezana z varnostnimi incidenti. Poleg tega je bilo veliko opozoril lažnih (ang. *false positive*). To so rešili z uporabo prej omenjenega korelatorja, s pomočjo katerega so pridobili majhno število zelo sumljivih dogodkov. Izkazalo se je, da so 37 milijonov opozoril reduciral v 200 združenih dogodkov.

### *3.2.1 Korelator*

Korelator je orodje namenjeno grupiranju opozoril IDS in s tem posledično zmanjšanju lažnih opozoril. Deluje tako, da najprej izločil majhno število dogodkov, ki kažejo na ponavljajoče se zlonamerne vzorce s specifičnimi vrstami opozoril. Nato združi podobna opozorila, ki so časovno blizu. Sledi klasifikacija združenih opozoril v skupine. Te skupine so *neposredni napad* (ang. *direct attack*), *ogrožen gostitelj* (ang. *compromised host*) in *kršitev pravilnika* (ang. *policy violation*). Nazadnje se izvedejo sklepanja in identifikacija gostiteljev.

Raziskovalci so korelator testirali na omrežju kampusa fakultete. V času devetih mesecev so pridobili 8632 milijonov opozoril in odkrili 12,912 okužb. Kot zanimivost podajamo pomembne ugotovitve, katere so pridobili ob testiranju korelatorja [5].

- Od skupaj 91 tisoč različnih aktivnih gostiteljev, jih je bilo 9% vsaj enkrat okuženih z zlonamerno programsko opremo.
- Dnevna verjetnost okužbe strežnika in odjemalca znaša 0.18% in 0.37%.
- Okuženi gostitelji imajo veliko večjo verjetnost nadaljnjih napadov.
- V praksi 5% gostiteljev znotraj omrežja prejme 70% vseh napadov. Strežniki so bolj pogosto napadeni kot odjemalci.
- Neokuženi gostitelji, ki so po naslovu IP sorodni okuženim gostiteljem, imajo večjo verjetnost okužbe.
- Okužena mesta kažejo dnevno ponavljajoče vzorce.
- Časovna analiza je pokazala, da so napadi na strežnike časovno neodvisni. Napadi na odjemalce so konsistentni, bolj opazni in časovno blizu skupaj.

Prišli so tudi do naslednjih prispevkov:

1. Predstavili so hevristiko za detekcijo okužb.
2. Predstavili so metodologijo, ki sistematično preveri sumljive okužbe. S tem so delež lažnih opozoril zmanjšali na 15% (prej 99%).
3. Karakterizirali so 9,163 okužb skozi obdobje devetih mesecev.

## 3.3 Poročila ranljivosti in pregledovanja

Raziskovalci so zbirali podatke o gostiteljih - o njihovih storitvah, programski opremi in posledično ranljivostih. Pridobljene informacije so bile komplementarne informacijam, ki so jih pridobili iz orodja Snort.

Za pridobivanje informacij so uporabili kombinacijo skeniranja omrežja in ocene ranljivosti. Uporabili so orodja kot so *IP sweeps* (pove kateri so aktivni gostitelji ali naslovi IP v omrežju), *NIC whois* in *skeniranje vrat TCP/UPD*. Whois je protokol uporabljen za poizvedovanje po podatkovnih bazah, ki vsebujejo imena domen in naslove IP. Vsebina je uporabniku podana v "berljivem" zapisu. Protokol je natančneje opisan v RFC 3912. S tem so ugotovili, ali je gostitelj izpostavljen zunanjim napadom. Z uporabo omenjenih orodij so tudi ugotovili vlogo gostitelja - DNS, spletni ali poštni strežnik.

Uporabili so *nmap* in s tem pridobili podatke o storitvah na gostiteljih. Poleg tega so pridobili verzijo in tip operacijskega sistema gostitelja, tipe sporočil ICMP, na katere odgovarja in učinkovitost požarnega zidu. Za iskanje ranljivosti so uporabili orodji *Nessus* (poglavje 3.1.2) in *OpenVas* (poglavje 3.1.3).

## 3.4 Črne liste

Z uporabo črnih list so raziskovalci želeli ugotoviti, kateri gostitelji pogosto vzpostavljajo povezavo z zlonamernimi domenami. Uporabili so pet črnih list. Uvrstili so jih v več skupin - aktivnost računalniškega robota (ang. *bot activity*), aktivni napad (ang. *active attack*) in nezaželena elektronska pošta (ang. *spamming*).

Na ta način so preverjali, kdaj se zunanji gostitelji povezujejo na gostitelje znotraj omrežja. Če je prišlo do zadetka, so notranjega gostitelja označili s pripadajočo črno listo. Metoda je učinkovita v primerih, ko sumljivi gostitelji znotraj omrežja pogostokrat vzpostavljajo povezavo z znanimi zlonamernimi domenami. Taka komunikacija se tipično pojavi, ko uporabnik obišče zlonamerne strani ali ko je na gostitelju nameščena zlonamerna programska oprema in poskuša izvajati akcije:

- preusmerjanje do drugih domen,
- posodabljanje jedra programa,
- deljenje ukradenih zaupnih podatkov,
- pridobivanje ukazov iz oddaljene naprave.

## 3.5 Iskalnik Google

Poleg uporabe omenjenih podatkovnih virov so raziskovalci pridobljene informacije preveri tudi na spletu. Gostitelj povezan z zlonamerno aktivnostjo pusti sledi. Te se največkrat

nahajajo na javno dostopnih virih kot so seznami DNS, beležke dostopov do strani, beležke proksijev, liste sledilcev P2P, forumi, ... Za iskanje informacij so uporabili iskalnik Google. Kot vhod so uporabili naslov IP in pripadajoče ime domene lokalnega in oddaljenega gostitelja. Nato so z uporabo oznak na avtomatiziran način preiskali izhod poizvedbe. Iskali so oznake "malware", "spam", "bot", "trojan", "worm", "pop3", "netbios", "banlist", "adaware", "irc", "undernet", "innernet" in "torrent". Te oznake so razkrile specifične akcije, ki jih je izvedel gostitelj (npr. prejemanje ukazov iz strežnika C&C, itd.) [6].

# 4. MEDSEBOJNA ODVISNOST DOKAZOV

V eksperimentu so raziskovalci uporabili opozorila programa Snort. Analizo so omejili na statične naslove IP, saj jih je lažje analizirati. Incidenti odkriti s pomočjo korelatorja so bili označeni z gostiteljevim naslovom IP, pripadajočimi opozorili in časovnim žigom. Razvili so tudi orodja, ki so kot vhod dobila naslov IP preiskovanega gostitelja, na podlagi katerega so pridobila podatke opisane v poglavjih 3.3, 3.4 in 3.5. Pridobljene podatke so prikazala analitikom, kateri so nato vsakodnevno analizirali novo odkrite incidente. Ročna analiza je zahtevala pregled velika števila opozoril programa Snort, pregled kakovosti signatur, zbiranje informacij o zlonamerni programski opremi in pregled medsebojne odvisnosti zbranih informacij. Analitiki so izpostavili 4 reprezentativne primere ročne analize za 4 tipe zlonamerne programske opreme.

## 4.1 Okužba Torpig

Torpig spada v skupino trojanskih konjev (poglavje 2.3) in je zelo razširjen po svetu. Največkrat se prenese s pomočjo ti. Drive-by prenosov. To so prenosi, pri katerih se oseba bodisi ne zaveda posledic prenosa ali se prenos zgodi brez vednosti osebe. Najpogosteje žrtev obišče neko stran, katera pošlje zahtevek na zlonameren strežnik. Ta v odzivu vrne JavaScript program. Program se nato izvede in Torpig se namesti na sistem žrtve. Torpig zaobide protivirusno zaščito s pomočjo programov za korenski dostop (poglavje 2.5). Namenjen je pridobivanju zaupnih informacij žrtve. Poročila ranljivosti in pregledovanja gostiteljev okuženih s Torpigom so pokazala, da imajo ti nameščen operacijski sistem Windows in imajo odprte storitve HTTP, Skype in FTP. To se sklada tudi z njegovim delovanjem, kajti Torpig uporablja protokol HTTP za komunikacijo s strežniki C&C (poglavje 2.8) in za preusmeritev uporabnikov na zlonamerne domene. Najpogosteje se vrine v aplikaciji Skype in CuteFTP. Ob komunikaciji Torpiga s strežniki C&C Snort proži opozorili z identifikatorjema 2002762 in 2003066. Pogostokrat pošilja tudi ukradene podatke preko zahtevkov POST, za katere Snort proži opozorila na intervalu [2404000, 2404300]. Domene, na katere Torpig najpogosteje pošilja ukradene podatke, v iskalniku Google najpogosteje vračajo besedi *trojan* in *bot* [12].

## 4.2 Okužba SdBot

SdBot je program, kateri oddaljenemu napadalcu omogoča neavtoriziran dostop do žrtvinega računalnika. Z njegovo pomočjo lahko napadalec bodisi izvaja napade ali ukrade uporabnikove podatke. Ugotovili so, da naprave okužene s SdBot najpogosteje uporabljajo Microsoftove storitve kot so WebDav, RPC in LSASS. WebDav je razširitev protokola HTTP namenjena urejanju in upravljanju datotek na spletu. RPC omogoča klic oddaljenih procedur oziroma metod. LSAAS je proces na operacijskem sistemu Windows, ki je zadolžen za uveljavitev varnostne politike. SdBot izkoristi pomnilniški preliv LSASS, nepravilno oblikovano sporočilo RPC o pomnilniškem prelivu in ranljivost WebDav, s katero ogrozi žrtev. Za komunikacijo s strežniki C&C uporablja IRC. Za omenjeno aktivnost Snort proži opozorila z identifikatorji na intervalu [2500000, 2500500]. SdBot se širi z iskanjem ranljivih gostiteljev. To stori s skeniranjem vrat (najpogosteje 445). Za opisano skeniranje Snort proži opozorila na intervalu [2011088, 2011089]. Za prenos dodatne zlonamerne programske opreme se SdBot povezuje na različne domene, katere v iskalniku Google najpogosteje vračajo besedi *malware* in *bot* [9].

## 4.3 Okužba Hotbar

Naprave v omrežju univerze so najbolj pogosto okužene z zlonamernim programom Hotbar. Pogostokrat se ga pridobi kot dodatek za brskalnike. V ozadju poskuša ugotoviti uporabnikove vzorce aktivnosti, kot na primer iskalne navade in uporabo aplikacij. Za opisane funkcionalnosti Snort proži opozorila na intervalu [2003305, 2003500]. Hotbar pogostokrat sporoča pridobljene podatke na domene, katere v iskalniku Google najpogosteje vračajo besedo *malware*.

## 4.4 Okužba Koobface

Koobface je zlonamerna programska oprema, katera se širi s pomočjo strani kot so Facebook in MySpace. Najpogosteje se veže na video posnetke objavljene na omenjenih straneh, s čimer želi prepričati uporabnike, da je nujno potreben kodek za predvajanje video posnetkov. Po namestitvi se širi s pošiljanjem sporočil žrtvinim stikom. To program Snort zazna s signaturami z identifikatorji 2009156, 2010335 in 2010700. Domene, na katerih se nahajajo namestitvene datoteke za Koobface, v iskalniku Google vračajo besedi *bot* in *worm* [6].

# 5. KOMPLEMENTARNOST IN OCENJEVANJE VARNOSTNIH VIROV

Tabela 1 prikazuje vire informacij, ki so bili uporabljeni pri identificiranju zlonamerne programske opreme. Zlonamerna programska oprema se v grobem deli na izogibanje avtentikaciji, vohunsko programsko opremo, računalniške črve in Trojanske konje. Pri tem je potrebno poudariti, da ima današnja zlonamerna programska oprema več izmed prej naštetih lastnosti, kar oteži delo pri klasifikaciji.

Pri analizi 30 varnostnih incidentov, raziskovalci niso mogli klasificirati oziroma niso uspeli zagotovo ugotoviti, za katero vrsto zlonamerne programske opreme gre. To jim ni uspelo, tudi če so uporabili informacije iz vseh štirih virov. Pri ostalih 170 varnostnih incidentih je bila klasifikacija uspešna. Ti so vključevali 85 Trojanskih konjev, 59 vohunskih programov, 18 izogibanj avtentikaciji in 8 črvov. Najbolj pogostokrat zaznani programi so bili AskSearch, FakeAV in Simbar. Zadnji štirje stolpci tabele 1 prikazujejo kombinacije virov podatkov, kateri so bili uporabljeni pri identifikaciji pripadajočega tipa zlonamerne programske opreme. Za visoko zanesljivost določanja tipa zlonamerne programske opreme sta bila v 41% primerih potrebna dva vira in v 14% vsaj trije viri. Uporaba več virov je bila potrebna predvsem pri naprednejši zlonamerni programski opremi, kot na primer

**Table 1: Tabela prikazuje uporabljene vire podatkov za identifikacijo različnih tipov zlonamerne programske opreme.**

| Tip zlonamerne programske opreme (št. incidentov) | Programi (št. incidentov) | Beležke IDS | Iskalnik Google | Črne liste | Aktivno skeniranje |
|---|---|---|---|---|---|
| Trojanski konji (85) | FakeAV (27) | x | x | | |
| | Simbar (26) | x | x | | |
| | Monkif (18) | x | x | | |
| | Torpig (10) | x | x | x | x |
| | Nervos (4) | x | x | | |
| Vohunska programska oprema (59) | AskSearch (50) | x | | | |
| | MySearch (9) | x | | | |
| Izogibanje avtentikaciji (18) | SdBot (5) | x | x | x | x |
| | ZBot (5) | x | x | | x |
| | Blackenergy (4) | x | x | x | x |
| | Parabola (2) | x | x | | x |
| | Ramsky (2) | x | x | | x |
| Računalniški črvi (8) | Koobface (6) | x | x | | x |
| | Conficker (2) | x | x | | x |

Torpig, SdBot in Polybot. Pri nekaterih programih, kot sta na primer AskSearch in MySearch, je zadostoval samo 1 vir [6].

Raziskovalci so preslikali proces forenzične analize v odločitveno drevo. Za treniranje klasifikatorja so uporabili rezultate prej opisane ročne preiskave. S tem so prenesli znanje ročno pridobljenih ocen varnostnih incidentov v orodje za podporo pri odločanju, kateri je lahko v veliko pomoč pri diagnosticiranju varnostnih incidentov v prihodnje [2].

Na sliki 1 je prikazano, kako pogosto je vir oziroma kombinacija virov uporabljena za diagnosticiranje okužbe. Prikazana je komplementarnost posameznih virov. Na podlagi eksperimenta so ocenili podane podatkovne vire:

1. **Opozorila IDS:** Opozorila Snorta so najboljši vir informacij in so osnova eksperimenta. Za 170 od 200 preiskanih incidentov so predstavljala uporabne dokaze.

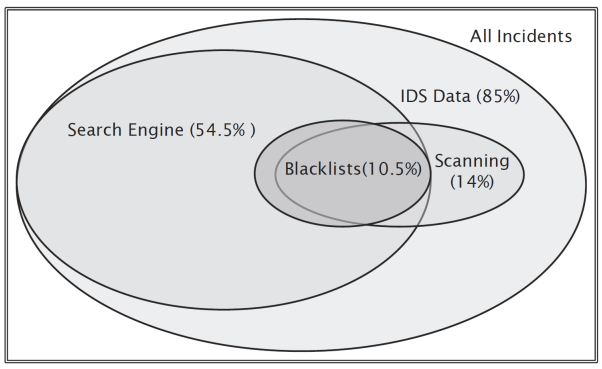



**Figure 1: Komplementarnost podatkovnih virov na preiskanih incidentih**

2. **Iskalnik:** Je drugi najbolj uporaben podatkovni vir. V 54,5% primerov so rezultati iskanja zagotavljali ključne informacije za analizo. Te informacije so bile v večini primerov komplementarne z vsebino opozoril IDS. To so informacije o aktivnih gostiteljih C&C, komunikacijskih kanalih omrežij računalniških robotov, zlonamernih ciljnih straneh, preusmerjanjih na zlonamerne domene in straneh za prenos zlonamerne programske opreme.

3. **Poročila ranljivosti in pregledovanja:** Informacije pridobljene s skeniranjem oziroma pregledovanjem sumljivih gostiteljev so v približno 14% primerih pripomogle pri klasifikaciji zlonamerne programske opreme. Ti primeri so bili najbolj zahtevni, saj vključujejo napredno zlonamerno programsko opremo, kot npr. Torpig ali SdBot.

4. **Črne liste:** Črne liste so prinesle najmanj informacij. Pridobljene informacije so se praktično prekrivale z informacijami pridobljenimi z opozorili IDS ali iskalnikom Google. Še več, vključevale so zelo veliko število lažnih opozoril (ang. *false positives*).

Raziskovalci so nato povzeli bistvo raziskave glede različnih varnostnih virov:

**1. Sklep** Samo en podatkovni vir zelo redko zagotavlja dokaz o incidentu. Z enim obrambnim mehanizmom oziroma virom podatkov lahko detektiramo samo grožnje. Namreč večina modernih groženj kaže kompleksno vedenje in zahteva več nadzornih točk varnostne arhitekture za učinkovito odkrivanje.

**2. Sklep** Senzorji IDS so bili močno kritizirani, ker proizvedejo veliko število lažnih opozoril in zaradi domnevne nezanesljivi. Raziskovalci so ugotovili, da s pomočjo statistike in korelacije opozoril postanejo podatki zelo uporabni, saj pomagajo identificirati kar nekaj primerov, kjer obstaja velika verjetnost okužbe.

**3. Sklep** Uporaba iskalnika je zagotavljala bolj koristno oceno od tradicionalnih virov kot so poročila o ranljivostih in črne liste. V postopku preiskave informacije o oddaljenih gostiteljih zagotovijo pomembne dokaze. Rezultati

iskalnika so bolj posodobljeni, kot informacije vsebovane v črnih listah.

# 6. LASTNOSTI DOBRIH SIGNATUR IDS

Pakete, ki potujejo preko omrežja, lahko primerjamo z vnaprej pripravljeno bazo signatur. Ta vsebuje lastnosti paketov, na podlagi katerih lahko ocenimo, ali je nek paket škodljiv. V nadaljevanju primerjajo lastnosti signatur, ki uspešno odkrivajo zlonamerne pakete, z lastnostmi, ki so pri tem manj uspešne. Identificirane dobre lastnosti uporabijo za prikaz primera dobre prakse pisanja signatur. Ugotovitve temeljijo na 170 primerih zlonamernih paketov, ki so bili ročno pregledani s strani strokovnjakov. S korelacijo so poiskali opozorila IDS, ki se pogosto prožijo skupaj. Nato so poiskali pripadajoče signature za opozorila. Skupaj je bilo najdenih 138 signatur Snort. Nekatere signature so si med seboj zelo podobne in se razlikujejo le v malenkostih. Zato so jih glede na medsebojno podobnost razdelili v 25 skupin. Kot dobre signature so izbrane tiste, za katere veljajo sledeče lastnosti: signatura je vsebovana v skupini dobljeni s korelacijo, signatura je bila uporabljena kot dokazno gradivo v kasnejšem ročnem pregledu in signatura je zajela incident, ki je bil s strani strokovnjaka označen kot zlonamerno delovanje. Signature, ki pogostokrat napačno ocenijo incident in pri zlonamernem delovanju naključno prožijo opozorila tako ne bodo označene kot dobre. V večini primerov bodo filtrirane s koleratorjem. V kolikor korelator tega ne zazna, so take signature odstranjene pri ročnem pregledu.

## 6.1 Prepoznavanje zlonamernih programov

Preučevana zlonamerna programska oprema po okužbi običajno opravi nekaj standardnih opravil. V veliko primerih poskuša kontaktirati strežnik C&C, da bi pridobila nadaljna navodila ali posredovala najdene podatke. V nekaterih primerih preusmeri uporabnika na zlonamerno spletno stran. To stori s spreminjanjem rezultatov v spletnih iskalnikih, z generiranjem pojavnih oken v brskalnikih in s preusmerjanjem zahtev HTTP. Večina obravnavanih virusov in črvov se poskuša širiti, zato iščejo nadaljne ranljivosti sistema, ki bi jih lahko izkoristili. Na podlagi opisanih dejavnosti zlonamerne programske opreme so signature klasificirali v pet kategorij:

- posodabljanje strojne kode zlonamerne programske opreme,
- pošiljanje zaupnih podatkov ustvarjalcu programske opreme,
- prenos dodatnih zlonamernih programov ali posodabljanje obstoječih,
- preusmerjanje uporabnika na zlonamerne spletne strani,
- ugotavljanje ranljivosti za nadaljnje širjenje in izvedba le-tega.

Glede na zgoraj opisane kategorije signature razdelijo v pet skupin. Nato primerjajo lastnosti dobrih signatur z lastnostmi povprečnih. Odkrili so 138 dobrih in 3198 ostalih signatur, ki so prožile opozorila tekom eksperimenta. Dobljeni podatki so zapisani v tabeli 2.

Opazimo lahko, da dobra signatura v povprečju poskuša najti ujemanje 23.5 bajtov podatkov za 2.8 lastnosti, med tem ko ostale signature primerjajo 11 bajtov za 1.2 lastnosti. Poleg tega 28% dobrih signatur nastavi odmik za iskano zaporedje bajtov. Regularne izraze za podrobnejši opis iskanega vzorca jih uporabi 50%. Vrata ciljne destinacije paketa jih pregleda 22%. Ujemanje po velikosti paketa jih opravi 15%. Za ostale signature so te številke sledeče: 8%, 15%, 17% in 7%. V primeru uporabe protokola HTTP so opazili, da dobre signature pogosteje opišejo lastnosti paketa kot so *header*, *method*, *URI* in *revision*. V splošnem kompleksnejše signature dosegajo boljše prepoznavanje zlonamernih paketov.

## 6.2 Dobre prakse

Kompleksnost dobrih signatur običajno prinese potrebo po večji računski moči. Dobre prakse za pisanje signatur IDS velevajo, naj bodo signature čim krajše in preprostejše. Glavni razlog za to je hitro delovanje sistema. Uporaba daljših signatur in s tem posledično več pregledanih lastnosti pomeni počasnejše delovanje in večjo zakasnitev pri procesiranju paketov. Drugi razlog je, da zlonamerna programska oprema pogosto spremeni svoje obnašanje in lastnosti, da bi se izognila odkritju. V kolikor imamo bolj preprosta oziroma splošna pravila, lahko ta zaznajo več različic programske opreme iz iste družine. Rezultati raziskave kažejo, da bolj splošna pravila pomenijo tudi zmanjšanje učinkovitosti prepoznavanja zlonamerne programske opreme. Splošna pravila pogosteje prožijo opozorila v primerih, ko ne gre za zlonamerno programsko opremo. Z bolj podrobnimi pravili dobimo manj opozoril, ki jih strokovnjaki za računalniško varnost lahko pregledajo v krajšem času [6].

# 7. PODOBNE RAZISKAVE

Precej študij je preverjalo varnostne incidente v produkcijskih omrežjih s pomočjo sledi v prometu. Pri tem niso sistematično validirali zaznanih incidentov. Niso imeli strokovnjakov, kateri bi ročno pregledovali varnostne incidente.

Veliko študij na področju avtomatizacije forenzične analize se je osredotočilo na analizo dokazov na strani končnega uporabnika, medtem ko se v tem članku osredotočijo na odkrivanje zlonamerne aktivnosti v omrežju uporabljenem v produkcijskem okolju.

# 8. ZAKLJUčEK

Strokovnjaki s področja računalniške varnosti so opravili kompleksno ročno preiskavo 200 domnevnih primerov zlonamerne programske opreme v produkcijskem omrežju. Te primere so pridobili s pomočjo korelatorja, kateri jih je izluščil iz množice 37 milijonov opozoril. Za identifikacijo sumljivih primerov so bili uporabljeni štirje viri podatkov. Pri analizi uspešnosti diagnosticiranja varnostih incidentov je bilo ugotovljeno, da je najboljši vir IDS - v našem primeru Snort. Za zelo dober vir se je izkazal tudi spletni iskalnik Google. Obnese se bolje kot poročila ranljivosti in pregledovanja ter črne liste, čeprav se te v praksi pogosteje uporabljajo na področju računalniške varnosti. Analiza je pokazala tudi, da en vir podatkov ne zadošča za dobro detekcijo, zato je potrebno kombinirati podatke iz različnih virov.

Raziskovalci so poiskali 138 signatur Snort, ki niso prožile lažnih alarmov. Nato so primerjali lastnosti dobrih signatur v

**Table 2: Primerjava dobrih signatur Snort z običajnimi. Zajetih je bilo 138 dobrih in 3198 običajnih signatur.**

| | Št. bajtov | Št. lastnosti | Zamik bajtov | Regularni izrazi | Vrata prejemnika | Velikost paketa |
|---|---|---|---|---|---|---|
| Običajne sig. | 11 | 1.2 | 8% | 15% | 17% | 7% |
| Dobre sig. | 23.5 | 2.8 | 28% | 50% | 22% | 15% |
| Povečanje | 2.14 × | 2.3 × | 3.5 × | 3.3 × | 1.29 × | 2.14 × |

primerjavi z običajnimi. Na podlagi rezultatov so predlagali priporočila za izdelavo dobrih signatur. Dobre signature natančneje zaznavajo zlonamerno delovanje, s čimer se zmanjša število lažnih opozoril in s tem količina dela varnostnih strokovnjakov, ki take primere ročno preiskujejo. V splošnem predlagajo uporabo kompleksnejših signatur Snort, pri katerih so lastnosti zlonamernih paketov natančneje opisane.

# 9. LITERATURA


[1] Concise Courses. Openvas. `http://www.concise-courses.com/security/openvas-tutorial/`. [Online; dostopano 29. aprila 2014].

[2] E. Raftopoulos, M. Egli and X. Dimitropoulos. Shedding light on log correlation in network forensics analysis. *In DIMVA'12*.

[3] Greenbone Networks GMBH. Openvas. `http://www.openvas.org/`. [Online; dostopano 29. aprila 2014].

[4] O'Reilly Media. Network security hacks. `http://oreilly.com/pub/h/1393`. [Online; dostopano 4. maja 2014].

[5] E. Raftopoulos and X. Dimitropoulos. Detecting, validating and characterizing computer infections in the wild. *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, pages 29 – 44, 2011.

[6] E. Raftopoulos and X. Dimitropoulos. Understanding network forensics analysis in an operational environment. *Security and Privacy Workshops (SPW)*, pages 111 – 118, Maj 2013.

[7] Sourcefire, Inc. *SNORT Users Manual 2.9.3*. Maj 2012.

[8] Tenable Network Security. Nessus. `http://www.tenable.com/products/nessus`. [Online; dostopano 28. aprila 2014].

[9] Wikia. Sdbot. `http://malware.wikia.com/wiki/Sdbot`. [Online; dostopano 25. aprila 2014].

[10] Wikipedia. Malware. `http://en.wikipedia.org/wiki/Malware`. [Online; dostopano 5. maja 2014].

[11] Wikipedia. Nessus. `http://en.wikipedia.org/wiki/Nessus_(software)`. [Online; dostopano 28. aprila 2014].

[12] Wikipedia. Torpig. `http://en.wikipedia.org/wiki/Torpig`. [Online; dostopano 25. aprila 2014].

# Forenzika v oblačnem računalništvu v teoriji in praksi s sistemom FROST*

Nejc Saje
Fakulteta za računalništvo in informatiko
nejc.saje@gmail.com

Nejc Sever
Fakulteta za računalništvo in informatiko
ns7827@student.uni-lj.si

Anže Sodja
Fakulteta za računalništvo in informatiko
as2508@student.uni-lj.si

## POVZETEK

V delu je povzeto delovanje, implementacija in analiza forenzičnega orodja FROST za oblačno platformo OpenStack, ki je opisan v članku Design and implementation of FROST[4].

FROST je orodje za pridobivanje API dnevnikov, dnevnikov požarnega zidu in slik virtualnih diskov. Njegova prednost je, da deluje znotraj OpenStack arhitekture, zato njegovo delovanje ni odvisno od navidezne naprave uporabnika. Orodje omogoča tudi pridobivanje podatkov brez posredovanja ponudnika oblačnih storitev.

Dodatno so opisani tudi problemi, s katerimi se soočajo forenzični preiskovalci pri pridobivanju dokazov iz storitev v oblaku in grob pregled področja forenzičnega preiskovanja v oblaku.

## Ključne besede

digitalna forenzika, oblačno računalnišstvo, OpenStack, FROST, zajem podatkov, integriteta

## 1. UVOD

Storitve v oblaku ponujajo uporabniku veliko prednosti. Uporabnik lahko do svojih podatkov ali storitev v oblaku dostopa praktično od kjerkoli. Ni omejen s strojno opremo svoje naprave, oblačne storitve pa prilagajajo strojne vire njegovim potrebam.

Vendar te prednosti omogočajo tudi veliko možnostjo izrabe oblačne infrastrukture za kriminalna dejanja. Npr. zlonamerni uporabniki tako lahko s pomočjo oblaka napadajo druge udeležence v internetnem omrežju. Lahko preobremenjijo oblak in onemogočijo dostop drugih uporabnikov oblaka do njihovih storitev. Lahko pa so napadene in poškodovane navidezne naprave uporabnikov iz zunanjih virov.

*Napisano z namenom izpolnjevanja standardov seminarske naloge pri predmetu Digitalna foreznika, FRI, 2014

Forenzični preiskovalci pri preiskovanju kriminalnih dejanj v oblaku naletijo na veliko težav dostopa do oblačnih storitev in predvsem zajema ustreznega dokaznega gradiva. Te težave se nanašajo predvsem na omejenost dostopa do naprav ponudnika oblačnih storitev. Tako se morajo zanašati na gradivo pridobljeno s pravnimi sredstvi direktno od ponudnika. Zaradi potrebe po izpolnjevanju pravnih formalnosti je to lahko časovno potratno. Dodatno, pridobljeno gradivo izgubi na vrednosti zaradi vprašljivosti o pravilnem zajemu in integriteti.

Forensic OpenStack Tools (FROST) je orodje za forenzično analizo na oblačni platformi OpenStack. Orodje je integrirano v samo arhitekturo platforme OpenStack. Zaradi integracije FROST ne deluje na ravni navideznih naprav uporabnikov, tako kot druga forenzična orodja. Pridobljeno dokazno gradivo ima tako večjo veljavo, saj je dokazovanje zaupnosti navideznih naprav uporabnikov netrivialno. Omogoča pridobivanje API dnevnikov, dnevnikov požarnega zidu in slik virtualnih diskov. Obenem skrbi za integriteto vseh podatkov s standardnimi zgoščevalnimi funkcijami. Zgoščevalne vrednosti hrani v posebni podatkovni strukturi. Pridobivanje dokaznega gradiva je enostavno ter zaobide težave sodelovanja s ponudnikom storitve. Dodatne obremenitve strežnika z namenom zbiranja podatkov za orodje FROST pa so predvidoma majhne.

Glavni cilj tega dela je spoznavanje problematik sodobne digitalne forenzike pri rokovanju z novimi tehnologijami, natančneje s storitvami v oblaku. Predstavljena je integracija forenzičnega orodja FROST za platformo OpenStack kot primer rešitve forenzike v oblaku. Delo vsebuje tudi pregled ostalih rešitev, njihovih prednosti in slabosti.

V poglavjih naprej si sledijo naslednje teme. V 2. poglavju je opisana implementacija orodja FROST, specifikacije, arhitektura in ocena delovanja. Nato v 3. poglavju sledi analiza sorodnih del, oris implementacij in slabosti. V 4. poglavju sledi diskusija o forenziki v oblaku, možnosti zlorabe oblačnih storitev in napadi na njih, ter probleme oblačne forenzike. V 5. poglavju je zajet pregled obravnavane tematike, ter kritična ocena uporabnosti forenzike v oblaku.

## 2. NAčRTOVANJE IN IMPLEMENTACIJA ORODJA FROST

Vsebina poglavja je povzetek članka Design and implementation of FROST: Digital forensic tools for the OpenStack cloud computing platform avtorjev Josiah Dykstre in Alan

T. Shermana[4]. Članek opisuje načrtovanje in implementacijo forenzičnega orodja, ki deluje na oblačni platformi OpenStack. Glavni namen orodja je pridobivanje informacij, ki so pomembne za forenzično preizkavo. Pridobivanje informacij poteka brez posredovanja ponudnika oblačne storitve, vsi podatki pa imajo zagotovljeno integriteto.

Pomembna predpostavka članka in orodja FROST je ta, da popolnoma zaupamo ponudniku oblačne storitve. To pomeni zaupanje v strojno opremo, gostiteljev operacijski sistem, navidezne stroje in zaposleno osebje ponudnika storitve.

## 2.1 Oblačna platforma OpenStack

OpenStack je odprtokodna oblačna platforma, ki je nastala kot kolaborativni projekt med National Aeronautics and Space Administration (NASA) in podjetjem Rackspace. Platforma je namenjena tako privatnim kot javnim oblačnim storitvam, ponuja velik nabor funkcionalnosti, je enostavna za implementacijo in skalabilna.

Platforma OpenStack sestoji iz šestih glavnih sestavnih delov. **Nova** je kontroler, ki je ključni element IaaS oblačnih sistemov. Odgovoren je za upravljanje in avtomatizacijo bazenov računalnikov in njihovih virov, povezan pa je tudi z virtualizacijo. **Swift** skrbi za hranjenje podatkov v različnih podatkovnih centrih in posledično pravilnost podatkov. **Glance** je upravitelj za slike diskov. **Keystone** je skupni avtentikacijski sistem za celotno oblačno platformo. **Horizon** je spletni uporabniški vmesnik za administracijo storitev, ki jih ponuja platforma. **Quantum** pa je upravitelj omrežnih storitev npr. dodeljevanje IP naslovov.

Implementacija orodja FROST se v večini nanaša na razširitev kontrolerja (Nova) in grafičnega vmesnika (Horizon).

## 2.2 Zahteve in specifikacije

### *2.2.1 Zahteve*

Implementacija orodja FROST mora ustrezati zahtevam na večih ravneh. Prva raven so **znanstvene zahteve**, ki se navezujejo na že uveljavjene prakse in postopke zagotavljanja verodostojnosti podatkov v digitalni forenziki. Druga raven so **tehnične zahteve** za možnost implementacije na oblačni platformi OpenStack. Nazadnje je potrebno zadovoljiti tudi **pravne zahteve** tako, da so pridobljeni podatki s pomočjo orodja FROST uporabni kot dokaz na sodišču.

V splošnem ne obstajajo verodostojni viri, ki bi natančno navajali način, kako izpolniti znanstvene zahteve pri razvoju novega forenzičnega orodja. Iz navedenega razloga se članek za zagotovitev teh zahtev sklicuje na že uveljavljene prakse in zapisane smernice iz katerih razberemo, da sta najpomembnejši dve stvari: celovitost in integriteta podatkov. Integriteto podatkov dosežemo z uporabo zgoščevalnih funkcij, medtem ko je celovitost določena s pravilnim postopkom zajema podatkov ter z zagotovilom, da so bili zajeti vsi možni podatki v danem trenutku.

Pri zadovoljevanju tehničnih zahtev se je potrebno ozreti na zahteve, ki so značilne za razvoj forenzičnih orodij na oblačnih platformah, kot je OpenStack. Orodje mora omogočati skalabilnost, enostavno prilagajanje in mora biti dostopno kot storitev na zahtevo. Da vse to omogočimo morajo biti izpolnjeni naslednji pogoji:

1. **Kompatibilnost z obstoječimi formati.** Pomembno je, da so podatki, ki jih proizvede orodje v takšnem formatu, da jih znajo obdelati tudi že obstoječa forenzična orodja. V digitalni forenziki se uporablja format Digital Forensic XML (DFXML), v katerega shranimo dokazno gradivo in vrednosti zgoščevalnih funkcij.
2. **Enostavna uporaba.** Orodje mora biti enostavno za integracijo s strani administratorjev, kot za pridobivanje potrebnih podatkov s strani uporabnikov. Integracija orodja je neodvisna od že obstoječih funkcionalnosti platforme, namestitev pa poteka z zagonom namestitvene skripte. Enostavnost uporabe orodja je zagotovljena z intuitivnim grafičnim vmesnikom v obliki spletne strani, kjer uporabnik le z enim klikom pride do forenzičnih podatkov.
3. **Odprtost in razširljivost.** Dokončano in delujoče orodje mora biti dostopno vsem OpenStack administratorjem. Prav tako mora biti omogočen doprinos ostalih razvijalcev za razširitve obstoječe rešitve. Izvorna koda bo združena v OpenStack projekt.
4. **Skalabilnost.** FROST mora v skladu s stalnimi praksami programov v oblaku delovati za več milijonov uporabnikov in virtualnih strojev. Orodje je omejeno le z zmogljivostjo procesiranja in pridobivanja podatkov gostiteljevega operacijskega sistema.
5. **Upoštevanje obstoječih praks in standardov.** Kjer je le mogoče je dobro razvijati po obstoječih standardih za razvoj forenzičnih orodij. To nam omogoča kompatibilnost in povezovanje orodja z ostalimi forenzičnimi produkti na trgu.

Pridobljeni forenzični dokazi s pomočjo razvitega orodja so brez vrednosti, če jih ne moremo uporabiti kot dokazno gradivo na sodišču. Ameriško pravosodno ministrstvo definira pravila o dokaznem gradivu, pridobljenem z računalniškimi programi oz. avtomatiziranimi procesi, v priročniku Šearch and Seizure Manual". Pravilo pravi, da je za uporabnost podatkov na sodišču potrebno predstaviti proces, ki je ustvaril te podatke ter dokazati, da proces vedno zagotavlja korektne podatke. V večini primerov je dovolj dokaz o tem, da uporabniki uporabljajo proces na dnevni ravni ali pa gre za proces, ki je uveljavljen kot stalna praksa. V primeru orodja FROST gre za pridobivanje slik diska in dnevniških zapisov, kar ni nič nevsakdanjega.

### *2.2.2 Specifikacije*

Orodje FROST je sestavljeno iz treh glavnih funkcionalnosti. V vseh treh primerih gre za zajem podatkov s strani uporabnika, brez poseganja ponudnika oblačne storitve. Prva funkcionalnost je zajem slik virtualnih diskov. Uporabnik lahko pridobi le podatke, ki pripadajo njegovim navideznim računalnikom. Preveri lahko integriteto slik diskov s pomočjo kriptografskih kontrolnih vsot. Druga funkcionalnost omogoča pridobitev dnevniških zapisov o vseh zahtevah na aplikacijski vmesnik (API) iz uporabniškega računa s katerim razpolaga. Zagotovljena je tudi integriteta podatkov.

Zahteve na aplikacijski vmesnik se uporabljajo za administracijo navideznih računalnikov npr. ustvarjanje, zaganjanje ali ugašanje virtualnega računalnika. Tretja funkcionalnost pa je pridobivanje celovitih dnevniških zapisov požarnega zidu za vse navidezne računalnike v uporabnikovi lasti. V dnevnikih požarnega zidu se nahajajo npr. poročila o blokiranih paketih namenjenih na določena vrata ali naslov. Te nastavitve lahko uporabnik nastavlja s pomočjo aplikacijskega vmesnika za požarni zid, ki teče na gostiteljevem računalniku.

Naštete specifikacije so zelo nizko nivojske in običajno uporabniku niso na voljo. Ponudniki oblačnih storitev pogosto omogočajo le pridobitev dnevniških zapisov na aplikacijskem nivoju npr. HTTP zahtevke. Avtor članka navaja neuradno informacijo o pogovoru z dvema ponudnikoma oblačne storitve, ki sta mnenja, da so takšne specifikacije zaželjene s strani ponudnikov.

Za zagotovitev glavnih treh specifikacij je potrebna implementacija grafičnega vmesnika in aplikacijskega vmesnika. Uporabnik lahko tako do podatkov neposredno dostopa preko spletnega brskalnika ali preko ukazne vrstice. Implementacija mora biti modularna za morebitne prihodnje nadgradnje sistema.

## 2.3 Arhitektura

V poglavju 2.1 so opisane glavne komponente platforme OpenStack. FROST orodje je integrirano v dve komponenti in sicer v kontroler Nova in spletni uporabniški vmesnik Horizon. Glavni del integracije je bil storjen na API storitvi. Na Sliki 1 je prikazana integracija v OpenStack arhitekturo.

Tako so bili dodani novi klici za dostop v kontrolor Nova. Te klice lahko uporabniki uporabljajo preko ukazne vrstici ali znotraj svojih programov. V vmesnik Horizon pa so bili dodani dodatni elementi za bolj prijazno uporabo orodja FROST. Na ta način FROST dostopa do podatkov, ki jih ustvari OpenStack. V grobem orodje FROST le poišče ustrezne shranjene podatke ter jih posreduje uporabniku. Ti podatki so lahko zapisi sistema, zapisi požarnega zidu ali diski navideznih naprav.

OpenStack za vsakega uporabnika ustvari direktorij na strežniku, kamor shrani navidezne naprave in diske. FROST omogoča pridobivanje posnetkov diskov ustavljenih naprav, kot naprav v teku. Diski so lahko v različnih formatih. Implementacija opisana v obravnavanem članku zajema le diske formata QEMU QCOW2. Dodano je, da bi bila razširitev za ostale formate možna in enostavna.

Za shranjevanje zapisov zahtev do vmesnika Nova, FROST že ustvarjene zapise platforme OpenStack ustrezno preoblikuje. S preoblikovanjem zapisov FROST poskrbi za integriteto podatkov. Dodatno FROST poskrbi za dostop do zapisov požarnega zidu za vsako navidezno napravo. OpenStack uporablja požarni zid iptables, katerega lahko vsak uporabnik nastavi za svoje potrebe. To orodju FROST omogoča, da nastavi beleženje stanj omrežnih povezav na požarnem zidu, ter zabeležena stanja shrani. Podrobnosti implementacije shranjevanja zapisov so opisane v poglavju 2.4.

### 2.3.1 Shema shranjevanja zapisov

Pridobljeni zapisi za forenzike nimajo velike veljave, če ni zagotovljene integritete. Pomembna izboljšava sistema beleženja v FROSTu so zgoščevalna drevesa. Zgoščevalna drevesa so posebna podatkovna struktura, ki omogočijo učinkovito shranjevanje zapisov in njihovih hash vrednosti, ter enostavno branje zapisov.

Iz korenskega vozlišča lahko pridemo v podrevesa uporabnikov. Tako ima vsak uporabnik OpenStack oblaka svoje podrevo. V podrevesu uporabnika se vrstijo drevesa uporabnikovih navideznih naprav, katerim sledi nivo leta, meseca in dneva. V listih so shranjena dejanski zapisi. V vsakem vozlišču, razen v listih, so dodatno izračunane hash vrednosti.

## 2.4 Implementacija

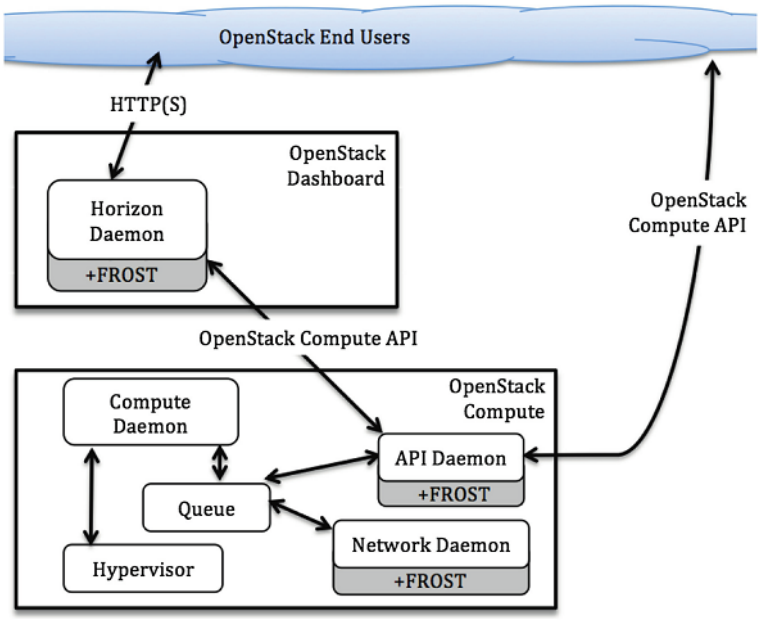
V poglavju je opisana implementacija orodja FROST. Sestoji iz treh glavnih delov. Prvi je algoritem za zagotavljanje integritete podatkov (Authenticated logging service), drugi je implementacija aplikacijskega vmesnika, tretji pa spletni uporabniški vmesnik.

### 2.4.1 Authenticated logging service

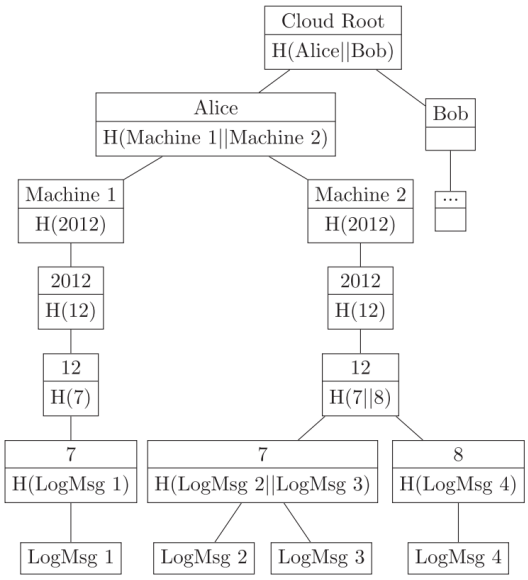
Gre za storitev, ki s pomočjo podatkovne strukture Merklovih dreves omogoča shranjevanje aplikacijskih dnevniških zapisov in dnevniških zapisov požarnega zidu.

Zgoščevalna drevesa (npr. Merklova drevesa) nudijo tri pomembne lastnosti. Prva lastnost omogoča shranjevanje kontrolnih vsot večje količine podatkov, brez potrebe po prenašanju le-teh, s še vedno optimalnim preverjanjem integritete podatkov. Zagotovimo jo tako, da se na delih podatkov izračunajo zgoščene vrednosti in nato nad temi zgoščenimi vrednosti nove zgoščene vrednosti, ki privedejo do zagotavljanja integritete celotnega drevesa. Iz tega razloga za zagotavljanje integritete deleža podatkov ni potrebno prenašati celotnega drevesa. Druga dobra lastnost zgoščevalnih dreves je struktura, ki omogoča enostavno povpraševanje po podatkih za določeno časovno obdobje. Tretja lastnost je ta, da lahko poleg vseh dnevniških zapisov zagotovimo integriteto podmnožice le-teh.

Drevo na sliki 2 predstavlja, kako ponudnik oblačne storitve hrani podatke o dnevniških zapisih. Dnevniški zapisi požarnega zidu in aplikacijske zahteve so predstavljene v istem drevesu, zato ponudnik oblačne storitve hrani dva drevesa s popolnoma enako strukturo. Iz slike je razvdno, da ponudnik storitve hrani dnevniške zapise za vsakega uporabnika posebej. Algoritem za dodajanje in računanje zgoščenih vrednosti je naslednji. Za vsakega na novo registriranega uporabnika se v drevo pod korensko vozlišče doda poddrevo, ki je namenjenu uporabniku. Koten uporabnikovega poddrevesa je podpisan z uporabnikovim javnim ključem, tako lahko iz njega bere le uporabnik. Podatki v uporabnikovem poddrevesu so razdeljeni v pet nivojev. V prvem nivoju se nahajajo instance navideznih računalnikov, nato sledijo nivoji za leto, mesec in dan za vsak dnevniški zapis. V listih drevesa pa se nahajajo vsebina dnevniških zapisov. V vozliščih se vsako minuto izračuna zgoščena vrednost združenih podatkov naslednjikov. Na koncu ponudnik storitve podpiše koren drevesa in vse korene uporabniških poddreves s korenskim Nova certifikatom.




**Slika 1: A sample black and white graphic (.eps format).**



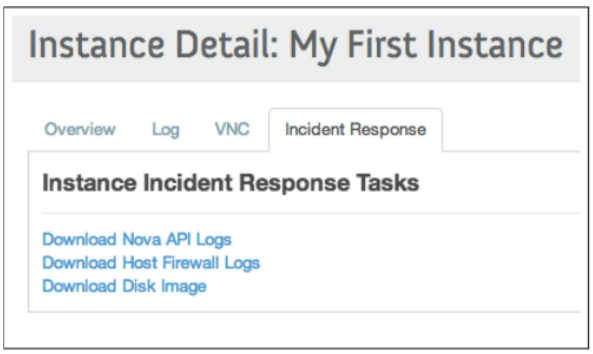

**Slika 2: Drevesna struktura za hranjenje dnevniških zapisov razvrščenih po uporabniku, virtualnemu računalniku, letu, mesecu in dnevu. Starši lista drevesa integriteto zagotavljajo z zgoščeno vrednostjo dnevniških zapisov, medtem ko starši takšnih vozlišč vsebujejo zgoščeno vrednost zgoščenih vrednosti sinov.**

Potek uporabnikovega zahtevka si lažje predstavljamo na primeru. Če bi uporabnica Alice (glej sliko 2) želela pridobiti podatke za 7.12.2012 na prvi instanci, bi prejela dnevniško poročilo LogMsg1 in zgoščeno vrednost v vozlišču Machine 1. Podatke ponudnik storitve podpiše s korenskim Nova ključem in tako zagotovimo avtentikacijo prejetih podatkov. Podatki se do Alice prenašajo v dnevniški datoteki DFXML. Alice bi nato iz prejetih podatkov izračunala končno zgoščeno vrednost in jo primerjala s prejeto. Ujemanje zgoščenih vrednosti pomeni uspešno preverjanje integritete prejetih podatkov.

### 2.4.2 *Implementacija aplikacijskega vmesnika*

Veliko uporabnikov upravlja s svojimi storitvami na OpenStack platformi preko ukazne vrstice. Ti ukazi kličejo funkcije preko aplikacijskega vmesnika. Končna destinacija zahtev je Nova API daemon. Za dodajanje novih razširitev Nova API daemona je potrebno kodo zapisati kot API razširitev, ki jo je potrebno registrirati kot Nova razširitev. Na zadnje je potrebno dodati še možnost klicanja dodanih fonkcionalnosti preko OpenStack-ove nadzorne plošče in ukaza `novaclient`. Nove API klice nato dodamo v OpenStack tako, da njihove funkcionalnosti dodamo v čontribution directory"in modificiramo ukaz `novaclient`, da le-ta uporabnikom omogoča klicanje dodanih ukazov. Za možnost manipuliranja dnevniških zapisov moramo "Nova logging handler"preusmeriti v nadomestno storitev orodja FROST, kjer lahko potem manipuliramo z aplikacijskimi dnevniškimi zapisi. Dnevniške zapise za požarni zid pa preusmerimo iz iptables upravljalca.

Nove funkcionalnosti API klicev lahko uporabljajo le avtenticirani uporabniki. Za to poskrbi Keystone, ki je del OpenStack platforme za upravljanje z identitetami uporabnikov. Vsak uporabnik se mora pred uporabo orodja FROST avtenticirati s privatnim ključem ali uporabniškim imenom in geslom, dostop pa je omejen le na podatke iz lastnih navi-




**Slika 6: Zavihek v nadzorni plošči za pridobitev forenzičnih dokazov.**

deznih računalnikov.

Aplikacijski dnevniški zapisi (Nova logs) se hranijo v datoteki `/var/log/nova` na gostiteljevem operacijskem sistemu. Ob uporabnikovem zahtevku za pridobivanje aplikacijskih dnevniških zapisov, orodje FROST preišče omenjeno datoteko za vnose, ki pripadajo uporabniku. Slika 3 prikazuje primer dnevniških zapisov za eno instanco, ki jih vrne orodje FROST. Poleg zapisov orodje prenese tudi DFXML datoteko z zgoščeno vrednostjo za preverjanje integritete.

Beleženje dnevniških zapisov požarnega zidu na OpenStack platformi ni samodejno omogočeno, zato je beleženje potrebno ročno omogočiti. OpenStack za vsak nov virtualni računalnik nastavi privzeta iptables pravila. Za delovanje orodja FROST je potrebno dodati dodatno pravilo, ki omogoči, da se vsakemu zapisu doda predpona identifikacijske virtualnega računalnika. To kasneje omogoči orodju iskanje zapisov le za določeno instanco. Zapisi se shranijo v datoteko `/var/log/syslog`. Primer generiranih zapisov požarnega zidu s pomočjo orodja FROST prikazuje slika 4. Tako kot pri aplikacijskih vnosih se kot odgovor uporabniku pošlje datoteka z vsebino in DFXML datoteka z zgoščeno vrednostjo za preverjanje integritete.

Zadnja specifikacija potrebna implementacije je pidobivanje slik diskov. Te so shranjeni na gostiteljevem operacijskem sistemu za vsak navidezni računalnik posebej. Orodje FROST omogoča le zajem slik ugasnjenih virtualnih računalnikov, zajem prižganih računalnikov pa OpenStack še ne podpira. Uporabnik prejme datoteko s sliko diska, ter podobno kot pri zajemu dnevniških zapisov, podatke o integriteti v datoteki DFXML.

### *2.4.3 Spletni vmesnik*

Spletni vmesnik za upravljanje z orodjem FROST je dodan zavihek v nadzorni plošči "Management Console for OpenStack Compute", ki vsebuje podatke o vseh instancah, ki jih je ustvaril uporabnik. Zavihek je prikazan na sliki 6 in vsebuje tri povezave, s katerimi uporabnik lahko pridobi dnevniške zapise Nova API-ja in požarnega zidu ter slike diskov. Prenesejo se dve datoteki, v prvi `.zip` datoteki se nahajajo podatki, v drugi DFXML datoteki pa podatki o integriteti.

## 2.5 Ocena delovanja

Delovanje FROSTa je bilo ocenjeno na dva načina. FROST je bil najprej testiran na podlagi stabilnost delovanja in vračanja pravilnih rezultatov. Tako je bilo ustvarjenih 100 uporabnikov, kjer je imel vsak uporabnik 5 navideznih naprav. Skupaj je bilo tako 500 navideznih naprav. Ustvarjena so bila pravila požarnega zidu, ki so dovoljevala le protokol SSH.

Za preverjanje delovanja zapisov požarnega zidu, je bil uporabljen pregledovalnik vrat (angl. port scanner), kateri je proizvedel prepovedan promet na požarnem zidu. Dodatno je bil izbran naključni uporabnik, ki je poskusil ustaviti naključno napravo na celotnem oblaku. S tem so se generirali zapisi uspešnih in neuspešnih poskusov za kontroler Nova. Nato je bilo izbranih 20 naključnih uporabnikov. Za vse te uporabnike se je poizvedlo po zapisih sistema, zapisih požarnega zidu in slik diksov. Preverila se je integriteta vseh zapisov in diskov. Ugotovljeno je bilo, da sistem deluje dobro.

Za večje število uporabnikov, bi sistem potreboval le več diskovnega prostora. Z uporabo SHA-1 zgoščevalne funkcije bi mehanizem zapisov lahko vseboval več kot 645 tisoč zapisov na 1GB diskovnega prostora. Shranjevanje zapisov tako ne bi preveč povečal strežniških potreb po diskovnem prostoru, ponudniki oblačnih storitev pa bi lahko FROST ponudili kot dodatno storitev. Prav tako je bilo ocenjeno, da bi izračun zgoščevalnih vrednosti minimalno obremenil strežnike. Pričakovano je, da bi uporabniki naenkrat zahtevali le majhno količino zapisov. Večjo količino zapisov in slike diskov pa le občasno.

Drug del testiranja je bil usmerjen na odziv ciljne skupine uporabnikov FROSTa. V ta namen je bil FROST predstavljen 12 uporabnikom in administratorjem večjih vladnih oblakov. Odziv je bil pozitiven. Večina je bila mnenja, da bi uporaba orodja FROST zmanjšala obremenitev administratorjev ob forenzičnih preiskavah.

## 2.6 Slabosti in možne izboljšave orodja FROST

V tem poglavju sledi pregled slabosti in možnih izboljaš za orodje FROST.

### *2.6.1 Slabosti*

V primerjavi z drugimi orodji, FROST omogoča forenziku pridobivanje dokaznega gradiva brez posredovanja upraljalca oblaka. Dodatno je FROST integriran v samo platformo oblaka in zaradi tega ni odvisen od virtualnih naprav uporabnikov. Vendar pa ima tudi pomankljivosti. FROST predpostavlja zaupanje v storitve ponudnika oblaka oz. v ponudnikovo strojno opremo, operacijski sistem in omrežje. Dodatno, mehanizem zapisov je mogoče izrabiti. FROST namreč ne more zaznati, če sistem za zapisovanje platforme OpenStack namerno ne beleži ali morda prilagodi kakšne dogodke. Nezaupanja vreden ponudnik storitve bi tako lahko sodeloval z zlonamernim uporabnikom in namenoma spremenil zapisovanje dogodkov, kar bi otežilo delo forenzika. Zato je zaupanje v ponudnika pomemben faktor pri preiskavi.

Shranjevanje zapisov podatkov ima tudi druge težave. Želja forenzikov je, da se dogodke shrani za čim daljše obdobje. Vendar pa pri tem naletu na problem shranjevanja velike

```
$ nova get-nova-logs 0afcfbcd-b836-4593-a02c-25d8d3a94b00 verify.xml
[truncated]
2012-12-01 13:30:49 INFO nova.api.openstack.wsgi [req-0afcfbcd-b836-4593-a02c-25d8d3a94b00 admin demo]
    POST http://10.34.50.142:8774/v2/5ee3040fa890428387f56111576cf819/servers
2012-12-01 13:30:49 DEBUG nova.quota [req-0afcfbcd-b836-4593-a02c-25d8d3a94b00 admin demo] Created
    reservations ['915e9c89-b3bc-4091-8b75-3b555961ec3e', '72c39d24-0a96-42ca-96f1-593da3aa9f81',
    '57843316-872b-4b40-a853-2aa7c730262e'] from (pid=16036) reserve /opt/stack/nova/nova/quota.py:697
2012-12-01 13:30:50 DEBUG nova.compute.api [req-0afcfbcd-b836-4593-a02c-25d8d3a94b00 admin demo] Going to
     run 1 instances... from (pid=16036) _create_instance /opt/stack/nova/nova/compute/api.py:492
[truncated]
```

**Slika 3: Nova API dnevniški zapisi za instanco z identifikacijsko številko 0afcfbcd-b836-4593-a02c-25d8d3a94b00 za uporabnika admin.**

```
$ nova get-firewall-logs 0a18799f-c198-4dbb-b369-b49184e3dfbc verify.xml
0a18799f-c198-4dbb-b369-b49184e3dfbc: Nov 28 11:13:38 domU-12-31-39-17-29-5D kernel: [  310.765760]
    IPTables-Dropped: IN=eth0 OUT= MAC=12:31:39:17:29:5d:fe:ff:ff:ff:ff:ff:08:00 SRC=130.85.36.72 DST
    =10.97.42.171 LEN=52 TOS=0x00 PREC=0x00 TTL=48 ID=29222 DF PROTO=TCP SPT=55739 DPT=443 WINDOW=1002
    RES=0x00 ACK URGP=0
0a18799f-c198-4dbb-b369-b49184e3dfbc: Nov 28 11:13:36 domU-12-31-39-17-29-5D kernel: [  309.623023]
    IPTables-Dropped: IN=eth0 OUT= MAC=12:31:39:17:29:5d:fe:ff:ff:ff:ff:ff:08:00 SRC=172.16.0.23 DST
    =10.97.42.171 LEN=103 TOS=0x00 PREC=0x00 TTL=64 ID=42188 PROTO=UDP SPT=33905 DPT=53 LEN=83
[truncated]
```

**Slika 4: Nova API dnevniški zapisi za instanco z identifikacijsko številko 0afcfbcd-b836-4593-a02c-25d8d3a94b00 za uporabnika admin.**

količine podatkov, bolj pomembno pa na kršenje pravic zasebnosti uporabnikov.

Ena izmed težav je tudi nadgradja in razvoj platforme OpenStack. Spremembe v platformi OpenStack lahko vplivajo na funkcionalnost orodja FROST. Zato se mora FROST prilagajati OpenStack spremembam in se skladno posodabljati.

### *2.6.2 Možne izboljšave*

Trenutna implementacija FROSTa podpira samo OpenStack platformo. Izvedljivo bi bila implementacija FROSTa tudi na druge IaaS oblačne platforme. Za PaaS in SaaS platforme pa bi bila implementacija težja oz. nemogoča.

V orodje FROST bi se lahko dodala podpora za večje število formatov diskov. Dodatno bi se lahko izvajalo avtomatsko shranjevanje posnetkov virtualnih naprav (angl. snapshots) in detektiranje sprememb v različnih posnetkih virtualnih naprav. Poleg tega pa bi bilo zanimivo dodati tudi avtomatski prenos zapisov za določeno časovno obdobje, npr. dnevne zapise, tedenske zapise ipd.

Nazadnje bi se lahko izvedlo bolj natančno testiranje stabilnost delovanja in sicer za večje število uporabnikov oz. v velikih oblakih. S tem bi lahko bolje analizirali skalabilnost orodja FROST in obremenjenost celotne oblačne infrastrukture. Z natančno analizo bi pridobili pomembne podatke za optimizacijo shranjevanja zapisov in zagotavljanja njihove integritete.

# 3. ANALIZA SORODNIH DEL

## 3.1 Dijkstra in Sherman

V članku na temo zajema forenzičnih dokazov iz IaaS oblačnega računalništva Dijkstra in Sherman [3] opozorita na izzive, s katerimi se soočajo preiskovalci pri preiskovanju zločinov, ki vključujejo oblačno računalništvo. Ker dokazi oz. strojna oprema navadno ni fizično prisotna, je potrebno veliko več zaupanja v različne entitete ter integriteto dokazov. Če je za zločin uporabljen navadni osebni računalnik in preiskovalci naredijo sliko diska, morajo zaupati elektroniki na disku, da vrača pravilne podatke. Če računalnik prižgejo in zaženejo forenzično programsko opremo, morajo poleg strojni opremi zaupati tudi integriteti operacijskega sistema. Če pa je preiskovani računalnik lociran v oblaku, pa je potrebno zaupati še večim plastem. Raziskovalca predstavita Tabelo 1, v kateri oblačno infrastrukturo razdelita na šest nivojev. Zajem dokazov je možen na vsakem od teh nivojev, ampak vsak zahteva drugačen pristop ter drugačno količino zaupanja v tretje osebe. Preiskovalec bi lahko npr. imel oddaljen dostop do virtualnega računalnika v oblaku. Dokaze bi lahko zajel oddaljeno z namestitvijo forenzične programske opreme v virtualnem računalniku. Na žalost pa zajem na tem nivoju zahteva zaupanje v gostujoči operacijski sistem, hipervizor, gostiteljski operacijski sistem, strojno opremo in omrežje.

V drugem delu članka raziskovalca osnujeta tri eksperimente, s katerimi ovrednotita obstoječo forenzično programsko opremo pri zajemu dokazov iz oblaka. Med ocenjenimi programskimi rešitvami so EnCase Enterprise, AccessData FTK in tri druga orodja. Prvi eksperiment preveri, ali je programska oprema zmožna oddaljeno zajeti forenzične dokaze v oblaku na nivoju gostujočega operacijskega sistema (plast 5). Drugi eksperiment preveri delovanje programske opreme na nivoju virtualizacije (plast 4) s podtaknitvijo oddaljenega agenta v prižgan virtualni računalnik. Tretji eksperiment pa je koristil storitev Amazon AWS Export, ki uporabniku omogoča, da podjetju Amazon pošlje podatkovni nosilec, ta pa mu na nosilec prenese uporabnikove podatke (v tem primeru sliko diska).

Vsa programska oprema v vseh eksperimentih je uspešno zajela dokaze, vendar pa količina zaupanja, potrebnega v dokaze, ni optimalna. V prvem eksperimentu moramo zaupati petim, v drugem pa štirim plastem oblaka. Čeprav tretji eksperiment na prvi pogled zahteva le zaupanje dvem plastem oblaka, pa moramo poleg tega zaupati še Amazono-

```
$ nova get-disk myvol-e9a5612d report.xml
MD5:  b17ee04095b2a3d81eed98628072eab6
SHA1: 399f5ffaccd09fe43d642d5f0d996875ca650c9f

$ sha1sum myvol-e9a5612d
399f5ffaccd09fe43d642d5f0d996875ca650c9f myvol-e9a5612d
```

**Slika 5: Kontrolne vrednosti za preverjanje integritete slike diska myvol-e9a5612d**

vemu tehniku, tehnikovemu računalniku, AWS programski opremi ter transportu podatkovnega nosilca.

Nazadnje raziskovalci predstavijo alternativne načine zajema dokazov v oblačnem računalništvu. Kot najbolj perspektivnega predstavijo zajem s pomočjo uporabnikove nadzorne plošče oblačne infrastrukture, praktična implementacija katerega je orodje FROST.

## 3.2 SUNDR

Liand in drugi v članku [5] opisujejo večuporabniški omrežni datotečni sistem, ki zagotavlja pravilno stanje datotek, tudi če je strežnik kompromitiran. To doseže tako, da vsak uporabnik ob modifikaciji datotečnega sistema podpiše ne le svoje spremembe, ampak vso zgodovino datotečnega sistema s svojim privatnim ključem. To kompromitiranemu strežniku onemogoči modifikacijo datotek. Vse, kar lahko stori, je to, da lahko enemu uporabniku prikrije dejanja drugega uporabnika. V takem trenutku se zgodovina razveji med obema uporabnikoma. Datoteke še vedno ne morejo biti spremenjene ali pre, so le nepopolne. Ob prvi komunikaciji med uporabnikoma pa lahko uporabnika ugotovita, da je strežnik kompromitiran.

Čeprav nam SUNDR zagotavlja integriteto, pa za uporabo v FROST ni primeren, saj zagotavlja integriteto le za vse podatke naenkrat. V FROST je bilo potrebno integriteto zagotoviti za vsakega uporabnika posebej.

## 3.3 Podatkovne strukture za zaupno hranjenje podatkov

V svoji doktorski dizertaciji[2] je avtor Scott Alexander Crosby raziskal rešitev podobnega problema, ki ga rešuje FROST vendar z drugim namenom. Nanaša se predvsem na spreminjanje podatkov na mestih, ki so lahko ogrožena s strani napadov in niso pod našim nadzorom (npr. strežniki, oblačne storitve). Dizertacija opisuje implementacijo različnih kriptografskih pripomočkov, kot so npr. avtentikacijska sredstva za hranjenje dnevniških zapisov.

Opisani algoritmi so namenjeni zaznavanju ponarejanja dnevniških zapisov s strani napadalca, ki se razlikuje od orodja FROST, ki uporablja zgoščevalna drevesa za namen pridobivanja forenzičnih podatkov iz oblačne platforme.

# 4. FORENZIKA V OBLAKU

## 4.1 Nekaj o oblaku

Računalništvo v oblaku ni nekaj novega. Prvi zametki oblačnih storitev so se pojavili v 50. letih prejšnega stoletja. Ti prvi oblaki so bili obstajali v večjih korporacijah in akademskih inštitucijah. Omogočali so dostop večjemu številu uporabnikov do glavnih računalnikov, ter s tem izboljšali izrabo teh računalnikov.

Danes so oblačne storitve široko dostopne in razširjene. Omogočajo učinkovito izrabo tuje infrastrukture za lastne potrebe. Danes se uporabljajo trije modeli oblačnih storitve [6] [1], to so Software as a Service (krajše SaaS), Platform as a Service (krajše PaaS) in Infrastructure as a Service (krajše IaaS).

Preko SaaS modela lahko uporabnik dostopa do aplikacij, katere upravlja ponudnik storitve. Najpogosteje je dostop enostaven preko brskalnika. Ker je uporabniku ponujen le dostop do vmesnika aplikacije, je za ponudnika upravljanje enostavno, saj sam skrbi za vse nižje nivo aplikacije. Primer SaaS storitve je Gmail.

PaaS model uporabniku omogoča zagon aplikacij v oblaku, ki so izdelane z orodji, ki jih podpira ponudnik oblačne storitve. Uporabnik nima dostopa do infrastrukture oblačne storitve, temveč ima nadzor nad svojimi aplikacijami, ter nad njihovo konfiguracijo. Primer PaaS storitve je Google-App engine.

IaaS ponuja uporabniku navidezne vire glede na uporabnikove zahteve. Uporabniku omogoča zagon lastnih sistemov v virtualnih okoljih. IaaS deluje na najnižjem nivoju oblačnih storitev. Primer IaaS storitve je Amazon Web Services.

## 4.2 Oblačne storitve kot orodje forenzika

Oblačne storitve so lahko za forenzika močno orodje. Tako forenziki lahko prilagajajo strojne potrebe primeru na katerem delajo. Dodatno oblak omogoča shranjevanje velikih količin podatkov, paralelno procesiranje in hitrejše deljenje dokaznega gradiva med sodelavci.

Težava pri uporabi oblaka kot orodja je zagotavljanje integritete gradiva in omejevanje dostopa do gradiva. Forenzik namreč s tem, ko dokazno gradivo prenese v oblačno storitev, tvega, da bi bilo gradivo prestreženo, spremenjeno ali kako drugače izrabljeno.

## 4.3 Preiskovanje oblačnih storitev

Pri preiskovanju oblakov se v glavnem pojavljata dve vrsti prekrškov. Oblačna storitev je lahko uporabljena kot orodje za izvajanje nelegalnih dejanj. Zlonamerni uporabniki lahko izrabijo oblačno storitev za napad na druge tarče v omrežju ali pa uporabijo oblačno storitev za hrambo prepovedane pornografije ali prepovedane programske opreme.

Oblačna storitev je lahko tudi napadena. Napadalci tako preobremenijo storitev ali onemogoči dostop do nje. Lahko pa so napadeni direktno uporabniki in njihovi shranjeni po-

**Table 1: Nivoji zaupanja**

| Plast | Plast oblaka | Metoda zajema | Potrebno zaupanje |
|---|---|---|---|
| 6 | Gostujoča aplikacija/podatki | odvisno od podatkov | omrežje, strojna oprema, gostiteljski OS, hipervizor, gostujoči OS |
| 5 | Gostujoči OS | forenzična orodja za oddaljeni zajem | omrežje, strojna oprema, gostiteljski OS, hipervizor, gostujoči OS |
| 4 | Virtualizacija | introspekcija | omrežje, strojna oprema, gostiteljski OS, hipervizor |
| 3 | Gostiteljski OS | dostop do navideznega diska | omrežje, strojna oprema, gostiteljski OS |
| 2 | Strojna oprema | dostop do fizičnega diska | omrežje, strojna oprema |
| 1 | Omrežje | prestrezanje paketov | omrežje |

datki ali osebni podatki (npr. email, uporabniško ime, geslo).

Zaradi karakteristik infrastrukture oblačnih storitev, forezniki naletijo na različne ovire pri preiskovanju. Največja ovira je dostop do samih podatkov, ki so shranjeni na strojni opremi ponudnika. Tu se morajo forenziki zanesti na ustrežljivost ponudnikov in na sodne naloge. V oblakih so podatki lahko tudi razpršeni po različnih napravah. Tako se lahko v različnih napravah nahajajo relevantni podatki, ali pa ima vsaka naprava svojo kopijo, kar lahko privede do težave pri zagotavljanju dokazne verige in integritete podatkov.

Oblačne storitve so dostopne različnim uporabnikom. Tako forenziki pri preiskovanju oblaka ne smejo motiti delovanja storitve oz. ponudniku povzročati poslovne škode. V takem primeru se lahko podjetja odločijo za tožbo in zahtevajo odškodnino zaradi izgub v poslovanju.

## 5. ZAKLJUčEK

Računalništvo v oblaku predstavlja nove tehnološke in pravne izzive za forenzične preiskovalce. Zaradi karakteristik oblačnega računalništva, kot je porazdeljenost in elastičnost sistemov, zajem dokazov s fizičnih podatkovnih nosilcev navadno ni mogoč. Najpogosteje trenutno dokaze zajamejo tehniki ponudnika oblačnih storitev, kar pa zahteva visoko stopnjo zaupanja v ponudnika. Taka stopnja potrebnega zaupanja navadno ni sprejemljiva na sodišču, zato je potrebno uporabiti alternativne oblike zajema. Za oddaljeni zajem je bil predstavljen sistem FROST, ki se integrira v oblačno platformo OpenStack. Na nivoju upravljanja oblaka omogoča zajem slik diskov, dnevnika dogodkov ter dnevnika požarnega zidu za posameznega uporabnika. Pri tem preiskovalcu zagotavlja integriteto pridobljenih podatkov. S tem se izognemo zaupanju tehnikom ponudnika oblačnih storitev, čeprav moramo še vedno zaupati ponudnikovi strojni opremi. Kljub vsemu pa sta avtorja mnenja, da je tak pristop zaenkrat najboljši način za zajem dokazov na oblačnih platformah.

## 6. LITERATURA


[1] R. Buyya, J. Broberg, and A. M. Goscinski. *Cloud Computing: Principles and Paradigms*. Wiley, 2011.

[2] S. A. Crosby. *Efficient Tamper-Evident Data Structures for Untrusted Servers*. PhD thesis, Rice University, Huston, Texas, December 2009.

[3] J. Dykstra and A. T. Sherman. Acquiring forensic evidence from infrastructure-as-a-service cloud computing: Exploring and evaluating tools, trust, and techniques. *Digital Investigation*, 9:S90–S98, Aug. 2012.

[4] A. T. S. Josiah Dykstra. Design and implementation of FROST: Digital forensic tools for the OpenStack cloud computing platform. *Digital Investigation*, pages S88–S95, 2013.

[5] J. Li, M. N. Krohn, D. Mazières, and D. Shasha. Secure untrusted data repository (sundr). In *OSDI*, volume 4, pages 9–9, 2004.

[6] N. I. of Standards and Technology. *The NIST Definition of Cloud Computing*. Dostopno na: http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf.

# Forenzika brezžičnih lokalnih omrežij

## Wireless Local Area Network Forensics

Peter Miklavčič
miklavcic.peter@gmail.com
Maj 2014

## Povzetek
Povzetek bo narejen po recenziji iz končne različice teksta.

## Abstract
Placeholder.

## 1. UVOD
Brezžična lokalna omrežja po standardih IEEE 802.11, z angleško kratico WLAN in s tržnim imenom Wi-Fi so danes že vseprisotna in so zaradi potreb po pasovni širini nujen komplement mobilnim omrežjem. Tipično domače oziroma manjše omrežje vsebuje v eno napravo integriran usmerjevalnik, stikalo, požarni zid, brezžično dostopno točko in vsaj še DHCP in DNS strežnik in se ji pogovorno reče *router*. Nekatere take naprave pa vsebujejo integriran tudi modem, datotečni strežnik ali VoIP centralo. Praktično vse te domače naprave programsko temeljijo na operacijskem sistemu Linux, saj le-ta predstavlja dober kompromis med zmogljivostjo, realnočasnostjo in kompleksnostjo. Omrežni protokoli in ostali mehanizmi za delovanje IP omrežja so namreč postali že tako obsežni, da si večina podjetij preprosto ne more privoščiti razvijati svoje implementacije TCP/IP sklada.

Brezžična lokalna omrežja so zaradi svoje vseprisotnosti tako tudi pogosta tarča napadalcev. Odprta javna omrežja so priročna točka priklopa v internet za napadalca, z napadom na omrežje samo pa napadalec lahko še dodatno prikrije sledi. Tudi domača omrežja so velikokrat napadena zaradi tega, pogosto pa so tudi zelo slabo zaščitena. Pomembnejša omrežja pa so največkrat končna tarča napadalcev. Na sliki je prikazana statistika uporabljenih varnostnih mehanizmov v brezžičnih omrežjih po Ljubljani, ki je kot bo razvidno iz naslednjega poglavja, zaskrbljujoča, saj večina omrežij nima primernih varnostnih nastavitev.

Zaradi zgoraj naštetih in podobnih razlogov tako forenzika brezžičnih lokalnih omrežjih postaja vedno pomembnejša. V tej seminarski nalogi pa bodo predstavljene ključne možnosti

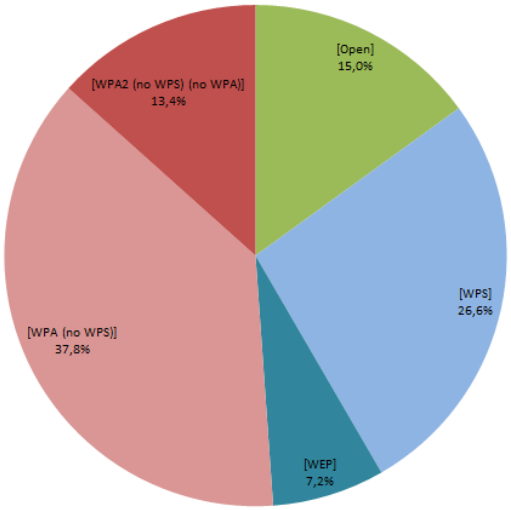



**Figure 1: Statistika uporabljenih WLAN varnostnih mehanizmov v Ljubljani.** [17]

pridobivanja dokazov v manjših omrežjih, ki uporabljajo integrirane naprave. Fokus bo predvsem na domačih usmerjevalnikih, ki so najpogosteje uporabljene naprave v takih omrežjih.

## 2. VARNOSTNI MEHANIZMI V WLAN
Tako kot v omrežjih Ethernet, imajo tudi uporabniki WLAN skupen medij. Vendar pa imajo le v slednjih skupen medij tudi različna omrežja, pod pogojem da delujejo na isti frekvenci (kanalu) ter so si v dosegu. Tudi če niso, lahko napadalec s pomočjo usmerjene antene dostopa do omrežja tudi iz večjih razdalj. Z ročno prenosljivo anteno je tako mogoče dostopati do omrežij na razdalji nekaj kilometrov, rekordi pa se gibljejo okoli več sto kilometrov. Zaradi ti. *broadcast* narave teh omrežij se v njih uporabljajo različni varnostni mehanizmi, katerih večina ima varnostne luknje. Najprej so našteti najbolj ranljivi.

### 2.1 Šibke zaščite
#### *2.1.1 Skrivanje SSID*
Skrivanje SSID se lahko vključi na praktično katerikoli dostopni točki posebej ali v kombinaciji z drugimi metodami. Ampak tudi v najnovejših verzijah standardov so okvirji, ki se pošljejo pred vzpostavitvijo šifrirane povezave, nešifrirani.

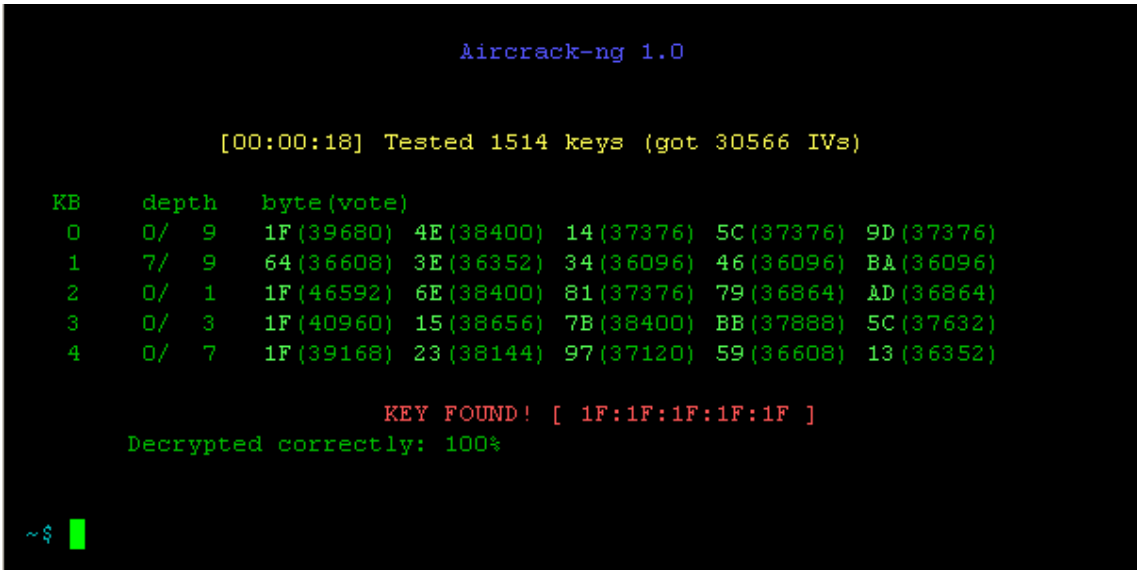



**Figure 2: Razbitje WEP ključa s programom *aircrack-ng* [14].**

Obstaja veliko orodij, ki iz spremljanja prometa na omrežju izluščijo SSID, tako da je pri tej vrsti zaščite praktično vseeno če je vklopljena ali ne, saj odbije le neizkušene napadalce.

### 2.1.2 Filtriranje MAC ali IP naslovov

Je dodaten mehanizem, ki se ga lahko vključi vzporedno ostalim in temelji na napačni predpostavki, da se MAC naslovi mrežnih vmesnikov ne spreminjajo. To zaščito se obide s prestrezanjem prometa drugih uporabnikov in uporabi njihovih MAC naslovov. Na enak način se zaobide tudi filtru IP naslovov. Tudi če je omrežje šifrirano, je zaobiti MAC filter po razbitju enkripcije trivialno, tako da se za ta način zaščite smatra podobno kot za zgornjega.

### 2.1.3 Wired Equivalent Privacy - WEP

WEP šifrira promet z algoritmom RC4, ki ima znane pomanjkljivosti. Avtentikacije ne uporablja oziroma v omrežje avtenticira kogarkoli, ki pošlje zahtevo. Šifrira le podatkovne okvirje. Že dolgo je znano, da je za zaščito neprimeren in dostopna so tudi že najrazličnejša orodja za razbitje, ki delujejo tudi na telefonih ter še manjših napravah in so že razmeroma hitra ter dodelana.

### 2.1.4 Captive portali

Nekatera omrežja imajo zaščito implementirano šele na 7. (aplikacijskem) nivoju OSI modela, in sicer s ti. captive portali, ki uporabnika ob prvi povezavi preusmerijo na spletni strežnik, kjer opravi avtentikacijo. V teh omrežjih ni enkripcije in gre v bistvu za odprta omrežja, tako da napadalec lahko prestreže MAC in IP naslove ter jih uporabi. Še ena možnost je napad na sam portal z npr. SQL injeciranjem ali drugimi znanimi napadi na spletne strani.

### 2.1.5 Wi-Fi Protected Setup - WPS

WPS je dodatna funkcija, ki se uporablja za lažje in hitrejše povezovanje naprav v omrežje. Privzeti način delovanja je, da s pomočjo 8-mestne PIN kode, ki jo vnese uporabnik, prenese glavni WEP, WPA ali WPA2 ključ. Koda je v bistvu 7-mestna, saj je zadnje mesto kontrolna vsota in ima tako možnih 10 milijonov kombinacij. Vendar pa se pri avtentikaciji preveri vsaka polovica kode posebej, tako da je za uspešno ugibanje PIN kode potrebnih največ 11.000 poskusov. Nekateri proizvajalci so imeli tako slabo narejen WPS da je avtor najdene ranljivosti članek naslovil *Brute forcing Wi-Fi Protected Setup: When poor design meets poor implementation.*

Srž problema pa še niti ni v tako majhnem dejanskem številu kombinacij, ampak implementaciji. Nekatere naprave namreč nimajo omejenega števila poskusov povezave preko WPS, tako da napadalec lahko v le nekaj urah poskusi vseh 11.000 kombinacij. Drug problem pri implementaciji pa se je pokazal po javni objavi napada. Nekaterim napravam se namreč funkcije WPS ne da izklopiti ali pa se funkcija dejansko ni izklopila, kar je bila v času po objavljenem napadu edina možna obramba.

### 2.1.6 Wi-Fi Protected Access - WPA

Tudi zaščita WPA ima že pokazane določene pomanjkljivosti. Ker ima korenine v WEP, so iz znanih napadov nanj izpeljali tudi nekaj napadov na WPA. Do sedaj je bilo pokazano, da je možno dekriptirati promet od dostopne točke do uporabnika ter injecirati pakete v omrežje. Druga vrsta napada pa je znani slovarski *bruteforce* napad, na katerega so občutljiva slaba gesla.

## 2.2 Močnejše zaščite

### 2.2.1 802.11i - WPA2

WPA2 je sicer edina zaščita, ki jo najdemo v uporabi v domačih omrežjih in nima objavljenih resnejših pomanjkljivosti, a je ob uporabi slabega gesla še vedno občutljiva na slovarski napad. Edina odkrita težava do sedaj je poimenovana Hole196 in omogoča prestrezanje prometa drugih uporabnikov v omrežju. Kljub temu da uporablja močan krip-

tografski algoritem AES, pa je znana težava še, da določeni tipi kontrolnih okvirjev niso kriptirani. To se je sicer delno izboljšalo s standardom 802.11w, a je del kontrolnih okvirjev še vedno nešifriran in je tako WPA2 med drugim še vedno občutljiva na DoS napad imenovan *disassociation flood*.

### 2.2.2 *Extensible Authentication Protocol - EAP*

Poleg zaščite z geslom (PSK), je to še drugi način delovanja WPA in WPA2. Deluje tako, da posreduje uporabnikove 802.1x pakete na RADIUS strežnik, le-ta pa opravi avtentikacijo. To pomeni, da deluje večina iz žičnih omrežij znanih napadov na EAP protokole. Ker gre za nabor različnih protokolov med njimi najdemo tako zastarele in razbite protokole kot tudi najbolj varne varnostne protokole v WLAN, ki uporabljajo kriptografijo z javnimi ključi (PKI).

## 3. ZGRADBA WLAN USMERJEVALNIKOV

Na sliki 3 je prikazana blokovna shema zgradbe tipičnega integriranega usmerjevalnika (po domače *routerja*). Delovni spomin, procesor in mrežni vmesniki so ponavadi integrirani v SoC, namenjen za take naprave. Notri je včasih integriran še flash spomin, ki je edini blok na sliki s poudarjeno enosmerno povezavo, saj se prebere ob zagonu in občasno med delovanjem, piše pa izključno med nadgrajevanjem programske opreme in shranjevanjem nastavljenih parametrov. Med ostalimi narisanimi povezavami pa poteka veliko prometa tudi ob čisto normalnem delovanju, saj naprave vedno pošiljajo kake podatke. Tudi če vsi uporabniki na omrežju ne počnejo nič na svojih napravah, se med temi pošiljajo vsaj kontrolni okvirji.

V SoC pa je lahko integrirano še stikalo, ki je v tem primeru še vedno strojno izvedeno, le da je na istem kosu silicija. Za razliko od usmerjevalnika in požarnega zidu, katerih funkcije se vršijo programsko v CPU. Funkcije usmerjanja so že del samega jedra Linux, le omogočiti je treba *IP forwarding*, medtem ko požarni zid največkrat vrši dodatek jedru *netfilter*, katerega uporabnik nastavlja z *iptables*. Tudi funkcijo stikala zna sistem Linux vršiti programsko in še več, omogoča tudi požarni zid na 2. (fizičnem) nivoju OSI modela z *ebtables*. Nekatere izvedbe usmerjevalnikov pa imajo celo WAN priključek povezan na stikalo in zanj samo ločen VLAN, kar se lahko izvaja strojno ali programsko. Programsko pa se seveda v vseh usmerjevalnikih izvajajo strežniki za protokole 7. (aplikacijskega) nivoja OSI modela (npr. DNS).

Integrirani (*domači*) usmerjevalniki največkrat poganjajo Linux, ki ima svoj korenski direktorij (/) na flash spominu in je med normalnim delovanjem zaradi možnosti izrabe pisalnih ciklov flash spomina priklopljen v bralnem načinu. Le pri nalaganju druge programske opreme ali ob spreminjanju nastavitev se flash spomin prepiše. Določeni modeli usmerjevalnikov pa imajo za nastavitve nekaj deset kilobajtov velik NVRAM kamor se zapisujejo nastavitve. Večina forenzično zanimivih podatkov se tako nahaja v delovnem spominu usmerjevalnika, kar pa pomeni, da le-ta ne sme izgubiti napajanja, ker se v tem primeru vsebina delovnega spomina izgubi. Tukaj nastopi dilema, saj je vsebino delovnega spomina zelo težko shraniti ali pregledati z drugo napravo, poleg tega pa so usmerjevalniki narejeni tako, da med normalnim delovanjem ne pišejo po spominu flash in NVRAM-u. Ročen zajem podatkov na kraju zločina tako ustreza načelu, da se

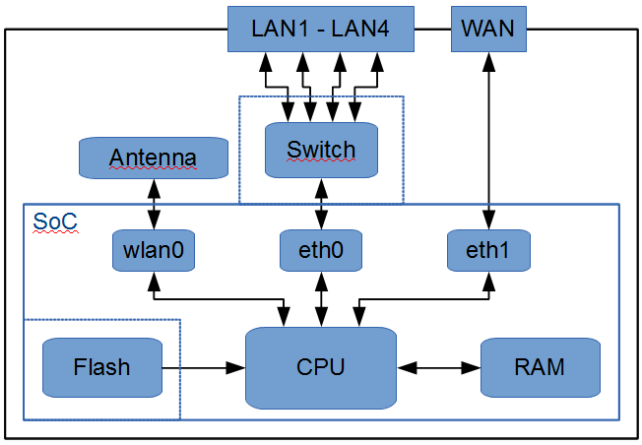



**Figure 3: Zgradba tipičnega integriranega usmerjevalnika.**

zajete podatke striktno le bere in se po njih nikoli ne piše. Večja težava je s podpisovanjem zajetih podatkov.

## 4. FORENZIKA WLAN

V spodnjih poglavjih so predstavljeni so forenzični podatki, ki se lahko nahajajo na mrežnih napravah, predvsem usmerjevalnikih. Le-ti imajo običajno do 1 GHz (tipično MIPS) procesor, do 128 MB delovnega spomina in do 64 MB flash spomina. Seveda obstajajo tudi izjeme, ampak tipični usmerjevalniki nimajo dovolj prostora, da bi beležili vse prometne podatke in se je za iskanje dokazov treba bolj potruditi. Predstavljenih pa je tudi nekaj lastnosti končnih naprav, ki so lahko uporabne pri forenzični preiskavi.

### 4.1 Forenzika mrežnih naprav

#### 4.1.1 *Usmerjevalnik je računalnik*

Na usmerjevalnik je potrebno gledati kot na računalnik kar je bilo predstavljeno že v prejšnjem poglavju, saj sta konec koncev tudi zmožna poganjati enak operacijski sistem. Tako se lahko npr. okuži in priklopi v botnet tudi usmerjevalnik, pri zbiranju dokazov pa se lahko uporabi metode za osebne računalnike s sistemom Linux. Zanimiva je tako celotna omrežna konfiguracija na usmerjevalniku, ki jo je še več kot na tipičnem osebnem računalniku, saj je usmerjevalnik dosti bolj aktiven na omrežju kot končne naprave. Če napadalec pridobi dostop do usmerjevalnika, lahko na njem tudi pusti sledi v obliki datotek (npr. bash skripte). Preiskovalec si lahko pomaga s proizvajalčevo sliko programske opreme (*firmware*), in sicer tako da jo primerja z zajeto. Tako lahko zanesljivo ugotovi, če je napadalec za seboj pustil kakšne sledi oziroma datoteke, ki bi se morale ohraniti po ponovnem zagonu usmerjevalnika.

#### 4.1.2 *Statistični podatki*

Že preprosti statistični podatki, kot je čas delovanja (uptime) ali čas delovanja povezave, so preiskovalcem lahko uporabni. V primeru da je storilec med zločinom izklopil elektriko ali telefonsko ali katero drugo povezavo, se to lahko vidi tudi na usmerjevalniku. Tudi navadni števci prometa, ki so aktivni na vseh vmesnikih, lahko indicirajo npr. da gre za osebo, ki zelo malo uporablja internetno povezavo. V

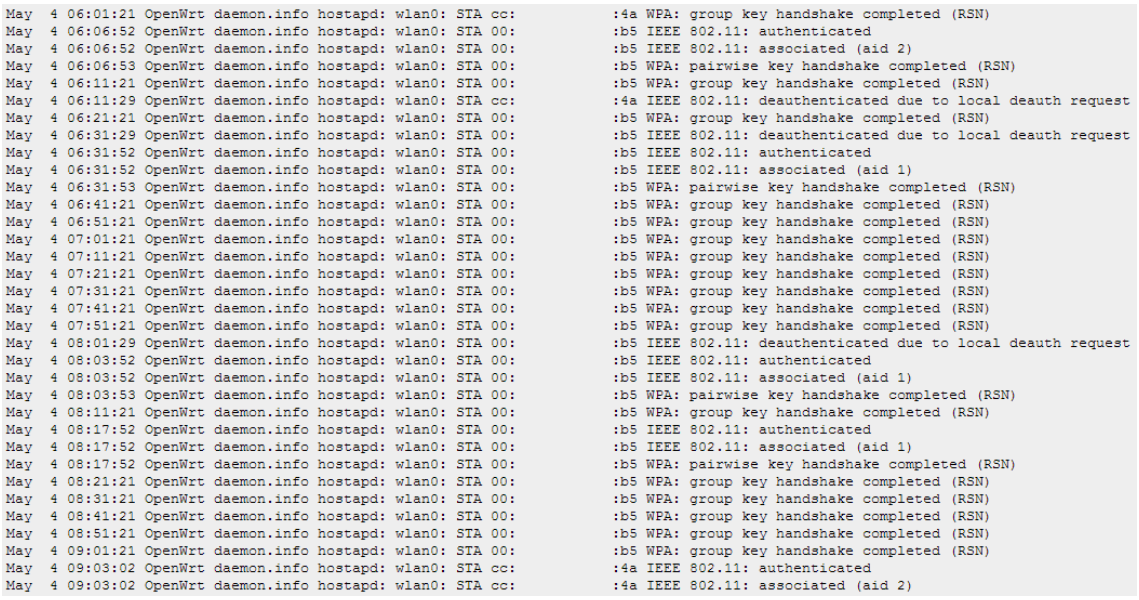



**Figure 4: Syslog usmerjevalnika z *OpenWrt*, vidi se prijavljanje in odjavljanje naprav v brezžično omrežje.**

tem primeru je tako manj verjetno, da gre za osebo, ki vodi botnet.

### 4.1.3 Dnevniki

Večina Linux usmerjevalnikov beleži dnevnike v *syslog*, nekateri pa celo izrisujejo grafe za promet, število uporabnikov, število povezav, ipd. Vse to so preiskovalcem pomembne informacije, saj lahko rekonstruirajo potek dogodkov ker se v syslog med drugim zapisuje tudi kdaj se katera brezžična naprava poveže na omrežje in do neke mere tudi do kdaj je povezana. Šifrirni ključ se namreč periodično obnavlja, kar se tudi zapisuje v sistemski dnevnik, v pomoč pa so tudi DHCP zapisi. Preiskovalec mora biti pozoren tudi na SNMP nastavitve, saj je možno da se dnevniki shranjujejo na drugem računalniku.

### 4.1.4 Požarni zid

Požarni zid ima lahko svoje dnevniške zapise in statistične tabele. Iz obojih se da sklepati še več o dogajanju na omrežju ali morebitnem preusmerjanju ali kakšni drugi manipulaciji s prometom. Funkcija požarnega zidu se v integriranih usmerjevalnikih vrši programsko, kar pomeni da je implementirana v programski opremi, ki teče na centralnem procesorju. Nekatere funkcije, kot npr. switching, bridging, IGMP snooping, ipd. pa se lahko na usmerjevalniku vršijo strojno, kar pomeni da ima usmerjevalnik vgrajen čip, ki podpira to funkcijo na strojnem nivoju. Vse funkcije, ki se vršijo programsko imajo nekje v OS ponavadi dostopne še dodatne parametre. Prav tako nekateri gonilniki, npr. za brezžični vmesnik, ponavadi omogočajo dostop do detaljnejših informacij.

### 4.1.5 DHCP strežnik

Praktično v vsakem manjšem omrežju je usmerjevalnik hkrati še DHCP strežnik in le-ta si zapomni povezane naprave vsaj za čas dodelitve, ki se spreminja, največkrat opažena vrednost pa je 24 ur. Nekateri usmerjevalniki v DHCP tabeli ohranijo vse zapise do ponovnega zagona, nekateri pa jih celo zapišejo na flash (kar je zelo redko). Tabela je zanimiva tudi zato, ker pove katere naprave je žrtev uporabljala.

### 4.1.6 UPnP strežnik

V UPnP tabeli sicer ni vseh naprav, vendar pa za tiste ki so daje večji vpogled v storitve, ki so bile na napravi v uporabi. Prav tako pa manj izkušen napadalec hitro pozabi na to tabelo oziroma računa, da je ročno nastavljen IP na njegovem vmesniku dovolj, da računalnik ne komunicira direktno z usmerjevalnikom ampak samo skozi njega, a temu ni nujno tako.

### 4.1.7 DNS predpomnilnik

Večina domačih usmerjevalnikov je hkrati tudi DNS strežnik za LAN in ima vlogo DNS predpomnilnika v primeru da več uporabnikov omrežja razrešuje iste domene. V njem se lahko skrivajo informacije do katerih strani je napadalec dostopal, če je za dostop do interneta uporabil to omrežje.

### 4.1.8 PPPoE gesla

Gesla za dostop do omrežja ponudnika ki uporablja avtentikacijo so dostikrat enaka geslom za e-poštni predal pri tem ponudniku. Čeprav se mora preiskovalec zavedati, pod katerimi pogoji lahko tako informacijo uporabi, ima le-ta še vedno veliko forenzično vrednost, saj lahko zelo skrajša čas v katerem preiskovalci pridejo do pošte od žrtve, saj ni potrebno čakati odziv ponudnika e-pošte.

### 4.1.9 Seznam povezanih in dovoljenih naprav

Tabela na brezžično omrežje povezanih naprav je preiskovalcem koristna, ker lahko vidijo katera naprava je zagotovo na kraju zločina. Prav tako pa je seznam dovoljenih MAC naslovov forenzikom lahko uporaben za preverjanje, ali so pregledali vse žrtvi znane naprave.

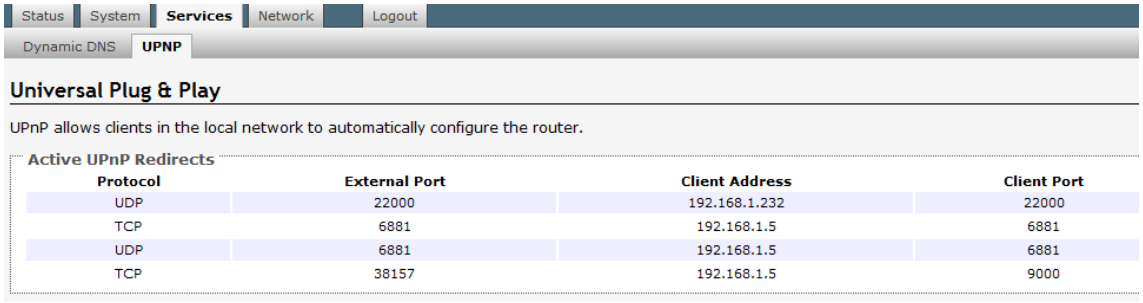



**Figure 5: UPnP tabela usmerjevalnika z *OpenWrt*, vidijo se odprte povezave napravam ki uporabljajo UPnP [16].**

#### *4.1.10 Ostala omrežja v dosegu*

V primeru, da so v dosegu kraja zločina druga odprta ali s privzetimi gesli zaščitena omrežja, se lahko storilčeve naprave avtomatsko povezujejo na njih in tako pustijo sledi. V primeru odprtih omrežij se lahko storilčeva naprava poveže sama od sebe, v primeru s privzetimi gesli zaščitenih omrežij pa je pogoj ta, da je storilec enkrat s to napravo že uporabljal omrežje, ki uporablja enako kombinacijo SSID in gesla.

### 4.2 Forenzika končnih naprav

#### *4.2.1 Računalnik je skoraj usmerjevalnik*

Tudi iz računalnika ali telefona je možno pridobiti dosti zgoraj opisanih informacij. Pomembni so dnevniški zapisi (OS, DHCP, požarni zid), IP nastavitve, DNS predpomnilnik, večina domačih usmerjevalnikov in ostalih mrežnih naprav pa se nastavlja preko spletnih vmesnikov. To pomeni da ob dostopu do nastavitev naprave ostanejo sledi na napadalčevem računalniku kakor ob brskanju po spletu. V poštev pridejo tako vse tehnike, ki se uporabljajo za ekstrahiranje dokazov iz podatkov o brskanju po spletu.

#### *4.2.2 Sondiranje - probe requests*

V splošnem brezžične dostopne točke oglašajo svojo prisotnost približno desetkrat na sekundo s ti. beacon okvirji. Pasivno skeniranje omrežij na odjemalcih zato poteka približno tako, da na vsakem kanalu poslušajo nekaj časa in sestavijo seznam omrežij v dosegu. V primeru, da izklopimo oddajanje SSID, se spremeni samo to, da se v beacon okvirjih ne oddaja SSID. Standard 802.11 pa pozna tudi aktivno skeniranje omrežij, kjer odjemalci pošiljajo ti. *probe request*, ki vsebujejo SSID željenega omrežja, torej se pošiljajo le probe request okvirji od znanih omrežij. Dostopna točka s pravim SSID se potem oglasi nazaj napravi s ti. *probe response* okvirjem. V primeru, da ima omrežje izklopljeno oddajo SSID, je to edini način da se omrežje in naprava najdeta.

Problem je v tem, da niti v standardu 802.11i (WPA2), niti v poznejši nadgradnji 802.11w, ki se ukvarja s to tematiko, ti okvirji niso kriptirani, kar je bilo že omenjeno v drugem poglavju. Tako naprave, ki oddajajo take okvirje izdajo, katera vsa omrežja poznajo in to naredijo z nešifriranim oddajanjem na vse strani. To sicer predstavlja zanimivo odtekanje informacij, ki v povezavi s kartiranjem dostopnih točk (kar npr. počne podjetje Google med slikanjem krajev za svojo storitev Street View), lahko pripelje do še bolj zanimivih vprašanj o zasebnosti, a v tem primeru je opisano, kako si preiskovalec lahko s to lastnostjo pomaga. In sicer če naprava ni povezana na nobeno znano omrežje, a ima vklopljen WLAN vmesnik, pošilja *probe request* okvirje. Zato mora preiskovalec napravo le izolirati (npr. s Faradayevo kletko) in poslušati katera omrežja le-ta sondira. Tako pridobi seznam znanih omrežij napravi brez najmanjše invazivnosti.

#### *4.2.3 MAC naslovi - skrivanje s programsko opremo*

MAC naslovi se uporabljajo za naslavljanje na 2. (fizičnem) nivoju OSI modela, torej so ključni v fizičnih omrežjih Ethernet in WLAN, ne pa logičnih (IP), ki tečejo preko prvih. Načeloma bi morala vsaka naprava imeti svoj unikaten MAC naslov, a se le-te da na dosti napravah poljubno spreminjati. V primeru napada na brezžično omrežje napadalec tako lahko spremeni svoj MAC naslov. Vendar pa ima dosti naprav svoj MAC naslov zapisan še kje, npr. na nalepkah na ali v napravi ali na embalaži, lahko pa tudi da ima naprava MAC naslov *zapečen* v strojni opremi, tako da se ob vsakem ponovnem zagonu ponastavi.

#### *4.2.4 Brezžični vmesniki - skrivanje s strojno opremo*

Napadalec lahko tudi uporablja drug WLAN vmesnik, torej zamenja vgrajeno kartico za čas napada z drugo ali doda USB vmesnik. Vendar vsaj pri OS Windows praktično vsaka kadarkoli priklopljena naprava na računalnik pusti sled, preko katere lahko preiskovalci ugotovijo katera strojna oprema je bila v uporabi. Uporabljeni mrežni vmesniki se nahajajo v Windows Registru na naslovu `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\NetworkCards`.

#### *4.2.5 Konfiguracije brezžičnih omrežij*

Praktično vsak OS si zapomni brezžična omrežja, na katera se poveže, da jih ni potrebno vedno znova nastavljati. Tudi če se vmesnik, preko katerega je bila povezava narejena fizično odstrani iz računalnika, so informacije o obiskanih omrežjih spravljene na računalniku in tako še vedno dostopne. V OS Windows se tako nahajajo na `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\NetworkList\Profiles`. V OS Linux se nahajajo v konfiguracijskih datotekah od programa, ki skrbi za povezave (npr. `/etc/NetworkManager`), zaščitena omrežja pa še v konfiguraciji programa wpa_supplicant (v `/etc/wpa_supplicant/wpa_supplicant.conf`), ki je potreben za WPA, WPA2 ali EAP avtentikacijo.

## 5. ZAKLJUČEK

Iz predstavljenih možnosti je razvidno, da imajo usmerjevalniki in ostale mrežne naprave veliko in vedno večjo foren-

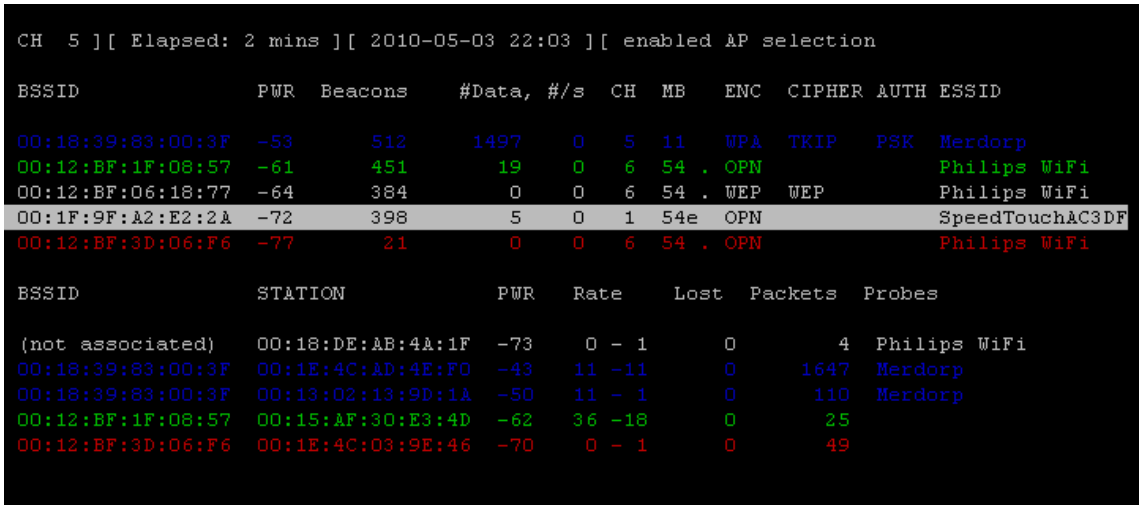
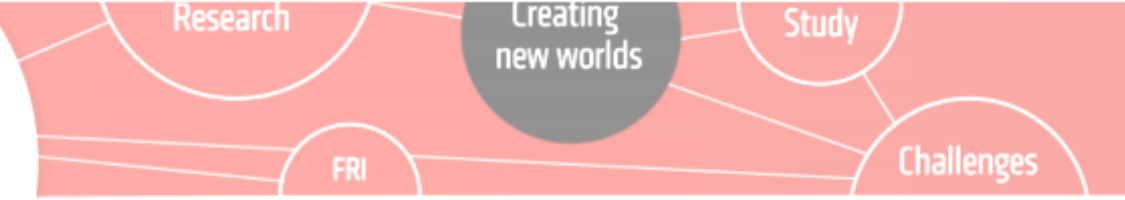



**Figure 6: Opazovanje prometa na WLAN kanalu s programom *airodump-ng*, vidijo se poslani probe request okvirji [15].**

zično vrednost. Že ob zbiranju gradiva pa sem naletel na dejstvo, da to področje v forenziki ne dobiva dovolj pozornosti, kar so tako ali drugače potrdili tudi številni citirani avtorji. Ocenjujejo, da se je glaven fokus raziskav v zadnjem času preusmeril iz računalnikov na mobilne naprave, mrežnim napravam in ostalim vgrajenim sistemom pa ni posvečeno dovolj pozornosti. Glede na to, da v računalništvu prihajajo časi vsesplošne uporabe SSD, kar nekateri strokovnjaki zaradi njihovih lastnosti (predvsem mehanizma TRIM) ocenjujejo za konec zlate dobe digitalne forenzike, bi se verjetno moralo fokus forenzičnih raziskav še bolj usmeriti v omrežja, saj se lahko zgodi da se bo glavnina dokazov morala iskati tam.

## 6. VIRI


1. Said, H. and Guimaraes, M. and Al Mutawa, N. and Al Awadhi, I., *Forensics and war-driving on unsecured wireless network*, Internet Technology and Secured Transactions (ICITST), str. 19-24, 2011.
2. Zhongli Liu and Yinjie Chen and Wei Yu and Xinwen Fu, *Generic network forensic data acquisition from household and small business wireless routers*, World of Wireless Mobile and Multimedia Networks (WoWMoM), str. 1-6, 2010.
3. Akers, P. and McGrew, W. and Dampier, D.A., *WiFi Stakeout: A network forensics tool for reconnaissance and first responders*, Engineering and Industries (ICEI), str. 1-4, 2011.
4. Szewczyk, P., *ADSL Router Forensics Part 1: An introduction to a new source of electronic evidence*, Proc. 5th Australian Digital Forensics Conf. Perth, str. 123-131, 2007.
5. Szewczyk, P., *ADSL Router Forensics Part 2: Acquiring Evidence*, Proc. 7th Australian Digital Forensics Conf Perth, str. 109-117, 2009.
6. Szewczyk, P., *The ADSL Router Forensics Process*, Journal of Network Forensics 2(1), str. 4-13, 2010.
7. Tsow, A., Jacobsson, M., Yang, L., Wetzel, S., *Warkitting: the drive-by subversion of wireless home routers*, Journal of Digital Forensics Practice 1(3), str. 179-192, 2006.
8. Szewczyk, P., *Analysis of Data Remaining on Second Hand ADSL Routers*, Journal of Digital Forensics, Security and Law 6(3), str. 17–30, 2011.
9. Viehboeck, S., *Brute forcing Wi-Fi Protected Setup: When poor design meets poor implementation*, dostopno na `http://sviehb.files.wordpress.com/2011/12/viehboeck_wps.pdf`, ver. 3, 2011.
10. Antoniewicz, B., *802.11 Attacks*, dostopno na `http://www.mcafee.com/us/resources/white-papers/foundstone/wp-80211-attacks.pdf`, ver. 1.
11. AirTight Networks, *WPA 2 Hole196 Vulnerability – FAQ*, dostopno na `http://www.airtightnetworks.com/fileadmin/pdf/WPA2-Hole196-vulnerability-FAQs.pdf`.
12. OpenWrt Dokumentacija, dostopna na `http://wiki.openwrt.org/doc/start`.
13. OpenWrt Wiki, dostopen na `http://wiki.openwrt.org`.
14. Slika, dostopna na `http://www.aircrack-ng.org/screenshots/aircrack-ng.gif`.
15. Slika, dostopna na `http://www.aircrack-ng.org/screenshots/airodump-ng_interaction.gif`.
16. Slika, dostopna na `http://advanxer.com/blog/wp-content/uploads/2012/06/upnp2.png`.
17. Vir je Laboratorij za telekomunikacije Fakultete za elektrotehniko Univerze v Ljubljani.
18. Mustafa, L., *Writing U-Boot directly to TP-Link WR741ND router flash*, dostopno na `http://wlan-si.net/en/blog/2013/01/07/writing-u-boot-directly-to-tp-link-wr741nd-`




# Digitalna forenzika na sistemih

# Rekonstrukcija manipuliranih poizvedb iz InnoDB Redo Log datotek

[Poglobljeni povzetek]

Dean Črnigoj
Fakulteta za računalništvo in informatiko
Gradišče pri Vipavi 51a
Vipava, Slovenija
dc9932@student.uni-lj.si

Nejc Bizjak
Fakulteta za računalništvo in informatiko
Podgrič 2
Podnanos, Slovenija
nb7232@student.uni-lj.si

## POVZETEK

InnoDB je podatkovni sistem za podatkovne baze MySQL, kateri je pridobil veliko uporabnikov v zadnjih letih. Ta članek[2] opisuje, kako lahko z forenzično analizo Redo log datotek pridobimo podatke kateri so bili manipulirani, če je imel napadalec administratorske pravice ali pa je bil sam administrator. Čeprav bi napadalec lahko spremenil log datoteke, podatkov ni lahko zakriti v Redo log datotekah. Glede na način poizvedbe, bomo v tem članku prikazali kako se obnovi manipulirane stavke tipa *Insert, Delete* in *Update.*

## Ključne besede

MySQL,InnoDB,digitalna forenzika,podatkovne baze,log datoteke

## 1. UVOD

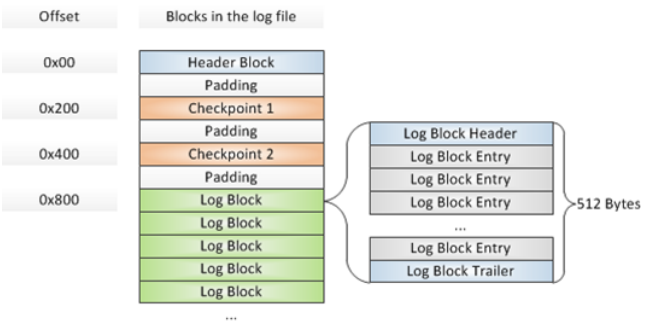
Ko izvedemo stavek SQL, InnoDB podatkovni sistem shrani delce stavka na različnih lokacijah. Če poznamo lokacije lahko z forenzično analizo odkrijemo brisane, ter spremenjene podatke in ustvarimo delni časovni potek sprememb. Ta članek opisuje nov način pridobivanja podatkov iz Redo log datotek, katere so bile do sedaj neznanka in omogočajo pridobitev več podatkov in te bolj zanesljivo, saj je le te težje ponarediti. Ker je InnoDB transaction-safe podatkovni sistem, mora biti sposoben razveljaviti vsake spremembe. To naredi tako da vsak stavek pretvori v atomske poizvedbe, in le te shrani v Redo log datoteke. Če pomislimo, da je MySQL najpopularnejša odprto-kodna podatkovna baza in da je z različico MySQL 5.5 InnoDB privzet podatkovni sistem, nam je ta način v veliko pomoč pri forenzični analizi večini novih podatkovnih baz.

## 2. INNODB PODATKOVNI SISTEM

MySQL ponuja veliko različnih podatkovnih sistemov. Med njimi trenutno prevladuje *InnoDB*[1] in to prav zaradi njegove hitrosti ter zanesljivosti. *InnoDB* polno podpira model ACID (Atomicity, Consistency, Isolation, Durability), kar pomeni, da so njegove operacije atomske majhne in s tem ali v celoti izvršene ali ne, da so podatki vedno konsistentni, da se vsaka transakcija izvaja kot bi bila izolirana oziroma izvršila serijsko, ter da enkrat ko je transakcija uspela naj bi bila shranjena v vsaki situaciji (Zato potreba po Redo log datotekah.) *InnoDB* prav tako podpira zaklepanje vrstic, kar pomeni da ni več potrebe po zaklepanju celotne tabele Te in še mnoge druge lastnosti so prepričale mnogo uporabnikov da so še pred uradnim privzetim podatkovnim sistemom tega pričeli uporabljati.

## 3. STRUKTURA REDO LOG DATOTEKE

V privzetih nastavitvah MySQL, InnoDB vsebuje dve datoteki Redo log *ib_logfile0* in *ib_logfile1* z velikostjo 5 MB. Obe datoteki imata enako strukturo (slika 1) in InnoDB prepisuje podatke v eno ali drugo datoteko. Podobno kot podatkovne datoteke MySQL so Redo log datoteke razdeljene na več blokov. Ti bloki so trije različni in sicer en *blok glave*, dve *kontrolni točki* in več *log blokov.* Blok glave skupaj z kontrolnimi točkami pogosto imenujejo tudi *glava datoteke* in ima skupno velikost 2048 bajtov.




**Slika 1: Struktura datoteke Redo log.**

### 3.1 Blok glave

Prvi del datoteke je blok glave in vsebuje glavne informacije o Redo log datoteki. Ta blok ima fiksno dolžino 48 bajtov in se začne na začetku (odmik 0x00). Tabela 1 prikazuje polja ter njihov odmik v bloku glave.

| Odmik | Dolžina( bajt ) | Razlaga |
|---|---|---|
| 0x00 | 4 | Številka skupine v log datoteki. |
| 0x04 | 8 | Prva log sekvenčna številka (lsn) v tevi datoteki. |
| 0x0C | 4 | Arhivirana številka datoteke. |
| 0x10 | 32 | To polje je uporabljeno samo za potrebe InnoDB Hot backup. Vsebuje *ibackup* in čas kreiranja. Uporabljeno je za prikaz informacij uporabniku kdaj je bil *mysqld* zagnan prvič po obnovitvi podatkov. |

**Tabela 1: Polja v bloku glave.**

| Odmik | Dolžina( bajt ) | Razlaga |
|---|---|---|
| 0x00 | 8 | Številka kontrolne točke. |
| 0x08 | 8 | Log sekvenčna številka (lsn) v kontrolni točki. |
| 0x10 | 4 | Odmik v log vnosu, izračunan s pomočjo *log_group_checkpoint()*. |
| 0x14 | 4 | Velikost datotečnega predpomnilnika (fiksna vrednost 2, 1024,1024). |
| 0x18 | 8 | Arhivirana log sekvenčna številka. Če *UNIV_LOG_ARCHIVE* ni omogočeno, je vrednost FF FF FF FF FF FF FF FF. |
| 0x20 | 256 | Prazen prostor za odmik. |
| 0x120 | 4 | Zgoščena vsota 1 (za preverjanje pravilne vrednosti od 0x00 do odmika 0x19F). |
| 0x124 | 4 | Zgoščena vsota 2 (za preverjanje pravilne vrednosti od 0x08 do odmika 0x124). |
| 0x128 | 4 | Trenutni neizkoriščen prostor v tablespace 0, podan v mega bajtih. Uporabljen za odločanje če v sistemu, kateri je brez podaljševanja datoteke se lahko neuporabljeni podatki stisnejo. |
| 0x12C | 4 | Polje ki pove ali obstaja polje nad njim. (Dodano v InnoDB različici 3.23.50). |

**Tabela 2: Polja v kontrolni točki.**

## 3.2 Kontrolni točki

InnoDB uporablja sistem kontrolnih točk. Kontrolne točke so vitalnega pomena za obnovitev podatkovne baze po sesutju sistema ali podatkovne baze. Kontrolni točki sta dve, zato da imamo v vsakem trenutku vsaj eno veljavno kontrolno točko(v primeru da zapis pred zadnjo ni uspel). Pisanje v kontrolne točke je po tem, ko so zapisani podatki v log blok in se izvaja izmenično v prvo ali drugo kontrolno točko. Odmik v Redo log datoteki je statičen za obe kontrolni točki in znaša (0x200 ter 0x400). Vsaka kontrolna točka ima dolžino 304 bajtov in vsebuje polja opisana v tabeli 2.

## 3.3 Struktura log bloka

Vnosi v Redo log datoteko so shranjeni v log blok. Vsak blok je velik 512 bajtov podatkov, njegova velikost je namerna saj je v času nastajanja bila enaka velikosti sektorja na disku in je ob zapisu porabila točno en sektor. Vsak blok je razdeljen na tri dele: *glava log bloka*, *podatke*, in *noga log bloka*. Struktura je namensko uporabljena za čim boljšo hitrost in in navigacijo skozi datoteko. V naslednjih podpoglavjih bomo podrobneje predstavili strukturo glave, podatkov in noge.

### *3.3.1 Struktura glave log bloka*

Prvih 14 bajtov vsakega log bloka se imenuje *glava log bloka*. Ta glava vsebuje vse informacije potrebne za InnoDB da ureja in prebira log podatke. Vsakih 512 bajtov InnoDB avtomatično ustvari novo glavo in nov log blok. Ker Redo log datoteka prvo vsebuje blok glave, kontrolni točki in imajo le te vmes še nekaj nedoločenega prostora je prva glava log bloka na poziciji 0x800. Tabela 3 prikazuje polja, ki jih vsebuje glava log bloka.

| Odmik | Dolžina( bajt ) | Razlaga |
|---|---|---|
| 0x00 | 4 | Številka glave log bloka (če je najbolj levi bit 1, potem to pomeni da je ta log blok prvi v pisalnem segmentu). |
| 0x04 | 2 | Število že zapisanih bajtov v tem bloku. |
| 0x06 | 2 | Odmik do prvega log record group od tega bloka (poglavje 3.3.3). |
| 0x08 | 4 | Število trenutno aktivnih kontrolnih točk. |
| 0x0C | 2 | Velikost Hdr. |

**Tabela 3: Polja v glavi log bloka.**

Kot že opisano v poglavju 3.2, trenutno aktivni log blok vedno kaže na trenutno aktivno kontrolno točko. Ta informacija je osvežena vsakič ko je log vsebina prepisana na disk.

### 3.3.2 Struktura noge log bloka

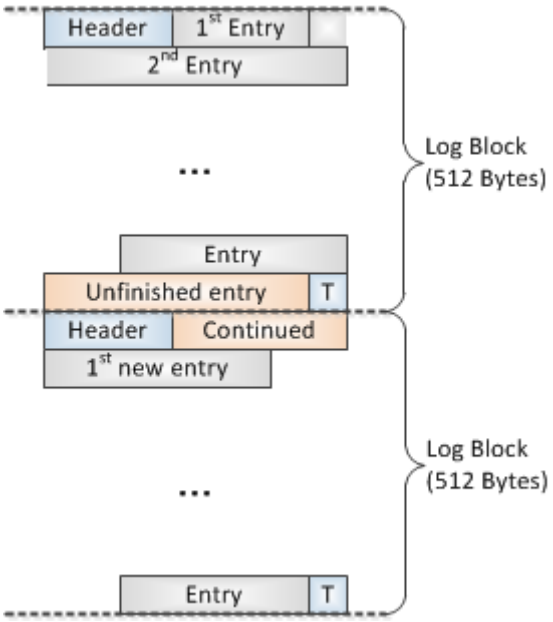
Noga log bloka vedno vsebuje zgoščeno vsoto celotnega log bloka. Vsebina je opisana v tabeli 4.

| Odmik | Dolžina( bajt ) | Razlaga |
|---|---|---|
| 0x00 | 4 | Zgoščena vsota celotnega log bloka. Do vključno različice InnoDB 3.23.52 to polje ni vsebovalo zgoščene vsote ampak vrednost Hdr iz glave bloka. |

**Tabela 4: Polja v nogi log bloka.**

### 3.3.3 Razlivanje vsebine log čez več blokov

V primeru, da je vsebina log prevelika, kot je še prostora v 512 bajtov velikem log bloku, se vsebina prelije čez dva log bloka. Vsebina trenutnega bloka se zasede do zadnjih štirih bajtov, kjer je potreben prostor še za nogo. Nov blok se ustvari in se po glavi zapolni s preostalo vsebino log zapisa. Odmik v glavi na poziciji 0x04 in 0x05 je uporabljen da pove kje v log bloku se začne nov zapis (Glej tabelo 3). To je potrebno da se identificira kje se začne naslednji zapis, brez da bi bilo potrebno pogledati v prejšnji log blok, s tem se poveča hitrost navigacije zapisa. Slika 2 prikazuje strukturo log bloka in kako se lahko vsebina log preliva čez dva log bloka.




**Slika 2: Struktura log bloka in prelivanje.**

## 4. REKONSTRUKCIJA STAVKOV

V tem poglavju demonstriramo kako pridobiti podatke iz Redo log datotek opisanih v prejšnem poglavju. Ker je več polj v stisnjeni obliki, ne moremo vedno pridobiti točne definicije dolžine polja. Ta dolžina je podana po metodi razširjanja. Ta polja so v tabelah označena z simbolom (o). V tem članku se osredotočamo predvsem na nov InnoDB kompakten format shranjevanja log vsebine, kateri je prepoznaven po *mlog_comp_* v log tipu. Starejše različice InnoDB log potrebujejo precej več prostora in niso opisane v tem članku. V naši analizi se osredotočamo na tri različne stavke, *Insert*, *Delete* in *Update*, ker pomenijo večino vnosov log ter so najzanimivejši za forenzično raziskavo.

### 4.1 Identifikacija stavkov

Vsi vnosi v log blok so identificirani po prvem bajtu vsakega vnosa. Celoten list vseh vrst vnosov je na voljo v izvorni kodi programa. Vselej bomo za forenzično analizo izpisali pomembnejše vrste (tabela 5).

| Prvi bajt | Ime | Razlaga |
|---|---|---|
| 0x14 | mslog_undo_insert | Identificira stavek za manipulacijo z podatki. |
| 0x26 | mslog_comp_rec_insert | Stavek za nov vnos podatkov. |

**Tabela 5: Pomembnejši tipi stavkov.**

Za vsak stavek za manipulacijo podatkov, InnoDB ustvari vsaj en vnos ki je tipa *mslog_undo_insert*. Ta tip log vnosa poda identifikacijo tabele katera je spremenjena, kakšen stavek je manipuliran (*Insert, Update, Delete ...*), kot tudi dodatne informacije, katere so pogosto odvisne na specifično vrsto stavka. Primer je prikazan v tabeli 6

| Številka polja | Dolžina | Razlaga |
|---|---|---|
| 1 | 1 | Vrsta vnosa (vedno 0x14). |
| 2 | o | Tablespace id. |
| 3 | o | Page id. |
| 4 | 2 | Dolžina log vnosa. |
| 5 | 1 | Vrsta manipulacije. |
| ... | spremenljivka | Preostanek log vnosa, odvisno od vrste manipulacije. |

**Tabela 6: Struktura *mslog_undo_insert* tipa log vnosa.**

Najbolj pomembno polje je tisto, katero pove vrsto manipulacije. V naši analizi se bomo osredotočili na vrednosti katere so prikazane v tabeli 7.

| Vrsta manipulacije | Razlaga |
|---|---|
| 0x0B | Stavek *Insert*. |
| 0x1C | Stavek *Update*. |
| 0x0E | Označi za brisanje. |

**Tabela 7: Identifikacija stavkov na podlagi vrste manipulacije.**

Oblika vsakega *mslog_undo_insert* log vnosa, je zelo odvisna kakšen tip stavka predstavlja. Zato ne moremo predvideti neke glavne strukture za vnos, ampak je vsak ovrednoten drugače. V primeru *Update* in *Delete* stavka, je preostanek vnosa celoten stavek, medtem ko v primeru *Insert*, moramo slediti vnosu *mslog_comp_rec_insert*, ter potem *mslog_undo_insert*.

## 4.2 Rekonstrukcija Insert stavkov

V primeru Update ali Delete stavkov je večina potrebne informacije shranjena v *mlog_undo_insert*

| Tip vnosa | Ime | Tip vnosa | Ime |
|---|---|---|---|
| 0x01 | 8byte | 0x1F | multi_rec_end |
| 0x18 | undo_hdr_reuse | 0x14 | undo_insert |
| 0x02 | 2byte | 0x26 | comp_rec_insert |
| 0x02 | 2byte | 0x02 | 2byte |
| 0x02 | 2byte | | |

**Tabela 8: Vsi dnevniški vnosi ob Insert stavku**

Večina vnosov ni pomembnih za forenzično analizo opisano v tem članku, le vnos tipa 0x26 (*comp_rec_insert*) vsebuje različne podatke, ki jih lahko uporabimo pri rekonstrukciji v dnevnik zabeleženega Insert stavka. Identificiramo ga na podlagi vnosa tipa 0x14, po katerem sledi zapis tipa 0x26. Tabela 8 pojasnuje večino polj znotraj prej omenjega *comp_rec_insert* zapisa.

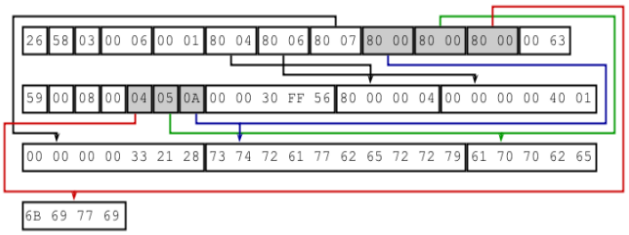
| Št. polja | Dolžina | Interpretacija |
|---|---|---|
| 1 | 1 | Vnos tipa (fiksna vrednost 0x26) |
| 2 | o | ID tabelnega prostora |
| 3 | o | ID strani |
| 4 | 2 | Število polj v vnosu ($n$) |
| 5 | 2 | Število unikatnih polj ($n_{unique}$) |
| 6 | 2 | Dolžina prvega unikatnega polja (primaryKey) |
| ... | 2 | Dolžina vnosov za unikatna polja |
| 7 | 2 | Dolžina zadnjega unikatnega polja |
| 8 | 2 | Dolžina ID-ja transakcije |
| 9 | 2 | Dolžina podatkovno-obnovitvenega kazalca |
| 10 | 2 | Dolžina prvega ne-unikatnegga stolpca |
| ... | ... | Dolžina definicij za ostale ne-unikatne stolpce |
| 11 | 2 | Dolžina zadnjega ne-unikatnega stolpca |
| 12 | 2 | Odmik |
| 13 | o | Dolžina konca segmenta |
| 14 | 1 | Informacije in statusi |
| 15 | o | Odmik izvora |
| 16 | 1 | Indeks neujemanja |
| 17 | o | Dolžina prvega dinamičnega polja (npr. varchar) |
| | ... | Dolžina vnosov za dinamična polja |
| 18 | o | Dolžina zadnjega dinamičnega polja |
| 19 | 5 | Neznano |
| 20 | 6* | Podatki za prvi unikatni stolpec |
| | ... | Podatki za unikatne stolpce |
| 21 | 7* | Podatki za zadnji unikatni stolpec |
| 22 | 8* | ID transakcije |
| 23 | 9* | Podatkovno-obnovitveni kazalec |
| 24 | 11* | Podatki za zadnji ne-unikatni stolpec |
| | ... | Podatki za ne-unikatne stolpce |
| 25 | 10* | Podatki za prvi ne-unikatni stolpec |
| ... | | |

**Tabela 9: polja v mlog_comp_rec_insert zapisu**

Struktura vnosa tipa *comp_rec_insert* je precej kompleksna. Začetna splošna polja (tip, ID tabele, ID strani) označujejo katera tabela iz zbirke podatkov je bila uporabljena. Podani sta dve pomembni polji: $n$ in $n_{unique}$, ki nosita informacijo o številu podatkovnih polj oziroma polj s primarnim ključem, ki jih lahko pričakujemo v tem zapisu. Število $n$ označuje vsoto števila polj v tabeli in števila internih sistemskih polj kot sta ID transakcije in podatkovno-obnovitveni kazalec.

Po definiciji za $n_{unique}$ sledi $2 \cdot n_{unique}$ bajtov, ki so rezervirani za definicijo dolžin teh unikatnih stolpcev (2 bajta za vsak stolpec). Nato so definirane še dolžine za ID transakcije in podatkovno-obnovitveni kazalec. Naslednjih $2 \cdot (n - n_{unique})$ bajtov vsebuje dolžine za stolpce, ki ne vsebujejo primarnih ključev. Te dolžine predstavljajo dolžino definirano v strukturi tabele in ne dejanske dolžine vstavljenih podatkov. V primeru statičnih podatkovnih tipov kot je naprimer *int*, podana dolžina predstavlja dejansko dolžino podatkov, kar pa ne drži za dinamične tipe, kjer podana

fiksna vrednost 0x8000. Dejanska dolžina podatkov je podana na kasneje v dnevniškem zapisu. Tretja slika prikazuje povezave med med definicijami dolžin in podatki.




**Slika 3: Povezave med polji v mlog_comp_res_insert dnevniškem zapisu.**

Bajti, ki sledijo tem definicijam vsebujejo razne informacije o zapisu, vendar niso potrebni za rekonstrukcijo.

Spodnja označena polja vsebujejo kompresirane informacije od dolžinah vseh stolpcev z dinamičnimi podatkovnimi tipi. Dolžine za omenjena polja pa so definirane s fiksnimi vrednostmi 0x8000. V naslednjih petih bajtih so shranjene razne informacije, ki pa niso potrebne v okviru predstavljene forenzične analize.

Vsebina vstavljenega zapisa je definirana po vrsti po stolpcih. Prvih $n_{unique}$ polj vsebuje podatke iz stolpcev s primarnimi ključi. Sledita dve polji z ID transakcije in podatkovno-obnovitvenim kazalcem, nato pa $n$-$n_{unique}$-2 polj, ki nosijo vsebino iz stolpcev, ki ne vsebujejo primarnih ključev. Kljub tem podatkom in znani dolžini polj, moramo za popolnoma pravilno interpretacijo poznati strukturo podatkovne tabele, ki jo dobimo iz analize .frm datotek.

## 4.3 Rekonstrukcija Update stavkov

Za rekonstrukcijo *Update* stavkov potrebujemo dva dnevniška vnosa. *mlog_undo_insert* vnos potrebujemo za rekonstrukcijo prepisanih podatkov, *mlog_comp_rec_insert* vnos pa potrebujemo za ugotavljanje vsebine novo-vstavljenih podatkov. Prvi vnos nam v primeru *Insert* stavka služi le prepoznavi vrste stavka, medtem ko se pri *Update* stavku v tem vnosu nahaja prepisana vsebina. V sklopu demonstracije bodo analizirani le taki *Update* stavki, ki ne spreminjajo vrednosti primarnega ključa. Posledica takih stavkov je večje število vnosov in sprememb v dnevniku.

### *4.3.1 Rekonstrukcija prepisanih podatkov*

InnoDB interno shranjuje prepisane podatke in tako varuje podatke pred izgubo in nudi možnost obnovitve podatkov in preklica stavkov. To nam daje dobro izhodišče za forenzično analizo take zbirke podatkov. Naslednja tabela prikazuje polja v z*mlog_undo_insert* zapisu.

| Št. polja | Dolžina | Interpretacija |
|---|---|---|
| 1 | 1 | Tip dnevniškega vnosa (fiksna vrednost: 0x94) |
| 2 | ∘ | ID tabelnega prostora |
| 3 | ∘ | ID strani |
| 4 | 2 | Dolžina vnosa |
| 5 | 1 | Vrsta ukaza (0x1c = posodobi obstoječ zapis) |
| 6 | 2 | ID tabele |
| 7 | 6 | ID zadnje transakcije na posodobljenem polju |
| 8 | ∘ | Podatkovno-obnovitveni kazalec zadnjih podatkov |
| 9 | 1 | Dolžina primarnega ključa |
| 10 | 9* | Posodobljen primarni ključ |
| ... | | |
| 11 | 1 | Število spremenjenega polj |
| 12 | 1 | ID prvega spremenjenega polja |
| 13 | 1 | Dolžina prvega spremenjenega polja |
| 14 | 13* | Prepisani podatki prveha spremnjenega polja |
| ... | | |

**Tabela 10: polja v mlog_undo_insert zapisu ob *Update* stavku**

Prvih pet polj je bilo pojasnjenih že v prejšnjem poglavju, zato bomo obravnavo tu izpustili. Dva bajta v šestem polju označujeta identifikator tabele, ki pa ga lahko najdemo tudi v definiciji tabel v .frm datoteki na naslovu 0x26. S pomočjo te informacije lahko ugotovimo ime tabele (.frm datoteka).

Naslednjih šest bajtov vsebuje ID zadnje transakcije pred klicem *Update* stavka. S pomočjo teh podatkov lahko rekonstruiramo celotno zgodovino sprememb v pravilnem vrstnem redu.

Sledeča polja vsebujejo informacije o posodobljenih primarnih poljih. Za vsak primarni ključ je v polju velikem en bajt informacija o dolžini nove vrednosti in v enem polju pa dejanska nova vrednost. Ker se to ponovi za vse primarne ključe, moramo poznati njihovo število. Naslednji bajt definira število ne-primarnih stolpcev, ki jih je *Update* stavek spremenil. Za vsak tak stolpec sledijo tri polja: ID spremenjenega polja, dolžina nove vrednosti in nova vrednost.

### *4.3.2 Rekonstrukcija izvedenega stavka*

InnoDB ustvari zapis v dnevnik *mlog_comp_rec_insert*, ki vsebuje informacije o novo-vstavljenih podatkih po *mlog_undo_insert* vnosu. Vstavljanje podatkov je tako zelo podobno v *Update* in *Insert* stavku. Struktura vnosa je enaka pri obeh stavkih. To pomeni, da lahko stavka ločimo le po analizi *mlog_undo_insert* vnosa, ki se nahaja pred samim *mlog_comp_rec_entry* vnosom.

## 4.4 Rekonstrukcija Delete stavkov

Rekonstrukcija *Delete* stavkov poteka na podoben način kot pri *Update* stavkih. Obstajata dve vrsti izbrisa. Fizičen izbris zapisa in izvedba zahtev, ki zapis označijo kot izbrisan. Fizičen izbris je nepredvidljiv in se lahko zgodi v poljubnem

času. V tej analizi se bomo posvetili drugi vrsti izbrisa, ki je forenzično bolj zanimiva.

Dnevniški zapisi stavkov, ki označijo zapis v tabeli kot izbrisan, so zelo kratki. Običajno se ustvarijo le štirje dnevniški vnosi, izmed katerih je za potrebe naše forenzične analize potreben le vnos *mlog_undo_insert*. Naslednja tabela prikazuje vnos ob izvedenem *Delete* stavku. Vidimo lahko, da je vnos zelo podoben tistemu iz *Update* stavka, le brez informacij o zbrisanih podatkih. Kljub temu jih lahko razberemo s pomočjo polja s številko 7, ki vsebuje ID zadnje transakcije na izbrisani datoteki.

Tudi tu moramo za pravilno interpretacijo podatkov poznati število primarnih ključev tabele. Brez tega podatka ne moremo izračunati števila spremenjenih primarnih ključev (polji 9 in 10).

| Št. polja | Dolžina | Interpretacija |
|---|---|---|
| 1 | 1 | Tip dnevniškega vnosa (fiksna vrednost: 0x94) |
| 2 | ∘ | ID tabelnega prostora |
| 3 | ∘ | ID strani |
| 4 | 2 | Dolžina vnosa |
| 5 | 1 | Vrsta ukaza (0x1e = izbriši zapis) |
| 6 | 2 | ID tabele |
| 7 | 6 | ID zadnje transakcije na izbrisanem polju |
| 8 | ∘ | Podatkovno-obnovitveni kazalec zadnjih podatkov |
| 9 | 1 | Dolžina primarnega ključa |
| 10 | 4 | Uporabljeni primarni ključ |
| ... | | |
| 11 | 3 | Neznano |
| 12 | 1 | Dolžina polja primarnega ključa |
| 13 | 4 | Primarni ključ izbrisanega polja |

**Tabela 11: Vnos *mlog_undo_insert* za *Delete* stavek**

# 5. DEMONSTRACIJA

V tem poglavju prikažemo dosedaj opisane tehnike rekonstrukcije na realnem primeru iz demonstracijske baze podatkov. Sledili smo postopkom, ki so jih raziskovalci uporabili v izvornem članku. Izpustili smo analizo .frm datotek in predpostavili, da poznamo strukturo tabele. Analizo smo izvedli na lokalnem Apache strežniku (XAMPP), na prenosnem računalniku z operacijskim sistemom Windows 7 (x64).

## 5.1 Demonstracijska tabela

Primere smo izvajali na naslednji tabeli.

**Listing 1: Demonstracijska tabela**

```
CREATE TABLE fruits (
primaryKey int(10) NOT NULL,
field1 varchar(255) NOT NULL,
field2 varchar (255) NOT NULL,
field3 varchar(255) NOT NULL,
PRIMARY KEY (primaryKey)
) ENGINE=InnoDB DEFAULT CHARSET= utf8;
```

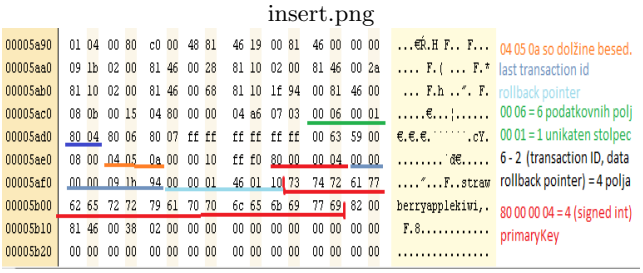
Uporabili smo dva preprosta podatkovna tipa (integer in varchar), da lahko ponazorimo rekonstrukcijo v primeru statičnega in dinamičnega podatkovnega tipa. InnoDB shranjuje polja tipa integer s fiksno dolžino 4 bajte. Polja z varchar imajo spremenljivo dolžino in lahko služijo kot splošen primer rekonstrukcije dinamičnih podatkovnih tipov.

## 5.2 Primer: Insert

Na naslednji sliki je prikazana vsebina dnevniškega zapisa *comp_rec_insert*. Izvedli smo naslednji *Insert* stavek.

**Listing 2: Insert stavek**

```
INSERT INTO fruits
(primaryKey, field1, field2, field3)
VALUES (4, strawberry, apple, kiwi);
```




**Slika 4: Analiza vnosa ob *Insert* stavku.**

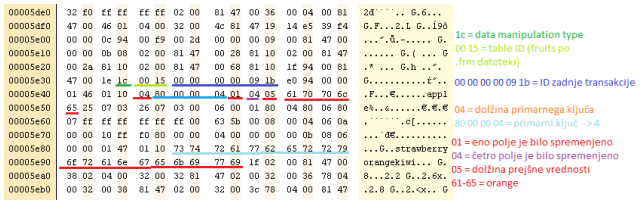
Vnos označen z vrednostjo 0x26 nam pove, da gre za *comp_rec_insert* (ni na sliki). Pomembna polja so na sliki označena z barvo. Prvo podčrtano polje nam označuje število podatkovnih polj (0x0006), drugo pa število unikatnih stolpcev (0x0001). Ker sta dve podatkovni polji rezervirani za ID transakcije in podatkovno-obnovitveni kazalec, lahko sklepamo, da so bili z *Insert* stavkom vstavljeni štirje stolpci in eden izmed teh vsebuje unikatno vrednost. Naslednje podčrtane vrednosti (0x04, 0x05 in 0x0a) označujejo dolžine vstavljenih besed. S pomočjo dolžin lahko določimo ostala polja v tem vnosu. Vrednost primarnega ključa nam predstavlja polje z vrednostjo 0x80000004 (4). Takoj zatem sledi ID transakcije, ki je v našem primeru 0x00000000091b94, Podatkovno-obnovitveni kazalec pa 0x000001460110. Nato pa se zvrstijo še vstavljeni podatki v ne-unikatne stolpce: 0x73747261776265727279 (strawberry, 3. stolpec), 0x6170706265 (apple, 2.stolpec), 0x6b697769 (kiwi, 1.stolpec).

Skupaj s predpostavljenim poznavanjem strukture tabele, lahko pravilno interpretiramo podatkovna polja in tako rekonstruiramo izvedeno *Insert* stavek: Primarni ključ ima vrednost 4, ne-unikatni stolpci imajo vrednosti (”kiwi”, ”apple”, štrawberry”) in iz tega izpeljemo stavek, ki je prikazan pod sliko Insert stavek.

## 5.3 Primer: Update

V tem delu bomo rekonstruirali prepisane podatke. Rekonstrukcija novo-vstavljenih podatkov bi potekala isto kot pri *Insert* stavku, zato bomo da del izpustili. Na izvlečku iz dnevnika so prikazana in podčrtana najbolj pomembna polja.

Stavek identificiramo na podlagi polja z vrednostjo 0x1c, ki označuje, da gre za *Update* stavek. Sledi ID tabele, s pomočjo katerega lahko iz .frm datoteke razberemo, da gre za




**Slika 5: Analiza vnosa ob *Update* stavku.**

tabelo z imenom ”fruits”. Sledita polji z ID-jem zadnje transakcije (0x00000000091b) in obnovitvenim kazalcem za zadnje podatke (0x0000014601). Polje z vrednostjo 0x04 označuje dolžino primarnega ključa, ki se nahaja takoj za tem poljem in ima že znano vrednost 0x80000004, ki predstavlja predznačeno celo število 4. To nam pove, da je bil zapis s primarnim ključem 4 spremenjen ob izvedbi *Update* stavka. Sledeča polja nam dajo naslednje informacije: koliko polj je bilo spremenjenih (0x01), katero polje je bilo spremenjeno (0x04), dolžina prejšne vrednosti (0x05), prejšna vrednost na spremenjenem polju (0x6170706c65 = ”orange”).

## 5.4 Primer: Delete

V tem primeru si pri interpretaciji pomagamo z izvlečkom iz *mlog_undo_insert* zapisa.




**Slika 6: Analiza vnosa ob *Delete* stavku.**

Iz polja z vrednostjo 0x94 razberemo, da gre za *undo_insert* vnos, iz polja z vrednostjo 0x0e, pa da gre za *Delete* stavek. Naslednje podčrtano polje prikazuje ID tabele (0x0015), sledi mu ID zadnje transakcije (0x000000000b08) in podatkovno-obnovitveni kazalec (0x000001470110). Nato preberemo dolžino primarnega ključa (0x04) in takoj zatem tudi dejanski primarni ključ izbrisane datoteke (0x800004). Nadaljna polja bi bila uporabna, če bi imeli več primarnih ključev. Iz opisanih polj lahko izpeljemo *Delete* stavek.

**Listing 3: Delete stavek**

```
DELETE FROM fruits
WHERE primaryKey=1;
```

# 6. ZAKLJUčEK

Uspešno smo ugotovili prejšnja stanja podatkovne baze z uporabo metode opisane v seminarski nalogi. Če bi bila omejitev velikosti Redo log datotek neskončna, bi s pomočjo le teh lahko kronološko ugotovili celotno zgodovino spreminjanja podatkovne baze.

Primer se osredotoča na preprosto tabelo in stavke, vendar bi lahko model analize razširili na ostale SQL poizvedbe in kompleksnejše tabele. S tehniko je mogoč globok vpogled v dogajanje v podatkovni bazi in posameznih tabelah. Prikažemo skoraj celotno časovnico dogajana v bazi.

# 7. VIRI

[1] M. Oracle. The innodb storage engine, May 2014.

[2] S. S. M. H. Peter Fruhwirt, Peter Kieseberg and E. Weippl. Innodb database forensics: Reconstructing data manipulation queries from redo logs. *Information Security Technical Report*, 17(4):227–238, May 2013.

# Forenzične sledi pri uporabi brskalnikov v zasebnem ali prenosnem načinu

## Predstavitev dveh forenzičnih raziskav iz področja zasebnega brskanja

Tine Mislej
tine.mislej@gmail.com

Blaž Meden
blaz.meden@gmail.com

## POVZETEK

V prispevku opiševa postopek in rezultate dveh izmed odmevnih analiz zasebnih načinov brskanja na najbolj uporabljenih brskalnikih[2, 6]. Predvsem se osredotočiva na količino informacij, ki jih zasebni načini brskanja pustijo za seboj na ciljnem računalniku. Opiševa tudi forenzične postopke, ki so bili uporabljeni med posamezno analizo. Hkrati pregledava uporabo prenosnih različic teh brskalnikov[1] ter rezultate primerjava z (največkrat uporabljenim) navadnim načinom brskanja.

## 1. UVOD

Internet je v zadnjih dveh desetletjih doživel skokovit razvoj in tako postal nepogrešljivo orodje za človeka v 21. stoletju. Uporabljamo ga za izvrševanje različnih vsakodnevnih dejavnosti, zato se je v zadnjem deseletju pojavila težnja k večji zasebnosti na internetu oz. spletu. Pri brskanju po spletu obstaja dva vidika zasebnosti, h katerima uporabnik navadno teži. Prvi je omogočanje zasebnega brskanja tako, da je identiteta uporabnika med brskanjem prikrita. Drugi vidik pa je omogočanje zasebnega brskanja brez puščanja sledi na napravi (računalniku). V tem članku bova predstavila širše področje zasebnosti pri spletnem brskanju in predstavila rezultate raziskave o zasebnem brskanju.

## 2. ZASEBNO BRSKANJE (PRIVATE BROWSING)

V raziskavi[2] so ugotovili, da ima zasebni način iskanja dva cilja pri ščitenju uporabnikove zasebnosti. Zasebni način ščiti uporabnikovo zasebnost na spletu (pred spletnim napadalcem),to pomeni, da uporabnikova identita ne sme biti razkrita na spletu. Zasebni način prav tako ščiti uporabnikovo zasebnost na računalniku (pred lokalnim napadalcem), kar pomeni, da podatki o brskanju ne smejo ostati na računalniku, ko uporabnik konča z brskanjem [6].

### 2.1 Lokalni napadalec

Lokalni napadalec je tisti, ki prevzame nadzor nad računalnikom. Varovanje pred takim napadalcem je možno le, če napadalec prevzame nadzor po tem, ko uporabnik zaključi z zasebnim brskanjem. Če napadalec prevzame nadzor pred ali med brskanjem v zasebnem načinu zaščita uporabnika ni več možna oz. zagotovljena. V teh primerih lahko napadalec namesti zlonamerno programsko opremo, ki beleži uporabnikovo aktivnost (npr. keylogger) in jo posreduje napadalcu [2].
V primeru varovanja pred lokalnim napadalcem se torej močno zanašamo na predpostavko, da uporabnikov računalnik ni bil ogrožen pred zaključkom brskanja v zasebnem načinu. Od zaključka brskanja naprej, pa se moramo zanesti na brskalnik, da bo vse sledi, ki so se ustvarile tekom brskanja izbrisal. Ta zahteva je vse prej kot preprosta, sploh pri dejstvu, da vse spremembe, ki so se zgodile med zasebnim brskanjem ne smejo biti izbrisane. V raziskavi [2] ločijo štiri kategorije trajnih sprememb:

1. Spremembe, ki jih izvede spletna stran brez vključevanja uporabnika (npr. dodajanje novega vnosa v zgodovino brskanja, nastavljanje piškotka, ...).
2. Spremembe, ki jih izvede spletna stran in zahtevajo uporabnikovo interakcijo (npr. dodajanje gesla v podatkovno bazo).
3. Spremembe, ki jih izvede uporabnik (npr. prenos datoteke).
4. Spremembe, ki niso specifične za določenega uporabnika (npr. posodabljanje spletnega brskalnika).

Vsi brskalniki ob zaključku zasbenega brskanja poskušajo izbrisati spremembe iz prve kategorije (1.). Spremembe v ostalih treh kategorijah pa niso nujno izbrisane in jih vsak brskalnik upošteva drugače [2].

Drugi problem zasebnega brskanja pa so kršitve zasebnosti na strani strežnika. Kljub brskanju v zasebnem načinu, lahko spletna stran objavlja različne podatke, ki jih uporabnih hoče ohraniti kot zasebne. Primer take spletne strani je npr. stran, ki javno objavi čas zadnje prijave posameznega uporabnika. Te kršitve so izven vpliva brskalnika, zato jih proizvajalci brskalnikov večinoma ignorirajo [2]. Kljub temu, pa obstaja nekaj možnih rešitev. V članku [2] avtorji predlagajo naslednje:

- Brskalniki bi lahko vsebovali seznam spletnih mest, ki jih med brskanjem v zasebnem načinu ne smejo obiskati.

- Spletne strani, ki so skladne z zasebnim brskanjem bi bile vidno označene. Tako se lahko uporabnik med zasebnim brskanjem takim stranem izogne.

- Spletne strani, bi izjavo o skladnosti z zasebnim brskanjem lahko posredovale v obliki P3P politik. Brskalnik bi onemogočil obisk spletne strani, ki je neskladna z zasebnim brskanjem.

## 2.2 Spletni napadalec

Brskalniki poskušajo zagotoviti zasebnost tudi v omrežju, torej izven računalnika in posledično dosega lokalnega napadalca. Spletni napadalec nima nadzora nad uporabnikovim računalnikom, vendar pa ima nadzor nad nekaterimi spletnimi mesti, ki jih uporabnik obišče. Spletni brskalniki poskušajo uresničiti naslednje cilje [2].

- Spletna stran ne more povezati uporabnika, ki brska v zasebnem načinu z uporabnikom, ki brska v privzetem načinu (t.j. javnem). Spletno mesto ne sme biti zmmožno povezati dve zasebni seji z istim uporabnikom. Večina brskalnikov to delno zagotavlja tako, da piškotki, ki so ustvarjeni v privzetem načinu niso dostopni v zasebnem načinu. Izjema je brskalnik Safari, pri katerem so vsi piškotki dostopni tudi v zasebnem načinu.

- Spletno mesto ne more povezati seje v zasebnem načinu z drugo sejo v zasebnem načinu istega uporabnika. Brskalniki to zagotavljajo tudi z brisanjem piškotkov ustvarjenih med določeno zasebno sejo.

- Spletno mesto ne sme biti zmožno ugotoviti, ali je uporabnik trenutno brska v zasebnem načinu. Tega cilja trenutno ne dosega noben brskalnik.

Tudi prva dva cilja je težko zagotoviti, saj lahko spletno mesto najde ustrezno povezavo z različnimi podatki, sploh pa IP naslovom.

# 3. ZASEBNO BRSKANJE V POPULARNIH SPLETNIH BRSKALNIKIH

Vsi večji spletni brskalniki imajo možnost brskanja v zaebnem načinu, vendar je v vsakem realizirana drugače.

## 3.1 Mozilla Firefox

V brskalniku Firefox, se način brskanja v zasebnem načinu imenuje **Private browsing**. Pri Mozilli so jasni, da ta način brskanja ne zagotavlja anonimnosti v spletu in zaščite pred keyloggerji, ipd. Brskalnik v tem načinu ne shranjuje piškotkov, zabeležk obiskanih strani (history), gesel, vnosov v poljih, predpomnjene vsebine, itd. Firefox v tem načinu tudi ne shranjuje podatkov o prenešenih datotekah, shrani pa datoteke, ki jih prenesemo in zabeležke strani (bookmark), ki jih ustvarimo v tem načinu. [5]

## 3.2 Google Chrome

Googlov Chrome ponuja t.i. **Incognito** način. Na podpornih straneh zagotavljajo, da se v tem načinu strani, ki jih obiščemo in datoteke, ki jih prenesemo ne beležijo nikjer. Ko zapremo brskalnik, se pobrišejo tudi vsi piškotki ustvarjeni tekom zasebne seje.
Tudi tu jasno poudarjajo, da Incognito način ne zagotavlja zasebnosti v spletu. Spletne strani, ki jih obiščemo si še vedno lahko beležijo podatke o našem obisku. Tudi datoteke prenešene tekom zasebne seje ostanejo na računalniku (se ne izbrišejo).

Incognito način je nekoliko drugačen le na napravah s sistemom IOS, zaradi sistemskih omejitev. Navadni in Incognito zavihki si tu delijo HTML5 predpomnilnik na strani odjemalca. To pomeni, da iste strani lahko dostopajo do podatkov v tem predpomnilniku tako v navadnem (privzetem), ko tudi Incognito načinu.[3]

## 3.3 Internet Explorer

V Internet Explorer je zasebni način brskanja na voljo pod imenom **InPrivate Browsing**. Internet Explorer nudi podobno zaščito kot ostali brskalniki. Zgodovina brskanja in prenešenih datotek se tekom brskanja v zasebnem načinu ne shranjuje, piškotki in začasne datoteke pa se ob zaključku brskanja pobrišejo.[4]

## 3.4 Apple Safari

Glede na Apple, Safari v zasebnem načinu ne zabeležuje obiskanih strani in podatkov o iskanjih, spletne strani ne morejo spreminjati informacij na našem računalniku. Po zaključku brskanja se tudi pobrišejo vsi piškotki [6].

Spletni brskalniki tudi omogočajo "Do not track"(DNT) možnost. DNT je dodatno polje v glavi HTTP paketa, ki od spletne aplikacije zahteva, da ne izvaja sledenja uporabniku. Do not track možnost ni obvezujoča, zato lahko spletne aplikacije to ignorirajo in nadaljujejo s sledenjem uporabniku.

## 3.5 Analiza sej različnih brskalnikov

Tukaj so predstavljeni rezultati analize brskalnikov iz [6]. V eksperimentu so bile z uporabo metod digitalne forenzike analizirane različne seje brskalnikov (tako v zasebnem kot tudi v prenosnem načinu).

#### Orodja in priprava

V analizi so bile uporabljene naslednje strojne komponente in orodja:

- Namizna delovna postaja (PC, 8GB RAM pomnilnika),
- 8 SATA diskov kapacitete 160GB,
- USB Flash pogon (8GB),
- USB zunanji disk (1TB WD Passport),
- SATA - USB pretvornik,

- USB Write Blocker (IDE/SATA),
- antistatične vrečke in antistatična zapestnica.

Poleg naštetih komponent je bila uporabljena naslednja programska oprema:

- Microsoft Windows 7 Professional (64),
- brskalniki Internet Explorer, Firefox, Safari, Chrome,
- orodje za virtualizacijo - VMware,
- orodje za preverjanje integritete datotek - DaemonFS,
- orodje za prepisovanje diskov - Disk Wipe,
- pregledovalniki zgodovine, predpomnilnika in piškotov - Nirsoft Internet Tools
- prenosna aplikacija - Launchpad, PortableApps,
- prenosni brskalniki - Firefox Portable, Chrome Portable, Opera Portable,
- orodje za ustvarjanje forenzičnih slik - FTK Imager, FTK Imager Lite (portable),
- orodje za analizo forenzičnih slik - AccessData FTK version 3.2 (Licensed)

Cilj celotne analize je bila izvedba standardiziranega testiranja različnih brskalnikov v nadzorovem okolju. Vsi postopki so bili izvedeni na enak način kot pri uradni forenzični preiskavi. Pri tem so bile posnete dokazne fotografije, ustvarjene so bile slike diskov, vsi postopki so bili dokumentirani, celotno dokazno gradivo pa je bilo shranjeno na varen način.

**Postopek priprave diskov**

Z uporabo orodja Disk Wipe so bili iz vseh diskov odstranjeni morebitni podatki. Disk Wipe omogoča več načinov prepisovanja podatkov. Prvi disk je bil analiziran po enkratnem prepisu z ničlami. V analizi so bili na disku še vedno najdeni podatki, zato je bilo brisanje opravljeno še enkrat - tokrat s trojnim prepisovanjem (DoD algoritem). Omenjen algoritem prepisovanja se je izkazal kot bolj učinkovit, zato je bil uporabljen na vseh diskih.

Po prepisovanju in brisanju podatkov je bila na vsakega od diskov naložen operacijski sistem Windows 7 Professional. Izbrana je bila 64-bitna različica, predvsem zato, da bi lahko kasneje bila analizirana večja količina RAM pomnilnika.

Na vsak disk je bil zatem naložen le en brskalnik, ki je bil prenešen iz zunanjega diska WD Passport. Naloženi so bili brskalniki Internet Explorer, Firefox, Chrome in Safari. Prenosne različice brskalnikov so bile testirane posebej. Vsak naložen brskalnik je bil konfiguriran, da se privzeto zažene v zasebnem načinu brskanja. Noben brskalnik ni imel prednaloženih razširitev ali drugih dodatnih komponent. Raziskava iz [2] je namreč pokazala, da naložene razširitve lahko predstavljajo potencialne varnostne luknje pri zasebnem brskanju. Vse konfiguracije so bile izvedene, preden je bil katerikoli od diskov priklopljen na omrežje.

**Tabela 1: Spremembe med normalnim brskanjem.**

| Brskalnik | Poglavitne spremembe |
|---|---|
| Internet Explorer 8.0 | Spremembe začasnih datotek (Content.IE, History.IE5, Cookies, Recovery, Custom Destinations, Index.dat) |
| Chrome 23.0.1271.95 | Spremembe direktorija Chrome\User Data (Safe Browsing Whitelist, Default\Cache, Current Session, Default\History, Default\Session Storage). |
| Firefox 17.0.1 | Spremembe direktorija Firefox\Profiles (Cache, jumpListCache, ...) in Win CustomDestinations. |
| Safari 5.1.7 | Spremembe direktorija AppleComputer\Safari (Cache, History, Webpage Previews, Cookies, WebpageIcons.db). |

**Tabela 2: Spremembe med zasebnim brskanjem.**

| Brskalnik | Poglavitne spremembe |
|---|---|
| Internet Explorer 8.0 | Izbrisani vsi relevantni podatki po zaprtju brskalnika. |
| Chrome 23.0.1271.95 | Spremembe v poljih Safe Browsing, Cookies, History; med brskanjem ni sprememb, ob koncu je datoteka chrome_shutdown_ms.txt zamenjana z novo časovno značko. |
| Firefox 17.0.1 | Spremembe v poljih Safe Browsing in Firefox\Profiles, med brskanjem ni sprememb. |
| Safari 5.1.7 | Ugotovljena sprememba le v datoteki NTuser.dat. |

**Postavljanje okolja in začetna analiza**

Po pripravi diskov je bilo postavljeno osnovno okolje za analizo, ki je vključevalo orodje za virtualizacijo VmWare in orodje za preverjanje integritete datotek DaemonFS. S pomočjo teh orodij, je bilo mogoče analizirati področja diskov, na katerih je prihajalo do sprememb med normalnim in zasebnim brskanjem ter med uporabo prenosnih brskalnikov. Orodje DaemonFS je bilo po zagonu nastavljeno na nadzor in beleženje vseh sprememb datotečnega sistema na posameznem lokalnem disku. Za tem je bil posamično pognan vsak od naloženih brskalnikov. Brskanje je potekalo v standardnih korakih in je zajemalo iskanje (člankov, slik, videov), obisk uporabniških računov (email in spletno bančništvo) ter tudi aktivnosti spletnega nakupovanja. Rezultati najdenih sprememb so prikazani v tabelah 1, 2 in 3.

**Zajem evidence zasebnega načina brskanja**

Vsak od diskov je bil ločeno priklopljen kot primarni disk. Od tega je bilo na prvih štirih diskih opravljeno testiranje zasebnih sej brskalnikov. Pri tem seja brskalnika predstavlja vse aktivnosti, ki so bile izvedene med zasebnim brskanjem. Aktivnosti so bile v veliki meri enake izvedenim aktivnostim med začetno analizo (normalnim brskanjem). Vključevale so iskanje člankov o vdorih v računalniške sisteme, iskanje slik in video materiala, brskanje po spornih kriminalnih straneh, poskus nakupa večjih količin streliva ter odtujene lastnine.

**Tabela 3: Spremembe pri prenosnih brskalnikih.**

| Brskalnik | Poglavitne spremembe |
|---|---|
| Opera Portable | Ustvarjene začasne datoteke, ki so ob koncu brskanja tudi izbrisane. |
| Chrome Portable | Spremembe v mapi GoogleChromePortable, Sys32\Winevt\Logs in Portable Chrome Cache |
| Firefox Portable | Spremembe v direktoriju Mozilla/Roaming, ustvarjene in izbrisane začasne datoteke v direktoriju Local AppData. |
| Safari Portable | Safari ni bil testiran, ker je prenosen le namestitveni paket. |

Pri tem sta bila uporabljena iskalnika Yahoo! in Google. Aktivnosti so bile izbrane na podlagi možnih realnih primerov, na podlagi katerih bi lahko bil vložen sum izvajanja kaznivega dejanja.

Po vsaki zaključeni seji brskanja so bili procesi brskalnika preverjeno zaustavljeni. S pomočjo orodja FTK Imager Lite (naloženega na USB pogonu) je bila na zunanji USB disk prenešena vsebina RAM pomnilnika. Vsebina tega pomnilnika je bila prenešena tudi pred izvedenim brskanjem, saj je bilo le na ta način možno pokazati, da v RAMu ni bilo ohranjenih podatkov iz prejšnjih sej. Pri pregledu vsebine RAMa je bila uporabljena programska oprema Nirsoft Internet Tools (pregledovalnik predpomnilnika, zgodovine in piškotov). Vsi pregledovalniki so ovrgli obstoj predhodnih podatkov ter tudi podatkov trenutne seje. Hkrati je bil iz diska prenešen tudi celoten register in vsebina začasne datoteke pagefile.sys. Zatem je bil računalnik izklopljen, trenutno aktiven disk pa je bil previdno odstranjen. Podrobni rezultati analize so (posebej, za vsak testiran brskalnik) prikazani v tabelah 4, 5, 6 in 7.

**Zajem evidence pri prenosnih brskalnikih**

Trije diski so bili uporabljeni pri analizi prenosnih brskalnikov. Brskalniki so bili tokrat nameščeni na USB pogonu. Uporabljene so bile prenosne različice brskalnikov Firefox, Chrome in Opera. USB pogon je bil konfiguriran s pomočjo programske opreme PortableApps, ki omogoča poganjanje različnih programov neposredno iz zunanjega pogona. Posamezen brskalnik je bil pognan ločeno (s svojim pripadajočim diskom), pri tem pa so bili koraki vsake seje brskanja enaki kot pri nameščenih brskalnikih. Ob koncu vsake seje, je bil disk odstranjen in shranjen v antistatično vrečko (tako kot pri analizi prvih štirih diskov). Pri rokovanju z elektronskimi komponentami je bila vedno uporabljena tudi antistatična zapestnica. Podrobni rezultati analize prenosnih različic brskalnikov so na voljo v tabelah 8, 9 in 10.

**Forenzična pridobitev evidence ter FTK analiza**

Zadnji izmed uporabljenih diskov je bil namenjen postavitvi forenzične delovne postaje. Nanj je bil nameščen operacijski sistem Windows 7 skupaj s programsko opremo Forensic Toolkit (FTK) 3.2. FTK je certificirano orodje za forenzično analizo digitalnih naprav. Posamezen disk je bil na postajo povezan preko write-block naprave, ki je preprečila pisanje operacijskega sistema na disk in s tem spreminjanje dokaznega gradiva ob njegovem priklopu na delovno postajo. Ohranjanje integritete digitalnega dokaznega gradiva je namreč najbolj pomemben faktor skupaj z dokazno verigo pri izvajanju forenzične preiskave. Z uporabo komponente FTK Imager je bila za vsak disk z dokaznim gradivom ustvarjena skompresirana slika v formatu .E01. Za vsako sliko je bilo izračunanih več izvlečkov z uporabo različnih zgoščevalnih funkcij. Generiranje vsake od slik je trajalo od 3 do 5 ur. V nadaljevanju je bila vsaka slika analizirana in klasificirana s FTK. Pri tem je ta analiza trajala od 6 do 72 ur. Trajanje je bilo odvisno tudi od naprednih parametrov orodja, ki so bili vključeni v analizi. Najdlje so se procesirale slike z nameščenimi brskalniki. Ko je orodje FTK dokončalo procesiranje, so bile v pridobljene podatke vložene še dodatne ure dela za organizacijo ter prebiranje podatkov relevantne vrednosti.

**Rezultati opravljene analize**

Rezultati kažejo, da tudi zasebni način brskanja ter uporaba prenosnih brskalnikov za seboj pusti uporabnikove sledi, vendar je količina teh podatkov različna od posameznega brskalnika. Pri tem določeni brskalniki za seboj pustijo dovoj sledi za vzpostavitev povezave do uporabnika. Od vseh analiziranih brskalnikov, je Internet Explorer za seboj pustil največ ostankov, vendar na neobičajnih lokacijah. Podobno velja tudi za druge brskalnike. Konkretno je bila pri brskalniku IE prazna datoteka Index.dat (ki hrani zgodovino), prav tako so bila prazna rudi polja TypedURLs v registru. Kljub temu, so bile uspešno obnovljene vse predpomnjene slike, zgodovina URL povezav ter tudi uporabniška imena s pripadajočimi računi. Video vsebine niso bile obnovljene. Večina podatkov je bila pridobljena iz praznega prostora (angl. free space) ter prostora na koncu particije (angl. slack space). Podobni rezultati so bili pri analizi slike pomnilnika, saj so bili iz nje obnovljeni podobni podatki. Oznake zasebnega brskanja ("InPrivate" in "Start InPrivate Browsing") so bile označene le na nekaterih mestih zgodovine ter URL povezav.

Iz ostalih brskalnikov je bilo nekoliko težje izluščiti ohranjene informacije o brskanju. Glede na rezultate je bila najbolj uspešna metoda izluščevanja podatkov iz slike RAM pomnilnika. Vendar se v realnosti zgodi, da forenzični preiskovalci te možnosti nimajo na voljo. Pri brskalniku Google Chrome je bilo najdenih kar nekaj indikatorjev ter časovnih značk, ki potrjujejo uporabo tega brskalnika. Za razliko od brskalnika IE pa tukaj ni bilo možno vzpostaviti povezave seje brskanja s končnim uporabnikom, saj ni bilo oprijemljivih sledi o uporabniških imenih in zgodovini brskanja. Enaki rezultati veljajo za brskalnik Mozilla Firefox. V obeh primerih pa so bili vsi začasno odprti in preneseni dokumenti uspešno obnovljeni. Informacija o obnovljenih dokumentih je pomembna za preiskavo, saj (skupaj s časovnimi značkami) lahko pojasnjuje manjkajočo zgodovino brskanja.

Applov brskalnik Safari se glede na skrivanje občutljivih informacij zasebnega brskanja nahaja na sredini področja, saj večino informacij uspešno obravnava kot zasebne podatke, po drugi strani pa še vedno na računalniku pusti določene sledi. Kot najbolj enostaven način pridobitve podatkov se je izkazal pregled baze "WebpageIcons". Omenjena baza namreč vsebuje podrobne vnose o obiskanih spletnih straneh skupaj z ostalimi podatki o brskanju. Tukaj je pomembno

izpostaviti, da se podatki o brskanju lahko nahajajo tudi na neobičajnih lokacijah, kar je pomembno tudi pri razlagi in posredovanju najdenega dokaznega gradiva sodnemu sistemu.

Pri analizi prenosnih brskalnikov se je izkazalo, da je (tako kot pri nameščenih brskalnikih) najlažji način pridobitve podatkov brskanja pregled slike RAM pomnilnika. Za razliko od RAMa, pa na diskih ni bilo mogoče najti enake količine informacij, kot pri nameščenih brskalnikih. Konkretno je brskalnik Google Chrome Portable pustil največ sledi na diskovnem pogonu. Obnovitev datotek iz diska je prikazala skoraj enako količino podatkov, kot če bi brskalnik bil dejansko naložen na sistemu. Obnovljeni podatki so vključevali predpomnjene slike, zgodovino in način brskanja ter tudi uporabniška imena skupaj z dodeljenimi računi. Vsi ti podatki so se nahajali na primarnem disku. Ob tem je pomembno omeniti, da je bila obnovitev izvedena brez prisotnega USB pogona, kjer se je Chrome Portable prvotno nahajal.

Prenosna različica brskalnika Opera, je razliko od Chroma za seboj pustila manjše količine sledi. Indikatorji brskanja so bili sicer najdeni, vendar so bili podatki o zgodovini brskanja in uporabljenih računih nedostopni in jih ni bilo mogoče restavrirati. Podobni rezultati veljajo za Firefox Portable, poleg tega pa so bili pri Firefoxu obnovljeni tudi podatki o uporabniku (uporabniška imena in email naslovi).

Pri klesanju slik iz RAMa, je bil majhen del slik prepoznaven, druge pridobljene slike pa so bile večinoma popačene. Ena izmed rešitev za identifikacijo popačenih slik, je bila primerjava z najdenimi slikami iz diska. Pri tem je bila uporabljena mehka zgoščevalna opcija (angl. fuzzy hash option) iz orodja FTK. Z vzpostavljeno povezavo med najdenimi slikami na disku ter slikami v RAMu bi lahko okrepili dokazno gradivo, ki potrjuje sporne aktivnosti uporabnika med brskanjem. Omenjeno primerjanje slik deluje tako, da skuša najti podobnosti izvlečkov posameznih slik. V nasprotju s pričakovanji se metoda ni izkazala kot zelo učinkovita. Uspešno je bilo le primerjanje nad določenimi ikonami, vendar nobena od njih ni bila v povezavi z uporabnikovimi aktivnostmi. Obstaja verjetnost, da bi metoda delovala bolje pri večji kapaciteti RAM pomnilnika.

# 4. BRSKANJE NA PRENOSNIH BRSKALNIKIH

V zadnjih letih je veliko brskalnikov postalo na voljo tudi v t.i. prenosni (ang. portable) obliki. Prenosne brskalnike lahko uporabimo na vsakem računalniku. Nameščeni so lahko na kakršnokoli prenosno napravo (največkrat USB ključ) iz katere se kasneje tudi zaženejo, zato ne zahtevajo namestitve na računalnik. Prenosni brskalniki so postali zelo priljubljeni, saj uporabniku omogočajo, da ima sebi prilagojen brskalnik vedno s seboj. V njem si uporabnik lahko ustvari svoje zabeležke (bookmarks), ki so posledično dostopne na vsakem računalniku. Ta vrsta brskalnikov je postala zelo priljubljena tudi zaradi prepričanja, da nudi večjo stopnjo zasebnosti. Ti spletni brskalniki naj bi na računalniku pustili precej manj ostankov brskanja kot njihove nameščene različice. To področje so raziskali tudi v raziskavi [1], ki jo bomo predstavili v nadaljevanju.

**Tabela 4: Internet Explorer 8.0 - InPrivate.**

| Ostanki | Ciljne lokacije |
|---|---|
| Indikatorji zasebnega brskanja | Memdump (slika RAMa);<br>Free/Slack Space ("Start InPrivate Browsing" - povezava z zgodovino URL povezav);<br>$I30 (. . . \Content.IE5- "inprivate[1]" - povezava s seznamom obstoječih *.jpeg datotek);<br>Pagefile; |
| Zgodovina brskanja | Memdump (slika RAMa); Free space;<br>File slack (Temporary Internet Folder, Roaming\...\Custom Destinations);<br>SysVol Info; $LogFile; $J;<br>AppData\...\IE\Recovery\Active; |
| Uporabniška imena oz. uporabniški računi | Memdump (slika RAMa); Freespace;<br>Temporary Internet Folder;<br>User\AppData. . . \IE\Recovery\Active |
| Slikovni material | Memdump (delna vsebina);<br>Free space (celotna vsebina);<br>File slack (celotna vsebina); |
| Video material | X |

**Tabela 5: Google Chrome 23.0.1271.95 - Incognito.**

| Ostanki | Ciljne lokacije |
|---|---|
| Indikatorji zasebnega brskanja | Memdump (slika RAMa);<br>Chrome\...\Installer\chrome.7z,<br>chrome.dll (ujemanje časovnih znamk);<br>$I30 (safebrowsing časovna znamka);<br>AppData\Local\Google\Chrome\User Data\chrome_shutdown_ms.txt (osvežena časovna znamka);<br>AppData\Local\Google\Chrome\User Data\Default\ExtensionState*.log (declarative_rules.incognito.-declaritiveWebRequest - časovna znamka se ujema z začetkom seje);<br>∼\SysVol Information (sledi novega Incognito okna, časovna znamka);<br>AppData\Roaming\Microsoft\Windows\Recent\CustomDestinations (sledi novega Incognito okna, časovna znamka);<br>Chrome\UserData\Safebrowsing-cookies.db (spremenjena časovna znamka); |
| Zgodovina brskanja | Memdump (slika RAMa);<br>SysVol Info (ujemanje časovnih znamk);<br>Pagefile.sys (sledi prenešenih datotek); |
| Uporabniška imena | X |
| Slikovni material | Klesanje vsebine iz slike RAMa (v večini delno rekonstruirane slike); |
| Video material | X |

**Tabela 6: Mozilla Firefox 17.0.1 - Private Browsing.**

| Ostanki | Ciljne lokacije |
|---|---|
| Indikatorji zasebnega brskanja | Memdump (slika RAMa, najden indikator uporabljenega načina brskanja); SysVolume Information (Enter Private Browsing in Window's User - ujemanje časovnih znamk datotek); |
| Zgodovina brskanja | Memdump (slika RAMa); Free space; AppData\...\Temp; Win\Prefetch (ugotovljen prenos začasne .rtf datoteke); AppData\...\Firefox\Profiles (blacklist.xml - ujemanje časovnih znamk); Firefox\Profiles\ (ujemanje časovnih znamk datotek); |
| Uporabniška imena | X |
| Slikovni material | Klesanje vsebine iz slike RAMa (v večini delno rekonstruirane slike); |
| Video material | X |

**Tabela 7: Apple Safari 5.1.7 - Private Browsing.**

| Ostanki | Ciljne lokacije |
|---|---|
| Indikatorji zasebnega brskanja | Memdump; ∼\SysVol Information (com.apple.Safari.PrivateBrowsing - prisotna časovna zmanka); |
| Zgodovina brskanja | Memdump (slika RAMa); Free/Slack Space (sledi zgodovine URL povezav); AppData\Local\AppleComp\Safari \WebpageIcons.db → tables; AppData\Local\AppleComp\Safari (spremembe časovnih znamk posameznih podatkovnih baz v datotekah) AppData\Local\AppleComp\Safari & Preferences (spremembe časovnih znamk pri nekaterih *.plist datotekah) Pagefile (zgodovina URL povezav in spremembe v časovnih znamkah); |
| Uporabniška imena | X |
| Slikovni material | Klesanje vsebine iz slike RAMa (v večini delno rekonstruirane slike); |
| Video material | X |

**Tabela 8: Google Chrome Portable.**

| Ostanki | Ciljne lokacije |
|---|---|
| Indikatorji zasebnega brskanja | Alociran in nealociran prostor NTFS particije; Prefetch; Pagefile; Memdump (slika RAMa); $Logfile; Users\AppData\Roaming\Microsoft \Windows\Recent\CustomDestinations; ∼\System Volume Information; AppData\Local\Temp; AppData\LocLow\Mic\CryptnetUrlCache; Win\AppCompat\Prog\RecentFileCache; Win\Mic.NET\Framework\log (nezaseden prostor oz. fileslack); Win\Sys32\LogFiles\WUDF (nezaseden prostor oz. fileslack); |
| Zgodovina brskanja | Alociran in nealociran prostor NTFS particije; Memdump (slika RAMa); Orphan Directory; Pagefile; Users\AppData\Roaming\Microsoft \Windows\Recent\CustomDestinations (izklesan .lnk) |
| Uporabniška imena | Direktorij [Orphan] in nealociran ter prost (Slack) prostor NTFS particije |
| Slikovni material | Klesanje NTFS nealociranega prostora in direktorija Orphan |
| Video material | X |

**Tabela 9: Opera Portable.**

| Ostanki | Ciljne lokacije |
|---|---|
| Indikatorji zasebnega brskanja | NTFS Allocated and Unallocated Space; Pagefile; Memdump (slika RAMa); $LogFile; ∼\System Volume Information; NTUSER.DAT; AppData\Local\Mic\Win\UsrClass.dat; Users\AppData\Roaming\Microsoft\ Windows\Recent\CustomDestinations (izklesan .lnk); Win\Prefetch; Win\Sys32\LogFiles\SQM\SQMLogger |
| Zgodovina brskanja | Memdump; AppData\Roaming\Mic\Win\Rec \CustomDestinations (izklesane .lnk datoteke z ohranjenimi zadnjimi časi dostopa); |
| Uporabniška imena | X |
| Slikovni material | Klesanje vsebine iz slike RAMa (v večini delno rekonstruirane slike s težko razločljivo vsebino) |
| Video material | X |

**Tabela 10: Mozilla Firefox Portable.**

| Ostanki | Ciljne lokacije |
|---|---|
| Indikatorji zasebnega brskanja | Memdump (slika RAMa); SysVol Information file - časovna znamka (v Firefox Portable appinfo); |
| Zgodovina brskanja | Memdump; SysVol Information (samo email naslov); |
| Uporabniška imena | Memdump; SysVol Information (zgodovina o aktivnostih email računa) |
| Slikovni material | Klesanje vsebine iz slike RAMa (v večini delno rekonstruirane slike s težko razločljivo vsebino) |
| Video material | X |

## 4.1 Raziskava

V tej raziskavi [1] so raziskovalci želeli ugotoviti ali prenosni brskalniki pustijo podobne forenzične artefakte na računlaniku, kot nameščeni brskalniki. Raziskavo so izvedli s testiranjem Google Chroma v naslednjih situacijah:

- Brskanje s prenosnim brskalnikom v privzetem načinu.
- Brskanje s prenosnim brskalnikom v zasebnem načinu.
- Brskanje z nameščenim brskalnikom v privzetem načinu.

V vseh treh načinih so izvedli podobna spletna brskanja in primerjali rezultate. Prvi cilj eksperimenta je bil določiti prednosti v smislu zasebnosti, če uporabljamo prenosni brskalnik, drugi pa določiti prednosti brskanja na prenosnem brskalniku v zasebnem načinu.

V nadaljevanju bomo v grobem opisali potek raziskave.

Za vsakega izmed treh poskusov so uporabili nov prenosni računalnik s sveže nameščenim operacijskim sistemom. Potek posameznega eksperimenta je v grobem potekal v naslednjih korakih.

**Priprava**
Na prenosni računalnik je bila nameščena sveža različica operacijskega sistema Windows XP SP3. V eksperimentu, kjer so spletno brskanje izvedli na nameščeni različici brskalnika, so po namestitvi operacijskega sistema namestili še brskalnik. V preostalih eksperimentih so na nov, prazen USB ključ namestili prenosno različico spletnega brskalnika.

**Spletno brskanje**
V spletnem brskalniku so izvedli krajše brskanje. V vseh treh eksperimentih so simulirali isto brskanje, v katerem so izvedli slednje:

- Ogled videoposnetkov na YouTube.
- Iskanje slik na Google Image Search.
- Iskanje na eBayu.

**Tabela 11: Tabela obiskani spletnih mest in na njih izvedenih dejanj.**

| Spletna stran | Aktivnosti | Ključne besede |
|---|---|---|
| www.youtube.com | Iskanje niza "Trikke", Iskanje niza "Parkour", Ogled nekaj zadetkov iskanja | "Trikke", "Parkour" |
| images.google.com | Iskanje "Dodge Viper", Iskanje "Mini Cooper", Ogled nekaj zadetkov iskanja | "Dodge Viper", "Mini Cooper" |
| www.ebay.com | Iskanje "Black-Berry Playbook", Iskanje "Galaxy i9100", Brksanje skozi seznam stvari | "Playbook", "Galaxy" |

V spodnji tabeli so prikazane obiskane spletne strani ter aktivnosti, ki so bile izvedene na posamezni strani.

Po zaključku brskanja so računalnik ugasnili ter izklopili napajanje.

**Pridobitev**

V tem koraku so iz ugasnjenega računalnika vzeli trdi disk in USB ključ ter ju priklopili v t.i. write-blocker (naprava, ki onemogoča kakršnokoli pisanje na disk). Po priklopu so z orodjem FTK Imager ustvarili sliki diska in ključa ter ju pretvorili v format dd raw za analizo.

**Analiza**

Analiza je bila izvedena z orodjem FTK. Vsaka slika je bila dodana v nov primer (case). Z FTK so nas sliko izvedli iskanje po ključnih besedah navedenih v tretjem stolpcu tabele 4.1.

**Rezultati**

Testirali so Google Chrome v privzetem in Incognito načinu. Ključne besede so našli na vseh treh pridobljenih slikah diskov. Velja omeniti, da se je količina in lokacija zadetkov iskanja razlikovala med slikami, vendar v majhnem obsegu.

Čeprav je iskanje po ključnih besedah največ zadetkov našlo na disku, kjer je bil brskalnik nameščen, je tudi iskanje na ostalih diskih našlo veliko sledov, tudi na disku, kjer je bilo brskanje izvedeno v zasebnem načinu.

Spodaj so naštete lokacije, kjer so bile najdene različne sledi.

- **Prenosna seja, HDD**
  \Documents and Settings\Administrator\Local Settings\Temp\GoogleChromePortable
- **Prenosna seja, USB**
  \GoogleChromePortable\Data\profile\Default
- **Prenosna zasebna seja, HDD**
  pagefile.sys

- **Prenosna zasebna seja, USB**

- **Navadna seja, HDD**
  \Documents and Settings\Administrator\Local Settings\ Application Data\Google\Chrome\User Data

Rezultati kažejo, da tudi uporaba prenosnih brskalnikov (tudi v zasebnem načinu), pusti forenzične sledi na uporabljenem računalniku.Tabela 4.1 prikazuje rezultate iskanja ključnih besed na vseh treh slikah. V privzetem načinu delovanja ni večjih razlik med nameščeno in prenosno različico brskalnika. Obe verziji pustita dovolj sledi na trdem disku, da lahko forenzični preiskovalec rekonstruira sejo. Prenosni brskalnik v Incognito načinu ne pusti nobenih sledi na disku, razen v datoteki pagefile.sys. Ta se uporablja za zapisovanje zamenjanih strani iz pomnilnika.Ko sistemu zmankuje RAM pomnilnika, operacijski sistem Windows premakne najmanj uporabljene strani pomnilnika v datoteko pagefile.sys z namenom sproščanja pomnilnika. To pomeni, da je pridobivanje dokazov iz te datoteke zelo težko, saj je datoteka velikokrat prepisana. Rezultati kažejo, da je odkrivanje sledi v tem primeru možno le, če se forenzična preiskava izvrši kmalu po zaključku brskanja.

| Ključna beseda | Nameščena verzija, HDD | Prenosna verzija, HDD | Prenosna verzija, USB | Prenosna Incognito verzija, HDD | Prenosna Incognito verzija, USB |
|---|---|---|---|---|---|
| Trikke | Da | Da | Da | Da | Da |
| Parkour | Da | Da | Da | Da | Da |
| Dodge Viper | Ne | Ne | Da | Da | Da |
| Mini Cooper | Da | Da | Da | Da | Da |
| Playbook | Da | Da | Da | Da | Da |
| Galaxy | Da | Da | Da | Da | Da |

**Tabela 12: Tabela zadetkov (iskanje ključnih besed) na posameznih slikah.[1]**

Rezultati raziskave kažejo, da nobena verzija spletnega brskalnika Google Chrome ne zagotavlja višje stopnje zasebnosti. Uporabniki prenosnega spletnega brskalnika katerih namen je prikrivanje spletnih aktivnosti se morajo zavedati, da bo tudi uporaba prenosnega brskalnika pustila sledi na gostiteljevem računalniku.

## 4.2 U3

Pred leti sta podjetji Microsoft in SanDisk sodelovali na projektu U3 Technology. Podobno kot pri ostalih prenosnih različicah programov je bila tudi tu ideja uporabnikom omogočiti prenašanja osebnih datotek in prilagojenih spletnih brskalnikov [6]. U3 pogoni so imeli prednameščen U3 Launchpad. U3 Launchpad, ki je izgledal podobno kot Start meni operacijskega sistema WIndows je nadzoroval nameščanje programa.Na U3 USB pogonu sta bili dve particiji, ena za shranjevanje podatkov (mass storage), druga pa je bila navidezen CD-ROM [6]. Navidezni CD-ROM je bila dejansko .iso datoteka, ki se je sistemu 'predstavila' kot fizični pogon. Na ta način je so omogočili 'autoplay' funkcijo na starejših sistemih. U3 aplikacije naj ne bi puščale za seboj nobenih sledi, vendar temu ni bilo tako. Glede na raziskavo [7] so U3 naprave na gostiteljevi napravi ustvarile nov direktorij in beležile uporabnikove aktivnosti. Ko je bila U3 naprava odstranjena se je zagnal program, ki je ustvarjene datoteke pobrisal. S pregledom t.i. prefetch datotek v sistemu Windows, so raziskovalci lahko ugotovili, katere U3 aplikacije so bile v uporabi.
Splošno gledano so U3 prenosne naprave uporabniku nudile večji občutek zasebnosti in možnosti prilagoditve. SanDisk je leta 2009 prenehal nuditi podporo U3 tehnologij. Med verjetnimi razlogi so bile tudi razne kritike kot npr.:

- Obnašanje podobno škodoželjnim programom (malware).
- Več programov v prenosni obliki.
- U3 naprave so izbrisale nekaj podatkov o uporabniku vendar ne vseh.
- Zaprta platforma.
- Nekompatibilnost z nekaterimi vgrajenimi sistemi.

# 5. ZAKLJUčEK

Forenzični analizi iz obeh prispevkov[1, 6] kažeta na to, da uporabnikova zasebnost nikoli ni zares skrita. Kot je bilo prikazano, je s pomočjo ustreznih orodij mogoče odkriti identifikatorje brskanja, ter v nekaterih primerih tudi podrobnejše podatke o seji uporabnika. To velja tako za brskanje v zasebnem načinu kot tudi pri uporabi prenosnih različic pribljubljenih brskalnikov. Za konec naj dodava, da dodatna previdnost pri brskanju po svetovnem spletu ni odveč. Hkrati opozarjava tudi na to, da zasebni način brskanja ne ponuja popolne zaščite uporabniških aktivnosti pred nepovabljenimi očmi.

## Literatura


[1] T. A. I. Andrew Marrington, Ibrahim Baggili and A. A. Kaf. Portable web browser forensics: A forensic examination of the privacy benefits of portable web browsers. *Computer Systems and Industrial Informatics (ICCSII), 2012 International Conference on*, December 2012.

[2] D. B. Gaurav Aggarwal, Elie Bursztein and C. Jackson. An analysis of private browsing modes in modern browsers. *In Proc. Of 19th Usenix Security Symposium*, August 2010.

[3] Google. Incognito mode (browse in private).

[4] Microsoft.com. Inprivate browsing.

[5] Mozilla.org. Private browsing - browse the web without saving information about the sites you visit.

[6] D. J. Ohana and N. Shashidhar. Do private and portable web browsers leave incriminating evidence? *IEEE CS Security and Privacy Workshops (SPW)*, December 2013.

[7] R. Tank and P. A. Williams. The impact of u3 devices on forensic analysis. *Proceedings of the 6th Australian Digital Forensics Conference, Edith Cowan University, Perth Western Australia*, December 2008.

# Android anti-forensics through a local paradigm

[Digitalna Forenzika - Seminarska, 1 inačica]

Blaž Jeršan
63090028
Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
Ljubljana, Slovenija
blaz.jersan@gmail.com

Zlatko Hrnčić
63090028
Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
Ljubljana, Slovenija
zhrncic@gmail.com

## POVZETEK

Pametni telefoni danes hranijo vse več informacij. Zato so lahko zelo koristni v forenzični analizi. Na kateri način lahko skrijemo in/ali uničimo dokaze znotraj telefona?

## Ključne besede

Android, Datotecni sistem, YAFFS, Forenzika, Anti-forenzika

## 1. UVOD

Preden začnemo je potrebno omeniti, da je članek že nekoliko zastarel, zato so lahko na določenih mestih nekoliko zastarele ali netočne informacije. Tekom pisanja, sva poizkusila preverjati ali so informacije točne in v primeru napak popraviti vsebino, vendar je možno da sva kaj spregledala.

Mobilne naprave, še posebej mobilni telefoni, hitro postajajo najbolj razširjena tehnologija na svetu. Število naročnikov je naraslo iz 2.6 milijard v letu 2008 na 6.8 milijard v letu 2013 ta trend pa se trenutno po svetu še vedno nadaljuje. Prav tako postajajo mobilni telefoni vedno bolj napredni in imajo vedno več funkcionalnosti. Za forenziko je najbolj zanimivo dejstvo, da imajo mobilni telefoni shranjenih vedno več osebnih podatkov (e-maili, bančni računi, zgodovina iskanja, etc.), ki jih je potrebno dobro varovati. Ameriški nacionalni inštitut za standarde in tehnologijo pozna tri različne klasifikacije mobilnih telefonov. Osnovni (basic), napredni (advanced) in pametni (smart) telefoni.

Za forenziko so torej najbolj zanimivi pametni telefoni, saj vsebujejo največ osebnih podatkov o uporabniku. Zaradi tega, se je začela uveljavljati anti-forenzika tudi pri mobilnih telefonih. Tekom seminarske naloge bova poizkusila predstaviti nekaj generalnih prijemov anti-forenzike in pogledati katere od teh podpira Android operacijski sistem.

| | Basic | Advanced | Smart |
|---|---|---|---|
| **Processor** | Limited Speed | Improved Speed | Superior Speed |
| **Memory** | Limited Capacity | Improved Capacity | Superior Capacity, Built-in Hard Drive Possibility |
| **Display** | Grayscale | Color | Large size, 16-bit Color (65,536 colors) or Higher |
| **Card Slots** | None | MiniSD or MMCmobile | MiniSDIO or MMCmobile |
| **Camera** | None | Still | Still, Video |
| **Text Input** | Numeric Keypad | Numeric Keypad, Soft Keyboard | Touch Screen, Handwriting Recognition, Built-in QWERTY-style Keyboard |
| **Cell Interface** | Voice and Limited Data | Voice and High Speed Data | Voice and Very High Speed Data |
| **Wireless** | IrDA | IrDA, Bluetooth | IrDA, Bluetooth, WiFi |
| **Battery** | Fixed, Rechargeable Lithium Ion Polymer | Removable, Rechargeable Lithium Ion Polymer | Removable, Rechargeable Lithium Ion |

**Slika 1: Klasifikacije mobilnih telefonov.**

## 2. ANTI-FORENZIKA

### 2.1 Definicija anti-forenzike

Anti-forenzika je relativno mlada veja znanosti, posebej kar se tiče mobilnih naprav. Za razliko od tipičnega okolja forenzike, imamo pri mobilnih napravah še nekaj dodatnih težav oz. omejitev na katere je potrebno biti pozoren in za katere v nekaterih primerih še ni rešitve, zato se tehnike anti-forenzike še spreminjajo in oblikujejo in s tem tudi sama definicija anti-forenzike.

Za nas naj velja da je anti-forenzika: ”Vsak poizkus ogrožanja dostopnosti ali uporabnosti dokazov v forenzičnem procesu.” To pa je možno doseči na več načinov. Lahko prestrežemo in ustavimo ustvarjanje dokazov, skrivamo dokaze, manipuliramo z dokazi ali jih poizkusimo zbrisati. Z razumevanjem metod anti-forenzike lahko izboljšamo obstoječa forenzična orodja in pristope.

### 2.2 Tipi anti-forenzike

Možne oblike anti-forenzike smo na hitro našteli že pri prejšni točki, sedaj pa si jih poglejmo nekoliko podrobneje.

#### *2.2.1 Uničevanje dokazov*

Pri uničevanju dokazov želimo doseči, da so le-ti neuporabni med forenzično preiskavo. Pozorni moramo biti na to, da lahko orodja, ki jih pri tem uporabljamo prav tako puščajo dokaze o lastnem delovanju.

### 2.2.2 Skrivanje dokazov

Sem štejejo vsi ukrepi s katerimi zmanjšamo ali izničimo vidnost dokazov med forenzično preiskavo. Ta tehnika velikokrat temelji na poznavanju orodij, ki se uporabljajo za zbiranje dokazov na napravah. Prav tako kot pri uničevanju, moramo biti pozorni na dokaze, ki jih za seboj pušča orodje.

### 2.2.3 Uničevanje izvorov dokazov

Pri tem pristopu se fokusiramo na izvore dokazov (npr. logerji, zgodovina iskanja, etc.) in želimo s tem uničiti dokaze še preden se te ustvarijo. Torej gre za preventivni ukrep.

### 2.2.4 Ponarejanje dokazov

Namesto, da uničimo vse dokaze, kar je prav tako lahko sumljivo, jih raje ponaredimo. S tem lahko preusmerimo oz. zavedemo forenzično preiskavo.

## 2.3 Anti-forenzika mobilnih naprav

Definicije, ki smo jih obdelali pri prejšnjih dveh točkah se lahko uporabi za katerikoli tip naprave. Fokus te seminarske naloge so mobilne naprave, zato si bomo sedaj pogledali, kje pride v upoštev dosedanje znanje ko govorimo o mobilnih napravah.

Kot smo že prej povedali, so mobilne naprave v zadnjih letih postale pomemben predmet forenzičnih preiskav zaradi vedno bolj bogatih osebnih informacij, ki jih vsebujejo. Prav tako smo omenili, da dosedanja orodja za forenzično preiskavo pri mobilnih napravah naletijo na več preprek in zato niso primerna. Glavni problem mobilnih naprav je, da ne omogočajo direktnega dostopa do notranjega pomnilnika. Odstranljive pomnilnike (SD kartica, sim kartica, etc.) lahko izoliramo in jih pregledamo na klasičen način, notranjega pomnilnika pa ne. Za ta problem že obstaja več komercialnih orodji (Paraben, Micro systemation, etc.), ki so tudi uporabljena v praksi, vendar pa ne zagotavljajo dobrih rezultatov preiskave.

Zaradi težkega dostopa je notranji pomnilnik popoln kandidat za uporabo prej omenjenih anti-forenzičnih tehnik, katerih specifično delovanje si bomo bolj podrobno pogledali kasneje.

# 3. ANDROID OPERACIJSKI SISTEM

Android je operacijski sistem namenjen mobilnim napravam, ki ga je razvil Google. Vsebuje operacijski sistem, middleware in skupek aplikacij. Preden se posvetimo Androidovim anti-forenzijskim tehnikam si na kratko poglejmo kako deluje. Članek tu napoveduje, da bo Android leta 2012 obsegal 15% tržnega deleža. Ta napoved je malce prenizka, saj je Android leta 2012 obsegal že skoraj 50% tržnega deleža. Torej je njegovo razumevanje še bolj pomembno kot predpostavlja članek.

## 3.1 Arhitektura Android OS

Android je sestavljen iz petih glavnih komponent.

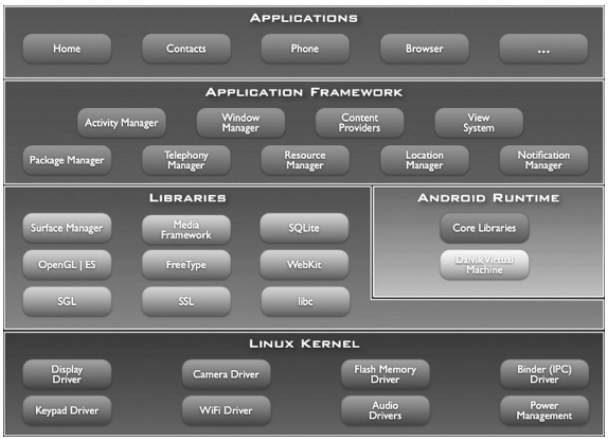
- Aplikacije: Skupek aplikacij, ki jih ponuja že osnovni paket programske opreme (npr. e-mail client, upravljanje z sms-i, brskalnik, upravljalec kontaktov, etc.), ki so spisani v Java programskem jeziku.
- Aplikacijski framework: Android ponuja bogato razvijalsko okolje, ki lahko izkorišča praktično vse funkcionalnosti telefona skozi različne programske vmesnike (API-ji), ta framework prav tako vsebuje različne varnostne omejitve.
- Knjižnjice: Različne knjižnjice za lažje delo z specifičnimi tipi podatkov (npr. 3D knjižnjice, knjižnjice za delo z zvokom, etc.), ki jih uporabljajo različne komponente Aplikacijskega frameworka, z njimi pa si lahko pomagajo tudi razvijalci.
- Android pogon (runtime): Sestavljen je iz skupka knjižnjic, ki so ključne za osnovno delovanje naprave in Dalvik virtualnega stroja (Dalvik Virtual Machine). Vsaka zagnana aplikacija uporablja oz. zažene svojo lastno instanco Dalvik-a in se izvaja v svojem lastnem procesu (ključna lastnost, ki jo izkorišča Androidova varnostna arhitektura).
- Linux jedro (kernel): Pod vsem tem leži jedro, ki uporablja Linux. Linux podpira vse ključne storitve kot npr. upravljanje s spominom, upravljanje s procesi, upravljanje z mrežo, gonilniki in varnost.

### 3.1.1 Android-ov datotečni sistem

Android privzeto uporablja YAFFS2 datotečni sistem (Yet Another Flash File System), ki je specifično narejen za uporabo z NAND flash čipi. Danes praktično vsi pametni mobilni telefoni uporabljajo NAND strukturo spomina (večja gostota, hitrost, nižja cena, etc.). YAFFS obstaja v dveh verzijah.

- YAFFS1: 512 byte-ov na stran (zastarel)
- YAFFS2: 2048 byte-ov na stran

YAFFS2 je prav tako napredoval na vseh ostalih področjih (hitrost, učinkovitost, zanesljivost, etc.).




**Slika 2: Arhitektura Android OS.**

## 3.2 Android-ova varnostna arhitektura

Kot smo že prej omenili, je Android več-procesna platforma, ki temelji in uporablja standardne Linux-ove prijeme za upravljanje z uporabniki in procesi. Prav tako je za večino varnostni med aplikacijami poskrbljeno na procesni ravni z uporabo standardnih Linux-ovih objektov. Poleg tega je za potrebe večkratne uporabe komponent in rezervacij storitev med različnimi aplikacijami realiziran še mehanizem dovoljenj.

### *3.2.1 Aplikacije in sandbox-i*

Android privzeto ne dovoli nobeni aplikaciji vplivati oz. izvajati operacije katerih namen je škodovati katerikoli drugi aplikaciji, operacijskemu sistemu ali uporanbiku samem. Zaradi tega so aplikacijam onemogočene kakršnekoli operacije na uporabnikovih privatnih podatkih (npr. sms-i, kontakti, etc.), dostopati do omrežij, spreminjati stanja naprave ali dostopati do datotek drugih aplikacij. To android doseže tako, da vsako zagnano aplikacijo poveže z zavarovanim sandboxom, ki ne more dostopati drugih aplikacij, razen v primeru, da ima izrecno dovoljenje za dostop do funkcionalnosti, ki jih sam sandbox ne ponuja. Dovoljenja so statična, se določijo ob inštalaciji aplikacije ter se kasneje ne morejo spreminjati. Vsaka aplikacija mora biti podpisana s certifikatom, katerega ustvari razvijalec, če želi dostopati do občutljivih podatkov oz. naprednih funkcij naprave.

### *3.2.2 Uporabniški ID-ji in dovoljenja*

Kot smo že prej omenili, Android tretira vsako aplikacijo kot ločenega Linux uporabnika. Ob inštalaciji dobi aplikacija svojo unikatno Linux uporabniško številko (ID). S to številko označi vse podatke ki jih shranjuje.

Privzeto so aplikacije brez kakršnih koli dovoljenj, če želimo dostopati do naprednejših funkcij moramo torej aplikaciji razglasiti dodatna dovoljenja. To naredimo med samim razvijanjem aplikacije z uporabo <uses-permission> tag-ov v manifestu aplikacije (manifest.xml). Med samo inštalacijo se aplikaciji dodeli zahtevana dovoljenja, seveda le v primeru da se s tem strinja uporabnik.

# 4. ANDROID ANTI-FORENZIKA

V tem poglavju bomo spoznali orodja in tehnike, ki lahko pomagajo v anti-forenziki. Prvo se bomo ukvarjali z že obstoječi orodji in aplikacijama, potem bomo spoznali vlogo zasebne mape in na koncu, predstavili sam potek anti-forenzike.

## 4.1 Možna orodja in tehnike

### *4.1.1 Android debug bridge(ADB)*

ADB omogoča interakcijo med napravo z Android operacijskim sistemom in oddaljeno delovno postajo, oziroma računalom. ADB deluje na principu oddaljenega izvajanja ukazov na mobilni napravi. Nekateri od teh ukazov so tudi dd, ls, pull, ki so lahko zelo uporabni v interakciji z notranjim pomnilnikom naprave.

### *4.1.2 NANDroid backup*

Orodje, ki nama omogoča prepisovanje in obnovitev podatkov z mobilne naprave. Z Nandroidom lahko preslikamo celoten NAND flash pomnilnik.

### *4.1.3 Fizično preslikovanje z dd*

Kot je že omenjeno, Android operacijski sistem v ozadju poganja Unix sistem. Torej, v komunikaciji z napravo lahko uporabljamo kaze Unix komandne linije. Predvsem, zelo je pomemben ukaz, oziroma orodje dd. Ukaz dd omogoča preslikovanje datotek na ravni bajtov.

### *4.1.4 Komercialna orodja - MIAT*

MIAT, oziroma Mobile Internal Acquisition Tool, je še eno v nizu orodij za pridobitev in obradu podatkov z mobilne naprave.

### *4.1.5 Mobilna aplikacija AFDroid*

AFDroid je mobilna aplikacija, ki so je naredili avtorji članka, Alessandro Distefano, Gianluigi Mea, Francesco Pace. Več govora o sami aplikaciji bo v nadaljevanju.

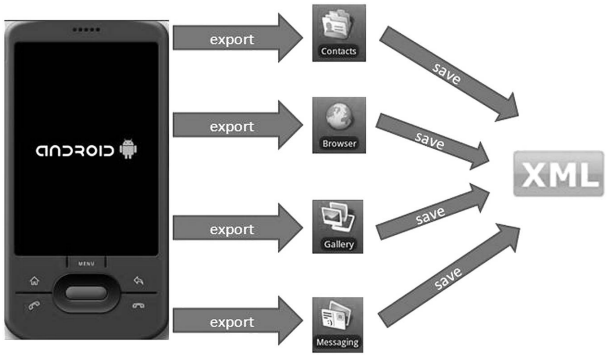
## 4.2 Zasebna mapa

Znotraj Android operacijskega sistema, vsaka aplikacija lahko uporablja svojo zasebno mapo. Do zasebne mape lahko dostopa samo lastnik, oziroma, druge aplikacije niso v možnosti brati ali pisati podatke vanjo. Zasebna mapa se kreira v notranjem pomnilniku naprave pri procesu nameščanja aplikacije. V zasebni mapi lahko so kakršne koli informacije, besedilni podatki, multimedija...

Zasebna mapa se samodejno briše pri procesu odstranjevanja aplikacije. Po brisanju zasebne mape iz notranjega pomnilnika mobilne naprave, do podatkov, ki so bili v mapi, je skoraj nemogoče dostopati.

Podatkov v zasebni mapi ni treba šifrirati, ker druge aplikacije nimajo dostopa do vsebine mape. To pomeni, da je postopek prenosa in branja podatkov preprost. Tudi časovna zahtevnost operacij znotraj aplikacije torej ni velika.

Iz vsega zgoraj navedenega, lahko zaključimo, da je zasebna mapa zelo dobra možnost v smislu anti-forenzike. Vsi ogroženi




**Slika 3: Proces izvoza dokazov**

podatki se lahko shranjujejo v zasebno mapo. Če se oseba želi znebiti takih podatkov, treba je samo odstraniti aplikacijo. Odstranjevanje aplikacije ne zahteva veliko časa (čas je na ravni sekund).

V aplikaciji AFDroid, zasebna mapa ima ključno vlogo v procesu anti-forenzike.

## 4.3 Izvoz dokazov

Izvoz dokazov je proces prenosa podatkov iz lahko dostopnih virov v zasebno mapo aplikacije AFDroid. Tako, podatki postanejo nedostopni za druge aplikacije in težko je narediti postopek obnove podatkov po odstranjevanju aplikacije.

Izvoz dokazov je popolnoma avtomatiziran postopek. Podatki se shranjujejo v datoteko export.xml.

Proces izvoza podatkov lahko prikažemo na primeru 4 tehnike anti-forenzike: brisanje dokazov, skrivanje dokazov, uničevanje virov dokazov, ponarejanje dokazov.

### *4.3.1 Brisanje dokazov*

Tehnika brisanja dokazov v aplikaciji vključuje brisanje sporočil, zaznamkov brskalnika in dnevnika klicev. Vsak podatek, ki lahko ogrozi lastnika mobilne naprave, trajno je izbrisan iz baze podatkov in shranjen v zasebno mapo.

### *4.3.2 Skrivanje dokazov*

Skrivanje dokazov izvedeno je na multimedijskih datotekah s spominske kartice. Ker je spominska kartica lahko dostopna (možno jo je odstraniti in pregledati s standardnimi postopki), podatki, ki so pomembni, shranjujejo se v zasebno mapo. Tako so dokazi skrivni.

### *4.3.3 Uničevanje virov dokazov*

Bogati viri dokazov so multimedijska sporočila in upravljanje pogovorov (Gmail e-mail upravljanje sporočil). Vsak pogovor je označen unikatnim identifikatorjem. To pomeni, da je vsako prejeto in poslano multimedijsko sporočilo mogoče povezati s pogovorom. Torej, vire dokazov lahko uničimo, če spremenimo identifikatorje v sporočilih, tako da niso več dostopni.

```
<database name='BROWSER'>
<table name='bookmarks'>
<row>
    <col name='_id'>1</col>
    <col name='title'>Google</col>
    <col name='url'>http://www.google.com/</col>
    <col name='visits'>7</col>
    <col name='date'>1263761875450</col>
    <col name='created'>0</col>
    <col name='description'>null</col>
    <col name='bookmark'>1</col>
</row>
</table>
</database>
```

**Slika 4: Del export.html datoteke**

### *4.3.4 Ponarejanje dokazov*

Android sistemi poleg klasičnih informacij o stikih, ponujajo tudi informacije o priljubljenih stikih in številu medsebojnih interakcij. V forenzični analizi ti dokazi lahko vodijo do zelo koristnih informacij (močna povezanost med stiki). Ponarejanje informacij o stikih lahko dovede do zavajajočih podatkov.

## 4.4 Uvoz dokazov

Uvoz dokazov je proces vrnitve podatkov iz zasebne mape v prvotno stanje. Drugače rečeno, uvoz dokazov je inverz procesa izvoza dokazov.

Če želimo podatke rekonstruirati, moramo jih nekako tudi shraniti. Podatki iz podatkovnih baz so v zasebni mapi shranjeni v datoteki export.html. Multimedijske datoteke iz spominske kartice so shranjene take, kot so.

V procesu vrnitve podatkov nastanejo težave, kot so dostopnost prejšnje slike, obvladovanje in dostopnost opreme za rekonstrukcijo. Zaradi tega je ta proces zapleten in časovno zahtevan. Pri tem procesu niso zajamčene tudi nekatere forenzične lastnosti, kot je vreme zadnje spremembe, ki je v procesu forenzične analize pogosto uporabljano. Tudi celovitost baze podatkov po obnovitvi ni zajamčena zaradi optimizacij, ki jih izvaja sistem za upravljanje baz.

## 4.5 Uničevanje dokazov

Kot je že prej omenjeno, se zasebna mapa, skupaj s celotno vsebino, samodejno izbriše pri procesu odstranjevanja dokazov. Ker se zasebna mapa nahaja v notranjem pomnilniku naprave, izbrisane podatke je težko izslediti.

Samodejni proces uničevanja dokazov pomeni izogibanje človeške napake, kot so na primer ostanki dokazov, pozabljena vsebina in podobno.

Največji vprašaj na to temo je, ali je vendar mogoče dobiti izbrisane podatke. Več o tem v naslednjem poglavju, Eksperimenti.




**Slika 5: Samsung Galaxy i7500.**

# 5. EKSPERIMENTI

Poglejmo si še nekaj eksperimentov, ki so bili planirani za testiranje anti-forenzičnih zmožnosti Androida. Testirali bomo EEP (evidence export process – proces izvoza dokazov) in EDP (evidence destruction process -> proces uničenja dokazov).

## 5.1 Cilj eksperimentov

Cilj obeh eksperimentov je testiranje moči izbranih procesov glede na trenutno dostopna orodja, ki so sposobna narediti sliko notranjega pomnilnika mobilne naprave. Cilj anti-forenzičnih tehnik je otežiti ali onemogočiti pridobitev in analizo dokazov na mobilni napravi. Moč anti-forenzične tehnike je zato obratno sorazmerna hitrosti oz. zmožnosti pridobivanja dokazov.

## 5.2 Pogoji eksperimentov

Eksperimente so izvajali na Samsung Galaxy i7500 napravi. Za pridobivanje so uporabili Nandroid orodje, s katerim so izdelovali slike notranjega pomnilnika. Za testiranje so razvili lastno aplikacijo, ki je naredila logično kopijo podatkov shranjenih v datotečnem sistemu na notranjem pomnilniku.

## 5.3 Potek eksperimentov

Proces izvoza dokazov (EEP):

- Slika naprave z Nandroid orodjem
- Izvedba AFDroid programa
- Pridobitev z MIAT orodjem
- Druga slika naprave z Nandroid orodjem

Proces uničenja dokazov (EDP):

- Slika naprave z Nandroid orodjem
- Izvedba AFDroid programa
- Druga slika naprave z Nandroid orodjem
- Brisanje AFDroid programa
- Pridobitev z MIAT orodjem
- Tretja slika naprave z Nandroid orodjem

## 5.4 Rezultati

Rezultate smo gledali na dva načina

- Bežen pregled:

  Pregled in analiza dokazov shranjenih na napravi s preprostim pregledom same naprave (sprehod skozi datoteke na napravi). Pri tej analizi se preiskovalec obnaša kot povprečen uporabnik mobilne naprave.

- Pridobitev in analiza notranjega pomnilnika:

  Uporabi prej omenjeni orodji za pridobitev slike notranjega pomnilnika. Te slike se nato pregleda z uveljavljenimi postopki forenzičnih preiskav. Tu preiskovalec opravlja pravo forenzično analizo.

### *5.4.1 Rezultati pri EPP (izvoz dokazov)*

Kot smo omenili že v četrtem poglavju se tekom tega procesa dokazi shranijo v privatno mapo (private folder). Po opravljenem procesu pokaže bežen pregled naprave naslednje stanje:

Pri kontaktih ni razlike SMS in MMS podatkovni bazi sta prazni Zgodovina klicanja je prazna Galerija je prazna

Analiza slike notranjega pomnilnika je bila zmožna najti vse relevantne dokaze (tako Nandroid kot MIAT). Za ekstrakcijo podatkov iz slike so uporabili še dodatno orodje unyaffs, ki zna iz slike rekonstruirati datotečno strukturo.

### *5.4.2 Rezultati pri EDP (uničenje dokazov)*

Dokler nismo izbrisali AFDroid aplikacije veljajo ista pravila kot pri procesu izvoza dokazov (torej iz druge slike pri tem poskusu lahko z orodji najdemo vse relevantne datoteke). Ko pa aplikacijo izbrišemo in stem zaključimo proces uničenja, se privatna datoteka aplikacije pobriše. Po tem ni nobeno od orodij bilo zmožno obnoviti izbrisane datoteke.

# 6. ZAKLJUČEK

Od prvotnih mobitelov, ki so imeli samo osnovne funkcije (pozivi, sporočila, stiki), današnji pametni telefoni so postali mali računalniki. Zaradi tega vsebujejo veliko koristnih informacij, ki so lahko dokazi v forenzični analizi.

Velik odstotek teh telefonov poganja Android operacijski sistem. Aplikacije na Android sistemoma lahko koristijo zasebne mape, ki lahko skrijejo dokaze pred drugimi aplikacijami in forenziki. Na tem principu narejena je aplikacija AFDroid.

Aplikacija AFDroid zelo uspešno je skrila/uničila morebitne dokaze pri testiranju. Trajanje procesa skrivanja je na ravni sekund ter ga lahko opredelimo kot zelo hitrega.

Iz vsega lahko zaključimo, da je dokaze na Android napravah mogoče skriti pred forenziki. Očitno bo treba nekaj narediti na tem področju. Verjetno bo v prihodnosti mogoče na učinkovit način obnoviti podatke tudi z notranjega pomnilnika naprave. Tako tudi zasebna mapa ni možna skriti dokaze pred forenzično analizo.

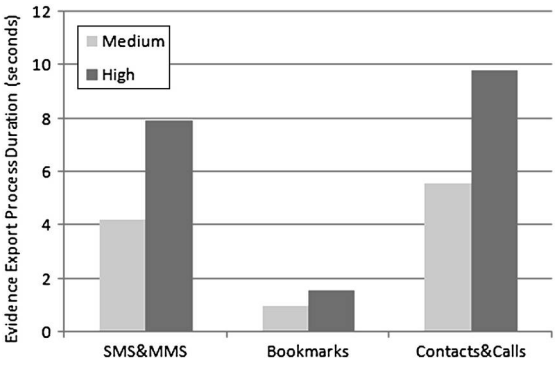



**Table 3 – Test Cases for the EEP described in terms of number of elements stored by the target databases.**

| Load | Contacts & calls | SMS & MMS | Bookmarks |
|---|---|---|---|
| Medium | 120 | 250 | 45 |
| High | 222 | 591 | 93 |

**Slika 6: Čas izvajanja procesa izvoza**

# Android forensics

## Automated data collection and reporting from a mobile device

Anton Semprimožnik
Fakulteta za računalništvo in informatiko, Magisterski študij
63100154

Andraž Pajtler
Fakulteta za računalništvo in informatiko, Magisterski študij
63100121

Matej Andolšek
Fakulteta za računalništvo in informatiko, Magisterski študij
63100037

## Povzetek

Članek opisuje prototip poslovnega sistema za monitoriranje pametnih mobilnih telefonov Android [2], ki v intervalih zajema različne tipe oz. množice podatkov, ki nas zanimajo pri iskanju morebitnih kršitev, varnostnem monitoriranju, forenzičnih preiskavah in podobno. Večina podatkovnih množic, ki jih opisani prototip zajema, drugi takšni sistemi niso zajemali. Sam prototip v osnovi ne zahteva privilegijev administratorja (ang. root), niti razkrivanja arhitekture samega Android sistema. Omogoča zmožnost medsebojne komunikacije različnih sistemov ter se izogiba klasifikaciji slabogramja oz. prisluškovalnih aplikacij (ang. spyware). Na sistemu je bila izvedena tudi anti-forenzična preiskava za identificiranje morebitnih ranljivosti s stališča možnega prirejanja podatkov, ki se zajemajo. Prispevek izvornega članka je povezan z izidom prve odprto-kodne poslovne aplikacije za monitoriranje Android sistemov, vodičem po veliki možici različnih podatkovnih množic, ki so na voljo za zbiranje brez zahteve po administratorskih pravicah in opis strukture Android aplikacij, ki so namenjene monitoriranju sistemov.

## Keywords

pametni telefoni, zbiranje podatkov, Android, upravlanje mobilnih naprav, razvoj Android aplikacij

## 1. UVOD

Delež uporabe pametnih telefonov se iz meseca v mesec hitro povečuje. Glavni igralci na trgu pametnih naprav so Google, Apple, Microsoft in do nedavnega tudi znani RIM (ang. Research in motion). Delež Googlovega operacijskega sistema Android še vedno raste, prav tako Applov iOS, med tem ko delež naprav podjetja RIM hitro pada. S trendom, kakršen je, bo ščasoma vedno več podjetji prisiljenih na prehod iz trenutnih mobilnih naprav, ki so dejansko osredotočene na uporabo v poslovnem svetu (znani BlackBerry, RIM) na novejše naprave, ki uporabljajo bodisi operacijski sistem Android ali iOS, saj tadva predstavljata največji tržni delež. Vedno pogostejša v podjetjih je tudi praksa imenovana "Prinesi svojo napravo" (ang. Bring your own device), ki še povečuje uporabo osebnih naprav Android/iOS v poslovnih okoljih. Problem, ki se tu pojavlja, je način oz. zaščita teh naprav znotraj oz. v okviru njihove uporabe v podjetjih.

Ciljna publika večine razvitih novih mobilnih naprav so posamezniki in ne več podjetja, kot je bilo to značilno za dobro poznan BlackBerry(RIM), ki je svojim poslovnih uporabnikom omogočal namestitve raznih poslovnih strežnikov za upravljanje teh naprav in podatkov na teh napravah. V nasprotju s tem naprave ostalih ponudnikov takšnih vgrajenih sistemov nimajo. Tem sistemom pravimo sistemi za upravljanje mobilnih naprav (ang. Mobile Device Management). Tu se pokaže tržna niša, ki hitro polni praznino sistemov za upravljanje z mobilnimi napravami na tem področju (ang. Third-party MDM - Mobile Device Management) [3], v nadaljevanju MDM. Trend kaže, da naj bi v naslednjih 5 letih kar 65 odstotkov podjetji pričelo z uporabo nekega MDM sistema zaradi morebitnih tveganj, povezanih z uporabo vedno novih, nepreverjenih naprav znotraj organizacij. Posledično, bodo vsa ta podjetja potrebovala natančne usmeritve in pravila za uporabo osebnih naprav zaposlenih v podjetjih (v okviru prakse "Prinesi svojo napravo").

Članek opisuje arhitekturno rešitev aplikacije DroidWatch [4] za naprave Android, ki omogoča samodejen zajem podatkov, pomembnih za notranje preiskave, ki vključujejo iskanje kršenja pravil, krajo intelektualne lastnine, ponarejanje, popravljanje podatkov, sabotaže in vohunjenje. Sam članek je zajem podatkov testiral na napravi Samguns Galaxy S 2.

Poglavje 2 opisuje stanje na področju sistemov za upravljanje mobilnih naprav.

Poglavje 3 opisuje terminologije razvoja aplikacij Android, varnost v aplikacijah Android, postopke pridobivanja administratorskega dostopa, raziskovanje pametnih mobilnih telefonov in razglablja o vprašanjih zasebnosti.

Poglavje 4 opisuje pregled nad obstoječimi komercialnimi in "third-party" aplikacijami.

Poglavje 5 opisuje arhitekturo in delovanje DroidWatch aplikacije.

Poglavje 6 opisuje rezultate testiranj aplikacije DroidWatch ter kako so lahko v pomoč pri notranjih preiskavah.

Poglavje 7 opisuje možne nadgraditve in nadaljni razvoj aplikacije.

## 2. SISTEMI ZA UPRAVLJANJE MOBILNIH NAPRAV

Mobilnost zavzema visok položaj med prioritetami v praktično vsaki organizaciji. Uporabniki zahtevajo dostop do mobilnih aplikacij in podatkov podjetja od koderkoli in kadarkoli, z različnih naprav, tudi lastnih. Tu ni več samo zajeta elektronska pošta, temveč tudi poslovne, spletne in mobilne aplikacije. Na drugi strani je običajno IT okolje, katerega skrbniki so stalno soočeni z varnostnimi in praktičnimi izzivi uporabe teh aplikacij in uporabniškega dostopa do podatkov z ne-upravljanih mobilnih naprav. S tem razlogom podjetja vse pogosteje uvajajo sisteme, ki pomagajo upravljati takšne mobilne naprave ter podatke na le teh.

Sistemi za upravlanje mobilnih naprav (ang. Mobile Device Management) varujejo, monitorirajo, upravljajo in podpirajo mobilne naprave ne glede na mobilnega operaterja in podjetje. Običajno omogočajo brezžično distribucijo aplikacij, podatkov in konfiguracijskih nastavitev za vse tipe mobilnih naprav. Uporabljajo se za vse, tako za naprave v lasti podjetji kot za osebne naprave zaposlenih. Z nadzorom in zaščito podatkov ti sisitemi zmanjšajo potrebo po podpori in zmanjšajo poslovne stroške. Cilj teh sistemov je optimizirati funkcionalnost in varnost mobilnih komunikacij v podjetjih ter zmanjševanje stroškov in časov nedosegljivosti (ang. downtime). V podjetjih strmo narašča uporaba osebnih naprav za dostop do podatkov podjetji. Sistemi MDM rešujejo problem upravljanja naprav, ki dostopajo in delajo s temi podatki. Osnovni cilj je torej upravljanje varnosti povezane z mobilnih dostopom do podatkov in ščitenje samega podjetja pred odtekanjem informacij.

Raziskava več vodilnih sistemov MDM je razkrila pomanjkanje funkcionalnosti, ki izvajajo avtomatiziran zajem forenzičnih podatkov iz naprav Android. Dosegljivost teh podatkov bi olajšala izvajanje vpeljanih varnostnih praks znotranj posamezne organizacije, torej odzive na incidente, beleženje varnostih logov, proaktivno varnostno monitoriranje in forenzične preiskave. 65 odstotkov podjetji uporablja lokalno varnostno beleženje (ang. Internal audits) kot varnostni mehanizem (vir: Computer Security Insistute Computer Crime 2010/2011). Velik del preostalih uporablja podobne tehnike za preprečevanje izgube podatkov in različne programske rešitve za zajem le teh. Večina podjetji se torej zaveda notranjih groženj v podjetju samem in uporablja različne metode in korake za ublažitev le teh. Splošno pomankanje funkcionalnosti za nadzor naprav in velika pomembnost zajetih podatkov za lokalne preiskave je vodila do izdelave prototipa in programa imenovanega DroidWatch, ki je opisan v nadaljevanju tega članka.

## 3. OZADJE

Poglavje opisuje osnovno terminologijo razvoja, varnostni model naprav Android, implikacijo rootanja naprav, upravljanje mobilnih naprav, trenutne raziskave in vprašanja zasebnosti.

### 3.1 Terminologija razvoja android aplikacij

Operacijski sistem Google – Android je razširjen odprtokodni operacijski sistem za pametne mobilne telefone, ki temelji na linuxovem jedru. Razvija ga Google v sodelovanju s podjetji združenja Open Handset Alliance (kratica OHA). Operacijski sistem je najpogosteje nameščen na mobilnih napravah, katerim je v prvi vrsti tudi namenjen, vendar ga kljub temu najdemo tudi na drugih napravah, kot so recimo tablični računalniki, ročne ure in podobno. Android ponuja platformo za izdelavo aplikacij in iger ter odprt trg za delitev le-teh. Razvojna orodja ponujajo vse, kar potrebujemo za razvoj odličnih, uporabniško prijaznih aplikacij, ki učinkovito uporabljajo strojne zmogljivosti naprav, na katerih tečejo.

Aplikacijske komponente sistema so sestavni deli programskega ogrodja Android, ki pomagajo definirati obnašanje aplikacij. Aplikacija, razvita v okviru tega članka uporablja komponente: aktivnosti (ang. activities), storitve (ang. services), ponudniki vsebine (ang. Content providers), sprejemniki sporočil (ang. Broadcast receivers), opazovalci vsebine (ang. Content observers) in alarmi (ang. alarms). Vsaka komponenta znotraj aplikacije služi svojemu namenu.

**Aktivnosti** tvorijo uporabniški vmesnik. Prikazujejo informacije, začenjajo nove, druge aktivnosti. V okviru aplikacije DroidWatch so aktivnosti uporabljene redko. Večina dela se opravi v zalednem delu in ne vpliva na običajno delo uporabnika.

**Storitve** so dolgo trajajoči procesi, ki tečejo v ozadju. Druge komponente sistema Android, kot so na primer aktivnosti, zaženejo storitve in jih ohranijo delujoče, tudi ko delujejo ostale aplikacije. DroidWatch se zanaša na trajno delujoče storitve za zbiranje podatkov in prenos oz. sinhronizacijo le teh na zaledni sistem aplikacije DroidWatch (server).

**Ponudniki vsebine** so aplikacijska komponenta, ki upravlja in deli podatke posameznih aplikacij. Aplikacije komunicirajo s ponudniki vsebine preko vnaprej določenih ponudnikov vsebine (ang. URI). Aplikacija DroidWatch jih uporablja za dva namena. Prvi je branje podatkov, shranjenih znotranj drugih aplikacij in drugi je pisanje podatkov, shranjenih znotraj aplikacije same.

**Sprejemniki sporočil** so komponente, ki v povezavi s ponudniki vsebine upravljajo in odgovarjajo na sistemske dogodke naprav Android. V okviru DroidWatch aplikacije detektirajo dogodke, kot so dohodna kratka sporočila, nameščanje aplikacij in podobno.

**Opazovalci vsebine** v povezavi s ponudniki vsebine prejemajo obvestila, ko je ciljna baza podatkov spremenjela. DroidWatch uporablja te komponente za zaznavanje sprememb podatkov v realnem času.

**Alarmi** so vnaprej določene oz. planirane operacije, ki v naši aplikaciji periodično preverjajo ponudnike vsebine in zahtevajo nove podatke. So zanesljivi, vendar tečejo le ob določenih trenutkih.

### 3.2 Varnost android aplikacij

Varnostni model Googla za aplikacije na Android platformi zahteva definicijo dovoljenja(ang. permissions) v datoteki

AndroidManifest.xml. V osnovi, aplikacija brez pravic ne more izvajati nobenih operacij, ki bi morebiti vplivale na druge aplikacije, operacijski sistem ali druge uporabnike. To pomeni, da aplikacija ne more dostopati do privatnih podatkov drugih apliakcij, uporabljati omrežnih storitev, pisati, brati iz internega pomnilnika ali uporabljati drugih osnovnih sistemskih operacij. Vsaka na novo naložena aplikacija mora pred namestitvijo obvestiti uporabnika o njenih zahtevah po pravicah in zahtevati njegovo potrditev.

## 3.3 Pridobitev administratorskega dostopa android aplikacij

Pridobitev administratorskega dostopa omogoča uporabnikom uporabo višje priviligiranih funkcij na napravi, ki so drugače možne le v določenih pogojih. Običajno je izveden s ciljem prekoračitve omejitev, ki so jih določili proizvajalci naprav. Rezultat je možnost spreminjanja in popravljanja sistemskih aplikacij in nastavitev, poganjanje specializiranih programov, ki zahtevajo administratorski dostop ali izvajanje ostalih operacij, ki so drugače za navadnega uporabnika nedovoljene. Na napravah Android omogoča administratorski dostop tudi popolno odstranitev operacijskega sistema in menjavo z drugim.

Tak dostop se lahko uporablja na legitimen ali nelegitimen način. Uporabniki aplikacije DroidWatch bi se lahko izognili varnostnim omejitvam oz. posegali v zajete podatke znotraj le te. Na drugi strani pa je lahko uporabljen legitimno s strani forenzičnih preiskovalcev za izluščevanje podatkov z naprave. V splošnem naj bo pridobivanje administratorskega dostopa zadnja možna opcija. Proces sam tipično odpre mnoge ranljivosti na specifični napravi ali operacijskem sistemu in lahko vodi do drugih varnostnih ranljivosti. Samo pridobivanje administratorskega dostopa vodi tudi do spreminjanja delov naprave, kar je konktradiktorno forenzičnim praksam. So pa primeri, ko je pridobivanje tega dostopa nujno.

Splošno, administratorski dostop bi povečal funkcionalnost aplikacij, kot je DroidWatch, vendar prinaša preveliko število posledic, kot so ogrožena sistemska varnost, zmanjšana interoperabilnost in morebitno prenehanje garancije s strani dobavitelja.

## 3.4 Raziskovanje pametnih telefonov

Vladne agencije in organi pregona igrajo glavno vlogo na področju mobilne varnosti, saj so pooblaščeni za preiskovanje in raziskovanje kriminalnih dejanj in zavarovanje pomembnih vladnih informacij. Korporacije pravtako kažejo velik interes glede varnosti mobilnih naprav za zaščito svojih podjetji pred razkrivanjem podatkov pred konkurenti, finančnimi krajami in krajami intelektualne lastnine. Ločitveni postopki, sodni postopki in podobno enako pridobivajo sorazmerno z razvojem tega področja.

Imamo več tipov raziskav, ki lahko pridobijo zaradi monitoriranja mobilnih naprav, kot so notranje raziskave, preiskave kršenja zakonika, privatne raziskave. Sama DroidWatch aplikacija rešuje problem notranjih raziskav v okviru podjetja s strani posameznika, pogodbenika ali drugo (npr. odziv na incident, varnostne zabeležke) glede potencialnih kršitev pravil, kraje intelektualne lastnine, ponarejanja, zlorabe, sabotaže, poneverbe, vohunjenja in ostalih nelegalnih dejanj. Notranji preiskovalci se niso vezani držati striktnih postopkov forenzičnih preiskav in postopkov ohranjanja dokazov, vendar je zaželjeno, da se držijo običajnih praks in tehnik forenzičnih preiskav.

**Table 1: Vprašanja zasebnosti na napravah**

| Primer | Zakon |
|---|---|
| U.S v Zieglerju (2007) | Organizacija ima pravico do monitoriranja njene lastne opreme, pod pogojem, da se uporabniki tega zavedajo in so o podrobnosti obveščeni |
| City of Ontario (2010) | Monitoriranje se lahko izvaja na vseh napravah podjetja, tudi če stroške uporabe plačuje uporabnik |
| Carrier IQ | Ni na voljo |

Za pridobivanje pomembnih podatkov z mobilnih naprav z namenom izvajanja notranjih preiskav je bil običajno zahtevan fizični dostop do preiskovane naprave. Velja omeniti izjemo, podjetje EnCase Enterprise [5], ki je izvedlo oddaljeno forenzično preiskavo naprav Android preko omrežja. Obstajajo tudi sistemi MDM, ki omogočajo omejeno oddaljeno monitoriranje in preiskovanje, vendar še vedno ne dovolj, da bi pokrili vse potrebe po podatkih, zahtevanih za preiskavo. Če predvidevamo, da je naprava fizično zasežena, obstaja vrsta orodij in tehnik za zajem slike trenutnega stanja naprave. Večina zahteva prenos podatkov preko povezave z USB kablom in administratorski dostop.

Splošno se mobilna forenzika osredotoča na pametne telefone kot vir dokazov, izvaja potrebne korake, ki kvalitetno rokujejo z napravami v postopku preiskave, razumejo različne tehnologije, iščejo slabogramje, analizirajo rezultate in pridobivajo digitalne dokaze z raziskovanjem do najnižjih nivojev sistemov mobilnih naprav.

## 3.5 Vprašanja zasebnosti

Aplikacija DroidWatch prikaže med samim zagonom mobilne naprave uporabniku obvestilo o soglasju uporabe le te ter aktivnosti, ki jih aplikacija izvaja. S tem se aplikacija izogne klasifikaciji "spyware"-a. Obstajajo posebna pravila, kako mora aplikacija pokazati obvestilo o zajemanju podatkov. Podjetja imajo torej pravico naložiti nadzorne oz. monitorirne aplikacije na vse naprave podjetja kot tudi na naprave, ki so v zasebni lasti, v kolikor le te delujejo v privatno nadzarovanem omrežju.

# 4. POVEZANA DELA

Poglavje prikazuje in opisuje pregled nad obstoječimi komercialnimi in "third-party" aplikacijami.

## 4.1 Komercialni produkti

Na internetu lahko najdemo veliko različnih MDM aplikacij. Aplikacije so razvite bodisi s strani velikih podjetji bodisi s strani posameznikov oz. nepotrjenih avtorjev (ang. third-party application). V večini primerov MDM aplikacije omogočajo samo osnovni zajem forenzičnih podatkov, kar je zelo slabo, saj se pri digitalni preiskavi velikokrat uporabljajo tudi ostali podatki. Zanimivo je, da večina vodilnih

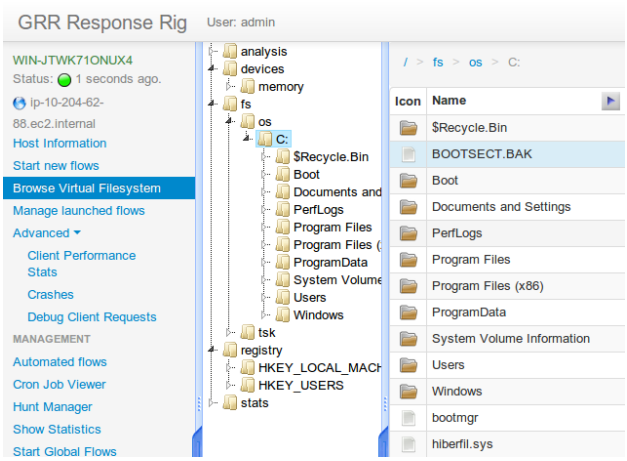



**Figure 1: Zaslonska maska Google GRR aplikacije**

aplikacij (Zenprise [6], AirWatch [7] in MobileIron [8]) na tem področju omogoča samo osnovni zajem podatkov (SMS, GPS lokacija itd. . . ). Med raziskavo aplikacij je bilo ugotovljeno, da aplikacija, ki spada med vodilne v svoji kategoriji zajame samo cca. 50% podatkov v primerjavi z aplikacijo DroidWatch. Eden izmed problemov je tudi, da se vsi podatki hranijo na strežnikih podjetja, ki je razvilo to aplikacijo.

Na voljo imamo tudi nekaj drugih produktov, ki shranjujejo forenzične dokaze. To so tako imenovane "personal spy" aplikacije. Prednost teh aplikacij je, da so zmožne zbirati toliko podatkov kakor aplikacija DroidWatch, vendar potrebujejo dodatne pravice za dostop do mobilne naprave kar pa ne želimo. Dodatna slabost takih aplikacij je, da brez uporabnikove vednosti zajemajo njegove osebne podatke, kar je v zelo velikih primerih v nasprotju z zakonom. Take aplikacije se lahko brez naše vednosti namestijo na sistem in pošiljajo naše osebne podatke podjetjem, ki jih lahko nato izkoristijo na veliko različnih načinov.

Na trgu se pojavljajo tudi orodja za oddaljeno zbiranje podatkov, ki naj bi povečala varnost v podjetju. Z orodji za oddaljeno zbiranje podatkov lahko podjetje, policija oz. digitalni preiskovalci v primeru, da pride do incidenta zajame forenzične podatke kar preko omrežja. To preiskovalcem omogoča, da lahko dokaj hitro ovržejo oz. potrdijo določene začetne sume in domneve, še predno fizično dostopijo do računalnikov, saj se lahko le ti nahajajo nekje na oddaljeni lokaciji. Poznamo večje število aplikacij, ki omogočajo oddaljeno zbiranje podatkov. Ena izmed bolj poznanih je aplikacija GRR [9] (Slika 1), katere lastnik oz. razvijalec je računalniški gigant Google.

Poznamo še mnogo drugih aplikacij, ki opravljajo podobno delo kakor GGR (EnCase Enterprise, AccessData Enterprise [10], . . . ). Problem je, da večina aplikacij, ki omogočajo oddaljen zajem forenzičnih podatkov ne podpira operacijskega sistem Android. To pomeni, da mora digitalni preiskovalec fizično dostopati do naprave ter iz nje zajeti vse uporabne podatke. Slaba stran takih aplikacij je, da forenzičnih podatkov ne zbirajo neprestano ampak so podatki zajeti šele takrat, ko je to zahtevano s strani operaterjev. Vse slabosti zgoraj naštetih aplikacij je možno izboljšati z Android mobilno aplikacijo DroidWatch, ki omogoča neprestano pošiljanje zajetih podatkov in komunikacijo z oddaljenim strežnikom.

## 4.2 Raziskava obstoječih aplikacij

Raziskava oz. pregled aplikacij je pokazal, da je nekaj razvijalcev že poizkušalo razviti aplikacijo, ki bi omogočala zajem forenzičnih podatkov. Eden izmed razvijalecv je prišel na zamisel, da bi na zunanjo pomnilniško kartico SD namestil aplikacijo, ki bi preko API-ja omogočala zajem forenzičnih podatkov na isto SD kartico, na kateri se nahaja sama aplikacija. V osnovi je zamisel dobra, vendar aplikacija ne podpira neprestanega zajema podatkov kar pa je ena izmed večjih slabosti. AFLogical [11] je odprtokodna aplikacija, ki uporablja podoben pristop s SD kartico in prav tako ne podpira neprestanega zajema podatkov. Poleg tega je eden glavnih problemov tudi ta, da aplikaciji ne podpirata oddaljenega zajema podatkov in je zato vedno potreben fizičen dostop do naprave. Mobilni preiskovalec mora fizično vzeti SD kartico ven ter jo pregledati na drugem računalniku. Tak način je lahko zelo nepraktičen saj se lahko zgodi, da nimamo oz. ne moremo dobiti fizičnega dostopa do mobilne naprave. Lahko se tudi zgodi, da oseba SD kartico zavrže oz. uniči, lahko je kartica šifrirana, lahko pride do napak pri izdelavi forenzične kopije itd. Razvita je bila tudi aplikacija, ki omogoča neprestan zajem podatkov na podoben način kot VNC [12] (ang. Virtual network computing). Aplikacija lahko deluje brez "root" pravic, vendar še vseeno ne dosega funkcij, ki si jih želimo pri dobri forenzični aplikaciji.

# 5. DROIDWATCH

DroidWatch je avtomatiziran sistem, ki je sestavljen iz Android aplikacije in zunanjega strežnika na katerega se shranjujejo podatki. Aplikacija omogoča neprestano beleženje in zajem forenzičnih podatkov. Ti se preko določenega protokola pošiljajo na zunanji strežnik, kjer se hranijo in so na voljo za nadaljnjo uporabo. Aplikacija za svoje delovanje ne potrebuje "root" pravic, kar nam olajša namestitev in uporabo.

## 5.1 Sistemska arhitektura

Celotna sistemska arhitektura (Slika 2) je sestavljena iz več plasti. Vsaka plast nižje predstavlja večjo abstrakcijo. Na najnižjem nivoju se nahaja Java aplikacija, ki se izvaja na mobilni napravi ter zunanji strežnik s katerim komunicira aplikacija. Java aplikacija uporablja Android storitve, ki komunicirajo z notranjo SQL [13] bazo, direktnim dostopom do podatkovnega sistema, mrežnim prenosom, opazovalci vsebine itd.

## 5.2 Obvestilo uporabniku, da se izvaja nadzor

Zaradi pravnih in formalnih razlogov je v primeru, da se stalno zajema uporabnikove podatke potrebno uporabnika o tem obvestiti. Ker aplikacija omogoča ročni in avtomatski zagon, je potrebno ob prižigu mobilne naprave oz. ob ročnem zagonu aplikacije uporabnika o tem obvestiti. Uporabnik lahko zajemanje podatkov tudi odkloni. V primeru, da se uporabnik strinja, se v ozadju prižge storitev, ki skrbi za celoten zajem in prenos podatkov. Ker je aplikacija odportokodna (ang. open source) to pomeni, da lahko vsak vidi njeno izvorno kodo. To posamezniku omogoča, da si lahko aplikacijo prilagodi po svojih potrebah tako, da odvzame oz.

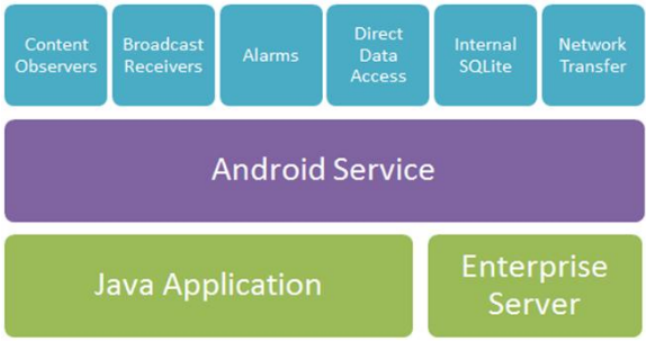



**Figure 2: DroidWatch sistemska arhitektura**




**Figure 3: Zajem forenzičnih podatkov**

doda kakšen modul, ki ga potrebuje. V primeru, da želi organizacija oz. posameznik spremeniti pogoje, ki se prikažejo ob zagonu naprave lahko to stori v datoteki "Strings.xml". Prikaz obvestila je realiziran s pomočjo Android aktivnosti (ang. Android activity), ki je v Android manifestu konfigurirana, da se izvede takrat, ko se zaključi boot proces (BOOT_COMPLETED). Zastavica "BOOT_COMPLETED" je zelo uporabna, saj je možno aplikacijo razviti tako, da v primeru ko se uporabnik ne strinja z navedenimi pogoji, ne mora uporabljati mobilne naprave. To še posebaj pride prav podjetjem, ki svojim zaposlenim nudijo službene telefone.

## 5.3 Strategija zajema podatkov

Zaradi poenostavitev razvojnega procesa DroidWatch aplikacije se pri razvoju uporablja tri že navedene komponente Android operacijskega sistema (sprejemniki sporočil, opazovalci vsebine in alarmi) - Slika 3. Sama strategija zajema je dokaj preprosta. Na začetku je potrebno preveriti nabor podatkov, ki jih želimo zajeti ali generirajo sistemske dogodke. V primeru, da nabor podatkov generira sistemske dogodke jih je možno zajeti s sprejemniki sporočil. Če se zgodi, da nabor podatkov ne generira sistemskih dogodkov, aplikacija za zajem uporablja opazovalce vsebine. Možno je tudi, da nabor podatkov ne generira ne sistemskih dogodkov in podatkov ni možno zajeti z opazovalci vsebine. Takrat se uporabi komponenta alarmov. Vse naštete komponente operacijskega sistema so bolj natančno opisane v poglavju Ozadje.

## 5.4 Lokalno shranjevanje podatkov

Kljub temu, da aplikacija komunicira z zunanjim strežnikom je potrebno zajete podatke nekam shraniti. Ker operacijski sistem Android nudi podporo za podatkovne baze, današnje mobilne naprave pa imajo več prostorskega spomina kot v preteklosti, se vsi podatki shranjujejo kar na mobilno napravo. Najbolj razširjena podatkovna baza za operacijski sistem Android je SQLite [14]. SQLite baza omogoča večino možnosti kot ostale podatkovne baze. Od ostalih podatkovnih baz se razlikuje po tem, da je del aplikacije (ostale so ločen proces). Celotna struktura baze, tabele, indeksi in ostalo je shranjeno v eni datoteki.




**Figure 4: Prenos podatkov med mobilno napravo in zunanjim strežnikom**

Shranjevanje podatkov v lokalno bazo na uporabnikovo mobilno napravo je lahko zelo velik problem. Če je uporabnik malo bolj vešč mobilne tehnologije oz. programiranja lahko poišče lokacijo podatkovne baze in spremeni določene podatke, ki bi bili lahko pomembni za forenzično preiskavo. Vsi zajeti podatki se periodično pošiljajo iz podatkovne baze do zunanjega strežnika (periodo lahko nastavimo v nastavitvah aplikacije). Pošiljanje se izvaja s pomočjo protokola HTTPS [15].

V teoriji bi lahko torej vsak napreden uporabnik med vsako periodo spremenil zajete podatke, kar bi privedlo do nekonsistentnih podatkov. Da se izognemo takim in podobnim problemom, je SQLite baza zaščitena na tak način, da lahko do nje dostopa samo aplikacija DroidWatch. Po končanem prenosu podatkov se prenešeni podakti izbrišejo iz lokalne baze.

## 5.5 Zunanji strežnik

Zunanji strežnik je namenjen shranjevanju in pregledu podatkov, ki so zajeti na mobilni napravi. Uporablja operacijski sistem Ubuntu [16] ter zagotavlja podporo za Apache [17], MySQL [18] in Splunk [19]. Spletni strežnik je konfiguriran s pomočjo SSL-a [20] saj je povezava med mobilno napravo in zunanjim strežnikom šifrirana. Povezava je šifrirana zaradi več razlogov. Eden izmed razlogov je, da morajo biti podatki, ki prihajajo iz mobilne naprave nespremenjeni ter pravilni. V primeru, da nekdo med prenosom spremeni datoteko, lahko privede do katastrofalnih posledic. Pomembna je tudi zasebnost. Če za prenos datoteke oz. podatkov izberemo HTTP protokol, ki ni šifriran, so vsi podatki vidni kot čistopis. To napadalcu omogoča nedovoljeno branje podatkov. Slika 4 prikazuje podatkovni prenos med mobilno napravo in zunanjim strežnikom.

Ročno vpisovanje vseh podatkov, ki se zajamejo na mobilni napravi v MySQL podatkovno bazo na zunanji strežnik je zelo zamudno. Za avtomatsko vpisovanje novih podatkov v MySQL skrbi skripta, ki je napisana v PHP jeziku. V primeru, da je aplikacija DroidWatch nameščena na mobilni napravi, ki se zelo veliko uporablja, se na njej nahaja zelo veliko forenzičnih podatkov. Ker je pri velikem številu podatkov zelo težko izluščiti bistvo je na zunanjem strežniku nameščena aplikacija Splunk. Aplikacija Splunk periodično prebira podatke iz podatkovne baze ter omogoča lepši in bolj efektiven prikaz podatkov in poročil.

## 5.6 Podatkovne zbirke

Collected data sets.

| Data set | App component used | | |
|---|---|---|---|
| | Broadcast receiver | Content observer | Alarm |
| App install/removal | ✓ | | |
| Browser navigation | | | ✓ |
| Browser search | | | ✓ |
| Calendar event | | | ✓ |
| Call log | | ✓ | |
| Contact list | | ✓ | |
| Device account[a] | | | |
| Device ID | | | ✓ |
| GPS location | | | ✓ |
| GPS location setting | ✓ | | |
| MMS | ✓ | | ✓ |
| Picture gallery | | ✓ | |
| Screen lock status | ✓ | | |
| SMS | ✓ | ✓ | |
| Third-party app log | | | ✓ |

**Figure 5: Prenos podatkov med mobilno napravo in zunanjim strežnikom**

DroidWatch omogoča zajem veliko različnih podatkov oz. podatkovnih zbirk. Vse podatkovne zbrike (Slika 5) so bile izbrane na podlagi zahtev pri forenzičnih preiskavah ter na podlagi dostopa do njih (tiste, ki za dostop do njih ne potrebujemo "root" pravic). V tabeli, ki prikazuje vse zajete zbirke lahko vidimo, da vsaka programska komponenta zajema določene podatke oz. več različnih komponent zajema iste podatke. Celotem zajem je prilagodljiv. Lahko se zgodi, da določena organizacija ne potrebuje, ne sme oz. ne želi zajemati vseh podatkov. V tem primeru lahko v datoteki "droidwatch.properties" spremeni nastavitve. Zaradi varnosti je datoteka vključena v sam projekt, kar onemogoča uporabniku mobilne naprave ročno spreminjanje nastavitev. Nastavitve lahko spremeni samo tisti, ki ima dostop do izvorne kode aplikacije katera je nameščena na mobilni napravi.

Opazimo lahko, da v tabeli manjkata dve podatkovni zbirki email sporočil in gesla računov. Razlog za to je, da mehanizmi, ki omogočajo dostop do email sporočil niso del standardnega Android razvojnega orodja. Drug problem je, da so tako gesla kot email sporočila zaščitena z "signatureOrSystem" pravico, ki določa, da do podatkov ni mogoče dostopati preko aplikacij posameznikov oz. nepotrjenih avtorjev.

# 6. ANALIZA IN OCENA

Ta razdelek opisuje rezultate testiranj aplikacije DroidWatch ter kako so lahko v pomoč pri notranjih preiskavah. Scenarij testiranj je temeljil na razpoložljivih zbirkah podatkov in avtorjevih predhodnih izkušnjah. Vsi rezultati so bili pridobljeni iz ene naprave in njenega uporabnika. Omeniti je potrebno, da se je v tej raziskavi uporabili Splunk zaradi svoje cene(brezplačen do 500 megabajtov na dan), konkurenčni produkti lahko opravljajo podobne funkcije.

Slika 6 prikazuje število zabeleženih dogodkov pridobljenih iz orodja Splunk v razponu enega dneva, razčlenjeni so po podatkovnih zbirkah. Vzorec vsebuje 442 zapisov, pridobljenih iz ene naprave in poudarja relativno majhno količino zabeleženih

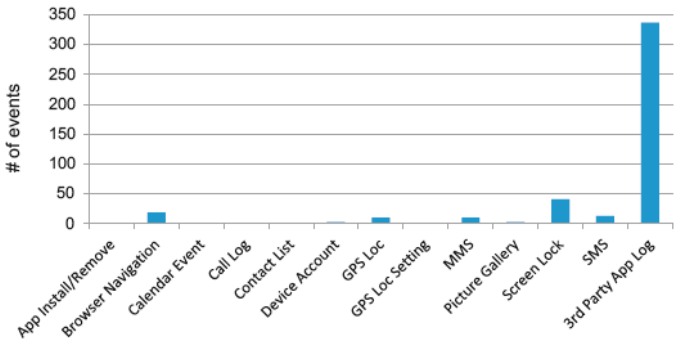



**Figure 6: Število zabeleženih dogodkov pridobljenih iz orodja Splunk**




**Figure 7: Dogodki "Screen Unlocked" - Splunk**

dogodkov. Seveda skupno število dogodkov na dan temelji na uporabnikovih navadah uporabe. Na napravi, ki je bolj v uporabi bomo videli večje število dnevniških zapisov. V času eksperimenta, kar je zelo pomembno, ni bilo zaznanih negativnih vplivov na uporabniško izkušnjo ali povečano porabo baterije

## 6.1 Splošni trendi uporabe

Iskanje po dogodku "Screen Unlocked" v orodju Splunk nam prikaže časovnico začetkov uporabnikovih akcij, ki kažejo na uporabo naprave (npr. vnos PIN kode, gesla, geste). Rezultate zbrane v razponu nekaj dni so vidni na sliki 7. Ti podatke se lahko uporabljajo tudi za določitev dejavnosti telefona na točno določen čas, odkritje vzorca uporabe ali odkritje neavtorizirane uporabe oz. uporabnika, ki mu ni bila dodeljena naprava.

## 6.2 Sumljivi kontakti in komunikacija

Kontaktna imena ali telefonske številke, ki predstavljajo tveganje za organizacijo se lahko odkrijejo z iskanjem ali vključitev Splunk sprožilcev (na voljo v Splunk Enterprise). To so lahko številke zunaj imenika organizacije ali predhodno ugotovljeni slabi posamezniki.

Tudi po zabeleženi vsebini SMS in MMS sporočil je mogoče iskati za sumljive dejavnosti. Dohodni SMS v DroidWatch lahko služi kot časovni vir, saj so časovni žigi vgrajeni v SMS neodvisni glede na uro telefona (Casey, 2009). Nekatera težave, na katere se je naletelo ob analizi podatkov so:

- Sporočila, poslana na več stikov so v dnevnikih navedena le enkrat
- Dohodna SMS sporočila ne vsebujejo časovni pas.
- Sporočilo v MMS ni na voljo.

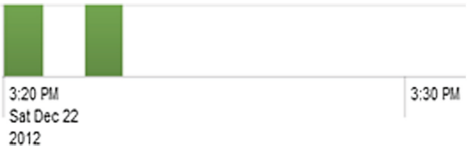



**Figure 8: Dnevniški zapis**

Če je bila slika posneta s kamero telefona takoj poslana preko MMS sporočila je to alarm za preiskovalce. Preveriti je potrebno okoliščine. Slika 8 prikazuje dnevniške zapise, kjer je bila slika zajeta v soboto, 22.12.2012, ob 15:20 kasneje priložena MMS sporočilu. Če se odkrije (po možnosti z GPS sledenjem), da je bil uporabnik sam v pisarni, je potrebno dodatno raziskati ali je prišlo do uhajanja podatkov.

## 6.3 Spremljanje lokacije

Zadnja znane lokacije, ki jih zabeleži DroidWatch vključujejo ID naprave, zemljepisno širino, zemljepisno dolžino in čas zajema. Pristop, ki ga DroidWatch uporablja za zbiranje lokacij varčuje z baterijo, kar pa se tudi odraža na številu zabeleženih lokacij. Le štiri lokacije so bile zabeležene v sedem-dnevnem obdobju. Čeprav je bila nastavitev GPS storitve vklopljena, zadnja znana lokacija ni shranjena na napravi v če se GPS ne uporablja aktivno (tj., aplikacija Google Maps prikazuje trenutni položaj). Poleg tega je zadnja znana lokacija telefona počiščena ob vnovičnem zagonu naprave, tako se lahko koordinat izgubijo še preden je bila možnost jih zabeleži.

Tudi zabeleženi dogodki v koledarju so lahko koristni za preiskovalce. Pomagajo lahko pri preverjanju alibija, načrtovanja nadzora ali identifikacijo potovalnih načrtov.

## 6.4 Zgodovina brskanja

DroidWatch zbira in daje na razpolago dogodki v okviru vgrajenega Android spletnega brskalnika. Vsak dogodek vsebuje tip (npr. brskanje ali iskanje), iskani niz ali URL, čas dogodka, ID naprave. Te podatki se lahko uporabijo za prepoznavanje sumljive uporabe brskalnika, kot na primer nalaganje intelektualne lastnine na zunanje spletne strani. Iskanja v brskalnik tudi lahko razčlenimo za boljše razumevanje uporabnikovih namer za zaznanimi dejanji.

## 6.5 Zlonamerne aplikacije

Pregled nameščenih aplikacij na napravi lahko razkrije, da je nameščena zlonamerna ali druge aplikacije, ki vzbujajo skrb. To bi upravičevalo dodatne skrbi in varnostne ukrepe med notranjo preiskavo. DroidWatch nam v rezultatu vrne ime aplikacije, dejanje in datum namestitev/odstranitve. DroidWatch tudi zbira dnevniške zapise ostalih aplikacij(Third-party apliakcij).

Več mehanizmov filtriranje omejuje dnevnike, na le ne-sistemske aplikacije. Filtriranje ni bilo popolno, a se je skupna količina zbranih dnevniških zapisov vseeno znižal na bolj obvladljivih 337 dnevnikov na dan (padec iz več kot 10.000).

## 6.6 Anti-forenzika

Naslednji razdelki so kategorije anti-forenzike, ki izhajajo iz prejšnjega dela, opravljenega na področju Android anti-forenzike in se uporabljajo za ocenjevanje ranljivosti DroidWatch aplikacije. Kategorije so: uničevanje dokazov, skrivanje dokazov, spreminjanja virov dokazov, ponarejanje dokazov, in odkrivanje forenzičnih orodij (Distefano et al, 2010;. Azadegan et al, 2012).

Treba je poudariti, da je DroidWatch, v sedanji obliki, povsem dovzeten na administratorski dostop, odstranitev aplikacije, zaustavitev procesa. Zunanja zaščita, ki jo ponujajo MDMs, kot so detekcija administratorskega dostopa, odkrivanje in uveljavljanje politike nameščanja aplikacij, so bistvenega pomena za celovitost podatkov v DroidWatch.

### *6.6.1 Uničevanje dokazov*

Podatki zbrani preko sprejemnikov sporočil, opazovalcev vsebine, so precej varni pred metodami uničenje saj se kopija dogodkov shrani v zaseben spomin aplikacije DroidWatch. Medtem pa so podatki zbrani preko alarmov bolj dovzetni na uničenje, mogoče je namreč odstraniti dokaz (npr. poslano MMS sporočilo, "third-party" aplikacijo, dogodek v koledarju, zgodovino brskanja, zadnjo znano GPS lokacijo) še pred načrtovanim zbiranjem.

Eden od povezanih pomislekov je, da lahko aplikacije registrirajo filter sporočil po meri ob namestitvi. Aplikacija, ki uporablja maksimalno vrednost prioritete($2^{31}$ -1), je tako sposobna prestreči vse sporočila preden jih lahko dobi druga aplikacija. DroidWatch, v fazi prototipa, uporablja privzeto vrednost prioritete.

### *6.6.2 Skrivanje dokazov*

Podatki, zbrani z rednimi alarmi so med tistimi, ki so najbolj dovzetni na taktike skrivanja dokazov. Če želi uporabnik skriti nekaj nedavnih poslanih MMS sporočil jih lahko ročno preusmeri stran od procesa zbiranja podatkov aplikacije DroidWatch.

Omenjena sposobnost aplikacij registrirati filter sporočil po meri, lahko podobno omogočajo aplikacijam skrivanje podatkov s prestrezanjem in preusmeritvijo. Na primer, SMS aplikacija GoSMS [21], se prijavi z največjo možno prioriteto na dohodna SMS sporočila in tako odpravi podvajanje obvestil sistema.(Kovačević, 2011).

### *6.6.3 Spreminjanja virov dokazov*

Spreminjanje virov dokazov (Distefano et al. 2010), gre za spremembo niza podatkov, da bi preprečili zbiranje podatkov. To je še eno področje skrbi za DroidWatch. Podatki, ki so zbrani prek alarmov so občutljivi, ker se proces zbiranja zanaša na prisotnost določenih vrednosti.

Na primer, ko se pregledujejo nova odhodna sporočila preko MMS ponudnika vsebine, polje "msg_box" označuje smer sporočilo, vrednost "2", predstavlja poslana sporočila. Če se vrednost tega polja spremeni v "5" se sporočilo v postopku zbiranja prezre.

### *6.6.4 Ponarejanje dokazov*

Ponarejanje dokazov na mobilnih napravah vključuje dodajanje lažnih podatkov v obstoječe, z namenom zavesti preiskovalce. DroidWatch zbirke so ranljivi, ker ni preverjanja, ki bi razlikoval med ponarejenimi in pravimi vnosi.

Dodajanje ogromnih količin fiktivnih podatkov v kratkem času (ang. denial of service attack) [22] lahko prav tako prepreči pravilno delovanje aplikacije DroidWatch.

### *6.6.5 Odkrivanje forenzičnih orodij*

Raziskava odkrivanja forenzičnih orodij,ki jo je izvedel (Azadegan et al., 2012) ni bilo mogoče neposredno prenesti na DroidWatch. (Azadegan et al., 2012) se je namreč osredotočil na poslušanje začetnih priključitvenih podpisov nekaterih znanih forenzičnih orodja za telefone Android. DroidWatch opravlja nadzor, za razliko od tradicionalnih forenzičnih orodij, in nima začetnih priključitvenih podpisov ali zahtev za fizični dostop do naprave.

Vendar pa se lahko idejo odkrivanja podpisov, uporabi proti načrtovanim prenosom aplikacije DroidWatch. Če avtomatizirano orodje zazna vzorec prenosa na strežnik, lahko prekine internetno povezavo na napravi in prenos se preloži na naslednji čas.

## 7. NADALJNJE DELO

Nadaljnje raziskave in razvoj bodo usmerjene v aplikacijo DroidWatch in strežnik. Naslednja poglavja opisujeta predlagane izboljšave za zbiranje dodatnih podatkov (poglavje 7.1) in implementacija Mehanizmi proti nedovoljenimi posegom (poglavje 7.2).

Poglej bodočih izboljšav v DroidWatch, bi integracija aplikacije v MDM rešitev prinesla veliko vrednost Android varnostni skupnosti. Trenutni MDM sistemom manjka nadzor uporabnika, ki bi lahko usmerjal interne preiskovalce. Celotna varnostna rešitev za podjetja, ki bi vseboval stroga pravila, oddaljen dostop do naprave, nadzor nad uporabnikom, bi odpravila pomisleke industrije in države v uporabo Android pametnih telefonov.

### 7.1 Dodatne zbirke podatkov

Forenzika na Android napravah je področje, ki se neprestano razvija z izdajo novih različic operacijskega sistema. Novi podatki in funkcije postanejo dostopni, ki imajo lahko veliko vrednost v preiskavah. Naslednje različice DroidWatch aplikacije bodo vključevale naslednje podatke: nastavitev USB razhroščevanja, ponovni zagonu telefona, dnevniški zapisi glasovne pošte, dodatni dnevniški zapisi aplikacij in jedra OS iz dumpsys, dumpstate, in dmesg.

### 7.2 Mehanizmi proti nedovoljenimi posegom

Podatki zbrani in shranjeni na telefonu trenutno zanašajo na Andrid varnostnem modelu, da bi preprečili zlorabe iz strani aplikacij in uporabnika.

Da bi zmanjšali ta tveganja se predlaga sledeče:

- Šifriranje dogodkov baz podatkov (z kontrolno vsoto).
- Visoka prioriteta vrednosti filtra sporočil.
- Periodično beleženje delovanja aplikacije.
- Zbirke, ki temeljijo na dogodkih in prenosih.
- Zgoščevanje podatkov podatkovne baze.

Šifriranje dogodkov v DroidWatch podatkovni bazi je mehanizem, ki bi odvrnil uporabnike od pregledovanja in spreminjanja zbranih dogodkov. Vsakemu dogodku bi lahko bil lahko dodelili kontrolno vsoto in ga šifrirali z javnim ključem (zasebni ključ bi bil shranjen na strežniku).

Navesti najvišjo vrednost filtra sporočil v AndroidManifest bi morda lahko preprečilo ostalim aplikacijam zatirati sporočila sistemskih aplikacij. Dodatno je potrebno raziskati kaj se zgodi če imata dve aplikaciji nastavljeno maksimalno vrednost. S periodično beleženjem delovanja bi lahko zaznali motnje, ki se bi izražali v manjkajočih zapisih. Z naključnim pošiljanjem podatkov bi se zaščitili pred preprečevanjem pošiljanja. Zadnji mehanizmi proti nedovoljenimi posegom je zgoščevanje podatkov podatkovne baze, ki je trden način odkrivanja nedovoljenih posegov.

## Zaključek

Sistem predstavljen v tej raziskavi služi kot prototip Android skupnost, kako načrtovati in implementirati aplikacijo za nadzor Android naprav za podjetja, brez administratorskih pravic. DroidWatch je prvi takšen odprtokodni sistem, zato je potreben nadaljnji razvoj, ki bi razširili in izboljšal njegove zmožnosti. Dodati bo potrebno mehanizme proti nedovoljenim posegom v delovanje aplikacije. Podatki, ki jih aplikacija lahko zbere, so lahko koristni v preiskavah z zaradi različnih razlogov. Nazadnje pa to delo služi tudi kot vodilo za dostop do nabora podatkov, ki so na voljo prek Android API.

## 8. REFERENCES


[1] J. Grover, "Android forensics: automated data collection and reporting from a mobile device", *DFRWS 2013*, 2013.

[2] Android OS. Dostopno na:www.android.com, 2014.

[3] Mobile device management. Dostopno na:http:/en.wikipedia.orgwikiMobile_device_management, 2014.

[4] DroidWatch. Dostopno na:https://github.com/jgrover/DroidWatch, 2014.

[5] EnCase. Dostopno na:https://www.guidancesoftware.com/products/Pages/encase-forensic/overview.aspx, 2014.

[6] Zenprise. Dostopno na:http://en.wikipedia.org/wiki/Zenprise, 2014.

[7] AirWatch. Dostopno na:http://www.air-watch.com/solutions/mobile-device-management, 2014.

[8] MobileIron. Dostopno na:http://www.mobileiron.com/en, 2014.

[9] GRR Rapid Response. Dostopno na:https://code.google.com/p/grr/, 2014.

[10] AccessData Enterprise. Dostopno na:http://www.accessdata.com/products/digital-forensics/ad-enterprise, 2014.

[11] AFLogical. Dostopno na:https://viaforensics.com/resources/tools/android-forensics-tool/, 2014.

[12] VNC. Dostopno na:http://en.wikipedia.org/wiki/Virtual_Network_Computing, 2014.
[13] SQL. Dostopno na:http://en.wikipedia.org/wiki/SQL, 2014.
[14] SQLite. Dostopno na:http://www.sqlite.org/, 2014.
[15] HTTPS. Dostopno na:http://sl.wikipedia.org/wiki/HTTPS, 2014.
[16] Ubuntu. Dostopno na:http://www.ubuntu.com/, 2014.
[17] Apache. Dostopno na:http://www.apache.org/, 2014.
[18] MySQL. Dostopno na:http://www.mysql.com/, 2014.
[19] Splunk. Dostopno na: http://www.splunk.com/, 2014.
[20] SSL. http://en.wikipedia.org/wiki/SSL, 2014.
[21] GoSMS. https://play.google.com/store/apps/details?id=com.jb.gosms, 2014.
[22] Denial of service attack. http://en.wikipedia.org/wiki/Denial-of-service_attack, 2014.

# Platforma za ocenjevanje forenzičnih orodji za kreiranje slike delovnega pomnilnika

[Povzetek članka] *

Jernej Grosar
Fakulteta za računalništvo in informatiko
Tržaška cesta 25
Ljubljana, Slovenija
grosar.jernej@gmail.com

Andraž Gregorčič
Fakulteta za računalništvo in informatiko
Tržaška cesta 25
Ljubljana, Slovenija
gregorcic.andraz@gmail.com

## POVZETEK

V tem članku bova najprej predstavila faze digitalnega preiskovanja in kakšne so nekatere tehnike za zajem pomnilniške slike. Nato sledi povzetek članka avtorjev Stefan Vomel in Johannes Stuttgen, v katerem sta opisala tehniko zajema pomnilniške slike z uporabo treh različnih aplikacij, katerih testiranja sat izvajala na emulatorju Bochs x86. Za konec pa vam bova predstavila še nekaj programskih in strojnih rešitev, ki se uporabljajo na tem področju.

## Kategorije in opis teme

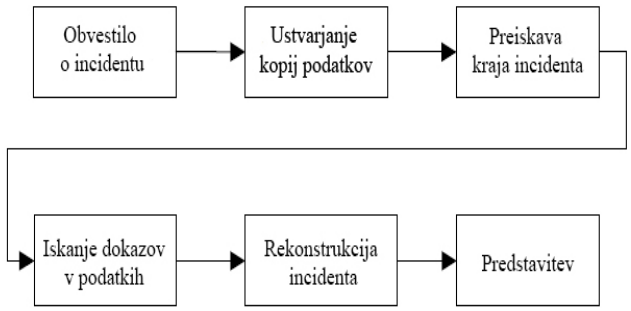
Digitalna forenzika **Zajem pomnilniške slike**

## Ključne vesede

Zajem pomnilniške slike, Forenzika pomnilnika, Digitalno preiskovanje, Pravilnost pomnilniške slike, Atomarnost pomnilniške slike, Itegriteta pomnilniške slike

## 1. FAZE DIGITALNEGA PREISKOVANJA

Če želimo pri digitalni forenziki pravilno pridobiti podatke moramo slediti fazam digitalnega preiskovanja. V digitalni forenziki obstaja več načinov kako pravilno pridobiti podatke. V nadaljevanju bova opisala faze, ki sta jih predstavila Carrier in Spafford. Faze si sledijo kot je prikazano na sliki 1.




**Slika 1: Faze digitalnega preiskovanja**

To so: *obvestilo o incidentu* pri kateri sestavimo ekipo za preiskovanje incidenta, potem ko le tega zaznamo. Ko je ekipa sestavljena in je ta preverila incident sledi *faza ustvarjenja kopij podatkov*, pri tem pa moramo paziti, da se podatki čim manj spreminjajo. Preiskovalci morajo ustvariti čim bolj natančno kopijo digitalnega kraja zločina, da ga lahko kasneje analizirajo v laboratoriju. Ko so kopije ustvarjene, preiskovalci *preiščejo kraj incidenta*, da vidijo če so kakšni očitni dokazi. Ko imajo nekaj tez o poteku digitalnega zločina, ki so jih sestavili iz dokazov, se prične natančnejše *iskanje dokazov v podatkih*, da lahko na podlagi teh potrdijo oziroma zavržejo teze. Potem ko so pridobili podatke v fazah preiskovanja kraja incidenta in iskanja dokazov v podatkih, se prične faza *rekonstrukcije incidenta*, kjer se pridobljeni dokazi in teze testirajo in oblikujejo v končno zgodbo o poteku zločina (v nekaterih primerih se dokazi iščejo tudi v tej fazi). Zadnja faza preiskave pa je *predstavitev teorije o poteku zločina* odgovornim osebam.

## 2. TEHNIKE ZAJEMA SLIK

Za zajem slik delovnega pomnilnika poznamo več tehnik, nekatere iz med njih bova predstavila v nadaljevanju.

### 2.1 Crash dumps

Crash dump vsebuje podatke, ki so bili v pomnilniku ob času, ko se je le ta zgodil (po navadi, ko se program nepravilno zaustavi). Poleg podatkov iz pomnilnika se po navadi shranijo tudi podatki iz registrov, ki vsebujejo programski števec, kazalec na sklad, informacije o upravljanju pomnilnika in druge zastavice procesorja ali operacijskega sistema. Crash dump se najpogosteje uporablja pri iskanju napak v programih lahko pa se uporabi tudi pri forenzični preiskavi.

Na moderni Unix operacijskih sistemih lahko pregledujemo zajete podatke z uporabo GNU Debugger (gdb) in objdump, ki uporabljata knjižnico GNU Binutils Binary File Descriptor (BFD). Ta knjižnica prikaže samo neobdelane podatke in ne zna rekonstruirati podatkov v bolj berljivo obliko, za to mora poskrbeti aplikacija, ki uporablja to knjižnico.

Če hočemo ročno prožiti crash dump moramo uporabljati tipkovnico z priklopom PS/2, popraviti pa je potrebno tudi zapis v registru

*[HKEY_LOCAL_MACHINESYSTEM\CurrentControlSet \Services\i8042prt\Parameters]*

na vrednost *"CrashOnCtrlScroll"=dword:00000001*

nato pa držimo desno tipko Ctrl in dvakrat pritisnimo tipko ScrollLock. To povzroči, da gonilnik za tipkovnico generira *STOP-Error 0x000000E2*, kar prisili operacijski sistem v izvedbo crash dump. Crash dump podatki se shranijo v swap datoteko, ki mora biti najmanj 1MB večja kot fizični pomnilnik in se mora nahajati na sistemski particiji diska. Med ponovni zagonom Windows zazna, da so bili podatki zapisani v to datoteko in jo zaklene, da prepreči kakršenkoli dostop. Nato swapdump.exe kopira podatke iz swap datoteke v navadno datoteko, katero lahko kasneje pregledujemo.

### 2.2 LiveKd dumps

LiveKD dump je program, ki omogoča kreiranje slike pomnilnika in jo shrani v crash dump formatu. Z razliko od crash dump-a ne zaustavi delovanje sistema, kar je iz vidika digitalne forenzike slabše, saj se lahko podatki v pomnilniku med zajemom spremenijo in pride pri preiskavi do napačne interpretacije.

### 2.3 Hibernacija

Hibernacija je način pri katerem operacijski sistem pri prehodu v hibernacijo shrani vsebino pomnilnika na disk, pri ponovnem zagonu pa to vsebino, ki je na disku prenese nazaj v pomnilnik. Slaba lastnost te metode je, da se lahko podatki vmes spremenijo. To vsebino lahko z diska pridobimo in jo lahko uporabimo za forenzične namene. Hibernacija je podrta v vseh operacijskih sistemih.

### 2.4 Firewire

Z firewire ali IEEE1394 napravami je mogoč direkten dostop do fizičnega pomnilnika. Te naprave dostopajo do glavnega pomnilnika preko DMA kontrolorja in pri tem ne uporabljajo glavnega procesorja in operacijskega sistema, kar pomeni, da se podatki med prenosom ne spreminjajo. Zaradi te lastnosti je mogoč zajem pomnilniške slike za namene pri preiskavi v digitalni forenziki, vendar ta način še ni dovolj varen.

### 2.5 "Mrzel in topel" zagon

Tehnika "toplega" zagona pomeni, da ponovno zaženemo računalnik brez izklopa napajanja tako, da pomnilnik ostane pod napetostjo. Nato z uporabo programov msramdump ali afterlife, ki jih lahko zaženemo iz USB ključa oziroma CD-ja, shranimo podatke iz pomnilnika na drugo napravo.

Pri tehniki "mrzlega" zagona uporabimo hladilni sprej, kateri mora biti posebej namenjen za uporabo na elektronskih komponentah. Z njim ohladimo pomnilnik, ko računalnik še deluje na temperaturo pod lediščem (tem nižja je temperatura boljše je). To omogoči, da se podatki v pomnilniku počasneje izgubijo, ko mu odklopimo napajanje. Ko pomnilnik ohladimo dovolj, sklopimo napajanje ter pomnilnik prenesemo v drug računalnik, kjer lahko zajamemo podatke. Pri zajemu si lahko pomagamo z enakimi orodji kot pri "toplem" zagonu.

### 2.6 Izdelava slike navideznega računalnika (virtual machine)

Navidezni računalnik (npr.: xen, qemu ,vmware, bochs) je program, ki simulira delovanje računalnika pri tem pa ga operacijski sistem, kateri teče v njem vidi kot običajen računalnik. Ta program ima dve možnosti, zajema podatkov iz pomnilnika navideznega računalnika. Prva možnost je da program ustavimo z možnostjo "pause/suspend/stop", kar povzroči da se shrani vsebina pomnilnika v pomnilniško sliko. Druga možnost, ki jih ti programi podpirajo pa je, da lahko shranimo vsebino pomnilnika navideznega računalnika tudi med delovanjem.

## 3. SPLOŠNO O čLANKU

Pridobivanje podatkov iz delovnega pomnilnika računalnika in njihovo analiziranje postaja v digitalni forenziki vedno bolj pomembno in se je v zadnjih letih zelo razvilo. K razvoju te tehnike je pripomogla potreba po pridobitvi ustrezne alternative klasični tehniki pregledovanja diskov (podatkovno orientirani tehniki) s katero je vse težje pridobiti potrebne informacije. Ker postajajo diski vse večji, hitrost podatkovnega vodila pa ostaja enaka, se čas preiskave povečuje, poleg tega pa postane preiskava nemogoča, če osumljenec uporablja šifriranje celotnega diska in ni pripravljen povedati gesla. Ta tehnika je vse bolj pomembna tudi zato, ker se vse več zlonamernih programov izvaja samo v pomnilniku in ne puščajo nobene sledi na disku. Posledica tega je, da lahko na koncu pridemo do napačnega zaključka.

Do sedaj je bilo na področju forenzične preiskave pomnilnika opravljenega že veliko dela, vendar pa sta tehniki pridobivanja podatkov iz pomnilnika in kreiranje slike pomnilnika slabše raziskani (predvsem kvaliteta in zmogljivost teh tehnik). V tem članku sta avtorja predstavila arhitekturo platforme, katera pomaga določiti stopnjo pravilnosti, atomarnosti in integritete pripomočkov za zajem podatkov v pomnilniku, ter nam pomaga odgovoriti na naslednja vprašanja:

- Ali pripomoček za kreiranje pomnilniške slike ustvari sliko, ki je enake velikosti kot fizični naslovni prostor?
- Ali pomnilniška slika vsebuje podatek, ki je bil v pomnilniški strani v času kreiranja slike te strani?
- Kako ti pripomočki obravnavajo napake in dele pomnilnika, ki niso dosegljivi?
- Kakšen je učinek teh na podatke v pomnilniku in koliko se jih spremenjeni, ko se to orodje naloži v pomnilnik?
- Koliko pomnilnika se spremeni v času kreiranja pomnilniške slike?

Avtorja sta za testiranje orodji uporabila Bochs x86 PC emulator, katerega sta priredila za njune potrebe. Prav tako sta pri testiranju posameznega orodja uporabila pristop "bele škatle" in sta preuredila izvorno kodo testiranega programa njunim potrebam. Dodala sta več klicev (hypercall), kateri so opozorili platformo na pomemben sistemske dogodke in operacije, kot so pričetek in konec kreiranja slike podatkov ali pa ko je prišlo do podvojitve pomnilniške strani. Ko so se sprožili ti klici se je shranilo trenutno stanje sistema, katerega sta kasneje primerjala z narejeno pomnilniško sliko in sta tako lahko ocenila učinkovitost orodja. Pri testiranju sta uporabila orodja win32dd (Suiche, 2009b), WinPMEM (Cohen, 2012) in mdd (ManTech CSI, Inc., 2009).

# 4. KRITERIJI ZA CELOVITOST SLIKE POMNILNIKA

## 4.1 Pravilnost (podatkov shranjenih v sliki pomnilnika)

Forenzična slika pomnilnika je pravilna glede na niz pomnilniških področji R $\subseteq \mathcal{R}$, če je "vrednost slike za vsa ta področja enaka vrednosti v pomnilniku ob času, ko je bila slika narejena" [1], stopnja pravilnosti pa je odstotek pomnilniških področji, ki so bila pravilno zajeta. Na prvi pogled se zdi, da to ni težko doseči, vendar lahko zlonamerni programi poizkušajo manipulirati ali ovirati operacijo kreiranja pomnilniške slike, programi za zajem slike pa lahko vsebujejo programske napake in ne preslikujejo pravilno fizičnega naslova v slikovno datoteko in tako povzročajo kasnejše napačne razlage podatkov.

## 4.2 Atomarnost

Poleg tega, da mora biti forenzična slika pomnilnika pravilna, nanjo ne smejo vplivati aktivnosti, ki se hkrati izvajajo. To pomeni, da mora slika vsebovati vse podatke, ki so bili v pomnilniku ob določenem točki v času, ne prej ne potem. Seveda pa to s programskimi orodji za zajem slike pomnilnika ni mogoče doseči, ker se izvajanje sistema ne da "zamrzniti". Stopnja atomarnosti je odstotek pomnilniških regij, katere ustrezajo tej definicij.

## 4.3 Integriteta

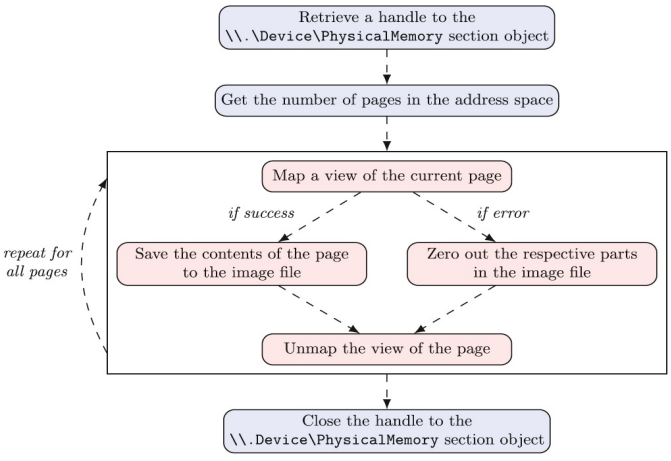
S pomočjo tega kriterija je mogoče opazovati stanje pomnilnika skozi določen časovni interval. Da pomnilniška slika zadostuje integriteti v določeni točki v času T pomeni, "da se vrednosti pomnilniških področji, katera so bila zapisana v sliko niso spremenila po času T" [1]. Točka T predstavlja čas na začetku postopka preiskave, ko pomnilnik še ni bil spremenjen zaradi kreiranja pomnilniške slike. Če to stanje pomnilnika ostane nespremenjeno do takrat, ko je shranjeno v pomnilniško sliko pomeni, da se je integriteta ohranila. Vpliv programa za zajem pomnilniške slike je zanemarljiv v primerjavi s spremembami, ki se zgodijo tudi ko sistem miruje. Stopnja integritete je odstotek pomnilniških lokacij, katere so ostale nespremenjene po času T.

# 5. METODE MERJENJA IN ARHITEKTURA PLATFORME

Avtorja sta za ta test uporabila odprtokodne programe, katere sta preuredila zato, da sta lahko kasneje ocenila pravilnost, atomarnost in integriteto dobljene slike pomnilnika.

## 5.1 Metode zajema forenzične slike pomnilnika na Microsoftovem operacijskem sistemu

Običajna tehnika pridobivanja forenzične kopije pomnilnika na operacijskem sistemu Microsoft Windows uporablja objekt \\.\Device\PhysicalMemory , kateri zagotavlja dostop do delov pomnilnika, vendar pa ta način dostopa do pomnilnika v uporabniškem prostoru ni več mogoč na sistemih novejših kot Microsoft Windows Server 2003 Service Pack 1. Zato morajo forenzični programi znati spremeniti uporabniške pravice, prav tako pa morajo vsebovati tudi gonilnike, kateri se namestijo v jedro OS-a. Ti gonilniki lahko s pomočjo funkcije ZwOpenSection dostopajo do pomnilnika in berejo iz njega, ter prebrane pomnilniške strani neposredno zapišejo v datoteko že v jedru ali pa jih preko predpomnilnika prenesejo v uporabniški prostor za nadaljnjo obdelavo. Kot je prikazano na sliki 2, je algoritem za zajem pomnilniške slike zelo enostaven, vendar pa se lahko pojavijo napake, če velikost fizičnega naslovnega prostora ni pravilno izračunana. Ta metoda zajema pomnilniške slike ni preveč varna, saj so njene operacije z lahkoto blokirane z uporabo preprostih antiforenzičnih metod. Da se temu izognemo, moramo uporabiti alternativne metode ali pa kličemo rutino MmMapIOSpace jedra operacijskega sistema.




**Slika 2: Algoritem zajema pomnilniške slike**

## 5.2 Arhitektura platforme uporabljene s strani avtorjev članka

Testna platforma je narejena s pomočjo odprtokodnega x86 PC emulatorja Bochs. Njegova največja prednost pred ostalimi podobnimi emulatorji kot je QEMU je ta, da ima okoli 50% krajšo izvorno kodo ( 250 000 vrstic kode). Avtorja sta napisala večino program v programskem jeziku C del, kateri pa vpliva na hitrost delovanja pa je napisan v asemblerju.

# 6. MERITVENI FAKTORJI ZA POPOLNO POMNILNIŠKO SLIKO

## 6.1 Pravilnost

Da bi avtorja ugotovila pravilnost zajete pomnilniške slike katera je bila zajeta s pomočjo forenzičnega programa, sta poleg le te ustvarila tudi sliko pomnilnika gostiteljskega operacijskega sistema kot jo vidi Bochs emulator. S pomočjo te metode sta lahko tudi ugotovila ali se velikost zajete pomnilniške slike ujema z velikostjo fizičnega naslovnega prostora in tudi ali se podatki na zajeti pomnilniški sliki ujemajo s podatki, ki so bili na pomnilniku ob času kreiranja slike.

## 6.2 Atomarnost

Zaradi zmogljivostnih razlogov sta avtorja izbrala indirekten pristop za merjenje stopnje kršitev atomarnosti. Kršitev se zgodi takrat, ko nit, katera se izvaja dostopa do dela pomnilnika, ki je že bil zajet v pomnilniško sliko in spremeni njegovo vsebino, zaradi te spremembe pa se mora spremeniti tudi vsebina pomnilnika, ki še ni bila zajeta. V tem primeru pomnilniška slika postane nekonsistentna. V praksi je zelo težko preveriti ali je sprememba enega dela pomnilnika vplivala tudi na spremembo drugega dela, ker bi to pomenilo, da bi morali spremljati dogajanje celotnega sistema. Zato sta merila samo potencialne kršitve atomarnosti, za katere se ne ve ali vodijo do pomembnih sprememb. Za merjenje tega sta uporabila *bx_instr_lin_access* funkcijo, ki spremlja operacije nad pomnilnikom. Ko se je pričel proces zajemanja slike, sta gradila seznam vseh strani, do katerih so programi dostopali in so bile že prej zajete v pomnilniško sliko (preden je program dostopal do nje). Poleg tega pa sta shranila tudi številko procesa, ki je dostopal do tega dela pomnilnika. Če je kateri od procesov spreminjal podatke na delu pomnilnika, kateri še ni bil shranjen, sta preverila ali je mogoče ta proces spreminjal tudi vsebino pomnilnika na delu, kateri je že bil zajet v pomnilniško sliko in če je, sta to zabeležila. S pomočjo teh zabeležk sta lahko ocenila zgornjo mejo nekonsistence pomnilniške slike.

## 6.3 Integriteta

Podobno kot pri procesu merjenja pravilnosti zajema pomnilniške slike, sta določila stopnjo integritete tako, da sta naredil več kopij pomnilnika ob različnem času izvajanja zajema pomnilniške slike (preden se je naložil program za zajem pomnilniške slike v pomnilnik, na koncu procesa zajema slike pomnilnika in vsakič ko se je prožil hypercall). Tako sta lahko kasneje primerjala koliko podatkov je bilo spremenjenih.

# 7. PREUčEVANJE

## 7.1 Preučevanje metodologije

V članku avtorja preučujeta zmogljivost in kvaliteto treh popularnih orodji za pregledovanje pomnilnika, ki se imenujejo win32dd(Suiich, 2009b), WinPMEM(Cohen, 2012), in mdd(ManTech CSI, INc., 2009). Medtem ko so bile vse aplikacije prvotno odprto kodne, win32dd je bil zamenjan z zaprto kodno različico in je razpoložljiv samo v programu imenovan MoonSols Windows Memory Toolkit. Zaradi tega sta lahko testirala samo zadnjo verzijo tega programa.

Vsi eksperimenti so potekali na standardnem računalniškem sistemu s procesorjem Intel i5-650 in 8GB delovnega pomnilnika. Edini dodatek je bil SSD disk, ki je bil uporabljen zaradi povečanja zmogljivosti binarnega zapisovalnika. Za platformo sta uporabila maksimalno 2GB fizičnega pomnilnika. Za emulator je bil uporabljen spomin med 512MB in 2GB. Zadnja vrednost predstavlja maksimalno vrednost spomina, ki ga je podpiral emulator Bochs v času pisanja članka, dejstvo, ki na žalost ni bilo dokumentirano v ustreznem uporabniškem priročniku in sta ga avtorja odkrila šele ko sta dokončala razvoj platforme. Za glavni operacijski sistem je bila nameščena prevzeta namestitev Microsoft Windows XP (SP3). Za najboljšo možnost simuliranja računalnika sistemske storitve niso bile prilagojene. Vsi začetni testi so bili pognani iz zagonskega procesa operacijskega sistema in so bili popolnoma dokončani.

Za vse velikosti pomnilnika, ki sta jih avtorja uporabila, sta izdelala 30 sistemskih slik za vsako velikost. Takšna slika je bila ustvarjena z začasno ustavitvijo gostujočega sistema. Ta funkcija je možna z uporabo grafičnega vmesnika emulatorja Bochs, ki dovoljuje shranjevanje trenutnega stanja procesov, pomnilnika in priključenih naprav na trdi disk. Če se simulacija ni izvedla v celoti in se je nadaljevala kasneje je zaradi programskih napak v emulatorju prišlo do nedelovanja uporabniških vhodov in izhodov. Iz tega razloga so bile vse aktivnosti povezane s sistemskimi slikami avtomatizirane brez posredovanja uporabnika.

Ob zaključku izvajajočega testa sta sliko in simulacijo povrnila v začetno stanje za preučevanje ostalih sistemskih slik. Rezultati uspešnosti različnih produktov z ozirom na določeno sliko so tako direktno primerljivi. V celoti sta izpeljala 270 eksperimentov (90 testov na posamično velikost pomnilnika) za tri orodja za pregledovanje pomnilnika. Za kreiranje log datoteke, je glede na razpoložljivo kapaciteto pomnilnika, potrebno od 4,37GB do 12,37GB prostega prostora na trdem disku. Glede na to, da sta ustvarila zunanjo in notranjo sliko(skupno 5) sta potrebovala za vsak eksperiment približno 6,87 GB – 22,37GB prostora na trdem disku.

## 7.2 Rezultati

### *7.2.1 Pravilnosti*

Fizični naslovni prostor računalnika si delijo operacijski sistem, aplikacije in strojne naprave, ki so potrebne pri operacijah preslikav pomnilnika (MMIO) in moramo pri izdelavi forenzične slike, vzeti v obzir tudi ta pomnilniški prostor. Priporočljiva metoda za prvo opravilo je klic nedokumentirane funkcije *MmGetPhysicalMemoryRanges*, ki prikaže strukturo pomnilnika kot je viden v Windows-ih. V testih, sta obe orodji – win32dd in mdd v začetku določali napačno velikost

naslovnega prostora. Na primer: mdd računa velikost pomnilnika na osnovi izhoda funkcije
*Global-MemoryStatusEx*, ki na žalost ne vključuje informacije glede MMIO prostora. Posledično, ko le ta pregleduje naslovni prostor, so pomnilniška področja prezrta in se zato ustvari manjša slika. Kar je še slabše, ker je ta slika zapisana zaporedno namesto smiselno, je posledično dostop do pomnilniškega prostora poškodovan. Pri analiziranju postopka, ki se nanaša na razlagi zamikov, lahko to pomeni pomemben problem v zaključni fazi raziskave. Z obzirom na orodje win32dd, je bila napaka kasneje popravljena ko so izdali zaprto kodno različico. Javno dostopna verzija orodja mdd pa ime še vedno napake. Z namenom, da sta ustrezno testirala obe orodji sta razvila popravke s katerimi sta uspela ustvariti slike, ki točno ustrezajo velikosti pomnilnika. Ena iz med posebnosti emulotorja Bochs je da je zgornji del gostujočega fizičnega naslovnega prostora vedno skrajšan za 16 strani (65,536 bytes). Tako na primer, za računalnik z 512MB pomnilniškega prostora celotno število strani ustreza 131,056 namesto 131,072. Avtorja članka verjameta, da je to lahko zaradi internih nastavitev, tudi pogovor z razvijalci emulatorja Bochs ni pripomoglo k rešitvi tega problema.

Rezultati njihovih preizkusov so predstavljeni v tabeli 1 v prilogi. Kot je razvidno vsa orodja za izdelavo slik dokazano dosežejo 99.3% pravilnost. To pomeni, da je rešitev sposobna pregledovanja vsebine pomnilnika.
Napake niso pokazatelj samo za pomnilniški prostor strojnih naprav, kot je bilo pričakovano, toda tudi za različne strani operacijskega sistema. Točni vzrok za takšno obnašanje, bo potrebno še podrobneje preučiti.

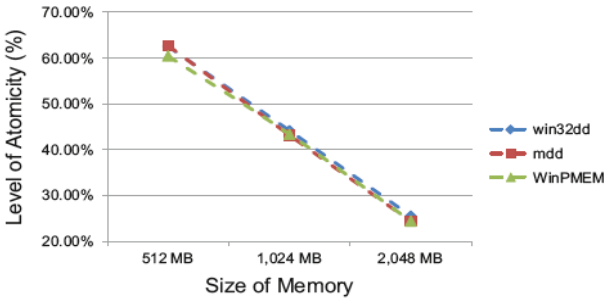
### 7.2.2 *Atomarnost*

V nunih poskusih se stopnja potencialnih atomarnih kršitev hitro povečuje z velikostjo nameščenega pomnilnika (glej tabelo 2 v prilogi). Medtem ko sta spremljala sliko na računalniku z 512 MB delovnega pomnilnika sta zaznala med 37,29% in 39,55% neskladji na strani, na računalniku z 2GB delovnega pomnilnika pa je število kršitev poskočilo na 75%. Prepričana sta, da je čas tisti, ki vpliva na strmo rast, seveda ob upoštevanju da je sistem vedno enako obremenjen. Ker je potreben daljši čas za končanje izdelave slike na računalniku z večjo velikostjo pomnilnika imajo tudi hkratno delujoče aplikacije daljšo obdobje za dostop do področji, ki so že bila podvojena. Na kratko z daljšim obdobjem izdelovanja slik, je vedno težje obdržati slikovno datoteko brez ”madeža”(glej tudi sliko 3).

V času tega pisanja je še vedno ne rešen problem v katerih primerih in v kolikšni meri hkratne aktivnosti med procesom izdelave slik v resnici vplivajo na rezultat kasnejših analiz. Po drugi strani, visoka stopnja ne atomarnosti je v nasprotju s klasičnem dojemanjem forenzične zanesljivosti. Vsa neskladja, ki so navedena v prejšnjem odstavku bi lahko bila sporna v primeru morebitne obravnave na sodišču.

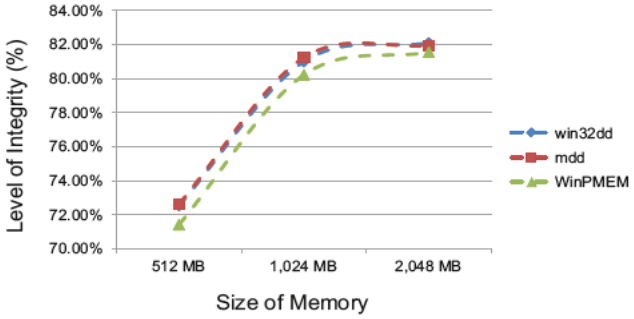
### 7.2.3 *Integriteta*

V nasprotju s stopnjo atomarnosti, se stopnja izdelanih slik zadovoljivo poveča z naraščajočo velikostjo razpoložljivega pomnilnika. Na računalniku z 512 MB delovnega pomnilnika je bilo v času izdelave slik v povprečju spremenjenih med 27,37% in 28,587% podatkov. V grobem je bilo prizadetih približno 47% - 49% strani. V nasprotju s tem, je bilo




**Slika 3: Stopnja atomarnosti dosežena pri posameznem orodju**

na sistemu z 2GB delovnega pomnilnika, spremenjenih le med 17,905% in 18,447% podatkov, čeprav število prizadetih strani neznatno narašča v večini primerov. Ta razvoj je bil pričakovan, ker na sistem s konstantno obremenitvijo, vendar pri višji pomnilniški kapaciteti, je potrebna sorazmerna manjša količina prostora za operacijski sistem in za aplikacije v delovanju. Rezultati so podobni tudi, če jih primerjamo z rezultati prejšnjih poskusov drugih avtorjev pri uporabi alternativnih orodji.




**Slika 4: Stopnja integritete dosežena pri posameznem orodju**

# 8. RAZPRAVA

## 8.1 Testiranje black-box (črna škatla) proti white-box (bela škatla)

Kot sta avtorja že razložila, temelji njihovo preučevanje platforme na pristopu bele škatle, to pomeni, da sta morala nekoliko prilagoditi izvorno kodo programa za izdelavo slik in vključiti več klicev, ki pokažejo določene dogodke kot so začetek in konec izdelave slik. Na žalost pa je veliko rešitev zaprto kodnih in se komercialno tržijo ( Memoryze (Mandiant, 2011), Fast Dump Pro (HBGary, 2013) ali FTK Imager (ACCESSData, 2012)). Z uporabo teh aplikacij, sta si najprej prizadevala za testiranje na pristopu črne škatle - to je merjenje kvalitete orodja navkljub pomanjkanju določenega znanja o njegovem internem načinu delovanja. Za ta namen sta uporabila tehniko zamenjave obstoječe kode z novo zato, da sta lahko razdelila kodo na več manjših v ustrezne binarne datoteke. Ti deli kode vsebujejo navodila za sprožitev klica in začetek komunikacije z vmesnikom (instrumentation interface). Vendar ta metoda ni zanesljiva

iz več razlogov. Prvič, ni zagotovljeno, da bo v vsakem primeru najdeno dovolj veliko območje nezasedenega prostora za navodila klicev. Drugič, z namenom, da bi določili natančne meritve, je potrebno vstaviti dva klica po, možnosti v bližini dela programa, ki izvaja izdelavo slik. Ta način bi zahtevalo obratni inženiring programa, proces, ki je lahko precej zapleten in je po navadi zakonsko prepovedan.

Poskušala sta tudi zmanjšati število klicev in prepoznati proces izdelave slike, ki temelji zgolj na vzorcu pomnilniškega dostopa. Vendar se v praksi ta postopek ni pokazal koristen. Avtorja sta mnenja, da je izvajanje testiranja na pristopu bele škatle najbolj primerno, čeprav sta lahko ocenjevala le manjše število izdelkov, ki so bili na voljo do takrat.

## 8.2 Omejitve platforme

V času, ko sta avtorja pisala ta članek sta se morala še vedno ukvarjati s številnimi slabostmi in omejitvami platforme. Zaradi nekaterih notranjih programskih struktur, je platforma žal sposobna poganjati samo 32-bitne aplikacije, poleg tega pa Bochs emulator ne omogoča dodelitve pomnilnika večji kot 2GB sistemu, kateri teče na njemu. To je pomembna slabost, ker so delovni pomnilniki v modernih računalnikih pogosto večji in bi bilo posebej zanimivo oceniti izdelavo slik v primeru meje na 4GB.

Kot smo že poudarili, medtem ko lahko izmerimo pravilnost pomnilniške slike dokaj natančno, smo sposobni le določiti minimalno stopnjo atomarnosti slike. Prav tako lahko le grobo ocenimo vpliv programa za izdelavo slik s primerjanjem stanja pomnilnika v času s stanjem pomnilnika tik pred začetkom operacije za izdelavo slik.

## 8.3 Operativne zmogljivosti orodji za izdelavo slik delovnega pomnilnika

V izvedenih poskusih so vsi testi orodji za izdelavo slik delovnega pomnilnika pokazali podobno zmogljivost. Posledično lahko forenzični preiskovalci izberejo izdelek, ki jim najbolj ustreza za njihovo delo. Če predpostavimo, da prevzeta tehnika za izdelavo neobdelane slike preko \\.\Device\PhysicalMemory ni okužena z zlonamerno programsko opremo, potem ne potrebujemo bolj sofisticiranih metod za izdelavo slik kot je funkcija MmMapIOSpac. Po drugi strani, pa je v večini primerov potrebno upoštevati potencialno visoko stopnjo kršitev atomarnosti in integritete.

Na računalniških sistemih, kateri tečejo v nadzorovanem okolju, lahko namesto uporabe prej omenjenih metod uporabimo kar vgrajeno funkcijo *CrashOnCtrlScroll*. Ta funkcija dovoljuje shranjevanje stanja pomnilnika, kakor tudi pomembne procesorske registre na disk ob pritisku bližnjice na tipkovnici. Vpliv na stopnjo atomarnosti je nižji, ker so sistemske prekinitve odkrite še pred izdelavo slike. Ta funkcija je privzeto onemogočena, zato jo potrebno najprej ročno aktivirati. Ker imajo gonilniki možnost sprejemati obvestila o bližnjih operacijah za izdelavo slik preko metod povratnih klicev, kot je *KeRegisterBugCheckCallback*, se lahko zlonamerni programi odstranijo iz pomnilnika, preden se prične zajem pomnilniške slike, zato je zelo pomembno, da preiskovalci pretehtajo katero metodo je bolje uporabiti.

Wang [2] ponazarja pristop za zanesljivo ustvarjanje pomnilniške slike z uporabo sistemskega načina upravljanja procesorja (SMM). V tem načinu je izvajanje operacijskega sistema začasno ustavljeno, medtem pa je možno preučiti stanje sistema. V nasprotju sta Martignoni [3] in Yu [4] predstavila koncept izdelave slik, ki temelji na osnovi virtualizaciske strojne opreme. Kolikor vemo, so zadnje raziskave bolj akademske narave in še niso bile obsežno testirane v praksi.

## 8.4 Možnosti za nadaljnje raziskave

Vsi testi avtorjev so bili začeti v stanju mirovanja sistema. Za obširnejše ocenjevanje kvalitete orodji za izdelavo pomnilniških slik je potrebno simulirati različne sistemske obremenitve in še posebno spremljati spremembo stopnje atomatnosti in integritete. Poskus bi bilo potrebno izvajati na različnih operacijskih sistemih tako na strežnikih in na namiznih, kakor tudi na sistemih, ki so bili namerno okuženi z zlonamerno programsko opremo.

Uporaba Bochs emulatorja kot osnova za njihovo rešitev se avtorjema ne zdi več tako optimalna kot na začetku. Čeprav jima je uspelo izdelati osnovni vmesnik precej hitro, je bilo za prilagajanje emulatorja njihovim potrebam in izdelave delujočega prototipa platforme potrebno veliko napora. To je predvsem posledica slabe dokumentacija, ki je v večini nepopolna in ne zajema bolj kompleksnih tehničnih konceptov. Poleg tega je bila berljivost izvorne kode zelo otežena zaradi velikega števila makro programov. Iz teh razlogov sta morala veliko originalnih pristopov za izvedbo testov pri izdelovanju slik delovnega pomnilnika zasnovati na novo. Za boljše kasnejše vzdrževanje priporočata kopiranje platforme v bolj stabilno in zrelo okolje kot je QEMU.

Sočasne aktivnosti v teku izdelovanja pomnilniške slike pomembno vplivajo na njeno stopnjo atomarnosti in integritete. Medtem, ko so kršitve teh dejavnikov v neposrednem nasprotju s klasičnem dojemanjem "forenzične zanesljivosti", pa so dejanske posledice teh kršitev z oziroma na nadaljnje preiskave še vedno v glavnem nejasne. Potrebno je še odkriti v katerih primerih in kakšna stopnja neskladnosti nas pripelje do različnih rezultatov. Iskanje odgovorov na ta vprašanja bo zanimivo področje raziskav v bližnji prihodnosti.

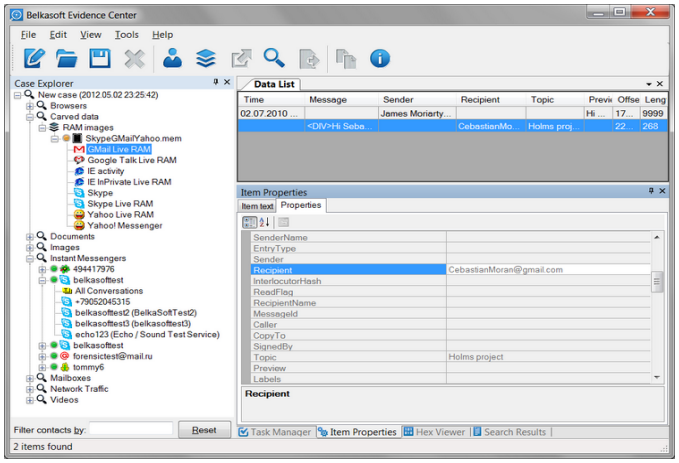
# 9. DRUGE REŠITVE

Za zajem pomnilniške slike imamo na izbiro, tako programske kot tudi strojne rešitve, katere vam jih bova v nadaljevanju predstavila.

## 9.1 Programske rešitve

### *9.1.1 Windows*

- **Belkasoft live RAM capture**
  Ta program je brezplačen za prenos ter zasede malo prostora, kar je pri zajemu pomnilnika zelo pomembno, deluje pa na operacijskih sistemih Windows in ga lahko zaženemo kar iz USB ključa. Program deluje v "kernel-mode", kar mu omogoča, da obide zaščito "anti-debugging" in "anti-dumping", katere uporabljajo moderne aplikacije kot so spletne igre in sistemi za nadzor vdora. Uporaba tega načina prinaša boljše rezultate, kot uporaba programov, ki delujejo v "user-mode".




**Slika 5: Orodje Belkasoft live RAM capture**

- **HBGary**
  Program je na voljo v dveh različicah – brezplačni in plačljivi, namenjen pa je uporabi na operacijskih sistemih Windows. Brezplačna različica je nekoliko okrnjena. Prav tako, kot prejšnji tudi ta ne zasede veliko prostora.

- **FTK Imager** Program je brezplačen in omogoča, da izberemo katere podatke želimo zajeti. S programom lahko poleg zajema pomnilniške slike naredimo tudi sliko diska ali drugih pomnilniških medijev, pregledamo vsebine na različnih pomnilniških medijih, priklopimo slike v načinu samo za branje (tega z uporabo vgrajenih orodji operacijskega sistema Windows ne moremo narediti) ali pa izdelamo digitalni izvleček datotek (hash).

### 9.1.2 *Linux*

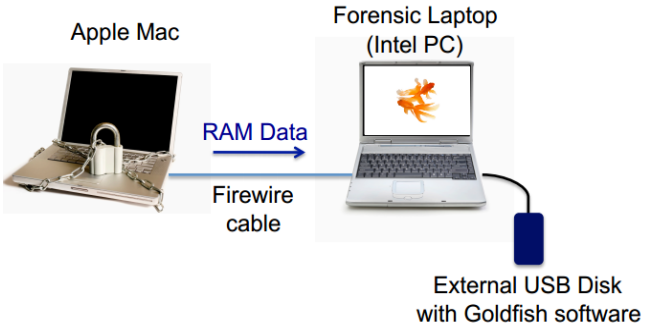
- **/dev/mem**
  /dev/mem je znakovna datoteka, ki je slika glavnega pomnilnika. Naslovi v njej so interpretirani kot fizični naslov pomnilnika. Na starejših Linux sistemih lahko pridobimo celotno vsebino te datoteke s programom dd, medtem ko je mogoče na novejših sistemih pridobiti samo del te vsebine.

- **/dev/crash**
  Na Red Hat sistemih in podobnih distribucijah lahko poženemo ”crash driver”, ki ustvari napravo s katero omogoča neobdelan dostop do fizičnega pomnilnika preko ukaza ”modprobe crash”. Z nekaj popravki lahko ta način uporabimo tudi na drugih Linux sistemih, vendar dostop do celotne slike, ki jo ustvarimo, ni varen. Dostopamo lahko samo do naslovov označenih s ”System RAM”, ki so shranjeni v /proc/iomem.

- **Second Look: Linux Memory Forensics** Program je komercialno forenzično orodje, ki že vsebuje ”crash driver” in skripto za varno kreiranje pomnilniške slike z uporabo originalnega ali spremenjenega gonilnika na vseh Linux sistemih.

### 9.1.3 *Mac OS X*

- **Goldfish**
  Goldfish je Linux orodje, ki temelji na Debian live distribuciji, katero poženemo z USB ključa. Da lahko uporabimo to orodje potrebujemo: USB ključ v velikosti najmanj 8GB, računalnik, kateri ni znamke Apple (na njem bomo pognali program), ter primeren FireWire / IEEE 1394 kabel s katerim se povežemo na Appl-ov računalnik. Ko je kreiranje pomnilniške slike končano, program samodejno pridobi uporabniško geslo za dostop do sistema.




**Slika 6: Shema načina uporabe orodja Goldfish**

- **Mac Memory Reader**
  Program poženemo v ukazni vrstici in nam zajame vsebino fizičnega pomnilnika, ter jo shrani v ”Mach-O” binarni datoteki oziroma v neobdelani datoteki. Z njim lahko dostopamo tudi do spomina zunanjih naprav, ki imajo preslikane pomnilniške naslove (memory-mapped device), kot je deljen video spomin. Program je brezplačen in je na voljo za 32 in 64 bitne različice Mac sistema.

### 9.1.4 *Virtual*

- **Xen in Qemu**
  Xen in Qemu sta programa s katerima ustvarimo in poganjamo več navideznih strojev na enem računalniku. Če želimo pridobiti pomnilniško sliko posameznega operacijskega sistema, ki teče v programu Xen, ga najprej poiščemo z ukazom ”xm list” in uporabimo preprost ukaz *”sudo xm dump-core –L /tmp/dump-core-6 6”*. V programu Qemu pa z ukazom *”pmemsave 0 0x20000000 /tmp/dumpfile”*.

## 9.2 Strojne rešitve

- **Tribble PCI Card**
  Ta naprava je produkt raziskovalnega projekta in ni na voljo za prodajo. Je PCI kartica, ki ima priklop za zunanjo pomnilniško napravo ter fizično stikalo za vklop. Ko je kartica vstavljena v računalnik in če fizično stikalo ni vklopljeno, jo sistem ne zazna. Ko želimo zajeti pomnilniško sliko moramo fizično vklopiti stikalo. Kartica ima funkcijo, ki je zelo pomembna v računalniški forenziki in sicer, da če je le mogoče zaustavi delovanje procesorja. Ko je zajem pomnilniške slike končan, naprava samodejno izračuna digitalni izvleček (hash) zajete slike ter ga shrani v datoteko.




**Slika 7: Capture GUARD (ExpressCard)**

- **Windows Scope CaptureGUARD**
  Gre za strojno rešitev, ki jo je razvilo podjetje Windows Scope in deluje samo na operacijskih sistemih Windows. Obstajata dve različici in sicer za namizne računalnike, ter strežnike v obliki PCI Express kartice, druga pa za prenosne računalnike – ExpressCard. Naprava je sposobna zajeti fizični pomnilnik računalnika na katerem je priklopljena. Zajeta slika se shrani v WinDD format, katerega lahko pregledamo z orodjem WindowsSCOPE Cyber Forensics Ultimate ali z orodjem, ki podpira takšen format. Naprava se poveže neposredno s pomnilnikom, pred zajemom pomnilniške slike pa je potrebno namestiti še manjši CaptureGUARD gonilnik.

## 10. ZAKLJUčEK

V digitalni forenziki je zelo pomembno, da pravilno pridobimo dokaze, zato sva v začetku predstavila faze digitalnega preiskovanja, ki sta jih predstavila Carrier in Spafford.

Predstavili smo platformo za ocenjevanje forenzičnih orodij za izdelavo slik delovnega pomnilnika glede na tri dejavnike: pravilnost, atomarnost in integriteto, ki sta jih opisala avtorja v svojem članku. Ti dejavniki določajo kvaliteto slike delovnega pomnilnika in so merjeni z uporabo testiranja na pristopu bele škatle, kar pomeni, da moramo poznati izvorno kodo posamezne rešitve. S pomočjo številnih vstavljenih klicev, ki se nanašajo na kode za izdelavo slik, lahko signaliziramo pomembne dogodke in operacije. Prilagojena različica x86 Bochs emulatorja prestreže ta obvestila in ustvari protokol procesa izdelave slik.

V začetku smo opisali zmogljivosti aplikacij win32dd, WinPMEM in mdd za izdelavo slik delovnega pomnilnika in sicer za velikost pomnilnika med 512MB in 2GB. Avtorja članka sta analizirala 270 slik sistema v stanju mirovanja. Raziskava je pokazala, da sprva niso vsi izdelki sposobni ustvariti kopijo celotnega fizičnega naslovnega prostora. Še slabše, nepravilne rešitve ustvarijo slikovno datoteko z neusklajenostjo podatkov z zamikom. Ker bi lahko analize teh datotek pripeljale do morebitnih napačnih rezultatov v kasnejših raziskavah, sta avtorja naredila programske popravke za ta orodja in tako rešila problem.

Ena iz med zanimivih spoznanj je tudi, da se stopnja atomarnosti zmanjša z naraščajočo velikostjo pomnilnika. Dokazala sta, da se s povečevanjem velikosti pomnilnika poveča tudi čas, ki je potreben za dokončanje procesa izdelave slik, posledica tega pa je, da skladnost slikovne datoteke vedno bolj težko vzdrževati zaradi sočasnih aktivnosti sistema. Sistemu s konstantno obremenitvijo, se spremeni sorazmerno manj pomnilniškega prostora in stopnja integritete slike se zadovoljivo povečuje.

Uspešnosti testiranih orodji za izdelavo slik delovnega pomnilnika se ne močno razlikujejo. To ni presenetljivo, ker vse rešitve za kreiranje kopije pomnilnika interno dostopajo do \\.\Device\PhysicalMemory oddelka v prostoru jedra. Te operacije so lahko prestrežene s strani zlonamerne programske opreme bodisi za preprečitev izdelave slike v celoti bodisi spremenjen prikaz sistemskega delovnega pomnilnika. Pomembno je tudi poudariti, da oba win32dd kakor tudi WinPMEM ponujata alternativno metodo izdelave pomnilniških slik, ki lahko preprečijo proti forenzične poskuse.

Na koncu sva raziskala še druga orodja (tako programska kot strojna) za zajem pomnilniške slike.

## 11. REFERENCE

[1] Vömel S, Freiling FC. *Correctness, atomicity, and integrity: defining criteria for forensically-sound memory acquisition.* Digital Investigation, 2009, str. 131-132.

[2] Su Z,Wang L. *Evaluating the effect of loading forensic tools on the volatile memory for digital evidences.* Seventh international conference on computational intelligence and security, 2011.

[3] Martignoni L, Fattori A, Paleari R, Cavallaro L. *Live and trustworthy forensic analysis of commodity production systems.* Proceedings of the 13th International conference on recent advances in intrusion detection (RAID), 2010.

[4] Yu M, Lin Q, Li B, Qi Z, Guan H. *Vis: virtualization enhanced live forensics acquisition for native system.* Digital Investigation, 2012, str. 22–33.

[5] (2014) Brian D. Carriera, Joe G. *A hardware-based memory acquisition procedure for digital investigations.* Dostopno na: http://grandideastudio.com/wp-content/uploads/tribble_paper.pdf.

[6] (2014) Naja D. *Live Memory Acquisition for Windows Operating Systems: Cover Page and Abstract Tools and Techniques for Analysis.* Dostopno na: https://www.emich.edu/ia/pdf/research/Live%20Memory%20Acquisition%20for%20Windows%20Operating%20Systems,%20Naja%20Davis.pdf.

[7] (2014) Tools:Memory Imaging. Dostopno na: http://www.forensicswiki.org/wiki/Tools:Memory_Imaging.

[8] (2014) int for(ensic){blog;}. Dostopno na: http://computer.forensikblog.de/en/2005/10/acquisition-2-crash-dump.html.

[9] (2014) Antonio M. *FireWire Memory Dump of a Windows XP Computer: A Forensic Approach.* Dostopno na: http://www.friendsglobal.com/papers/FireWire%20Memory%20Dump%20of%20Windows%20XP.pdf.

[10] (2014) Michael G, Tilo M. *On the Practicability of Cold Boot Attacks*. Dostopno na: http://www1.cs.fau.de/filepool/projects/coldboot/fares_coldboot.pdf.
[11] (2014) Belkasoft Live RAM Capturer. Dostopna na: http://forensic.belkasoft.com/en/ram-capturer.
[12] (2014) Slika 5. Dostopno na: http://forensic.belkasoft.com/en/.
[13] (2014) Slika 6. Dostopna na: http://digitalfire.ucd.ie/wp-content/uploads/2012/10/Goldfish-presentation.pdf.
[14] (2014) Slika 7. Dostopna na: http://www.windowsscope.com/.

| | | 512 MB | 1024 MB | 2048 MB |
|---|---|---|---|---|
| Number of pages in the address space | | 131,056 | 262,128 | 524,272 |
| *win32dd* | Number of correctly imaged pages | 130,269 (99.40%) | 260,572 (99.41%) | 521,183 (99.41%) |
| | Number of pages with observed differences | 19 | 20 | 17 |
| | Number of bytes changed in total | 1446 | 1484 | 1388 |
| | Number of indicated read errors (incl. MMIO) | 768 | 1536 | 3072 |
| *mdd* | Number of correctly imaged pages | 130,286 (99.41%) | 260,590 (99.41%) | 521,198 (99.41%) |
| | Number of pages with observed differences | 1 | 1 | 1 |
| | Number of bytes changed in total | 1074 | 1075 | 1076 |
| | Number of indicated read errors (incl. MMIO) | 769 | 1537 | 3073 |
| *WinPMEM* | Number of correctly imaged pages | 130,188 (99.34%) | 260,492 (99.38%) | 521,100 (99.39%) |
| | Number of pages with observed differences | 1 | 1 | 1 |
| | Number of bytes changed in total | 14 | 14 | 14 |
| | Number of indicated read errors (incl. MMIO) | 867 | 1635 | 3171 |

**Tabela 1: Ocene pravilnosti rezultatov testov**

| | | 512 MB | 1024 MB | 2048 MB |
|---|---|---|---|---|
| Number of pages in the address space | | 131,056 | 262,128 | 524,272 |
| *win32dd* | Number of pages unaffected by concurrent activity | 81,979 (62.55%) | 115,763 (44.16%) | 133,871 (25.54%) |
| | Number of pages affected by potential atomicity violations | 49,077 (37.45%) | 146,365 (55.84%) | 390,401 (74.46%) |
| *mdd* | Number of pages unaffected by concurrent activity | 82,180 (62.71%) | 113,199 (43.18%) | 127,648 (24.35%) |
| | Number of pages affected by potential atomicity violations | 48,876 (37.29%) | 148,929 (56.82%) | 396,624 (75.65%) |
| *WinPMEM* | Number of pages unaffected by concurrent activity | 79,217 (60.45%) | 113,697 (43.37%) | 127,964 (24.41%) |
| | Number of pages affected by potential atomicity violations | 51,839 (39.55%) | 148,431 (56.63%) | 396,308 (75.59%) |

**Tabela 2: Atomarnost rezultatov testov**

# Druga področja digitalne forenzike

# Brodolom potniške ladije Costa Concordia

## [Forenzična preiskava]

Vlada semenova
Fakulteta za računalništvo in informatiko
Ljubljana, Slovenija
vs2726@student.uni-lj.si

Gregor Cimerman
Fakulteta za računalništvo in informatiko
Ljubljana, Slovenija
gc9623@student.uni-lj.si

## ABSTRACT
V članku je opisan postopek pridobivanja informacij s črne skrinjice brodoloma potniške ladje Costa Concordia. Črne skrinjice imenujemo VDR (angl. Voyage Data Recorder). VDR beleži podatke ladijskega navigacijskega in komunikacijskega sistema. Modernejše ladje beležijo tudi podatke o lokaciji, stanju in dogajanjem na poveljniškem mostu in so ključnega pomena pri rekonstrukciji dogodkov v primeru nesreč. Glavni problem pri forenzični preiskavi takšnih naprav je v nestandardnem način beleženja podatkov, kar lahko podaljša forenzično preiskavo.

## Keywords
Costa Concordia, brodolom, VDR, križarka, digitalna forenzika, beleženje podatkov

## 1. UVOD
Beleženje podatkov na ladji je ključnega pomena za rekonstrukcijo zadnjih momentov pred nesrečo. V zadnjih 50 letih se je samo v Združenih Državah Amerike potopilo kar 55 potniških ladij križark. Glede na statistiko [**?**] ZDA do leta 2014 z ladjo križarko potovale povprečno preko 20 milijonov ljudi letno. Potniki so običajno starejši, povprečna dolžina potovanja traja sedem dni. Od leta 1979 pa do leta 2014 je na krovu umrlo že 172 ljudi. Tukaj so šteti primeri naravne smrti in nezgode med potovanjem. Do leta 2014 je na izdelavo čakalo 26 novih naročnikov križark.

Pri najhujši nesreči ladje križarke je umrlo več kot 1500 ljudi[**?**]. Rekonstrukcija nesreče lahko pomaga pri iskanju krivca in pripomore k gradnji varnejših potniških ladij. Problemi s katerimi se srečujejo digitalni forenziki so predvsem tehnični, saj se skoraj vsak proizvajalec varnostne opreme odloči za svoj način beleženja podatkov na podatkovno enoto.

## 2. VOYAGE DATA RECORDER (VDR)
Danes ima skoraj vsaka ladja VDR. VDR sisteme so privzeli iz letalstva. Podatki zajeti s pomočjo VDR sistema oz. koncepta črne skrinjice se izkaže za zelo koristno v primeru nesreč. Podatki, ki so zabeleženi s pomočjo VDR sistema, lahko omogočajo podrobnejši vpogled v dogodek ter boljše razumevanje dogajanja. VDR je računalniški sistem, ki shranjuje digitalne podatke z ostalih naprav, ki opravljajo pomembne funkcije v vozilu. Za rekonstrukcijo teh podatkov je potrebno poseči po digitalnih forenzičnih načinih obdelave podatkov. Standardni forenzični postopek je sledeč: zbiranje, shranjevanje, pregledovanje, analiziranje ter rekonstrukcija. Te postopki so podobni splošnim digitalnim forenzičnim preiskavam.

### 2.1 Predpisi za uporabo VDR
Konvencije SOLAS se šteje kot najpomembnejša in najvplivnejša konvencija za mednarodne pogodbe o varnosti trgovskih in potniških ladij. Prva različica predpisov o uporabi in zgradbi VDR sistemov, je bila sprejeta leta 1914, kot odziv na Titanik. Druga različica je bila objavljena leta 1929, tretja leta 1948, četrta pa leta 1960. Različica, objavljena leta 1974, vključuje postopek, ki zagotavlja, da spremembe v predpisih začnejo veljati na navedeni datum. Kot rezultat je bila ta konvencija velikokrat posodobljena in nekajkrat spremenjena. Predpisi, ki so veljavni še danes se imenovanujejo SOLAS, 1974.

## 3. COSTA CONCORDIA
Križarka Costa Concordia je bila izgrajena leta 2005[**?**]. Prejme lahko 3780 potnikov, za katere skrbi 1100 članov posadke. Concordia je bila križarka svetovnega kova in novica o brodolomu je pretresla evropsko javnost. 13. januarja 2012 je v jasnih vremenskih razmerah in mirnem morju trčila v otok Isola del Giglio, v bližini katerega je tudi nasedla. Okoliščine nesreče so bile dolgo časa vprašljive, saj je bila raziskava nekajkrat podaljšana.

## 4. VDR SISTEM
VDR sistem skrbi za beleženje stanja ladje. Sistem je samostojna enota, ki je priključena na nekatere ključne ladijske sisteme. Shema priključenih sistemov je vidna na sliki **??**. Na VDR so priključeni elementi:

- Concentrator - običajno indrustrijski računalniški sistem, ki sprejema podatke s senzorjev in procesira oz. kodira prejete podatke. Obdelane podatke tudi zabeleži na podatkovni medij.
- Sensors - vsi senzorji, ki jih beleži VDR sistem.

- Sensor interface unit - Opcijska strojna oprema, ki razširja podatkovno vodilo naprave, da lahko sprejema podatke z večjega števila naprav.
- Final recording medium (FRM) - podatkovna enota, ki trajno shranjuje podatke in jih je mogoče v primeru nesreče mogoče pridobiti z naprave. Narava mora zdržati vse obremenitve na katere je lahko izpostavljena med normalnim delovanjem in med nesrečo.
- Dedicated power source - izvor električne energije, ki omogoča delovanje VDR sistema v primeru izpada glavnega in rezervnega napajanja na ladiji in zagotavlja vsaj dvourno delovanje.
- Bridge alarm unit - povezava z VDR panelom na poveljniškem mostu, ki omogoča zagon in ustavitev enote ter osnovno diagnostiko o delovanju VDR enote.
- Replay stations - opcijske naprave, k omogočajo pregled podatkov shranjenih na VDR podatkovni enoti za morebitno obdelavo ali arhiviranje.

V primeru nesreče se z ladje odstrani VDR v kateremu so shranjeni zabeleženi podatki, prav tako velja omeniti, da vsak priključen sistem sam vodi evidenco in zgodovino delovanja, ki je prav tako lahko v pomoč pri preiskavi vzrokov za nesrečo. Glavni zahtevi VDR sistema sta, da mora podatkovni medij vsebovati vsaj 12 ur posnetkov in mora delovati vsaj 2 uri po izgubi dotoka električne energije. Poleg tega mora sistem zagotavljati način prenosa podatkov s podatkovnega medija, če se pojavi potreba po verzioniranju varnostnih kopij.

## 4.1 Kako VDR zbira podatke

Centralni sistem zbira podatke z ostalih navigacijskih, komunikacijskih, informacijskih in alarmnih sistemov na ladji. Viri podatkov so precej razpršeni vendar se vse informacije zbirajo v poveljniškem mostu. VDR večino svojih podatkov prejem z naprav na poveljniškem mostu. Te naprave so različnih tipov in oddajajo različne vrste signalov, ki jih je potrebno predhodno obdelati preden se lahko shranijo na podatkovni medij.

- Mikrofoni
- Digitalni signali
- Analogni signali
- Video signali
- Serijski podatki
- Kompleksnejši podatki

Modernejše ladje vsebujejo večinoma digitalne naprave, ki imajo pripravljene namenske izhode za sisteme kot je VDR. Več problemov predstavljajo starejše ladje, ki so bile opremljene s starejšimi napravami, ki namenskih izhodov niso imele, prav tako so tudi te naprave lahko oddajale različne signale, ki jih je bilo težavno shraniti na enoten podatkovni medij.

## 4.2 Vrste podatkov

NMEA (angl. National Marine Electronics Association) je priporočila format hrambe podatkov na podatkovnem mediju v sistemu VDR. Format hrambe podatkov je v obliki NMEA 0183 nizov. NMEA določa tudi način komunikacije med elektronskimi napravami namenjeni za pomorstvo in so vgrajene v vse vrste vodnih plovil. Specifikacija določa način komunikacije med napravami kot so sonar, giroskop, avtopilot, GPS oddajniki in ostali merilni inštrumenti, ki se nahajajo na krovu ladje. NMEA standarde preveri in upravlja nacionalna združba za elektronske naprave v pomorstvu. Standard NMEA 0183 počasi zamenjuje nov NMEA 2000 standard.

NMEA 0183 določa uporabo enostavnega ASCII serijskega komunikacijskega protokola, ki določa kako naj bo izveden prenos podatkov. S pomočjo množilnikov in vmesnih členov, ki razširjajo izhode lahko naprava, ki oddaja podatke namenjene enemu sistemu, pripeljemo do VDS sistema, ki bo te le beležil in skrbel za hrambo in zgodovino. Na ta način je mogoče poceniti in poenostaviti vgradnjo VDR sistema na plovilo, kar je ključnega pomena, če je omenjena naprava obvezna in regulirana s strani nacionalne organizacije. Specifikacija določa tudi kako naj poteka komunikacija, da vsi poslušalci podatkov z naprave prejmejo celoten paket in kaj se zgodi v primeru neuspešne dostave. VDR sistem je običajno naprava ki samo posluša, medtem ko so vse ostale merilne enote na ladji oddajniki, ki proizvajajo podatke.

Spodnji primer prikazuje NMEA niz, ki vsebuje podatke o lokaciji plovila glede na GPS sprejemnik:
`$GPGGA,123519,4807.038,N,01131.000,E,1, 08,0.9, 545.4,M,46.9,M,,*47`
Niz sestoji s sledečih podatkov:

- $ - začetni simbol, ki je lahko tudi klicaj v posebnih priemrih.
- GP - prva simbola, ki opisujeta vrsto podatkov. GP predstavlja tip podatkkov, ki so z GPS sprejemnika (angl. GPS receiver).
- GGA - opis podatkov z določene naprave. Opis vrste podatkov. GGA predstavlja podatke z GPS sprejemnika, ki so bili uspešno pridobljeni (angl. global positioning fix data).
- 123519 - časovni žig zajema podatkov (12:35:19 UTC).
- 4807.038,N - pasovna dolžina (48 deg. 07.0380 North).
- 01131.000,E - pasovna širina (11 deg. 31.000 East).
- 1 - kvaliteta GPS signala (1 stands for GPS fix).
- 08 - število vidnih satelitov s katerih je bila pridobljena informacija
- 0.9 - horizontalna natančnost sistemov za pozicioniranje
- 545.4,M - višina nad srednjo nadmorsko višino (545.4 m).
- 46.9,M - višina nad WGS84 ellipsoidom.

- Empty field - prazno polje, čas v sekundah od zadnje meritve.
- Empty field - ID številka DGPS postaje
- *47 - kontrolna vsota (vedno se začne s simbolom *).

Kontrolna vsota je heksadecimalni zapis osem bitne ekskluzivne OR operacije celotnega niza med znakom $ in *.

NMEA standard omogoča tudi dodajanje lastnih zapisov, ki jih proizvajalci VDR sistemov prilagodijo svojim potrebam, pri tem morajo le paziti na notacijo. Običajno le zapise, ki so nestandardni označi s črko P, ki ji sledi ID proizvajalca ki je omogočil beleženje podatkov.

Ena izmed večjih težav pri forenzični preiskavi nestandardnih podatkov je v iskanju pomena. Običajno proizvajalci samo določijo tip in obliko podatkov, ki pa ni najbolj informativne narave. Pristop, ki ga uporabljajo digitalni forenziki pri obdelavi takih podatkov, je primerjanje z ostalimi podatki s podobnih virov ali s pomočjo proizvajalca sistema. Običajno so sistemi takšnega tipa slabo dokumentirani ali pa je dokumentacija skrita za javnost.

V primeru brodoloma Coste Concordije se je v VDR sistemu, za beleženje stanja vodnotesnih vrat uporabljala nestandardna notacija, kar je razvidno z dnevniških zapisov. Primer takšnega zapisa je:

- $PSWTD,01,O——,*XX
- $PSWTD,24,OFV—,*XX

Ker se niz začne z znakom "P", kar označuje nestandarden niz proizvajalca za katerega je namenjena oznaka "S" je bilo potrebno posvetovanje s proizvajalcem za translacijo in opis podatkov. V term primeru se je oznaka proizvajalca ujemala z njegovim imenom; Seanet. Znaki ki sledijo "PS", označujejo vrsto zabeleženih podatkov in sicer "WTD" označujejo vodotesna vrata (angl. Wather Thight Doors). Kratici sledi število, ki označuje vrata. Števila so v rangu od 1 do 25. Tekom preiskave so ugotovili, da ladja vsebuje 25 vodotesnih vrat, na kar so sklepali, da so števila od 1 do 25 oznake vrat. Eno število za vsako od 25 vrat. Forenziki so se, poleg pomena oznake, srečali tudi s problemom številčenja vrat, saj je moral nekdo vsakim vratom nameniti številko, kar bi lahko predstavljal problem, saj teh oznak na samih vratih ni bilo. Oznaki vrat sledi niz petih znakov, ki predstavljajo stanje vrat. Simbol "O" predstavlja odprta vrata (angl. Open), ozaka "C" pa zaprta (angl. Closed). Predpostavka o prvem simbolu se je izkazala za resnično šele kasneje v preiskavi. S pomočjo proizvajalca vrat in konkretnega sistema VDR so lahko potrdili pomen ostalih oznak. Po pravilni interpretaciji podatkov zabeleženih na podatkovnem mediju so lahko rekonstruirali celotno sliko stanja vrat. Sodelovanje s proizvajalcem strojne opreme se je izkazalo za ključno pri natančnosti in uspešnosti forenzične priskave.

## 4.3 Problemi pri analizi podatkov s sistema VDR

Problemi s katerimi se srečujejo forenzični preiskovalci, so predvsem povezani s formatom zapisanih podatkov in programska oprema s katero je mogoče pridobiti podatke s podatkovnega medija. Običajno so proizvajalci VDR sistemov sami poskrbeli za hrambo podatkov, kar pa ni bilo vedno skladno s forenzično higieno. Pri forenzični preiskavi podatkovnega medija digitalni forenziki sodelujejo s proizvajalcem, ki mora omogočiti, kot to zagotovi lastniku ladje, prenos podatkov s podatkovnega medija. V starejših verzijah, ko prenos podatkov z naprave ni bil zahteva IMO (angl. International Marine Organization), je bilo potrebno podatkovni medij poslati proizvajalcu, ki je pridobil podatke s same naprave in jih vrnil preiskovalcem. Z novo politiko pa so pravila zaostrili in omogočili lastniku kot tudi forenzičnim preiskovalcem ekstrakcijo podatkov s podatkovnega medija. IMO je določil performančne zahteve za VDR sisteme ter standarde glede formata zabeleženih podatkov in hrambo le-teh.

### 4.3.1 *Problemi pri forenzični preiskavi Coste Concordije*

Brodolom Coste Concordije je bil dober primer digitalne fornezične preiskave ter rekonstrukcije podatkov z VDR sistema. Križarka je bila opremljena z modernimi navigacijskimi in upravljavskimi sistemi. Ladja je imela tudi sistem za avtomatično krmiljenje, ki je omogočal vnos poti, kateremu je sledil in ladjo ustrezno krmilil.

Ladja je bila izdana v uporabo leta 2005 s strani Fincantieri Sestri Ponente - italijanski ladjedelec, Julija leta 2006 pa je prestala obsežen servis. Januraja 2012 je v mirnem morju nasedla na otok Isola del Giglio na zahodni obali Italije. Takoj po trku s skalo na obali otoka je v trup ladje začela vdirati voda, zaradi česar je tudi kmalu za tem izgubila električno napajanja in pogon. Nasedla je na vzhodni del otoka, severno od mesta trka, v plitvih vodah, kjer je obležala na desni strani. Večina njene desne strani je bilo potopljene, leva stran (stran trka) pa je bila obrnjena gor. Leta 2013 so začeli z reševanjem ruševin, letos so jo razglasili za odpadni material, zaradi česar bo predana v razrez.

Preiskovalci so se med forenzično preiskavo brodoloma srečali z nemalo težavami. Na srečo preiskovalcev je VDR sistem preživel nesrečo in zabeležil vse prejete podatke. Pridobitev VDR sistema z ladje je bil enostaven, saj naprava ni bila potopljena ali se nahajala pod samim brodolomom. Slika zajeta med ekstrakcijo naprave VDR je prikazan na sliki 1.

### 4.3.2 *Težava pri forenzični preiskavi Maersk Kendal incidenta*

Tovorna kontejnerska ladja Maersk Kendal je bila zgrajena leta 2007 v Severni Koreji in leta 2009 doživela prvo nesrečo. Tovorna ladja je prikazana na sliki 2. Nesreča se je zgodila med prevozom tovornega blaga proti Singapurskem pristanišču Tanjung Pelepas (Malaysia). Ladja je namreč nasedla na Monggok Sebarok greben v Singapurskem Strait. V času nesreče je bilo na ladji 25 članov posadke, od katerih ni bil nihče huje poškodovan. Ladja je bila poškodovana v jekleni konstrukciji sprednjem delu. Poškodovan je bil sprednji balastni rezervoar števila 1, sprednja jeklena konstrukcija, prostor v katerem se nahaja trupni stranski pogon in



**Figure 1: Prevzem naprave VDR z brodoloma potniške ladje Costa Concordia. Ker ladja ni bila povsem potopljena, iskanje in pridobitev naprave ni bila zahtevna.**

sam pogon.




**Figure 2: Tovorna kontejnerska ladja Maersk Kendal. Slika je bila zajeta leta 2009**

Pri preiskavi dogodkov pred trkom je bilo ugotovljeno, da kapitan ni upošteval opozoril pristaniškega osebja. Prav tako, med prihodom v pristanišče ni upošteval hitrostnih omejitev v pristanišču. Vso komunikacijo med Ladjo Maersk Kendal in pristaniškim osebjem je priskrbel nadzorni sistem prometa v Singapurskem pristanišču, Singapurski VTIS (angl Singapore Vessel Traffic Information System).

Kapitan je po trku ukazal vzvratni manever s katerim bi lahko ladjo znova pripravil do plovbe, vendar mu to ni uspelo. Po neuspešnem poizkusi je od strojnih inženirjev prejel opozorilo o nevarnosti preobremenitve motorjev. Kapitan je nato iztaknil gumb za shranjevanje podatkov na VDR sistem. Komandna plošča sistema VDR na ladji Maersk Kendal je vidna na sliki 3.

Pri forezični preiskavi je bilo ugotovljeno, da je kapitan ladje nepravilno ravnal s komandno ploščo VDR sistema in s tem




**Figure 3: Kontrolna plošča VDR sistema na tovorni ladji Maersk Kendal**

uničil pomembne dokaze za rekonstrukcijo dogodkov pred nesrečo. Najdenih je bilo tudi nekaj nepravilnosti pri upoštevanju navodil VTIS osebja in protokolov v primeru nasedanj. Kapitan je prav tako prepozno obvestil lastnike o nesreči in stanju ladje.

## 5. PRIMER FORENZIčNE PREISKAVE VDR PODATKOV BRODOLOMA COSTE CONCORDIJE

Pri analizi podatkov z VDR sistema so preizkovalci našli 80GB velik podatkovni medij. Podatkovni medij je vseboval eno particijo, ki ni bila zagonska, formatirana na QNX4.0 datotečni sistem. Glavna direktorija, ki sta vsebovala podatke sta bila "frame" in "NMEA". Direktorija sta vsebovala radarske slike in NMEA zapise podatkov z različnih senzorjev.

Podatkovne datoteke so poimenovane z numeričnimi vrednostmi, kot si sledijo po kronološkem zaporedju. Vse datoteke so bile brez končnic, šele naknadna forenzična preiskava s pregledovalnikom vsebine datotek je razkrila tip datotek. Datoteke so bile skompresirane v GZip format. Radarske zaslonske slike so bile kompresirane tako, da se je pri kompresiji ohranila končnica datoteke, le dodala se je končnica .GZ. Datoteka je bila preimenovana v ustrezno zaporedno številko in shranjena na diskovno enoto. Podobno so bile obravnavane tekstovne datoteke, le da je bila prvotna končnica teh datoteka .LOG, po novem pa .LOG.GZ.

Preiskovalci so za lažje filtriranje podatkov v programskem jeziku Python napisali nekaj skript, ki so pridobljene podatke ustrezno obdelali in prikazali le relevantne.

Radarske zaslonske slike so bile shranjene kot slike, pridobljene z dveh radarskih zaslonov na poveljniškem mostu v rednih intervalih. Intervali so trajali približno sedem sekund. Zbranih je bilo 11,759 radarskih slik ki so razkrivale stanje radarskih zaslonov približno 24 ur pred nesrečo. Vsaka slika vsebuje informacijo o času zajema v lokalnem času.

NMEA nizi so shranjeni kot ASCII besedilne datoteke. Vsaka besedilna datoteka vsebuje število vrstic. Vsaka izmed vrstic je sestavljena s časovnega žiga zabeleženega po Unix notaciji. Vsaka izmed datotek vsebuje približno pet minut podatkov o različnih senzorjih po ladji, ki jih je zbral VDR sistem. Skupaj so datoteke vsebovale podatke, ki niso bili starejši od dveh tednov. Domena NMEA zapisov je prav tako kot radarske slike zabeležena na način rotiranja podatkov, kar pomeni, da se stari podatki izbrišejo za potrebe shrambe novih.

Po uspešni pridobitvi podatkov so preiskovalci morali pridobljene podatke interpretirati. Težava pri interpretaciji takih podatkov je v nekonsistentnosti med posameznimi proizvajalci in pomanjkljiva tehnična dokumentacija.

## 5.1 Podatki s krmila

Prva zahteva za preiskovalni postopek je bila razjasnitev kaj se je dogajalo na poveljniškem mostu pred nesrečo in kdo je v tem času krmaril ladjo. V preiskavi je bilo ugotovljeno, da je krmar napačno ravnal in krmilo usmeril v napačno smer, kot je to zahteval kapitan. Po kapitanovem ukazu je krmar nekaj sekund napačno usmerjal krmilo ladje. Za pojasnitev dogodkov so morali preiskovalci pregledati podatke, zabeležene na sistemu VDR, povezane z krmilnim sistemom ladje. Stanje krmila se je beležilo v nestandardni obliki NMEA notacije. Podatki vsebujejo analogna in digitalne vrednosti senzorjev v obliki:
`$PAVBADC, a, xx.xxx,[a times xx.xxx], name *hh`
kjer posamezni deli kodatkov predstavljajo:

- a - število ponovitev sledečih podatkov (1–8). Lahko tudi število podatkov v tej vrstici.
- xx.xxx - analogna vrednost
- name - opis izovra podatkov
- hh - kontrolna vsota

Grafični prikaz stanja krmila tik pred nesrečo in nekaj minut po tem je prikazan na sliki **??**.

Pri rekonstrukciji podatkov so preiskovalci našli konfiguracijsko datoteko, ki pojasnjuje zapise zabeležene na podatkovnem mediju VDR sistema. Konfiguracijska datoteka opisuje dva tipa informacij: `Rudder order`, ki opisuje stanje krmila kot ga obrača krmar in `Ruder response`, ki opisuje stanje krmila ki ga je nastavila mehanika krmila glede na ukaze krmarja in krmarskega obroča.

Graf, ki prikazuje stanje krmila v zadnjih sekundah pred trkom. Preiskovalci so potrdili napako krmarja, saj so s poslušanjem posnetkov s poveljniškega mostu in podatkov krmila opazili neskladnost akcij in ukazov. Do časa 21:44:40 je krmar upošteval ukaze kapitana. Po tem času je, kot je z zvočnih posnetkov jasno razvidno kapitan zahteval spremembo smeri v desno, na kar je krmar začel obračati krmilo v levo. Obračanja krmila ni popravil nadaljnjih deset sekund nakar je v podatkih s krmila razvidna nagla sprememba smeri z leve na desno. Če je omenjena napaka res vzrok za nesrečo je popolnoma prepuščeno odločitvi sodišča.

## 5.2 Vodotesna vrata

Tekom preiskave je bilo potrebno raziskati tudi stanje vodotesnih vrat v trupu ladje. Hipoteza je bila, da so bila vsa vrata v trupu tekom potovanja in v času nesreče odprta. Vodotesna vrata morajo, glede na pravilnik o plovbi, biti ves čas plovbe zaprta. Na ta način preprečijo vdiranje vode v primeru preboja trupa, v druge nepoškodovane prostore in s tem zmanjšajo tveganje potopa. Stanje teh vrat je ključnega pomena pri številu nesreč plovil, zaradi česar se te podatki običajno hranijo tudi na VDR sistemu.

Vodotesna vrata na potniški ladji so upravljana hidravlično. Upravljanje vrat zahteva delovanja oljnega sistema ter električnega sistema, ki upravlja z ventili za vrata. Za zapiranje vrat v primeru nesreče je zagotovljene redundančna električna enota, ki zagotavlja nekaj zapiranj in odpiranj tudi v primeru, da odpove glavni dotok električne energije na ladji. Hidravlična vrata je mogoče odpreti ali zapreti neposredno na lokaciji ali na daljavo s poveljniškega mostu ali varne sobe. Vsaka vrata so opremljena s posebnim NMEA oddajnikom, ki oddaja stanje vrat v obliki nestandardnega NMEA niza.

Podatki zabeleženi na diskovno enoto VDR sistema, v povezavi z osmimi vodotesnimi vrati:

- 2012/01/13-21:00:07 - $PSWTD,08,C——,*35Õ A
- 2012/01/13-21:19:15 - $PSWTD,08,O——,*31Õ A
- 2012/01/13-21:19:30 - $PSWTD,08,C——,*35Õ A
- 2012/01/13-21:22:07 - $PSWTD,08,O——,*31Õ A
- 2012/01/13-21:22:54 - $PSWTD,08,C——,*35Õ A
- 2012/01/13-21:26:01 - $PSWTD,08,O——,*31Õ A
- 2012/01/13-21:26:17 - $PSWTD,08,C——,*35Õ A
- 2012/01/13-21:42:45 - $PSWTD,08,O——,*31Õ A
- 2012/01/13-21:43:01 - $PSWTD,08,C——,*35Õ A
- 2012/01/13-21:46:56 - $PSWTD,08,CFV–,*37Õ A
- 2012/01/13-22:32:26 - $PSWTD,08,CFV-P,*3AÕ A
- 2012/01/13-22:32:41 - $PSWTD,08,CFV–,*37Õ A
- 2012/01/13-22:33:13 - $PSWTD,08,OFV–,*33Õ A
- 2012/01/13-22:33:28 - $PSWTD,08,?????,*39Õ A

Pri čemer podatki predstavljajo:

- 2012/01/13-21:46:56 - časovni žig
- $PSWTD - oznaka, ki predstavlja nestandarden zapis stanja vodotesnih vrat prizvajalca Seanet
- 08 - številka vrat (oznaka)
- CFV– - stanje vodotesnih vrat. Prva oznaka predstavlja stanje vrat: O - odprta (angl. open) in C - zaprta (angl. closed), sledi oznaka okvare: – - delujoče stanje, F - težava z vrati (angl. faulty) nato pa sledi še opis napake: V - težava z električnim sistemom (angl. Low Voltage), P - tažva s pritiskom olja (angl. Low Oil Pressure) in L - nizek nivo olja v sistemu (angl. Low Level of Oil)
- *37 - kontrolna vsota

S podatkov je razvidno, da se stanje vrat periodično posreduje VDR sistemu vsakih 16 sekund. Zgornji podatki vsebujejo le spremembe stanja vrat, kar so raziskovalci izluščili z vseh zbranih podatkov. Podatki razkrijejo, da osma vrata v trupu ladje, v času trka niso bila popolnoma zaprta. Odprtost teh vrat je prav tako lahko vplivala na brodolom ladje o čemer bo odločalo sodišče.

## 6. ZALKJUčEK

Večina digitalnih forenzičnih preiskav brodolomov bo imelo opravka z VDR sistemom. Sistem je zelo uporaben pri diagnosticiranju in rekonstrukciji dogodkov pred nesrečo. Različni prijemi digitalnih forenzičnih preiskav se lahko direktno uporabijo pri preiskavi vendar so hkrati vsebovane tudi prodrobnosti, ki zadevajo same podatke v tesni povezavi z ladjevjem. Forenzični preiskovalci bodo pri preiskavi običajno potrebovali tehnično dokumentacijo ali pomoč s strani proizvajalca VDR sistema. Preiskovalci brodoloma Coste Concordije so se posluževali vseh omenjenih pristopov za forenzično preiskavo. Omenili smo načine hranjena podatkov in probleme s katerimi se srečujejo preiskovalci, kar običajno podaljša preiskovalni postopek. V prihodnjih verzijah zahtev za gradnjo in izdelavo VDR sistemov pričakujemo večjo pozornost podatkom, ki so uporabni za forenzično preiskavo ter le te hraniti na določen način, ki preiskavo olajša in ne oteži. Interpretacija podatkov naj bo enostavna in jasna, če je možno naj bo priložena tudi dokumentacija.

## 7. LITERATURA


**Carrier Brian** , Defining digital forensic examination and analysis tools using abstraction layers , International Journal of Digital Evidence, 2014

**Costa Concordia Wiki** , http://en.wikipedia.org/wiki/Costa_Concordia, 2014

**Worst ship disasters** , http://www.usnews.com/news/articles/2013/02/14/the-eight-worst-cruise-ship-disasters, 2014

**Crouse Junkey** , http://www.cruisejunkie.com/events.html, 2014

**Cruise Ship Industry Statistics** , http://www.statisticbrain.com/cruise-ship-industry-statistics/, 2014

**International Maritime Organization** , International convention for the Safety Of Life At Sea SOLAS 1974.

**International Maritime Organization** , Resolution MSC. 163(78) “Perfor- mance standards for shipborne simplified Voyage Data Recorders (S-VDRs)”. Adopted on May 17, 2004.

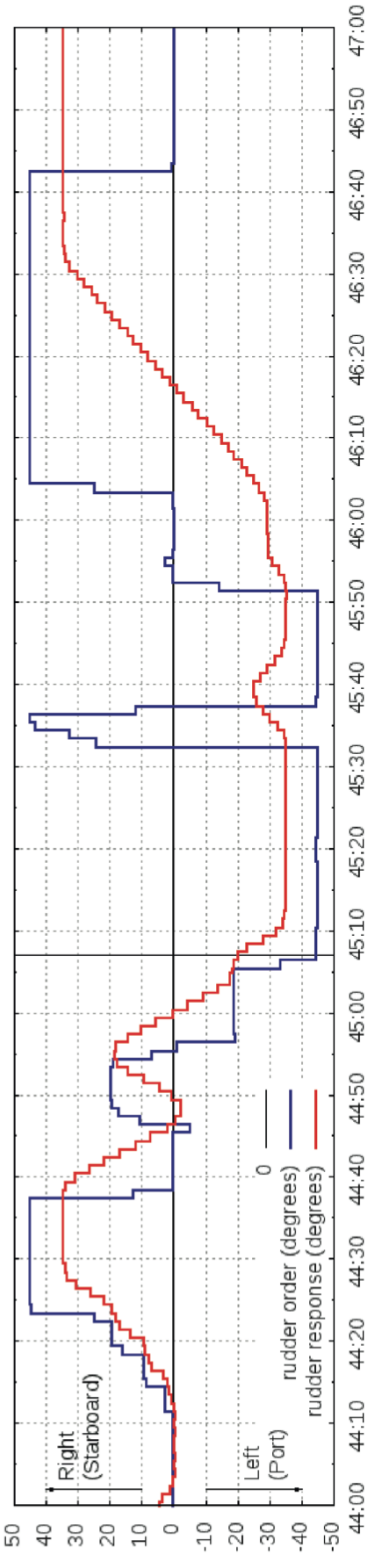



**Figure 4: Podatki o stanju krmila na potniški ladji Costa Concordia. Prikazani podatki so stopinje in čas, pri čermer časovni žig sestavljajo minute in sekunde, 12. januarja 2012 ob 21h.**

# Kdo sem jaz? Analiza digitalnih oseb v preiskavah spletnega kriminala

Jernej Jerin
Fakulteta za računalništvo in informatiko
Univerza v Ljubljani

Tomaž Bartol
Fakulteta za računalništvo in informatiko
Univerza v Ljubljani

Tadej Vodopivec
Fakulteta za računalništvo in informatiko
Univerza v Ljubljani

## POVZETEK

Spletne dejavnosti kaznivih dejanj pogosto vključujejo kriminalce, ki se skrivajo za večjim številom digitalnih identitet (t.i. digitalne osebe). Na drugi strani imamo kriminalne organizacije, kjer večje število ljudi uporabljaja eno digitalno identiteto. Odkrivanje kdo se v resnici nahaja za temi digitalnimi osebami je ne-trivialen problem zaradi velike količine pridobljenih podatkov. Tu seveda najbolj izstopajo podatki v obliki komunikacije znotraj socialnih omrežij. Breme preiskovalcev spletnega kriminala je ogromno - obstoječa orodja ne vsebujejo funkcionalnosti, potrebnih za analizo digitalne osebnosti. V tem članku bomo predstavili Isis orodje, ki obravnava prav to področje. V nadaljevanju bomo predstavili tehnike analize za odkrivanje organizacijskih struktur na podlagi prikritih mrež.

## Ključne besede

Zasebnost, analiza, forenzika, spletni kriminal, digitalna oseba, pedofilija, ISIS orodje

## 1. PREDSTAVITEV

Digitalizacija in posledično Internet je omogočil medsebojno komunikacijo na globalni ravni, s tem pa tudi nehote omogočil kriminalcem nove načine za dostop do potencialnih žrtev. Z pojavom socialnih omrežij se je razširil tudi pojem digitalne osebnosti, ki je pred tem obstajala samo znotraj zaprtih skupnosti kot so bile strani z sporočilnimi deskami in forumi. Digitalne osebnosti v socialnih omrežjih igrajo ključno vlogo. Za kaznivim dejanjem se lahko skrivajo različne digitalne osebnosti, ki pa v resnici predstavljajo isto fizično osebo. V tem primeru lahko rečemo, da imamo relacijo 1:n. V nasprotnem primeru je lahko ena digitalna oseba skupna za celotno kriminalno skupino. Sedaj govorimo o relaciji n:1. Poleg tega lahko zaradi narave identitete na spletnih socialnih omrežjih kriminalci prikrijejo lastno identiteto z relativno lahkoto ter s tem pridobijo zaupanje potencialnih žrtev. Primeri, ki vsebujejo izkoriščanje digitalnih osebnosti v nelegalne namene:

- Pedofili, ki se z uporabo lažne digitalne osebe pretvarjajo da so vrstniki žrtve, katero nadlegujejo. V tem primeru so najpogosteje lažni podatki kot so starost in spol ter seveda namen.
- Prevaranti na podlagi romance, ki preko strani za zmenke z lažno digitalno identiteto pridobijo njihovo zaupanje. V tem primeru je lažen namen.
- Skranje ideološke skupine, ki z pomočjo lažnih digitalnih identitet podpihujejo sovraštvo in napeljuje ostale na kriminalna dejanja zoper posameznike in/ali skupine.

Največji problem, s katerimi se srečujejo kriminalisti je analiza komunikacij. Namreč večina forenzičnih orodij se dobro obnese pri pridobivanju podatkov (Encase), medtem ko so orodja za podporo analize pridobljenih podatkov precej manj sposobna/razvita. Največ kar omogočajo je iskanje po ključnih besedah ali detekcija določenih fraz. Preiskovalci morajo tako ročno pregledati veliko količino podatkov in iz njih izluščiti pomembne informacije za dani primer. Ob tem lahko seveda zaradi prevelike količine podatkov spregledajo ključne podatke. Zaradi tega so začeli preiskovalci dajati vedno večji poudarek na metode rudarjenja iz zajetih podatkov, kjer bi s pomočjo strojnega učenja in umetne inteligence orodja avtomatično izpeljala dejanske atribute osebe kot so npr. spol in starost, ki se skriva za lažno digitalno osebo.

### 1.1 Internet – zaveznik pedofilije?

Pojav interneta je ponudil izjemno priložnost za delovanje pedofilskim mrežam. Pred širitvijo Interneta je bil pedofilski material dostopen le v tiskani ali filmski obliki. Pojav Interneta pa je ponudil vrsto prednosti za pedofilske mreže, ki so jih le-te hitro izkoristile. Internet namreč postaja najbolj množičen medij, dosegljiv vsakomur in od povsod.

V klasičnih medijih je imel možnost ponuditi informacijo, v našem primeru pornografijo, samo tisti, ki je imel znanje, denar in tehnologijo, v svetovnem spletu pa je to lahko vsakdo, ki ima kakršnokoli pornografsko vsebino.

Prednosti interneta so:

- Tekoč in hiter prenos podatkov (npr. pedofilske vsebine). Namreč možnost prenosa video vsebine hitro in

ne glede na geografsko razdaljo je pomenilo pomembno pridobitev za pedofilske mreže. Pošiljanje vsebine po navadni pošti je bilo veliko bolj tvegano, saj je pri počasnejšem prenosu več možnosti za odkritje. Prav tako je veliko lažje v večini primerov hitro uničiti digitalne dokaze kot pa fizične.

- Relativno visoka stopnja varnosti. Internet ponuja veliko možnosti zaščite. Šifriranje in skrivanje podatkov (steganografija) predstavljata najpomembnejši del.

- Skoraj popolna anonimnost. Člani se lahko predstavljajo pod različnimi imeni, dostopajo preko različnih računalnikov in jih je zato težje odkriti.

## 1.2 ISIS orodje

Projekt Isis financira EPSRC (Engineering and Physical Sciences Research Council) in ESRC (Economic and Social Research Council). Cilj financiranja je, da se razvije programsko okolje in orodja za podporo organom pregona ter da bodo z njimi nadzirali socialna omrežja za namen zaščite otrok na spletu.

V zadnjih letih je opaziti kako hitro narašča število in uporaba spletnih socialnih omrežij. Ta socialna omrežja predstavljajo dva pomembna tveganja v zvezi z izkoriščanjem otrok:

1. Pedofili, ki prežijo nad otroci: Otroci aktivno sodelujejo v klepetalnicah in spletnih forumih. Pedofili lahko uporabljajo take forume za preženje nad otroci ali celo načrtujejo dejavnosti, ki so povezane z pedofilijo.

2. Pedofili, ki distribuirajo in dellijo gradiva, ki vsebujejo podatke o zlorabi otrok: Za ta namen pedofili uporabljajo mehanizme, kot so sistemi za izmenjavo datotek. Nedavne študije so pokazale, da je na stotine iskanj nezakonitih slik na vsako sekundo.

V zvezi s tem želi Isis odgovoriti na tri glavne raziskovalne izzive:

- Kako prepoznati aktivne pedofile preko spletnih skupnosti?

- Kako identificirati ključne distributerje (avtorje) gradiva, ki vsebujejo podatke o zlorabi otrok?

- Kako zagotoviti, da takšni razvoji ohranjajo etične prakse?

## 1.3 Delovanje ISIS orodja

Isis orodje (slika 1) deluje na principu statističnih metod za procesiranje naravnega jezika. Takšne metode omogočajo naslednje funkcionalnosti:

- Profiliranje ključnih besed

- Primerjanje frekvenc določenih besed

Orodje se zanaša na bazo učnih podatkov za uspešno učenje modelov in primerjavo med besedili. To pa predstavlja tudi težavo za uporabo teh orodij v tujih jezkih, kjer je obstoj takšnih podatkovnih baz bolj izjema kot pravilo. Namreč od razširjenosti jezika je odvisno kako dobro so izpopolnjene te baze. Orodje nam omogoča pridobitev jezikovnega stilskega profila osumljenca oziroma lahko tudi žrtve. Pri razvoju orodja je bil velik izziv integracija statistično dovršenih ampak pomanjkljivih tehnik temelječih na obstoječem znanju. Orodje omogoča naslednje zmožnosti preiskovanja:

- Vzpostavljanje jezikovnega stilskega profila, ki ga uporablja oseba. S tem lahko ugotovimo, ali preiskovano digitalno identiteto uporablja ena ali več oseb.

- Vzpostavljanje starosti in spola osebe. Iz podane komunikacije lahko orodje z visoko gotovostjo ugotovi ali se recimo starejša oseba skriva za identiteto otroka (uporaba neobičajnih besed za otroke, preveč bogat besedni zaklad, itd.).

- Vzpostavljanje vzorca prisotnosti na internetu in ostalih vzorcev komunikacije.

## 1.4 Jezikovni stilski profil

Iz vsake povedi oziroma besede lahko izluščimo določene vzorce, ki so značilni samo za določeno osebo. Najbolj značilne so naslednje vsebine:

- Uporaba ločil. Število le teh in pravilna ter tudi nepravilna uporaba. Predvsem na podlagi nepravilne uporabe lahko zelo dobro ugotovimo ali gre za isto osebo.

- Uporaba emotikonov. Ali oseba uporablja emotikone med besedami oziroma na koncu stavka.

- Frekvenca besed glede na slovar. Oseba lahko zelo pogosto uporablja nekatere manj znane besede (besede, ki niso v splošni uporabi), kar poveča verjetnost, da gre za isto osebo. Predvsem uporaba slenga in narečnih besed je močan identifikator.

Torej pomembna je uporaba metrik, ki bodo čim bolj pripomogla k ustvarjanju enoličnega profila. Primeri napogostejših metrik, ki se uporabljajo:

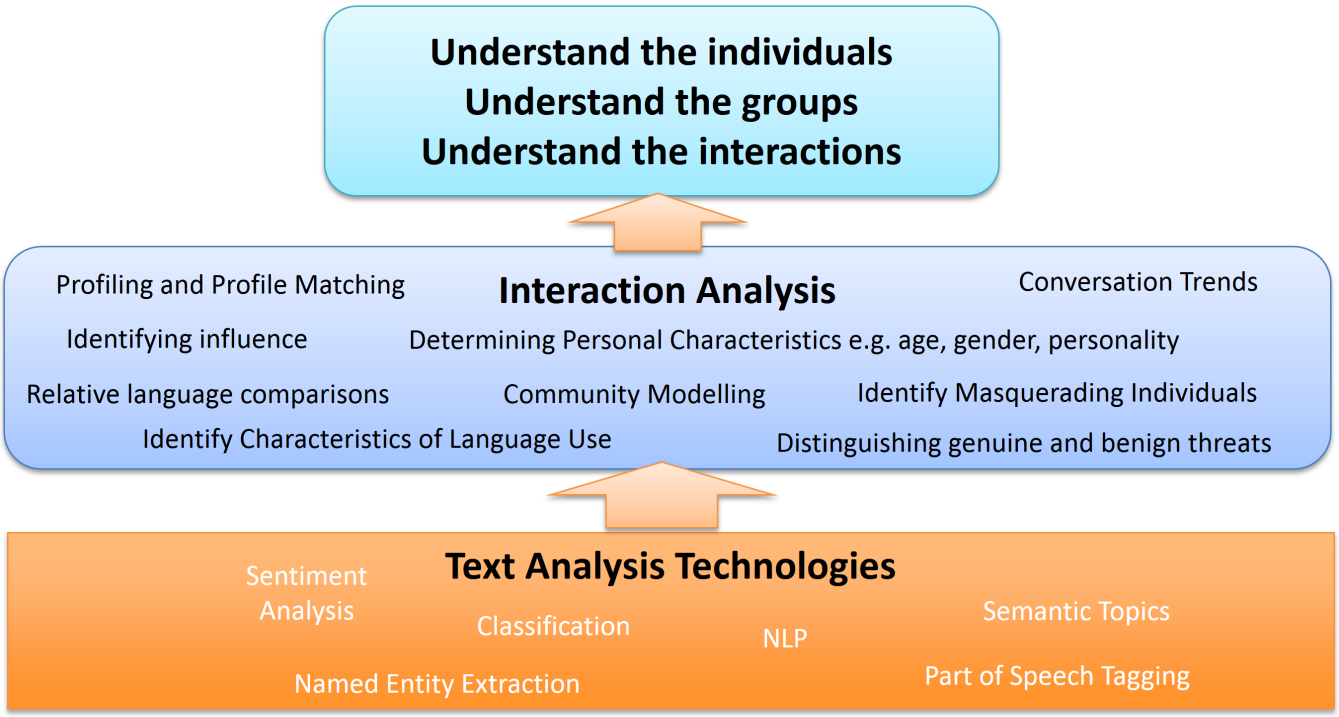
- Število klicajev, vprašajev, vejic

- Indikator širine nabora besed

- Dolžina sporočil

- Tip sporočila glede na lokacijo objave

Predvsem tip sporočila glede na lokacijo objave je zelo pomembna metrika, saj so ponavadi e-poštna sporočila daljša kot pa besedila objavljena v sobah za čvekanje. Namreč daljše je sporočilo, več osebnih indikatorjev bo vsebovalo napram krajšim sporočilom. Za analizo besedila je potrebno dobiti učni vzorec, za katerega vemo naslednje:




**Slika 1: Interaction Analysis Engine.**

- Kakšna oseba ga je napisala:
  - spol
  - starost
  - izobrazba
  - spolna usmerjenost
  - ostale osebne karakteristike
- Kje je bilo sporočilo objavljeno
- V kakšnem kontekstu je bilo objavljeno

Na podlagi množice takšnih sporočil lahko primerjamo dobljene vrednosti metrike za sporočilo neznane osebe z izračunanimi metrikami znane osebe. Če so rezultati podobni, lahko z veliko verjetnostjo trdimo, kakšna oseba je napisala to sporočilo.

## 1.5 Odkrivanje potegavščin, prevar in preslepitev pri spletnem pisanju

Pri digitalni forenziki se pogosto pojavljajo vprašanja glede avtorjev dokumentov: njihova identiteta, demografsko ozadje ter ali jih je mogoče povezati z drugimi dokumenti. Področje stilometrije uporablja jezikovne značilnosti in tehnike strojnega učenja s katerimi želijo odgovoriti na ta vprašanja.

Stilometrija je veda, ki preučuje možnosti merjenja lingvističnega stila avtorjev pisane besede. V besedilih išče vzorce, ki enolično definirajo njihovega avtorja.

Medtem ko lahko stilometrija identificira avtorja z visoko natančnostjo, brez kontradiktornih scenarijev, je njena točnost manjša, ko se sooča z avtorji, ki hočejo namenoma zamegliti svoj stil pisanja, ali poskušajo posnemati drugega avtorja. Medtem ko je to dobro za zasebnost, se postavlja vprašanje glede goljufije. Trdimo, da se nekatere jezikovne lastnosti spremenijo, ko ljudje skrivajo svoj stil pisanja. Pomemben prispevek tega dela je metoda za odkrivanje slogovne prevare v pisnih dokumentih.

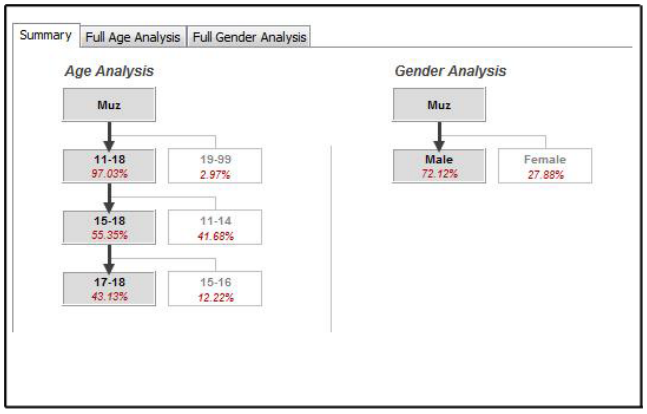
Stilometrija lahko pomaga odgovoriti na vprašanja kot naprimer kdo je avtor nekega dokumenta. Je zelo dobrodošlo orodje za preprečevanje prevar, potegavščin in goljufij.

Primer: Ameriški blogger Thomas MacMaster se je izdajal za sirsko homoseksualno žensko Amina Arraf v blogu "Lezbika v Damasku"(ang. "A gay girl in Damascus"). Pisal je o sirski politiki ter družbenih vprašanjih. Veliko medijev, vključno z The Guardian in CNN, so mislili, da je blog resnično iskren ter njegova besedila objavili. Čeprav še nihče ni govoril z njo ali jo poznal je "Amina"hitro postala zelo priljubljena bloggerka.

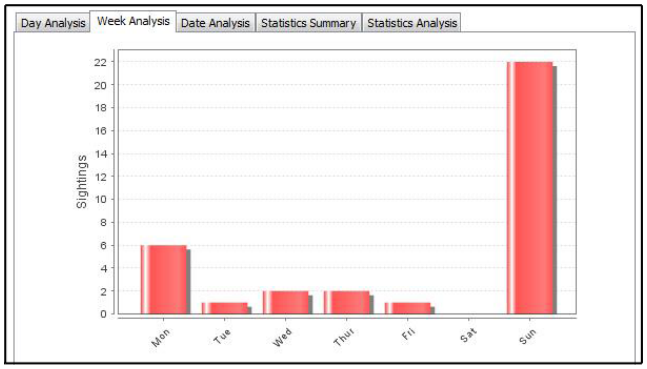
## 1.6 Analiza na podlagi jezika in starosti

Odvija se v štirih korakih in sicer prvi trije zajemajo uporabo orodja za analizo naravnega jezika, torej izračun vrednosti zgoraj naštetih metrik medtem ko v zadnjem koraku uporabi bazo znanja za primerjavo vrednosti metrik. Predstavitev posameznih korakov:

1. Vhodno sporočilo je razbito na posamezne besede in ločila. Vsaka beseda je ozačena glede na tip (povedek, glagol, samostalnik,...).
2. Vsako besedo označimo glede na pomen. Značke so




**Slika 2: Drevo odločitev za starost in spol s pomočjo Isis orodja.**




**Slika 3: Tedenska analiza opažanj s pomočjo Isis orodja.**

definirane vnaprej.

3. Štetje značilnosti posameznih besed.

4. Pripravimo referenčne podatkovne baze, na podlagi katerih bomo primerjali vrednosti metrik. Dobro je imeti ločene referenčne baze glede na prej definirane atribute učnih vzorcev. Na podlagi vrednosti metrike uporabimo algoritme strojnega učenja (recimo linearna regresija), ki nam kot izhod podajo verjetnostno porazdelitev v posamezne ciljne skupine. Rezultate lahko predstavimo v obliki odločitvenih dreves, ki intuitivno predstavijo moč uteži za posamezno skupino.

### *1.6.1 Primerjava stilnih odtisov*

Jezikovne stilne odstise najlažje primerjamo z t.i. radar grafi. Ob tem seveda potrebujemo učno množico, oziroma že obstoječe jezikovne odtise stilov. Ob primerjavi z obstoječimi lahko razberemo kako dobro se vsebina, ki smo jo zajeli iz nekega vira ujema z nekim obstoječim jezikovnim stilnim odtisom. Če se področji pokrivata, potem gre za podoben jezikovni stilni odtis.

Poleg primerjave jezikovnih stilnih odtisov za ugotovitev starosti in spola lahko primerjavo uporabimo za ugotovitev ali gre pri dveh različnih besedilih za delo iste osebe. Oziroma v primeru deljene digitalne osebe, ali so besedila ob različnih časovnih intervalih delo iste ali več oseb.

### *1.6.2 Medmrežni vzorci sodelovanja*

Iz zajetih log datotek lahko pridobimo strukturirane podatke, ki nam razkrijejo določene vzorce vpletenih oseb. Pomembnejši so naslednji vzorci:

- Datum in čas povezave na komunikacijko storitev.
- Število objavljenih sporočil.
- Število deljenih povezav.

Tako lahko spremljamo dinamiko pomembnih vzorcev na podlagi katere lahko sklepamo, ali je to digitalno osebo res uporabljala določena oseba. Recimo, če se oseba vsak dan poveže na komunikacijsko storitev zvečer ob 19:00 uri, nakar je bila poslana grožnja iz strani te digitalne osebe zjutraj ob 08:00, potem nam ti izsledki nakazujejo, da je nekdo v imenu te osebe poslal grožnjo. Druga pomemba posledica je sledenje preklaplanju med različnimi digitalnimi osebami iz strani iste osebe. Namreč s pomočjo časovnice lahko ugotovimo ali obstaja sum, da je ista oseba povezana z večjim številom digitalnih oseb na podlagi nekaj sekundnem zamiku med prijavo x digitalne osebe in odjavo y digitalne osebe.

## 1.7 Profiliranje kibernetskih kriminalcev in žrtev

Isis orodje omogoča avtomatsko profiliranje kriminalcev in žrtev na podlagi zajete komunikacije. Celoten profil je zgrajen na podlagi naslednjih elementov:

- Uporaba jezika. Uporaba določenih izrazov (žargon), ljudi, krajev, pogosto uporabljene besede, URL naslovi, itd.
- Starost/spol analiza. Uporaba odločitvenega drevesa, kjer orodje poudari najverjetnejšo pot/vejo, toda ob tem prikaže še verjetnosti za preostale starostne intervale in tako omogoči preiskovalcu natančen vpogled kako je orodje razdelil verjetnosti na posamezne časovne intervale.
- Aktivnost na medmrežju. Kdaj se je digitalna oseba napogosteje pojavljala v komunikacijski storitvi v 24h časovnem oknu in v katerih dnevih v tednu.

# 2. RAZLIKOVANJE PRISTNIH DIGITALNIH OSEB OD ZAVAJUJOčEGA OBNAŁANJA: ISIS ORODJE V PRAKSI

Primer uporabe Isis orodja v praksi in rezultati. Naredili so dva testa:

- Test v katerem ni zavajujočega obnašanja, torej digitalne osebe se ne izdajajo za 3. osebo.
- Test v katerem se oseba izdaja za 3. osebo.

## 2.1 Klasifikacija starosti in spola osebe, ki se ne izdaja

Uporabili so bazo British National Corpus (BNC), ki vsebuje kar 100 miljonov besed in zajema večji del Britanske angleščine. Za testiranje so uporabili 1684 oseb, kjer so podatki o njihovi starosti in spolu že znani. Uporabili so navzkrižno testiranje, kjer so trenirali model na podlagi vsebine vseh 1684 oseb razen enega, za katerega so želeli klasificirati starost in spol. Za takšno klasifikacijo nam orodje pripravi razpon verjetnosti za posamezna starostna obdobja kot so 11-18, 19-30, potem recimo znotraj intervala 15-18 itd. Za analizo pravilne klasifikacije so uporabili precision in recall. Pomemben je tudi določen prag oziroma mejna vrednost, s katerim določimo večjo natančnost toda ob tem žrtvujem število opravljenih klasifikacij (t.i. neznane vrednosti).

## 2.2 Klasifikacija oseb, ki so se izdajale za drugo osebo

Naredili so nadzorovani test, ki je po naravi podoben Turingovem testu. Vzeli so skupino otrok, v starosti 11-18 let in skupino 10 oseb, kjer jih je polovica bila otrok iste starostne kategorije, medtem ko ostalih 5 se je izdajalo za otroke, torej so bile v resnici starejše osebe. Izkazalo se je, da je Isis orodje precej bolj natančno klasificiralo dejansko starost in spol osebe na drugi strani komunikacije kot pa otroci.

# 3. UPORABA TEHNIK ANALIZE NA PODLAGI OMREŽJA ZA ODKRIVANJE ORGANIZACIJSKIH STRUKTUR NA PODLAGI PRIKRITIH MREž

## 3.1 Uvod

Predstavili bomo tehniko, kako pridemo iz množice teksta (besedila) do mrežne predstavitve prikritih socialnih mrež, ki so implicitno vsebovane v podatkih. Z posebno metodo imenovano Meta-matrična besedilna analiza pridobimo meta-matrični model. To metodo potem apliciramo za odkrivanje socialnih povezav in prikritih mrež za eno izmed bližnjevzhodnih držav.

## 3.2 Metodologija

Strojno berljiva besedila so v današnjem času po zaslugi razvoja informacijske tehnologije na voljo v velikih količinah. Tako so se sčasoma pojavile metode za odkrivanje organizacijskih struktur iz velike količine besedil. Ena izmed takšnih metod je Mrežna tekstovna analiza (Network Text Analysis - NTA), ki deluje na principu kodiranja relacij med besedami v besedilu in izgradnjo omrežja povezanih besed. Temelji na predpostavki da lahko jezik in znanje modeliramo kot omrežje besed in povezav med njimi. Obstaja več različnih NTA metod, kot so:

- Centering Reasonance Analysis
- Functional Depiction
- Knowledge Graphing
- Map Analysis
- Network Evaluation

Poleg tega je zelo pomembna vizualizacija prikritih omrežij, saj s tem dobimo dober vizualni pregled kdo je s kom povezan.

Za analizo povezave med besedami so uporabili analizo preslikave, ki izvleče in analizira povezave med besedami. To tehniko so implementiali v programski rešitvi z imenom AutoMap. Koncepte iz besedila so predstavili kot entitete in povezave med entitetami. Primeri takšnih entitet so:

- ljudje
- skupine ljudi
- aktivnosti
- viri

Uporabijo meta-matrični model v kombinaciji z metodologijo analize presikave. Tej kombinaciji pravijo Meta-matrična besedilna analiza in omogoča tudi izvleči socialno in organizacijsko strukturo iz besedil. To tehniko so implementirali v prej navedeno rešitev AutoMap. Implementirali so naslednje entitete iz meta-matričnega modela:

- agenti
- znanje
- viri
- opravila/dogodki
- organizacije
- lokacije
- akcije
- vloge
- atributi

## 3.3 Podatki

Podatki oziroma besedila so bila izbrana iz različnih virov kot so:

- Časpoisi
- Magazini
- Članki
- Knjige
- Besedila iz spletnih strani
- Transkripti iz sojenj

Besedilne vire so pridobili glede na imena in priimke 116 oseb, ki so bila ključna za Bližnje vzhodno državo v zadnjih 25 letih na področju terorizma. Vsega skupaj so pridobili 247 besedil.

## 3.4 Nastavitve za omrežno besedilno analizo

Prvi koraki je pred-procesiranje podatkov. S tem omogočimo lažjo interpretacijo in primerjavo med besedili. Najprej so uporabili tehniko brisanja, kjer so odstranili koncepte, ki so nepomembni za vsebino besedila kot so ločila. Nato so poenotili izraze, torej različne besede z istim pomen so poenotili. Z uporaba funkcionalnosti prepoznavanja entitet so dobili končni seznam imen, numeričnih vrednosti in kratic vsebovanih v besedilih. Uporabili so thesaurus kolekcijo, ki poveže besedilne koncepte z koncepti na višjem nivoju.

Obstajata dva glavna namena takšne generalizacije:

- Torej tako predstavimo aliase, sinonime, napačno črkovane besede z istimi besedami.
- Pretvorba zveze besed, ki predstavljajo več konceptov v en sam koncept.

Nato zgradimo meta-matrični thesaurus (tabela), ki klasificira koncepte iz zajetega besedila v entitetne razrede metamatrike. Namen tega je izvleči socialno in organizacijsko strukturo iz besedila. Po izvršitvi zgoraj navedenih nastavitev izvedemo meta matrično analizo na besedilu.

## 3.5 Rezultati analize

Na podlagi analize dobimo unikatne koncepte iz posameznega besedila ter entitete kot so vloga in lastnosti. Na podlagi tega lahko zgradimo socialno mrežo. Kot lahko vidimo na dejanskem primeru na spodnji sliki imamo ločeno socialno mrežo, ki ni povezana z ostalimi. Ta mreža ljudi je dejansko bila odgovorna za bombni napada na Khobar stolpnico v Saudovi Arabiji leta 1996.

# 4. SPLETNE DATING PREVARE

Spletne dating prevare so relativno nova oblika goljufije, ki se je pojavila okoli leta 2008. Pri tem kaznivem dejanju se kriminalci pretvarjajo, da preko internetnih dating strani iščejo romantično razmerje, potem pa ogoljufajo svoje žrtve za velike vsote denarja. Članek predstavlja nekaj opisne statistike odkrivanja in pregona internetnih dating prevar v Veliki Britaniji. Študija opisana v članku je pokazala, da je, kljub relativni novosti, ocenjeno število žrtev teh kaznivih dejanj okoli 230.000 britanskih državljanov. Sklepamo, da bi bilo potrebno premisliti o tem, kako zagotoviti možnosti prijavljanja tovrstnih kaznivih dejanj ali vsaj narediti postopke žrtvam bolj prijazne.

Britanska Nacionalna agencija za boj proti goljufijam je leta 2011 ocenila, da stroški, povezani z goljufijami, znašajo več kot 38 milijard funtov letno. To se je dramatično povečalo od leta 2006, ko je Urad Združenega kraljestva za pošteno trgovanje napovedal da Britanski potrošniki zaradi prevar izgubijo približno 3,5 milijard funtov letno, povprečni znesek na prevaro pa znaša 850 funtov. V istem poročilu, napisanem leta 2006, je bilo ugotovljeno, da je delež ljudi, za katere bi bilo verjetno, da bodo tarče tovrstne prevare, kar 48 odstotkov odraslih prebivalcev v Veliki Britaniji (23,5 milijona ljudi), in 8 odstotkov odraslih prebivalcev je priznalo, da so že bili žrtve take prevare. Na osnovi teh ugotovitev so ocenili, da se bo delež ogoljufanih v Veliki Britaniji vsako leto povečal za 6,5 odstotkov.

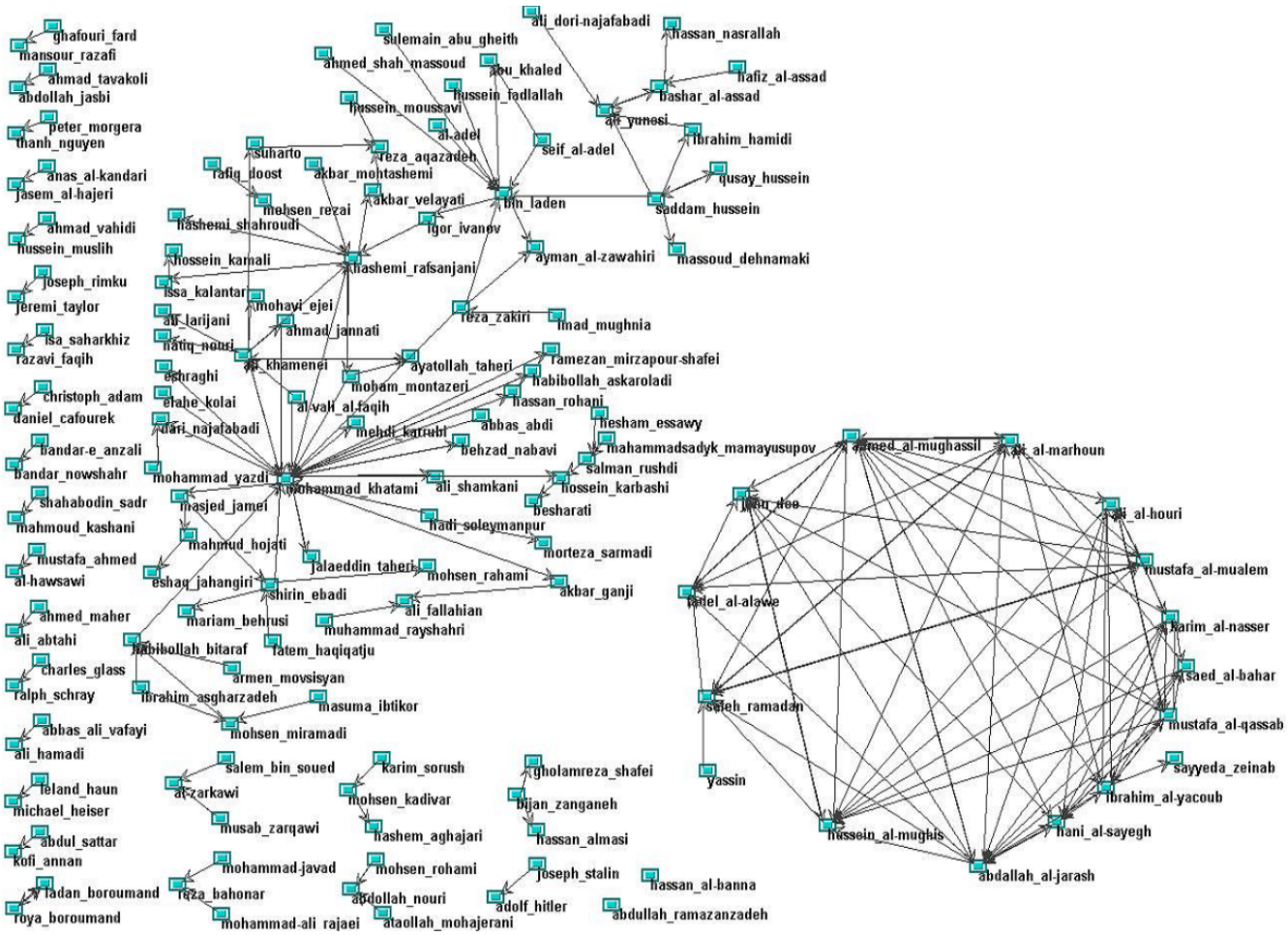
Članek se osredotoča na eno hujših kaznivih dejanj, storjenih preko interneta: internetne romantične prevare, ki se je pojavila okoli leta 2008 in ima svoje korenine v poštni goljufiji. Pri tem se kriminalci pretvarjajo, da jih zanima romantično razmerje preko internetnih dating strani, potem pa goljufajo svoje žrtve za večje vsote denarja. Žrtve tako zaradi tega zločina prejmejo "dvojni udarec": izgubo denarja, in tudi izgubo odnosa. Med 1. 4. 2010 in 1. 4. 2011 so v Veliki Britaniji na Action Fraud identificirali 592 žrtev tega kriminalnega dejanja. Od tega je 203 žrtev izgubilo več kot 5000 funtov. Na Action Fraud verjamejo, da je tovrstnega kriminala dejansko veliko več, vendar ga žrtve ne prijavijo (prijava kaznivega dejanja praviloma zahteva stik z lokalno policijo in podajanje izjave). Po ocenah SOCA (Agencija za organiziran kriminal v Veliki Britaniji) bi lahko finančne izgube segale od 50 pa vse do 240 tisoč funtov.

Rege, v enem od redkih znanstvenih člankov na to temo, iz spletnih strani črpa informacije, z namenom opisati goljufijo. Goljufivec kontaktira žrtev in z njo vzpostavi močne vezi, ki lahko trajajo 6-8 mesecev. Potem prosijo žrtev za denar, s tem da ustvarijo zgodbo o tragičnih ali obupnih okoliščinah kot na primer kraja osebnih dokumentov med potovanjem ali nepričakovanih potnih, oziroma bolnišničnih stroških. Nastala škoda je lahko ogromna preden je prevara razkrita. Rege opisuje to kot čustveno uničujočo goljufijo.

Action Fraud verjame, da število žrtev narašča. To se zdi verjetno, glede na to, da so internetne zmenkarije postale zelo priljubljena dejavnost in skoraj normativ v zahodni družbi. Dutton in Helsper poročata, da je 23% uporabnikov interneta v Veliki Britaniji že spoznalo koga preko interneta, ki ga niso poznali že prej. Dutton, Helsper, Whitty, Buckwalter in Lee so odkrili, da je 6% poročenih parov, ki uporabljajo internet v Veliki Britaniji, svoje partnerje spoznalo na spletu.

Ta študija ima dva cilja. Prvi je, da se ugotovi razširjenost spletnih ljubezenskih prevar v Veliki Britaniji. Trenutno lahko SOCA in Action Fraud število samo ocenjujeta na podlagi prijav žrtev in sledenja od znanih goljufov nazaj k žrtvam. Da bi se lahko ocenila resnost problema in zagotovile izhodiščne številke za postavitev prihodnih preprečevalnih ukrepov bi bilo potrebno natančneje ugotoviti stopnjo razširjenosti. Drugi cilj je ugotoviti, koliko je splošno znanega o tem kaznivem dejanju in oceniti, kako se posamezniki seznanjajo z problemom. Goljufije imajo veliko manj verjetnosti, da bodo uspešne, če posamezniki slišijo o njih preden so kontaktirani. SOCA in Action Fraud so se do sedaj posluževale naslednjih medijev za obveščanje javnosti: televizija, časopisi, revije, online dating strani, in internet. Razumevanje učinkovitosti teh medijev pri ozaveščanju lahko nakaže prihodnje strategije.

Študija je osnovana na reprezentativnem vzorcu odraslih v Veliki Britaniji. Vzorec je zajemal 2028 britanskih odraslih, starih 18 let in več. Neobtežena demografija je bila sledeča: v vzorec je bilo vključenih 902 moških in 1.126 žensk. Med udeleženci je bilo 205 starih od 18. do 24. leta, 300 starih 25-34, 285 starih 35-44, 352 starih 45-54 in 886 v starosti nad 55 let. Glede na status o zaposlitvi je vključevala 1087 zaposlena za polni delovni čas, 216 s krajšim delovnim časom, 75 študentov, 476 upokojencev, 39 brezposelnih oseb




**Slika 4: Socialna mreža ljudi.**

in 135 ostalih. Status vključuje 1.047 poročenih / civilnih partnerjev, 225, ki živijo v izvenzakonski skupnosti, 208 ločenih, 67 vdovelih, 472 nikoli poročenih in 9, ki niso želeli odgovoriti.

Vsi udeleženci so morali odgovoriti na naslednja vprašanja: "V internetnih dating prevarah kriminalci prevzemajo lažne identitete z uporabo ukradenih fotografij (pogosto modelov ali častnikov) in se pretvarjajo, da želijo vzpostaviti romantičen odnos z njihovo žrtvijo. Pogosto pri tem uporabijo spletne strani za zmenkarije in socialna omrežja. Na neki točki se začnejo pretvarjati, da nujno potrebujejo denar in prosijo za pomoč. Številne posameznike uspejo prepričati v to, da jim dajo večje vsote denarja, preden karkoli posumijo. Ali ste, preden ste privolili v sodelovanje pri tej raziskavi, že slišali za internetne dating prevare?"Udeleženci so lahko odgovorili bodisi: "Ja, sem", bodisi "Ne, nisem".

Po tem so bili udeleženci, ki so že slišali za internetne dating prevare, nadalje vprašani: "Kje ste v preteklosti že slišali za internetne dating prevare, preden ste sodelovali pri ter anketi?"Možni so bili naslednji odgovori: od prijatelja, po televiziji, v časopisu, v reviji, na internetu, na online dating strani, preko radia, drugo ali se ne spomnim.

Na koncu so vsi udeleženci bili vprašani še: "Ali ste vi ali kdorkoli ki ga poznate kdaj bili žrtev internetne romantične prevare (tj. Ogoljufani za kakršenkoli znesek denarja)?"

Pri raziskavi smo uporabili storitve YouGov (www.yougov.co .uk), strokovno raziskovalno in svetovalno organizacijo, ki omogoča kontakt s skupino ljudi, ki so pripravljeni sodelovati v spletnih anketah. Vprašanja so bila postavljena kot del razširjene ankete, skupaj z drugimi vprašanji na več nepovezanih temah. Podatki so bili zbrani od 6. do 8. julija 2011. K sodelovanju so bili povabljeni kandidati, naključno izbrani iz bazena ljudi iz cele Velike Britanije, v raziskavo je bilo vključenih približno 300.000 posameznikov. Vzorec odgovarjajočih je bil dovolj obtežen, da se zagotovi reprezentativen vzorec britanskih odraslih, starih 18 let in več (na podlagi podatkov popisa). Stopnja odzivnosti na anketo je približno 33,3 odstotkov.

Vsi sodelujoči so bili vprašani, ali so oni izgubili denar ali osebno poznajo koga, ki je izgubil denar zaradi spletne dating goljufije. Na splošno smo ugotovili, da je bilo ogoljufanih 0,65 % sodelujočih. To je ekvivalentno skoraj 230.000 ljudi.

Študija je pokazala, da je 51.66 % Britancev slišalo za spletni dating prevare.

Na splošno je 1051 udeležencev izjavilo, da so slišali za spletne dating goljufije preden so odgovarjali na anketo. Ti posamezniki so bili potem vprašani, kje so v preteklosti že slišali za spletne dating prevare, in da naj izberejo čim več možnosti. V tabeli 2 so predstavljene splošne ugotovitve za to vprašanje. Chi-square analiza je pokazala, da je več žensk kot moških slišalo za prevaro prek revij; x $\hat{2}$ (1, N=1048) = 6,52, p $<$ 0,01.

Njihovi podatki kažejo, da bi do sredine leta 2011 lahko bilo približno 230.000 britanskih državljanov žrtev tega novega kriminala. Te številke so precej višje kot jih ocenjujeta SOCA ali Action Fraud in potrjuje sume Action Frauda, da se tovrstnega kriminal ne prijavi vedno in je torej bolj resen kot so najprej mislili. To je zaskrbljujoče, ne samo zato, ker ljudje izgubljajo velike vsote denarja zaradi teh zločincev, temveč tudi zaradi psihološkega učinka, ki ga doživijo žrtve ljubezenskih prevar. Žrtve teh prevar, za razliko od vseh ostalih, prejmejo "dvojni udarec": izgubo denarja in romantičnega razmerja. Najverjetneje je, da žrtve sramoto in razburjenje, ki ga doživijo ob tem, odvrača od prijave kaznivega dejanja. Prepričani smo, da so zato potrebni novi načini poročanja kaznivega dejanja.

Rezultati kažejo, da opozorila o goljufiji dosegajo približno polovico britanskega prebivalstva. To je en pokazatelj za prihodnje raziskave o preventivnih ukrepih. Prav mogoče je, seveda, da samo znanje o kaznivem dejanju, tega ne preprečuje, vendar pa je pomembno primerjati poznavanje kriminala in število žrtev v prihodnjih študijah.

Ugotovitve dokazujejo, da je sporočilo uspešno dostavljeno prek večine medijev, ki so jih uporabili pri SOCA in Action Fraud. Revije so izjema, in morda je to zato, ker kot je razvidno iz razlik med spoloma ki smo jih omenili tukaj, so veliko bolj osredotočeni na revije, ki jih berejo ženske. Online dating strani so bile precej nizko, verjetno zato, ker le majhen delež vzorca uporablja online dating strani (trenutno se izvaja študija, za namensko vzorčenje za bolj poglobljen vpliv in vidik žrtev prek dating strani). Podatki kažejo, da bi bilo mogoče radio še boljše izkoristiti za širjenje znanja o spletnih dating prevarah.

Končno, spletne dating prevare so relativno nove in še veliko več raziskav je potrebnih, da se naučimo tako o temu kateri ljudje so ranljivi, kot tudi o psihološkem vplivu na žrtve. Študija je pokazala, da so stopnje razširjenosti precej višje od predhodnih ocen, ki so v glavnem temeljile na poročanju o kaznivih dejanj pri policiji. Ta velika razlika kaže, da se kaznivo dejanje premalo poroča, in opozarja, da mora policija žrtvam omogočiti, da lahko lažje prijavijo kaznivo dejanje.

# 5. POVZETEK

Z prihodom socialnih omrežij je postala zloraba digitalne identitete pomemben del taktike kriminalnih organizacij. Tako je postala potreba po orodjih, ki so zmožna analizirati in klasificirati pomembne atribute digitalne osebe, kot je spol in starost. Pokazili smo, da je s pomočjo takšnih orodij možno doseči zelo dobre rezultate. Izkazalo se je, da orodje deluje bolje, kadar se oseba izdaja za drugo osebo, torej uporablja taktiko zavajanja. Toda zaradi narave človeškega jezika ta orodja ne dosegajo 100% natančnosti pri klasifikaciji. Tako pridemo do zadnje ključne točke, da je ekspertno znanje še vedno nepogrešljivi del raziskovanja. Ugotovimo lahko, da so ta orodja najmočnejša ravno v kombinaciji z bogatimi izkušnjami kriminalistov.

# 6. LITERATURA


[1] Rashid, Awais, et al. "Who am I? Analysing Digital Personas in Cybercrime Investigations."(2013): 1-1.

[2] Rashid, Awais, et al. "Technological solutions to offending."(2012): 228-243.

[3] Rashid, Awais, et al. "Isis: Protecting Children in Online Social Networks."(2009).

[4] Whitty, Monica T., and Tom Buchanan. "The online romance scam: A serious cybercrime.ČyberPsychology, Behavior, and Social Networking 15.3 (2012): 181-183.

[5] Weimann, Gabriel, and Katharina Von Knop. "Applying the notion of noise to countering online terrorism.Študies in Conflict and Terrorism 31.10 (2008): 883-902.

[6] Davulcu, Hasan, et al. "Analyzing sentiment markers describing radical and counter-radical elements in online news.Šocial Computing (SocialCom), 2010 IEEE Second International Conference on. IEEE, 2010.

[7] Diesner, Jana, and Kathleen M. Carley. "Using network text analysis to detect the organizational structure of covert networks."Proceedings of the North American Association for Computational Social and Organizational Science (NAACSOS) Conference, Pittsburgh, PA. 2004.

[8] Bisgin, Halil, Nitin Agarwal, and Xiaowei Xu. "A study of homophily on social media."World Wide Web 15.2 (2012): 213-232.

[9] Afroz, Sadia, Michael Brennan, and Rachel Greenstadt. "Detecting hoaxes, frauds, and deception in writing style online.Šecurity and Privacy (SP), 2012 IEEE Symposium on. IEEE, 2012.

# Ocenjevanje velikosti človeka na podlagi posnetka nadzorne kamere

## Študija zanesljivosti

Matevž Černe
Fakulteta za računalništvo in informatiko
Tržaška cesta 25
Ljubljana, Slovenija
matevz.cerne@student.uni-lj.si

Klemen Marolt
Fakulteta za računalništvo in informatiko
Tržaška cesta 25
Ljubljana, Slovenija
klemzi.marolt@gmail.com

## POVZETEK

Cilj študije je bil ovrednotiti merjenje telesne višine s pomočjo metode SVM in preučiti vpliv telesne drže pri stanju, hoji in teku.

Deset moških so posneli s pomočjo 2D spletne kamere in sistema za analizo 3D premikanja po prostoru. Naredili so šest poskusov, tri v stoječem položaju in tri med hojo in tekom. Telesna višina je bila izmerjena s pomočjo metode SVM in Qualisys Track Managerja (QTM). Na koncu so izmerjene telesne višine primerjali z dejansko višino.

Telesna višina izmerjena z metodo SVM je bila znantno višja kot tista izmerjena s pomočjo QTM. Izmerjena višina pri stoji na miru in hoji pa je bila kljub temu znatno nižja od dejanske telesne višine. Pri teku pa razlike ni bilo zaznati.

Metoda SVM je v povprečju izračunala telesno višino z napako +2.30 cm. Telesna drža vpliva na telesno višino. Najboljši položaj za merjenje telesne višine pri hoji je v fazi prenosa noge naprej, pri teku pa v fazi, ko se nobena od nog ne dotika tal.

### Ključne besede

ocenjevanje telesne višine, metoda SVM, telesna drža

## 1. PREDSTAVITEV

Telesna višina je pomembna pri odkrivanju identitete storilcev kaznivih dejanj na posnetkih nadzornih kamer. Definicija telesne višine je vertikalna višina od tal do vrha glave, ko je oseba v pokončnem položaju, noge skupaj, stegnjena kolena. Ta pozicija se sicer redko pojavlja pri našem naravnem gibanju. Zaradi različnih drž pri stanju na miru in hojo ter izgubo informacije pri zajemanju 2D slike je ocena telesne višine na posnetkih nadzornih kamer otežena.

Število nadzornih kamer vsakodnevno narašča, tako v javnih kot tudi zasebnih prostorih. Kljub temu pa je kvaliteta teh kamer v številnih primerih še vedno preslaba za identifikacijo storilcev.

Na Švedskem posnetek osumljenca pošljejo v analizo Nacionalnemu laboratoriju za forenzično znanost v Linköpingu. V kolikor je kvaliteta posnetka preslaba, da bi lahko z njegovo pomočjo identificirali storilca, potem ocenijo njegovo telesno višino, pri čemer pa ne upoštevajo storilčeve drže.

## 2. OZADJE PODROČJA

Obstaja več metod s katerimi ocenjujemo telesno višino na slikah. V tej študiji so se lotili metode SVM (single view metrology method), ki jo uporabljajo v Švedskem nacionalnem laboratoriju za forenzične vede.

Metoda zahteva, da policist s posebno kalibracijsko palico obišče prizorišče zločina. Namenjena je kalibraciji metode SVM saj poznamo njene mere. Palica je prikazna na sliki 1

Popraviti je potrebno tudi popačenja, ki jih naredi leča. To pomeni, da linije, ki so dejansko ravne, take naredimo tudi na sliki. Potrebujemo dovolj linij da lahko ustvarimo perspektivo znotraj slike. Poleg tega moramo s pomočjo algoritmov izračunati bežišče in horizont. Kako je bilo to narejeno na testni sobi prikazuje slika 2

Bežišče je točka kjer se v 2D svetu sekajo linije, ki so vzporedne v 3D svetu. Horizont je premica, ki gre skozi bežišči horizontalnih robov v prostoru. Laično je to prikazano na sliki 3

S pomočjo kalibracijske palice dobimo referenčno višino, ki nam omogoči merjenje telesne višine.

## 3. CILJI ŠTUDIJE

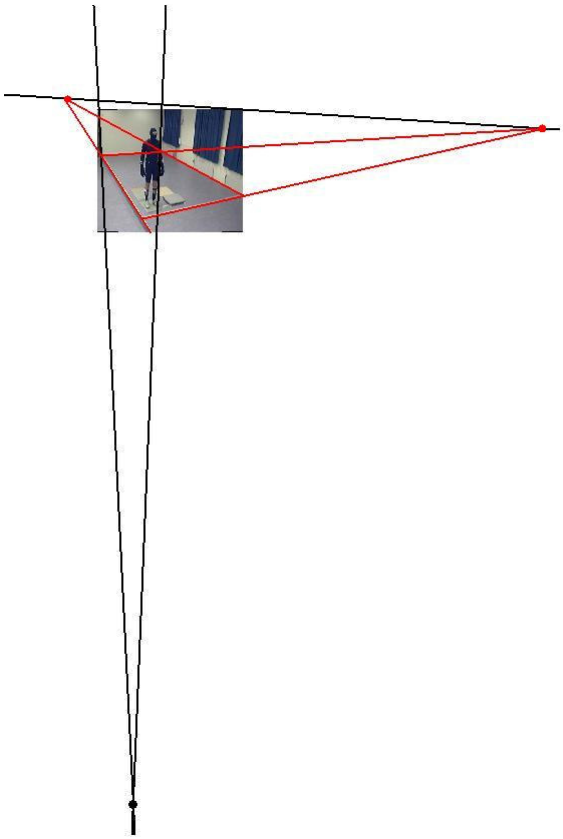
Cilj študije je bil oceniti meritve telesnih višin, ki jih dobimo z metodo SVM ter raziskati vpliv drže na oceno telesne višine pri stanju na miru, hoji in teku.

Odgovorili so na naslednja vprašanja:

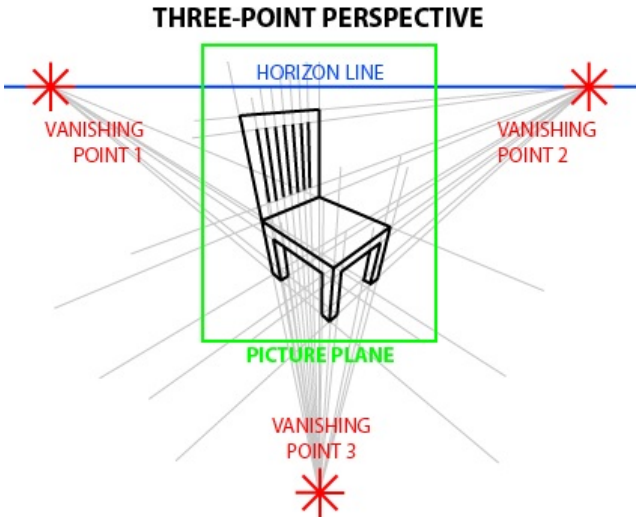
- Kako natančna je metoda SVM pri ocenjevanje telesne višine oseb na slikah?



**Slika 1: Prikaz testne sobe. Poleg tega je v levem sponjem kotu testnega področja prikazana tudi kalibracijska palica.**



**Slika 2: Bežišče in horizont testne sobe.**




**Slika 3: Laični prikaz bežišča in horizonta.**

- Kako drža vpliva na višino osebe v primerjavi z dejansko telesno višino te osebe ko le-ta stoji, hodi in teče?
- Kateri položaj med hojo in tekom se najbolje ujema z dejansko višino?

Za ovrednotejnje metod so postavili tudi naslednji hipotezi:

- H0: Ni bistvene razlike med metodami za merjenje telesne višine.
- H1: Je bistvena razlika med metodami za merjenje telesne višine.

Za primerjavo med telesno višino in vplivom telesne drže na le-to pa naslednji hipotezi:

- H0: Ni bistvene razlike v telesni višini pri različnih držah.
- H1: Je bistvena razlika v telesni višini pri različnih držah.

## 4. METODE

V raziskavi so uporabili deset zdravih moških brez znanih patoloških pojavov pri hoji. Izbrani moški so morali biti stari med 15 in 30 let, saj je to prevladujoča skupina osumljencev v kazenskih preiskavah. Seznam oseb in njihovi podatki so prikazani v tabeli 1

Povprečna starost moških je bila 25,6 let, povprečna višina 180,7 cm in povprečen indeks telesne mase 23,43.

Osebe so bile oblečene v enako roparsko opravo - črna oblačila, nosile so rokavice in podkapo, da so si bile čim bolj podobne. Vsak moški je nosil svoj par čevljev, da bi se v njih počutil udobno in bi tako hodil bolj sproščeno in normalno.

**Table 1: Podatki oseb**

| Št. osebe | Višina [cm] | Teža [kg] | ITM | Starost |
|---|---|---|---|---|
| 1 | 176.2 | 68.0 | 21.90 | 21 |
| 2 | 164.5 | 66.5 | 24.57 | 28 |
| 3 | 186.6 | 85.5 | 24.55 | 21 |
| 4 | 185.9 | 82.0 | 23.72 | 27 |
| 5 | 185.0 | 84.9 | 24.82 | 25 |
| 6 | 175.4 | 67.3 | 21.87 | 21 |
| 7 | 189.7 | 70.9 | 19.71 | 33 |
| 8 | 168.0 | 62.7 | 22.22 | 28 |
| 9 | 182.7 | 83.5 | 25.01 | 24 |
| 10 | 172.0 | 75.7 | 25.60 | 28 |



**Slika 4: Moški v roparski opravi.**

Razliko v višini zaradi obutve je težko oceniti, zato so po navadi meritve višine osumljencev opravljene z obutimi čevlji. (Primer osebe je na sliki 4)

Osebe so bile posnete v šestih različnih pozicijah. Meritve se je izvajalo v treh pozicija, ko oseba stoji in treh, ko se oseba premika. Pozicije so podrobneje opisane v tabeli 2. Te pozicije so bile izbrane zato, ker se najbolj pogosto pojavljajo na posnetkih nadzornih kamer in so zato najbolj zanimive za raziskovanje. Prve tri pozicije so bile združene v en posnetek vpodanem zaporedju. Hoja in tek pa sta bila posneta ločeno. Pri tem pozicija številka 1 predstavlja dejansko višino osebe. Vsi posnetki so bili narejeno vsaj dvakrat. Na posnetku gibanja pa sta posneta vsaj koraka pri hoji oz. vsaj en korak pri teku.

Za vsako sliko iz varnostne kamere so s pomočjo metode QTM izmerili višino osebe v tistem trenutku.

**Table 2: Šest pozicij v katerih so bile osebe posnete.**

| Pozicija | Opis |
|---|---|
| Stanje pokonci | oseba stoji pokonci in ima stopala 5 cm narazen |
| Stanje sproščeno | oseba stoji sproščeno, noge so bolj narazen in teža na obeh stopalih |
| Teža na eni nogi | oseba stoji sproščeno s težo na eni nogi |
| Počasna hoj | oseba hodi počasi |
| Hitra hoja | oseba hodi hitro |
| Tek | oseba teče |

Med hojo se velikost osebe na sliki spreminja glede na položaj v katerem je. Največja razlika je takrat ko je oseba v koraku (je v najnižji poziciji) in takrat ko pride do prenosa noge naprej (najvišja pozicija). Zaradi teh razlik se je višina merila v šestih različnih položajih:

- Prvi dotik stopala s tlemi,
- stopalo na tleh,
- prenos noge naprej,
- dvignjena peta,
- dvignjeno stopalo,
- prenos druge noge naprej.

Za boljšo predstavo so položaji skicirani na sliki 5

Za ugotavljanje kateri položaj pri hoji se najbolj ujema z dejansko višino osumljenca so izmerili maksimalno višino in jo potem primerjali s telesno višino osebe, ko sproščeno stoji s težo na obeh nogah. Rezultati so pokazali, da se z dejansko višino najbolje ujema položaj, ko se prenese noga naprej.

# 5. REZULTATI

## 5.1 Ovrednotenje metode SVM

Povprečna višina osebe na sliki izmerjena z metodo SVM je bila 176.7 cm, povprečna dejanska višina osebe v istem trenutku pa je bila 174.4 cm. Izkazalo se je, da je bila izmerjena višina z metodo SVM pri 57 od 60 meritev višja kot dejanska višina. Povpračna napaka pa je bila 2.30 cm.

Med metodama je visok koeficient korelacije. Primerjava metod je prikazana na sliki 6, kjer opazimo, da regresijska premica leži nad premico $x = y$. Tudi iz tega lahko sklepamo, da so bile meritve z metodo SVM višje od dejanskih. Je pa lepo vidno, da je napaka precej konstantna.

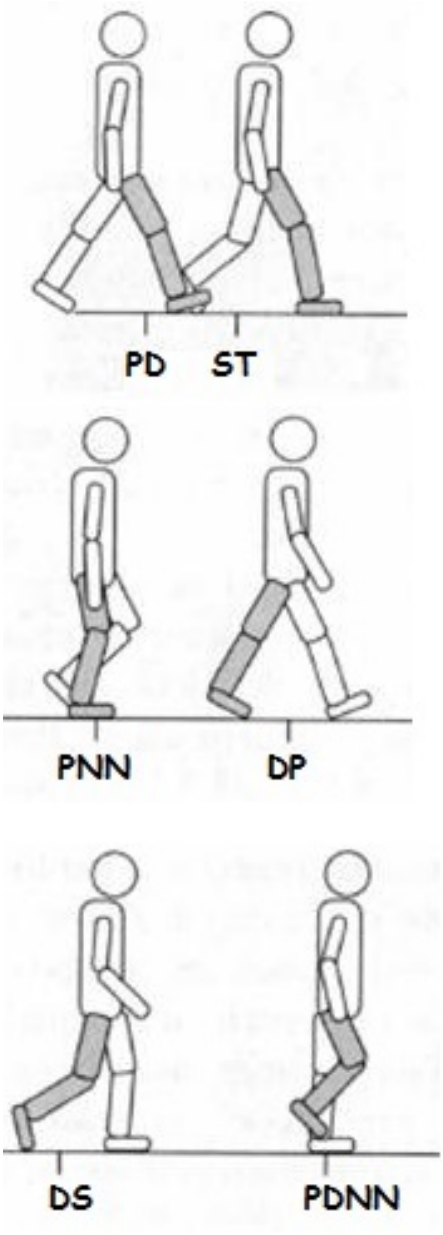
Slika prikazuje korelacijo med višino izmerjeno z metodo SVM in dejansko višino v določenem trenutku. Regresijska linija je nad linijo y = x, kar prikazuje, da so bile meritve z metodo SVM konstantno višje.

## 5.2 Vpliv drže na višino

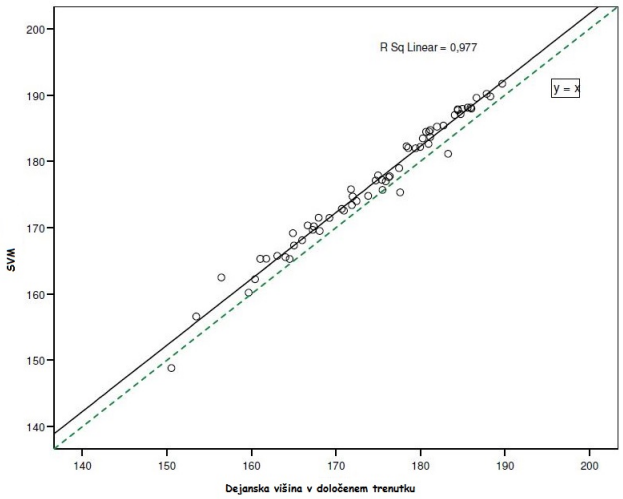
Izvedene so bile primerjave višine v določenem trenutku v različnih gibanjih.

Na sliki 7 je prikazana razlika med dejansko višino oseb, višino ko stojijo sproščeno s težo na obeh nogah in ko stojijo s težo na eni nogi. Vidimo, da je višina osebe na sliki, ki stoji s težo na eni nogi pri 9 od 10 primerov manjša kot pri stoji s težo na obeh nogah.

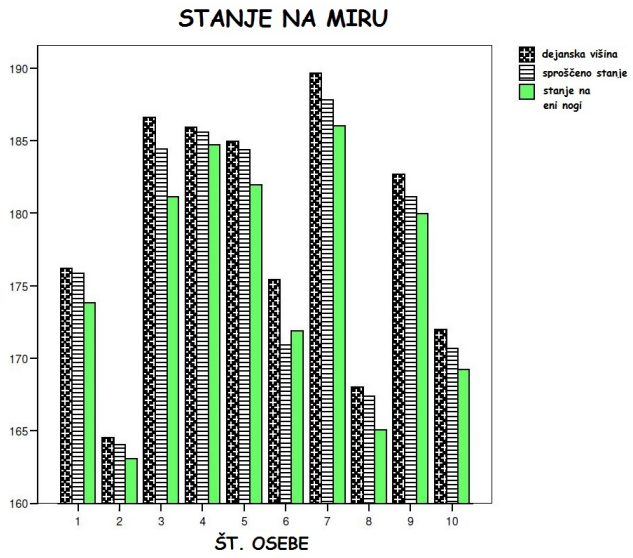
V povprečju izmerjena vertikalna višina pri teži na obeh nogah predstavlja 99,2% dejanske višine, ko pa je teža na eni nogi pa 98,4%. V tabeli 3 so prikazane vrednosti za vseh 10 oseb. Tudi iz teh podatkov, lahko sklepamo, da je vertikalna višina osebe, ob stoji s težo na eni nogi, manjša.




**Slika 5: Izbrani položaji pri hoji.**




**Slika 6: Korelacija med dejansko in izmerjeno višino. Črtkana črta predstavlja premico $x = y$.**




**Slika 7: Primerjava višine oseb med sproščeno stojo in stojo na eni nogi.**

**Table 3: Lev stolpec prikazuje koliko procentov predstavlja vertikalna višina pri sproščeni stoji v primerjavi z dejansko višino. Desni stolpec pa koliko procentov predstavlja stanje s težo na eni nogi.**

| Osebek | % dejanske višine (sproščeno stanje) | % dejanske višine (teža na eni nogi) |
|---|---|---|
| 1 | 99.8 | 98.6 |
| 2 | 99.7 | 99.1 |
| 3 | 98.8 | 97.1 |
| 4 | 99.8 | 99.4 |
| 5 | 99.7 | 98.4 |
| 6 | 97.4 | 98.0 |
| 7 | 99.0 | 98.1 |
| 8 | 99.6 | 98.2 |
| 9 | 99.2 | 98.5 |
| 10 | 99.2 | 98.4 |
| Povprečje | 99.2 | 98.4 |

Pri počasni in hitri hoji opazimo, da se maksimalna vertikalna višina (slow_max in fast_max na slikah 8 in 9) pojavi pri vseh osebah pri prenosu noge naprej (PNN), minimalna višina (slow_min, fast_min) pa ko je dvignjena zadnja peta(DP).

Pri obeh načinih hoje se opazi, da se v položaju prenosa noge naprej (PNN) vertikalna višina najbolje ujema z dejansko višino osebe. Kljub temu pa lahko vidimo opazno razliko med dejansko višino in višino osebe na sliki v poziciji PNN. V povprečju izmerjena vertikalna višina pri počasni hoji predstavlja 99,3% dejanske višine, pri hitri hoji pa 99,0%. V tabeli 4 so prikazane vrednosti za vseh 10 oseb. Iz teh podatkov lahko sklepamo, da so meritve bolj natančne, če oseba hodi počasneje.

Pri teku se maksimalna vertikalna višina(run_max na sliki 10) pojavi, ko je oseba z obema nogama v zraku. Minimalna višina (run_min) pa, ko seba prense nogo naprej (PNN). Kot je bilo že prej omenjeno, lahko tudi na podlagi slike 10 potrdimo, da se maksimalna višina med tekmo najbolje ujema z dejansko višino osebe.

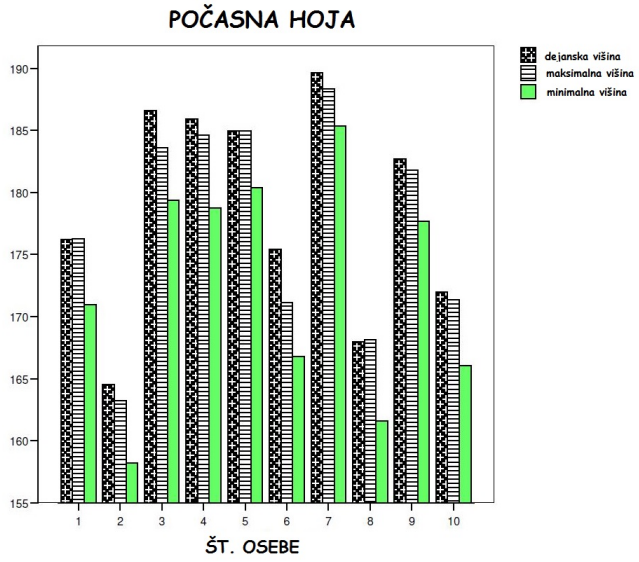
Kot prikazuje tabela 4 je v povprečju izmerjena vertikalna višina pri teku predstavljala 97,4% dejanske višine. Tukaj opazimo, da je izmerjena višina pri teku precej manjša kot pri hoji ali pri stanju.

## 5.3 Spreminjanje višine pri hoji in teku

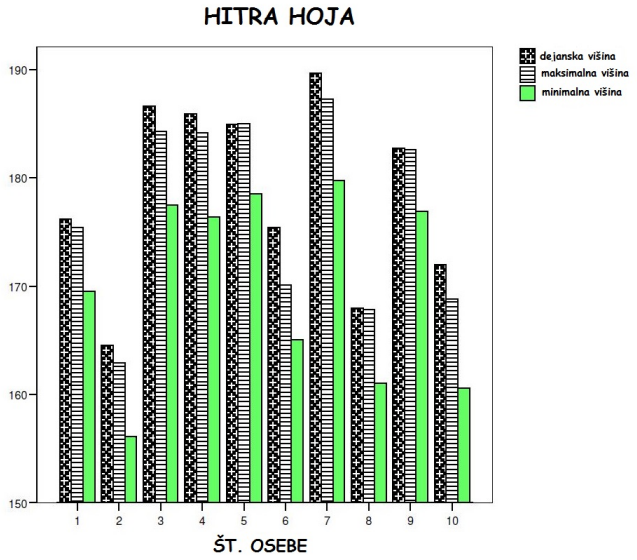
Vse osebe so imele na glavi marker s pomočjo katerega so beležili spreminjanje višine vrha glave. Grafi na sliki 11 prikazujejo spreminjanje višine pri osebi 5 pri počasni hoji, hitri hoji in teku. Vsaka točka prikazuje na kateri višini se v nekem času nahaja marker. Vsaka horizontalna črta označuje en cm.

Opazimo, da višina med gibanjem niha sinusno. Amplituda pa je odvisna od hitrosti gibanja - hitrejša kot je oseba, večja bo amplituda.

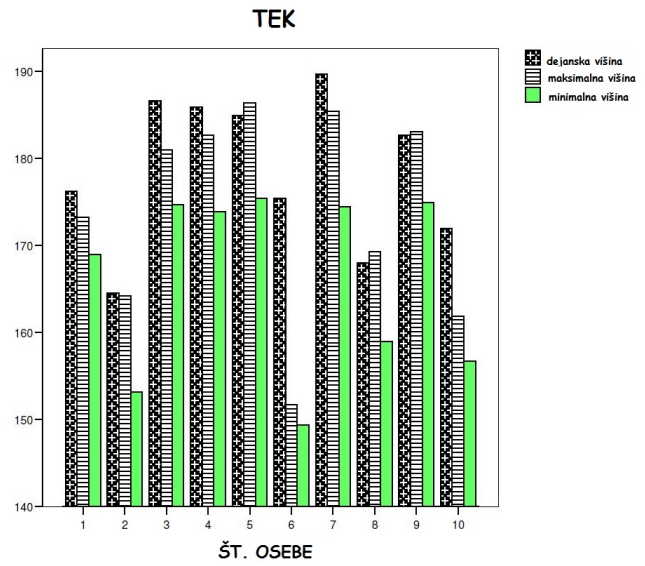
# 6. ZAKLJUČEK




**Slika 8: Primerjava maksimalne, minimalne in dejanske višine v cm pri počasni hoji.**



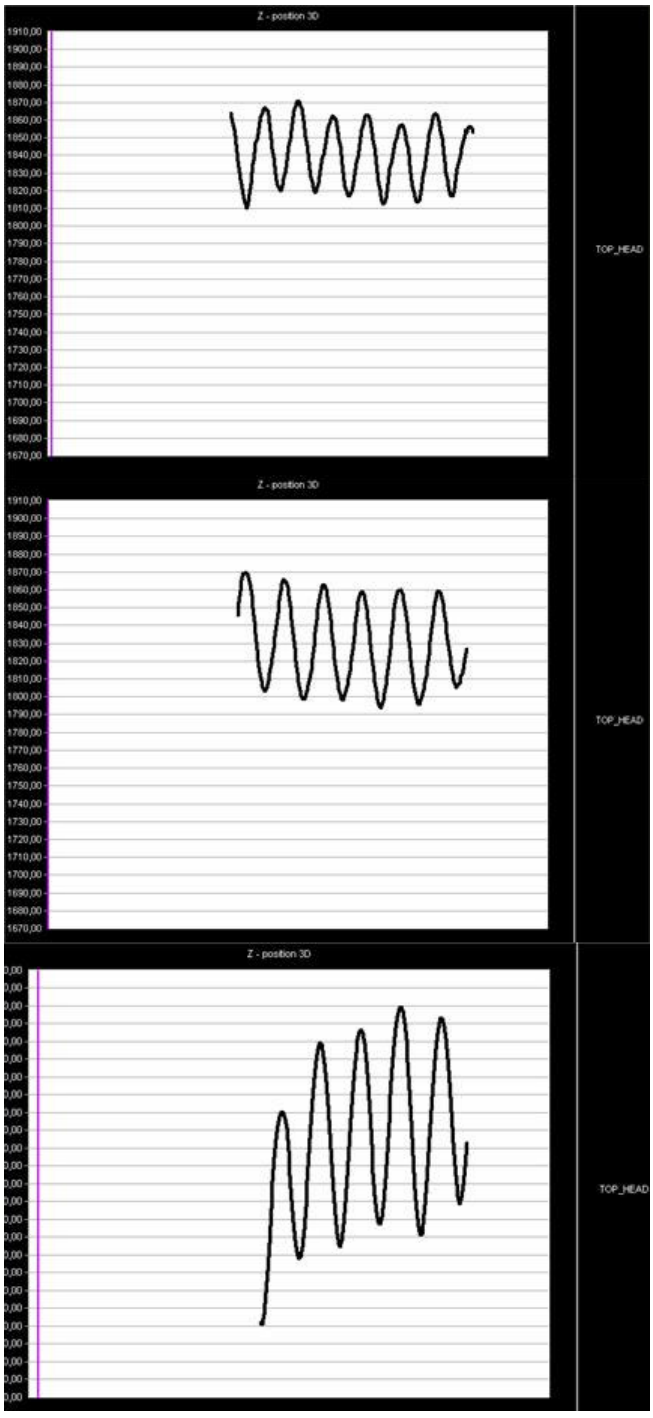

**Slika 9: Primerjava maksimalne, minimalne in dejanske višinev cm pri hitri hoji.**




**Slika 10: Primerjava maksimalne, minimalne in dejanske višine v cm pri teku.**

**Table 4: Lev stolpec prikazuje koliko procentov predstavlja vertikalna višina pri največji izmerjeni višini pri počasni hoji v primerjavi z dejansko višino. Srednji stolpec prikazuje koliko procentov predstavlja maksimalna višina pri hitri hoji. Desni stolpec pa pri teku.**

| Osebek | % dejanske višine(max pri počasni hoji) | % dejanske višine (max pri hitri hoji) | % dejanske višine (max (pri teku) |
|---|---|---|---|
| 1 | 100.0 | 99.6 | 98.3 |
| 2 | 99.2 | 99.0 | 98.8 |
| 3 | 98.4 | 98.8 | 97.0 |
| 4 | 99.3 | 99.0 | 98.3 |
| 5 | 100.0 | 100.0 | 100.8 |
| 6 | 97.6 | 96.9 | 86.5 |
| 7 | 99.3 | 98.7 | 97.8 |
| 8 | 100.1 | 99.9 | 100.8 |
| 9 | 99.5 | 99.9 | 100.2 |
| 10 | 99.7 | 98.2 | 94.1 |
| Povprečje | 99.3 | 99.0 | 97.4 |




**Slika 11: Slike predstavljajo premikanje markerja na glavi med počasno hojo (zgoraj), hitro hojo (na sredi) in tekom (spodaj).**

Problem pri tej raziskavi [11] je bila omejeno število oseb. Če bi želeli bolj natančna opažanja in primerjave med izmerjeno višino in dejansko višino, bi bilo bolje, da bi imeli več kot 10 oseb. Težava pri tej raziskavi je bila tudi kvaliteta kamere. Ta je imela namreč slabšo resolucijo kot večina nadzornih kamer.

Meritve narejene z metodo SVM so opazno višje kot dojanske višine oseb na slikah.

Metoda SVM izmeri višino s povprečno napako +2.30 cm. Največja napaka pa je bila 6.03 cm. Ugotovljeno je bilo tudi, da metoda SVM najbolje oceni višine, ki so večje kot 180 cm.

Vertikalne višine oseb na slikah so bile manjše kot dejanske višine oseb tako pri sproščenem stanju kot tudi pri stanje s težo na eni nogi. Ko oseba stoji sproščeno se njena višina zmanjša na 99.2%, ko stoji sproščeno na eni nogi na 98.4%, ko hodi počasi na 99.3 %, ko hodi hitro na 99.0 %in ko teče na 97.4 %.

Ugotovljeno je bilo, da je najboljše ujemanje med dejansko višino in vertikalno višino na sliki med hojo v položaju premika noge naprej, pri teku pa ko sta obe nogi v zraku.[11]

# 7. LITERATURA


[1] I. Alberink and A. Bolck. Obtaining confidence intervals and likelihood ratios for body height estimations in images. *Forensic Science International*, 177(2-3):228–237, May 2008.

[2] D. Angelis, R. Sala, A. Cantatore, P. Poppa, M. Dufour, M. Grandi, and C. Cattaneo. New method for height estimation of subjects represented in photograms taken from video surveillance systems. *International Journal of Legal Medicine*, 121(6):489–492, Oct. 2007.

[3] J. E. Birren. *Encyclopedia of Gerontology, Two-Volume Set: Age, Aging, and the Aged*.

[4] J. M. Bland and D. G. Altman. Statistical methods for assessing agreement between two methods of clinical measurement. *Lancet*, 1(8476):307–310, Feb. 1986. PMID: 2868172.

[5] P. Brolund. *Forensisk längdmätning i bilder*. PhD thesis, Linköping University, Linköping, 2006.

[6] P. Brolund and P. Bergström. Vidareutveckling av längdmätning i bild, 2006.

[7] P. Brolund and P. Bergström. Längdmätning i bild: En utvärdering av längdmätningsmetod i övervakningssystem. *Linköping: Statens Kriminaltekniska Laboratorium, Dokument- och Informationsteknikenheten*, page 1, 2007.

[8] C. Cattaneo. Forensic anthropology: developments of a classical discipline in the new millennium. *Forensic Science International*, 165(2-3):185–193, Jan. 2007.

[9] A. Criminisi, A. Zisserman, L. Van Gool, S. Bramble, and D. Compton. A new approach to obtain height measurements from video. In *Proceedings of SPIE*, volume 3576, 1998.

[10] J. Lee, E.-D. Lee, H.-O. Tark, J.-W. Hwang, and D.-Y. Yoon. Efficient height measurement method of surveillance camera image. *Forensic Sci. Int.*, 177(1):17–23, May 2008. PMID: 18096339.

[11] J. Ljungberg and J. Sönnerstam. Estimation of human height from surveillance camera footage - a reliability study. *Examensarbete i ortopedteknik*, 15, 2008.

[12] F. Massaad, T. M. Lejeune, and C. Detrembleur. The up and down bobbing of human walking: a compromise between muscle work and efficiency. *The Journal of Physiology*, 582(2):789–799, May 2007.

[13] Novacheck. The biomechanics of running. *Gait Posture*, 7(1):77–95, Jan. 1998. PMID: 10200378.

[14] A. Qualisys. Marker placement guide.

[15] J. G. Richards. The measurement of human motion: A comparison of commercially available systems. *Human Movement Science*, 18(5):589–602, Oct. 1999.

[16] M. Whittle. *Gait analysis: an introduction*. Butterworth-Heinemann, Edinburgh ; New York, 3rd ed edition, 2003.

# Triaža in nov model za selektivno iskanje digitalnih dokazov

Anže Škerjanc
Fakulteta za računalništvo in informatiko
Tržaška cesta 25
Ljubljana, Slovenija
anze.skerjanc@gmail.com

Nejc Škerjanc
Fakulteta za računalništvo in informatiko
Tržaška cesta 25
Ljubljana, Slovenija
nejko90@gmail.com

Svetlana Nikić
Fakulteta za računalništvo in informatiko
Tržaška cesta 25
Ljubljana, Slovenija
svetlana.nikic@gmail.com

## ABSTRACT

Digitalni dokazi predstavljajo vedno večjo vlogo pri preiskavi primerov in obsodbi osumljencev. Zaseg medijev za shranjevanje in ustvarjanje njihovih slik za preiskavo in analizo je ena najpogostejših in najboljših metod za najdbo novih sledi in dokazov. Ker pa je v splošnem na medijih veliko takih podatkov, ki niso povezani s preiskavo, je treba te nakako izločiti, da prihranimo čas in zavarujemo osebne podatke preiskovancev. S triažo želimo uravnovesiti dve nasprotji - na eni strani ne smemo preiskovati podatkov, nerelevantnih za preiskavo, po drugi strani pa se ne smemo preveč omejiti, da ne izgubimo za preiskavo pomembnega dokaza.

Do sedaj je bilo predstavljenih že nekaj digitalnih preiskovalnih modelov. Noben model do sedaj ni upošteval smernic zakonodaje, ko sodnik zahteva selektivno preiskovanje na kraju zločina samem.

Nov triažni model, ki ga predlagajo avtorji odpravlja nejasnosti, ki jih ima forenzik na kraju preiskovanja. Opisan je tudi vrstni red tipa datotek ob omejenem času pregledaovanja digitalne naprave. Nov triažni sistem poudarja, da je zelo pomembno pri preiskavi tudi dokumentiranje posameznik korakov. Ti podatki so zelo koristni pri kasnejšem sojenju. Celoten model poskuša združiti 2 nasprotujoča si pola - varstvo zasebnosti in temeljita preiskava.

## 1. UVOD

Na osebnih računalnikih in drugih elektronskih napravah shranjujemo ogromne količine podatkov. Ko preiskujemo nek zločin, iščemo podatke, ki bi nam pomagali rešiti primer, pri osumljencu in drugih vpletenih osebah – bodisi na osebnih računalnikih, računalnikih podjetja, tablicah, telefonih in drugih elektronskih napravah. Del podatkov, ki je relevanten za primer, ki ga rešujemo, je običajno zelo majhen in se prepleta z veliko količino nepomembnih podatkov.

Dolgo časa je veljalo, da so preiskovalci zasegli in pregledali vse medije, ki so jih našli. Ta pristop je močno kritiziran, saj posega v zasebnost, hkrati pa lahko tudi onemogoča delovanje podjetja. Osumljenec ima pravico do varnosti pred nediskretnim preiskovanjem in zasegu njegovih podatkov s strani države.

Posledično so se po svetu zelo povečale omejitve glede preiskovanja in zasega digitalnih dokazov, saj izvedenci v preiskavi avtomatično pregledajo vse podatke in s tem kršijo zasebnost.

Istočasno se po vsem svetu laboratoriji za digitalno forenziko ukvarjajo s vse večjim problem biti v koraku z zahtevami za forenzično preiskavo brez večje denarne pomoči. Brez ustreznih ukrepov laboratoriji tvegajo, da ne bi mogli več pravočasno rešiti primerov. Več kot je pregledanih elektronskih naprav, več podatkov je ustvarjenih, ki potrebujejo prostor in upravljanje. Ena izmed rešitev, ki poskuša reševati ta problem, je forenzična triaža.

## 2. TRIAŽA

Ime triaža izhaja iz medicinskega sveta in sicer je triaža (SSKJ [3]): ”razvrščanje bolnikov, poškodovancev v skupine glede na vrsto in težo poškodbe, obolenja, nujnost obravnave”. V svetu digitalne forenzike je triaža slabo definirana in čeprav v osnovi enaka, ima pri uporabi za različne ljudi različen pomen. Zelo na široko je triaža postopek eliminiranja elektronskih naprav, ki bodo pregledana, oziroma postavljanje prioritete napravam, da dobimo vrstni red, v katerem bodo pregledane naprave. Podobno potem za vsako napravo posebej lahko določimo v katerih delih bomo iskali dokaze in kaj vse bomo iskali med nekimi podatki. Praksa, da se s prioriteto določi samo vrstni red pregledovanja naprav, naredi bolj malo v tej smeri, zato je potrebno poseči globlje.

Istočasno se je treba zavedati tudi slabih lastnosti triaže. Z njeno uporabo tvegamo, da bo nek (lahko ključen) dokaz spregledan, ker sploh ni bil pregledan, ali pa ker ni bil podvržen celotni forenzični preiskavi. Poleg tega obstajajo tudi problemi kot to, da ljudje, ki bi izvajali triažo, ne bi bili dovolj tehnično podkovani, cene podporne programske opreme in usposabljanja in pravni problemi, ki bi nastali, ko nekateri dokazi v primeru ne bi bili analizirani. Torej v bistvu je največji nasprotnik triaži tveganje.

Seveda je močan argument za triažo, da alternativni pristop – pregled in analiza vseh podatkov predstavlja enako ali še

večje tveganje. Mediji so večkrat poročali o problemih, ki jih je povzročalo dolgo čakanje na analizo podatkov. Sodišča so redno kritizirala to čakanje, obtožencem so naknadno zmanjševali zaporne kazni, ko so po dolgotrajni analizi dobili nove podatke, osumljenci so naredili samomor med čakanjem analize na njihovem posestvu, ljudje, krivično obsojeni pedofilije so bili ločeni od svojih otrok, itd.

Kot pomeni triaža različnim ljudem različno, je podobno tudi s celostnim pregledom. Kaj za nekoga pomeni celostni pregled, je odvisno predvsem od znanja in izkušenj preiskovalca, zmožnosti programske opreme in časa, ki je na voljo. Tudi ob celostnem pregledu, je zelo verjetno, da nekateri dokazi ali potencialni dokazi ne bodo pregledani. Na primer, če je nekdo obtožen posedovanja neprimernih slik otrok, ali je smiselno pregledati vse tekstovne in pdf dokumente, če vsebujejo kakšno tako sliko? Ali je v splošnem potrebno rekonstruirati in pregledati stisnjene datoteke v nedodeljenem prostoru? V splošnem se te stvari ne izvajajo, saj je razmerje med časom, ki bi ga za to potrebovali in učinkom prevelik. Ravno to razmerje pa je bistvo triaže, s čimer lahko argumentiramo, da so preiskovalci večino časa izvajali neformalno triažo.

## 2.1 Administrativna triaža

Administrativna triaža ocenjuje okoliščine novega primera, na primer tip ali resnost kaznivega dejanja, pogosto preden je kakršenkoli dokaz analiziran. Čeprav imajo nekateri forenzični laboratoriji kriterij za sprejetje ali prioritizacijo pregledov, ni nobenega splošno sprejetega pristopa za proces administrativne triaže. Nekateri laboratoriji uporabljajo administrativno triažo samo za postavljanje prioritet, kar ne zmanjša število podatkov, ki jih je treba pregledati, medtem ko drugi laboratoriji odločno zavračajo primere in sprejmejo samo tiste naprave, ki ustrezajo določenim kriterijem. V tem scenariju, mora skupina, ki daje naprave v pregled pokazati močne argumente, ki pričajo o tem, da se dokaz res nahaja v teh napravah. Ta način predstavlja očitno tveganje, da se spregleda dokaze. Večina laboratorijev zato poskuša najti neko ravnovesje med tema dvema pristopoma.

## 2.2 Tehnična triaža

Na trgu obstaja mnogo komercialnih produktov, ki se oglašujejo kot orodja za tehnično triažo. Obstaja tveganje, da preiskovalci, ki bi pregledovali rezultate (na primer vnose v Registry), ne bi imeli zadostnega znanja za interpretiranje, saj takšno preiskovanje zahtevna visoko stopnjo znanja in izkušenj iz področja digitalne forenzike.

Pravilne nastavitve orodij so prav tako pomembne. Nepravilne nastavitve ali nepravilno razumevanje lahko pripeljeta do napačnih zaključkov.

Druge pogoste šibkosti tehnične triaže so še nezmožnost zaznavanja šifriranih datotek, procesiranje nedodeljenega prostora, izmenjevalnih datotek in drugih pomnilnikov ambientnih podatkov.

Še en možen pristop s tehnično triažo, je pregled rezultata forenzičnega orodja, s pomočjo katerega identificiramo kos dokaza. V tem trenutku se pregled ustavi in se izvede na tem mediju celostna forenzična preiskava. S tem pristopom tako zmanjšamo število naprav, ki jih je potrebno pregledati, ne zmanjša pa dela na sami napravi.

V splošnem je triaža zelo pogosto v uporabi, ne zato ker bi bila tako dober pristop, ampak ker zaenkrat še ni boljšega. Za vsak primer je potrebno oceniti ali se bolj splača triaža ali celoten pregled (če je treba na primer pregledati manjši medij) in če se odločimo za triažo, v kakšni meri jo je potrebno izvesti.

# 3. POTREBA PO NOVEM MODELU

V nadaljevanju se bomo predvsem osredotočili na probleme vdiranja v zasebnost preiskovancev. S triažo torej želimo omejiti iskanje na manj datotek, kot prvo zaradi hitrosti, kot drugo pa se želimo omejiti zgolj na datoteke, ki so relevantne za primer, ki ga preiskujemo, saj obratno pregledujemo ogromno drugih zasebnih podatkov, za kar pa pravic nimamo.

Pri preiskovanju so zato pomembne odločitve, s katerimi se odločamo kaj preiskovati, a nikjer ni regulirano kako enotno izvajati te odločitve. Torej je zelo potrebna nova metoda, ki bo uravnovesila dve nasprotujoči si vrednosti – varovanje človekovih pravic in hkrati doseg učinkovite preiskave, kar pomeni, da se vseeno ne smemo omejiti preveč, da ne bi izpustili pomembnih dokazov.

Katere podatke je dovoljeno zaseči in pregledati, je določeno z odredbo. Ampak ker je nemogoče vnaprej določiti kateri podatki natančno to so, oziroma kje se nahajajo, je opis bolj ohlapen. Nato pa je odvisno od subjektivne ocene izvedenca, kaj bolj natančno sodi pod ta opis. Problem se lahko potem pojavi na sodišču, kjer pa odločijo, da je ta opis določal druge podatke in lahko zavrnejo celotno tožbo, ker ne obstajajo standardi, po katerih bi se odločali na kraju preiskave. Potrebni so objektivni standardi, s pomočjo katerih bi določili ali je bila neka odločitev razumna. Te standarde je potrebno določiti ob popolnem razumevanju trenutne in realne situacije, saj so v nasprotnem primeru nepomembni in nerealnimi zahtevami.

Zato je potrebno predlagati nov triažni model za pridobivanje in preiskovanje podatkov na lokaciji preiskovanja, in sicer s pomočjo ocenitve trenutnega realnega stanja. Rezultati ocene lahko vsebujejo pomembne podatke o tem koliko ljudi in opreme je potrebnih, kakšne so omejitve, kakšni so pričakovani postopki in druge uporabne informacije. Glaven cilj je postavitev realističnih standardov za selektivno iskanje digitalnih podatkov, s pomočjo katerih se lahko izvedenci lažje objektivno odločajo in hkrati spoštujejo pravico do zasebnosti. Tako bi lahko vsak, ki bi to želel, lahko videl potek odločitev, na primer ljudje, katerih podatki so preiskovani, sodnik in drugi.

# 4. PRAVNO OZADJE PO DRŽAVAH

Kasneje opisani model je bil tudi izdelan za določen pravni sistem, in sicer za korejskega.

18. julija 2011 so sprejeli v Koreji nov zakon o kazenskem postopku, ki je prišel v veljavo s 1. januarjem 2012. Zakon vsebuje nove zahteve, ki se tičejo preiskovanja in zasega digitalnih dokazov.

Preden je prišel v veljavo, se je zgodila pomembna sodba, kjer so potrdili, da je cilj odredbe za preiskovanje in zaseg informacij, dejanska digitalna informacija in ne medij, ki ima to informacijo shranjeno. Sodišče je reklo, da "Relevantna digitalna informacija, ki se nanaša na sum, naveden v odredbi, bi morala biti pridobljena z zbiranjem natisnjenih dokumentov ali kopiranjem datotek na medij za shranjevanje s strani policista na lokaciji, kjer se nahaja iskana informacija. Ko to ni mogoče, lahko izjemoma prenesemo originalne medije ali njihove kopije v prostore organa pregona. Proces zbiranja natisnjenih dokumentov ali kopiranja datotek mora spadati pod odredbo za iskanje in zaseg."

V skladu s tem je v zakonu o kazenskem postopku jasno zapisano, da mora biti obseg zbirke natisnjenih dokumentov oziroma kopij omejena na del podatkov, ki se nanašajo na sum. Posledično so posodobili zakon o kazenskem postopku. Po spremembi zakona, sta se pojavila dva primera in obsodbi, ki sta pokazala na preiskovanje medijev izven obsega, ki ga je določala odredba za preiskavo. V enem primeru so zavrnili vse pridobljene dokaze z iskanjem, ker je to presegalo obseg, v drugem pa je preiskovanec tožil državo in primer dobil, saj so potrdili, da so nelegalno preiskovali preiskovančeve emajle.

V splošnem so v Koreji pozitivno sprejeli spremembe, saj bolj spoštujejo zasebnost ljudi. S spremembo zakona agencije potrebujejo nov model, ki se bo skladal s trenutnimi zakonskimi zahtevami.

V Sloveniji imamo Zakon o kazenskem postopku, ki obravnava preiskovanje in zaseg elektronskih naprav in podatkov predvsem v dveh členih – 219.a in 223.a. Člena govorita o tem kdaj se lahko izvede preiskava, na katerih nosilcih elektronskih podatkov, kaj mora vsebovati odredba za preiskavo, kdaj se uničijo pridobljeni podatki, na kakšen način se napiše zapisnik, o zavarovanju podatkov in podobno. O zasebnosti piše, da morata zaseg in zavarovanje podatkov biti opravljena na način, s katerim se v najmanjši možni meri posega v pravice oseb, ki niso osumljenci ali obdolženci, in varuje tajnost oziroma zaupnost podatkov ter se ne povzroča nesorazmerna škoda zaradi nezmožnosti uporabe elektronske naprave. V tem zakonu ne piše, da bi kakorkoli morali omejiti iskanje samo na podatke, ki so relevantni za kaznivo dejanje, na katerega se veže odredba. Ta zakon daje večjo moč preiskovalcem, saj deveti del člena 219.a [4] govori tudi o tem, da ob najdbi dokazov za nek drug zločin ni potrebno pridobiti novega naloga za preiskovanje, temveč se najdbe samo sporočijo državnemu tožilcu.

Na drugi strani pa varuje človekove pravice Zakon o varstvu osebnih podatkov [5]. 13. člen govori o obdelavi občutljivih osebnih podatkov. Ti se lahko pregledujejo samo s privolitvijo, če je to potrebno za varovanje življenja, če jih je njihov lastnih že javno objavil in podobno. 16. člen pa o tem, da se lahko osebni podatki zbirajo samo za določene in zakonite namene.

Zakon na splošno govori še o obveščanju lastnika podatkov, o roku hranjenja, posredovanju in zavarovanju podatkov pa tudi o pravicah posameznikov.

Torej smo tudi v Sloveniji vsaj uradno precej zaščiteni pred pretiranim preiskovanjem, čeprav v praksi žal temu ni vedno tako.

Evropska unija se podobno zaveda problemov kršenja pravic in je skupaj s Svetom Evrope izdala navodila za države članice [6]. Ta naj bi se uporabljala pri sprejemanju novih zakonov, prav tako pa je namenjena tudi posameznikom – policistom, sodnikom, odvetnikom in širši javnosti.

Med navodili najdemo tudi poglavje 6.1.5, ki opozori na to, da je lahko med zajetimi podatki veliko osebnih podatkov, ki so nerelevantni za samo preiskavo, a so zasebni in morda tudi zakonsko zaščiteni. Upoštevanje omejitev preiskave lahko predstavlja izziv, saj ni vedno jasno, kje v sistemu se nahajajo zasebni podatki, če niso ustrezno označeni. Prav tako pa je lahko med temi podatki vseeno tudi nekaj takih, ki pa so nujni za preiskavo.

Če preiskovalec med preiskovanjem naleti na zasebne podatke izven določil odredbe, mora pod nekaterimi pristojnostmi takoj končati s preiskovanjem teh podatkov in jih označiti kot zasebne, da se ne bi spet preiskovale. Nekateri zločinci pa lahko uporabljajo take datoteke za skrivanje podatkov, kar predstavlja še dodaten problem za preiskovalce.

Ena izmed možnih rešitev je sodiško zapečatenje podatkov, o katerih se potem obe strani pogovorita ali je potrebna preiskava ali ne. Če preiskovanec verjame, da njegov sistem vsebuje zaščitene podatke, ki ne bi smeli biti del preiskave, lahko to zagovarja pred sodnikom preden izvedenec analizira podatke.

# 5. TEŽAVE V PRAKSI

Argumenti za omejevanje obsega zasega so se pojavili po vsem svetu. Istočasno se močno kritizira različne standarde med sodišči, zaradi katerih so dokazi tretirani na različne načine. V ZDA je zato Ministrstvo za pravosodje objavilo smernice za preiskovanje s široko paleto primerov. Smernice z naslovom Searching and Seizing Computers and Obtaining Electronic Evidence in Criminal Investigation, vsebujejo nasvete pri reševanju problemov ter pripravi in izvajanju odredb.

Povzetek vsebine:

- Določitev verjetnega vzroka – če je računalniška strojna oprema nelegalna, dokaz, proizvod ali sredstvo kaznivega dejanja, naj odredba opiše samo strojno opremo. Če pa se vzrok nanaša na informacije, naj odredba opisuje informacijo, ki jo je treba zaseči in nato podati zahtevek oblasti, da zaseže te informacije v kakršnikoli obliki so pač shranjene.
- Podrobno opiši stvari, ki naj se zasežejo – Pri preiskovanju elektronskih medijev za shranjevanje, ki vsebujejo informacije, ki so dokaz kaznivega dejanja, naj se opisi stvari, ki jih je potrebno zaseči, rajši osredotočijo na vsebino relevantnih datotek kot pa na fizične medije za shranjevanje.

- Pokazati potrebo po ustavarjanju slike diska in preiskovanju zunaj lokacije – Po navadi preiskovanje trdih diskov in drugih medijev traja predolgo, da bi se izvedlo na sami lokaciji med izvajanjem odredbe. Odredba naj razloži zakaj je potrebno narediti sliko celotnega diska (ali ga zaseči) v namen kasnejšega preiskovanja in uporabe informacij.
- Več hkratnih odredb in iskanje na različnih lokacijah – Če preiskovalci že pred samim iskanjem izvejo, da je del ali vsi podatki, ki so opisani v odredbi, shranjeni na drugi lokaciji kot tista v odredbi, je najboljša možna pot odvisna od lokacije teh podatkov. Če so podatki na primer shranjeni po večih zveznih državah, naj preiskovalci pridobijo odredbe za vsako državo posebej.
- Sprememba fokusa in potreba po novih odredbah – en računalnik je lahko uporabljen za več različnih kaznivih dejanj, torej lahko disk vsebuje dokaze različnih dejanj. Ko preiskovalec išče podatke na računalniku z odredbo, običajno ta odredba dovoljuje iskanje dokazov samo nekaterih kaznivih dejanj. V tem primeru naj preiskovalci zahtevajo novo odredbo, da lahko legalno zasežejo tudi ostale podatke.
- Ovire glede očitnega neupoštevanja odredbe – Preiskovalci morajo pogosto zaseči tudi strojno opremo ali datoteke, ki niso opisane v odredbi. Odvetniki preiskovancev velikokrat to uporabijo za obrambo kot očitno neupoštevanje odredbe, s čimer želijo izničiti pridobljene dokaze. Najboljši odgovor tožilca na to je pokazati, da je zaseg stvari, ki v odredbi niso bile zapisane, rezultat t.i. odziva v dobri veri (good faith response) zaradi neločljivih praktičnih težav in ne poizkus preiskati celotne medije preiskovanca pod pretvezo selektivnega iskanja. Sodišča so prepoznala praktične težave, s katerimi se srečujejo preiskovalci pri iskanju določenih datotek in rutinsko odobravajo preiskovanja zunaj lokacije kljub nenamernemu zasegu dodatne lastnine.

Čeprav smernice določajo podrobna navodila, ostaja še vedno veliko nerešenih problemov. Kraji zločina so naključni in nepredvidljivi, torej je normalno, da so pravila lahko včasih dvoumna. Na primer, odredbe za iskanje in zaseg ne morejo biti pred preiskavo izdani z dovolj podrobnostmi, saj ne moremo povedati katere datoteke točno je treba zaseči ali katere metode iskanja uporabiti, saj ne vemo kako izgledajo mediji in ne vemo kakšne metode bodo potrebne. Torej lahko trdimo, da so omejitve pred samim iskanjem neprimerne in bi bilo boljše spremeniti pravila glede dopustnosti dokazov v dejanskem stanju. Če odredba za iskanje in zaseg ne specificira kaj zaseči, potem so preiskovalci deležni nekaj diskretnosti pri določanju obsega in bi bila dejanska ureditev uporabljena za oceno zakonitosti izvršitve. Hkrati se pojavljajo tudi druga vprašanja. Kateri so relevantni podatki, ki jih preiskovalci smejo zaseči? Kaj vse spada pod očitno neupoštevanje odredbe? Kaj se smatra za odziv v dobri veri zaradi neločljivih praktičnih težav? Kako lahko preiskovalci odločijo ali trenutna situacija potrebuje ustvarjanje slike diska ali pa preiskovanje zunaj lokacije? Vsaka oseba organa pregona se lahko odloča glede na svoje standarde in vsak standard je posledica lastnega razumevanja omejitev in zmožnosti na lokaciji. Ker smo si ljudje različni, pride do različnih odločitev, kar lahko povzroči resne probleme, saj policisti ne vedo, ali bodo njihove odločitve na sodišču sprejeli kot razumne in poleg tega tudi preiskovanci ne morejo meriti ustreznosti preiskovanja in zasega. Zato potrebujemo objektivne standarde, ki so narejeni na podlagi razumevanja dejanske situacije.

# 6. PREDLAGAN PRISTOP

Do sedaj je bil problem v tem, da še ni bil definiran noben standardni triažni model, ki bi bil nasplošno uporabljen pri računalniški forenziki. Mejo med človeškimi pravicami in učinkovitostjo preiskave, je bila odvisna od subjektivne ocene preiskovalca in sodnika. Celotni zajem in pregled digitalnih gradiv je takrat veljal kot lažji in časovno manj obremenjujoč, kot selektivno iskanje in zajemanje na terenu. Z vedno bolj razvitimi forenzičnimi programi, je postalo selektivno iskanje bolj efektivno in predvsem manj invazivno. Vse to je pripeljalo do tega, da se ja lahko uvedel standard, ki bo definiral triažni model za iskanje digitalnih dokazov na samem mestu zločina. S tem modelom, bo policist že na terenu lahko zasegel le za zločin relevantne dokaze.

## 6.1 Področje

Do sedaj je bilo predstavljenih že nekaj digitalnih preiskovalnih modelov. Noben model do sedaj ni upošteval smernic zakonodaje, ko sodnik zahteva selektivno preiskovanje na kraju zločina samem. Zato so predlagali strukturo procesa, ki je skupna do sedaj vpeljanih proceduram v sodstvu(Figure 1).

Postopki so [8]:

- Priprava: Priprava strategije poteka preiskovanja.
- Zbiranje in hramba: Opisan triažni model spada v ta postopek. V tem postopku na kraju zločina zberemo za primer relevantne digitalne podatke.
- Pregled in analiza: Podatke, ki smo jih zbrali na kraju zločina sedaj pregledamo in analiziramo.
- Predstavitev in poročanje: V tem koraku napišemo poročilo o naših ugotovitvah, ter to predstavimo sodnikom v njim razumljivem jeziku.
- Razširjanje primera: Razširimo primer z ugotovitvami iz postopka 'predstavitev in poročanje'.

## 6.2 Osnova algoritma

Predlagan triažni model ima v grobem tri faze(Figure 2):

- Presoja
- Vzpostavitev triažnega modela
- Periodično vzpostavljanje nove presoje

## 6.3 Presoja

V fazi presoje se na terenu zberejo podatki, ki nam nato pomagajo vzpostaviti triažni model. Na začetku si je potrebno zastaviti cilje, ki nam določijo cilje triažnega modela (presoje). V tej fazi mora preiskovalec na terenu razumeti pogoje na kraju zločina. Cilj tega je, da si oblikujemo pristop, ki nam bo pridobil bistvene podatke za vzpostavitev triažnega




**Figure 2: Faze predlaganega triažnega modela**

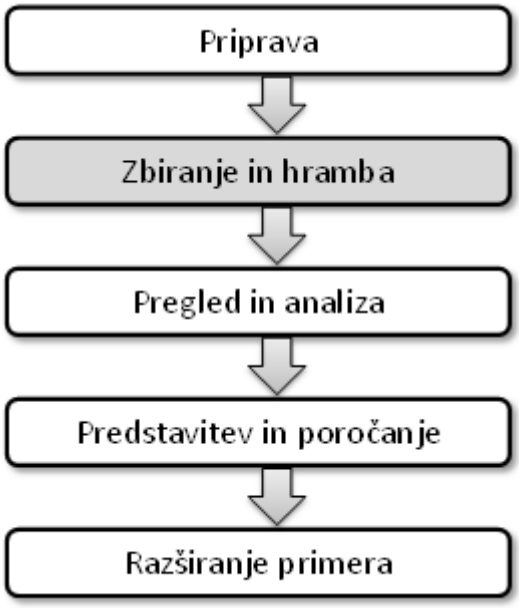



**Figure 1: Postopki preiskovanja**

modela. V tem koraku moramo identificirati splošni proces iskanja in zajema elektronskih podatkov. Pri fazi presoje se mora preiskovalec dobro zavedati zmožnosti njegove ekipe na terenu in na podlagi tega prilagoditi model. Preveč natančno preiskovanje lahko povzroči, da ekipi na terenu zmanjka časa. Poleg tega mora oceniti, koliko za primer relevantnih podatkov bo našel. Prav tako se mora zastaviti prioritete v zbiranju podatkov.

## 6.4 Vzpostavitev triažnega modela

Pri fazi vzpostavitve triažnega modela, v prejšnji fazi zastavljen triažni model, sedaj vzpostavimo. Pri tem moramo biti pozorni, da ima nov triažni model naslednje lastnosti:

- Vsebovati mora utemeljitev njegove odločitve, ki jih lahko sprožijo pravna vprašanja
- Biti mora učinkovit v selektivnem zbiranju podatkov, ki so relevantni trenutno preiskovanem zločinu
- Vsi postopki v triažnem modelu morajo biti forenzično čisti.

Faza presoje in vzpostavitev triažnega modela sta tesno povezani med sabo. Vse to nam zagotovi, da bo naš triažni model dejansko deloval tako, kakor smo si zamislili.

Triažni model je razdeljen na stopnje načrtovanja, izvajanja, kategorizacije in odločitve.

V stopnji načrtovanja policist vzpostavi strategijo, ki jo je zasnoval v fazi presoje. Pri tem upošteva razpoložljiva sredstva, vrsto kriminalnega dejanja in situacije na sceni.

V fazi izvajanja se zastavi objektivno merilo, ki nam pove ali gre za relevantno informacijo. V tej fazi se prav tako določi tehnična merila za razlikovanje med relevantnimi in irelevantni informacijami in na podlagi tehničnih meril, se vzpostavi učinkovita metoda izbiranja podatkov.

V fazi kategorizacije podatke klasificiramo v 5 različnih kategorij. Te kategorije so:

- Relevantni podatki: To so podatki, ki so vezani na primer in lahko pomagajo pri odločitvi o obstoju preiskovanega kriminala
- Irelevantni podatki: So podatki, ki niso pomembni za primer
- Drugi podatkovni vir: Podatek, ki nakazuje da obstaja še drugi vir, na katerem se lahko nahajajo pomembni podatki.
- Sprememba fokusa: So podatki, ki skoraj zagotovo kažejo na obstoj kriminalnega dejanja, a ti podatki niso vezani na trenutno preiskovan zločin.
- Nad omejitvijo virov: V to kategorijo spadajo podatki, ki zaradi omejitev (tehnologije, opreme, časa,...) ne morejo biti kategorizirane v ostale kategorije.

V fazi odločitve preiskovalec glede na rezultat prejšnjega koraka odloči, kaj narediti s podatkov. V primeru da smo našli relevantne podatke, se ti podatki zberejo. Irelevantne podatke zavrnemo. Če smo pri prejšnjem koraku, klasificiral podatek kot 'drugi podatkovni vir' ali 'sprememba fokusa', to pomeni, da potrebujemo novo odredbo. Če podatek klasificiramo kot 'Nad omejitvijo virov' pa podatek kasneje obdelamo za preiskovanje v laboratoriju.

## 6.5 Periodično vzpostavljanje nove naloge

Predlagan okvir temelji na predpostavki, da se postopek ponovi, če se v preiskavi pojavijo spremembe. V ta namen je potrebno kraj zločina periodično na novo oceniti. S tem dosežemo, da faza presoje in vzpostavitev triažnega modela ustreza sedanjemu stanju na terenu.

# 7. NOV TRIAŽNI MODEL

Avtorji so naredili anketo v kateri so spraševali forenzike o sami zbiranju forenzičnih dokazov na terenu.

Po opravljeni analizi odgovorov na anketo avtorji predlagajo nov triažni sistem. Pri tem poudarjajo, da je model narejen glede na trenutne potrebe v Južni Koreji.

Pri tem ima veliko vlogo trenutna zakonodja (v Južni Koreji). V posameznem primeru lahko preiskujejo samo digitalne dokaze, ki so povezani s samim zločinom. Model poudarja veliko vlogo preiskovanju samo relavantnih podatkov, ki so pomembni za preiskovani primer (crime relavant information).

Nov triažni sistem je razdeljen na 4 glavne korake:

- načrtovanje,
- izvedba,
- kategorizacija,
- odločanje/sklepanje.

## 7.1 Načrtovanje

V koraku načrtovanja poskušamo določiti obseg preiskovanja. Poskušajo določiti tip, velikost in stanje naprave ali podatkovnega nosilca, ki ga želimo preiskovati. Pri tem moramo upoštevati omejitve, ki jih ima posamezna naprava. Seveda pa selekcija ni vedno možna na kraju zločina, zato se lahko odločimo za zaseg naprav ali podatkovnih nosilcev. Te potem v kasnejši fazi pregledamo v forezničnem laboratoriju. Čas preiskovanja posameznih medijev je običajno zelo dolg. Posamezni medij lahko pregledujemo tudi nekaj ur. Čas pregledovanja medija se razlikuje glede na vrsto zločina.

Računalniški forenziki so na kraju zločina lahko tudi več kot 9 ur, kjer se ukvarjajo z zajemom in obdelavo dokaznega gradiva.

Samo preiskovanje digitalnih dokazov včasih ni mogoče izvesti zaradi uničenih naprav , zaščitenimi podatki, orodij za prepreče-vanje forenzične analize, itd.

## 7.2 Izvedba

V tem koraku podatke razdelimo na dva dela. Prva skupina so podatki, ki jih ni potrebno obdelati, da bi prišli do njihove vsebine. Druge vrste podatki pa potrebujejo predprocesiranje, da pridemo do njih.

V to spada:

- iskanje podatkov na formatiranih diskih,
- iskanje podatkov na nelociranih mestih,
- enkripcija zašitenih datotek,
- iskanje datotek, ki vsebujejo skrivne podatke (stenografija),
- iskanje po pokvarjenih datotekah,
- iskanje po stistnjenih datotekah (.zip),
- itd.

Naslednji korak v tej fazi je zmanjševanje količine preiskovanih podatkov. V primeru, da ne iščemo datoteke točno določenega tipa, model priporoča razvrščanje pregledanih podatkov glede na vrsto zločina.

V primeru splošnih kriminalnih dejanj model priporoča naslednji vrstni red (od najbolj do najmanj pomembnega):

- zgodovina brskanja,
- dokumenti,
- elektronska pošta,
- programi za neposredno sporočanje,
- video posnetki,
- datoteka s povezavami,
- itd.

V primeru tehnoloških kriminalnih dejanj model priporoča drugačen vrstni red, zaradi večje računalniškega znanja preiskovanca. Vrsti red je (od najbolj do najmanj pomembnega):

- zgodovina brskanja,
- dokumetni,
- register,
- zabeležka dogodkov,
- elektronska pošta,
- izvršljive datoteke,
- itd.

V primeru kriminalnih dejanj v gospodarskih družbah ločimo dve skupine. Prva skupina so kriminalna dejanja, ki niso neposredno povezana z računalniško tehnologijo. V to skupino spadajo vsa kriminalna dejanja, katere računalnik ni bil sredstvo za izvedbo zločina.

Se pa lahko na njem nahajo informacije o storjenem zločinu. Ko imamo opravka s takim zločinom preiskujemo podatke v naslednjem vrstnem redu (od najbolj do najmanj pomembne):

- dokumenti,
- računovodstke podatke,
- spletna pošta,
- programi za neposredno sporočanje,
- zgodovina brskanja,
- register
- itd.

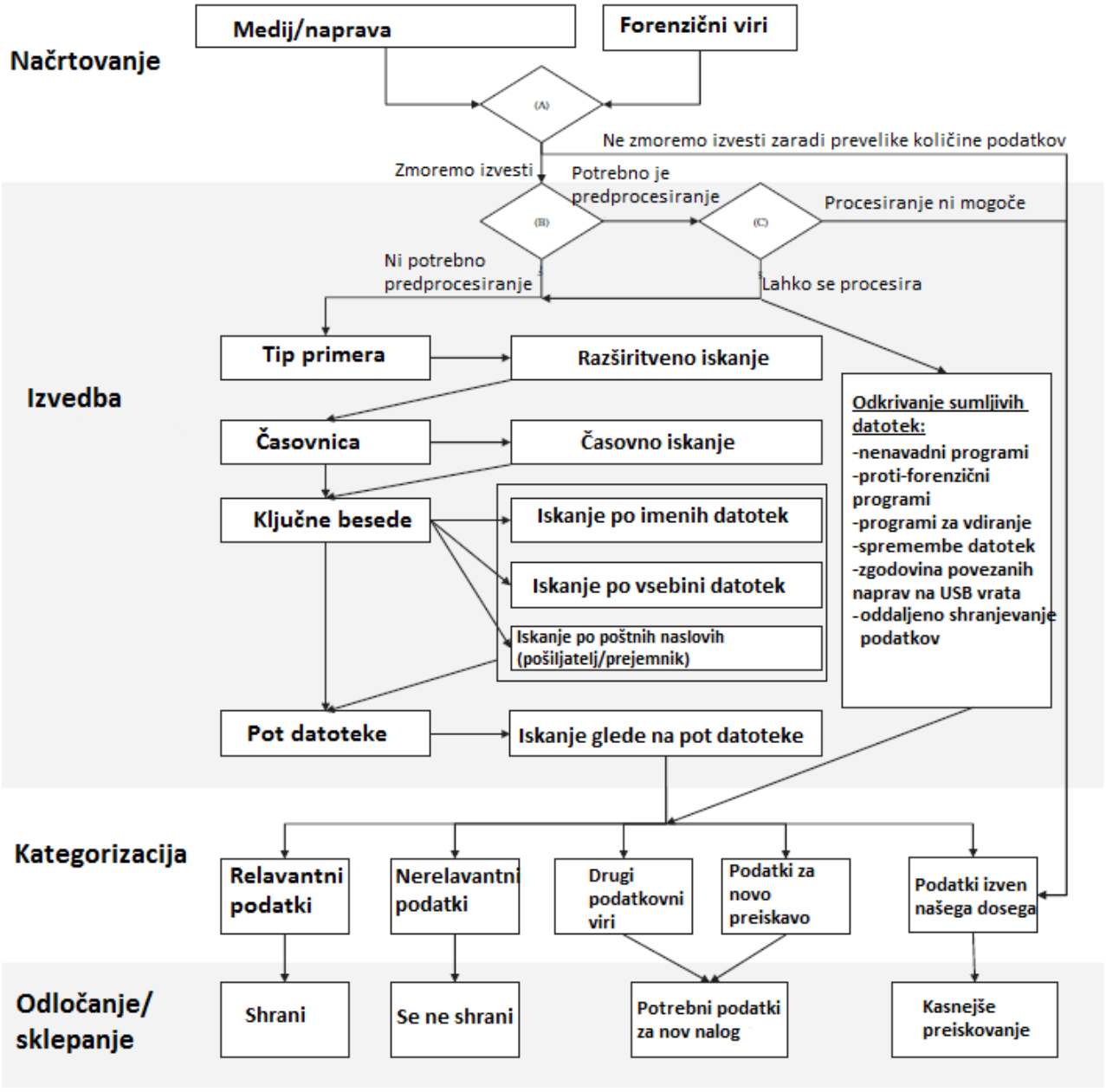



**Figure 3: Shema novega triažnega modela**

V drugo skupino spadajo zložini, katera značilnost je uporaba visokotenoloških naprav za izvedno zločina. Lahko je tudi predmet napada visokotehnološka naprava. V primeru, da imamo opravka s takim zločinom je prioriteta preiskovanja(od najbolj do najmanj pomembna):

- spletna pošta,
- dokumenti,
- inženirski načrti,
- zgodovina brskanja,
- programi za neposredno sporočanje,
- zgodovina tiskanih strani,
- itd.

Drugi pogled na preiskovanje je iskanje glede na čas izvedbe kriminalnega dejanja. V tem primeru iščemo podatke, ki so bili ustvarjeni, spremenjeni ali dostopani v določenem časovnem obdobju. Za to nam pri NTFS sistem najbolj koristi MFT tabela (Master File Table). Tu imamo zapisane podatke o kreiranju, zadnji spremembi in zadnjem dostopu do datoteke. Moramo pa pri tem biti pozorni, ker lahko nekoliko bolj vešči uporabniki spreminjajo te podatke. Ko ocenimo da obstaja možnost spreminjanja teh podatkov nam ta metoda ne koristi. Tega dejstva se moramo zavedati, da smo pripravljeni na morebitno načrtno zavajanje podatkov.

Prioriteto preiskovanja lahko določimo še na en način, ki je zelo podoben prejšnemu. S tem načinom preiskujemo metapodatke, ki so zapisani v v delu dateke. Nekateri programi v sklopu svojih funkcionalnosti zapisujejo različne podatke, ki znajo zelo pomagati pri preiskavi. V to štejemo zgodovino brskalnikov (Internet Explorer, Firefox, Chrome, Opera), časovne značke v programih za pisarniško delo (Microsoft Office, OpenOffice, LibreOffice). Zelo kortiste informacije pa lahko pridobimo tudi iz slik s pomočjo EXIF formata (Exchangeable image file format). Vsaka slika ponavadi vsebuje metapodatke o resoluciji in datumom kreiranja. Poleg tega nekatere naprave shranjujejo še podatke o osvetlitvi, tipu fotoaparata, kompresiji, uporabljeni bliskavici, itd. Tudi pri tem preiskovanju moramo imeti v mislih, da so lahko podatki namerno spremenjeni.

Po podatkih pa lahko iščemo tudi s pomočjo ključnih besed. Le-te lahko izbiramo z analizo osumljencev (na primer: ime podjetja, osebno ime, elektronski naslov, ime projekta, ime naprave, itd). Pri iskanju moramo biti pozorni na sklanjatve besed v določenem jeziku. Ponavadi iščemo besede v nedoločniku ali pa zapišemo samo koren besede. Tako iskanje v večini primerov vrne veliko boljše rezultate.

Možno je iskati podatke tudi s pomočjo poti do določenih direktorijev. Za iskanje teh direktorijev nam ni potrebno raziskati vsak direktorij posebej. Ponavadi so poti potenicalno zanimivih direktorijev shranjene na točno določenih mestih. Te lahko najdemo na primer v registrih, zgodovini iskanja in posebnih datotekah z vpisanimi potmi. V teh direktorjih lahko najdemo datoteke do katerih je uporabnik dostopal v zadnjem času in periodo dostopa do teh datotek.

Poleg vsega naštetega pa je pomemben člen tudi iskanje sumljivih datotek. Namen tega je, da preiščemo morebitne sledove namernega skrivanja ali prikrivanja podatkov. Na tem koraku preiščemo če je katera od datotek sumljive oblike(neberljive) in pa če se pojavljajo anomalije v datotekah. Na računalniku poskušamo poiskati programe oziroma sledi programov, katerih namen je skrivanje oziroma uničevanje podatkov. V to skupino sodijo proti-forenzični programi, sledovi zlonamernega dostopa, spreminjanje datotek in meta podatkov, nenavadni programi, sledovi priklopa nenavadne zunanje naprave, itd.

Preverimo tudi če je uporabnik dostopal do zunanjih naprav oziroma diskov. Na teh napravah so lahko naloženi programi, ki so omogočali kaznivo dejanje ali pa se je uporabnik zavedal morebitne forenzične preiskave in je podatke hranil na ■varnem■ mestu. Zakonodajno gledano je forenziku zelo težko najti in preiskati vse možne lokacije do katerih je dostopal osumljenec. Tu je problem varstva osebnih podatkov. Teh lokacij je lahko zelo veliko (USB ključki, oblačne storitve, prenosni mediji, itd).

V tej fazi preverimo tudi datoteke, katere je uporabnik izbrisal. Za ta namen obstaja že zelo veliko število programov.

Preiščemo tudi programe, ki bi lahko bili uporabljeni z zlonamernimi nameni. Lahko so namenjeni škodovanju ostalih naprav preko mreže ali za škodovanju preiskovanemu računalniku.

Forenzik mora tudi preveriti, katere programe je uporabnik najpogosteje uporabljal, ter ali je bil kdaj uporabljen program s sumljivo vsebino/delovanjem. Če to zaznamo moramo ugotoviti namembnost in uporabo programa.

Posebna zvrst pregledovanja pa je stenografija. O tem govorimo, ko so podatki zapisani v neke druge podatke. Velikokrat se v ta namen uporablja slike ali videoposnetke.

Poglejmo na primeru videoposnetka. Vsak video posnetek je sestavljen z večih slik, ki si sledijo v točno določenem zaporedju. Vsaka slika je sestavljena s pikslov. Vsak piksel opisuje svojo barvo s tremi različnimi kanali(rdeča, zelena, modra). V nekaterih formatih obstaja še četrti format (alfa kanal). Vsak izmed teh kanalov je zapisan z 8 biti. Če želimo skriti informacijo lahko spreminjamo posamezne bite teh kanalov z vsebino.

Algoritem vzame vnaprej znano sliko, jo binarno prebere ter na vnaprej določena mesta zapiše skrito vsebino. Ponavadi si ti zapisi ne sledijo linerno, zaradi težje detekcije uporabljene stenografije. Na drugi strani lahko program, ki pozna algoritem na točno določenih mestih prebere bite in jih pretvori v znano vsebino.

Ko gledamo sliko ali video posnetek odstopanja barv ne bomo zaznali s prostim očesom. Obstajajo pa programi, ki to zaznavajo in opozarjajo na morebitne spremembe v posnetku.

Forenzik mora tudi preiskati podatke skrite izven datotečnega sistema (na primer med particijami). Operacijski sistem jih običajno ne najde, a uporabnik lahko s posebnimi programi

dostopa do njih in na njih hrani podatke.

### 7.3 Kategorizacija in sklepanje

Po izvedbi postopka so podatki razdeljeni v pet katergorij:

- podatki, ki so pomembni za preiskovan zločin,
- podatki, ki niso pomembni za preiskovan zločin,
- zunanji podatki,
- podatki za novo preiskavo,
- podatki, ki so izven našega dosega.

Pri določanju pomembnih informacij (za preiskovan primer) so najbolj pomembni podatki, ki so neposredno povezani s primerom. Ti podatki kažejo motiv, namen ali priložnost in jih shranimo za kasnejšo obdelavo.

Poleg tega imamo tudi podatke, ki niso neposredno povezani s primerom. Služijo nam kot potrdilo, da so bili tisti neposredni podatki resnično napisani, uporabljeni ali spremenjeni. Taki podatki potrjujejo pravilnost neposrednih podatkov.

Tretje vrsti podatki so tisti, ki se nahajajo v istem direktoriju kot podatki, ki so neposredno povezani s primerom.

Četrte vrste podatki so tisti za katere sumimo, da si bili načrtno spremenjeni ozrima so bili na njih uporabljeni protiforenzični programi. Te dokaze je prav tako potrebno zbrati, da nam še dodatno potrdijo hipotezo.

V primeru, da v podatkih ugotovimo še kako drugo kriminalno dejanje jih razvrstimo v skupino za novo preiskavo. V tem primeru se uredi vse zakonske zahteve, ter pričnemo s preiskovanjem drugega kriminalnega dejanja.

Ko selektivno iskanje in zaseg ni mogoč se te podatke razvrsti v skupino podatkov izven našega dosega. V to skupino vključujemo velike količino podatkova li podatke katere ni mogoče razvrstiti. Te podatke ponavadi obdelamo kasneje v forenzičnem laboratoriju.

### 7.4 Dokumentacija

Celotno triažo je potrebno izvajati ob upoštevanju načel digitalne forenzike [7]. Pri tem moramo paziti na integriteto, verodostojnost in zanesljivost. Pri tem je zelo pomembno, da vsi vpleteni dokumentirajo postopke in odločitve. Pri tem morajo vedno napisati datum in čas izvajanja določnega postopka oziroma odločitve.

## 8. ZAKLJUčEK

Celotni triažni sistem predstavlja osnovo, kateri naj bi forenziki poskušali dosledno slediti. Seveda pa to ni mogoče v vseh primerih. Lahko pride do odstopanj pri posamezni vrsti kriminalnega dejanja. Lahko se zgodi, da ima posamezna država drugačne protokole pri obravnavi posameznega kriminalnega dejanja.

Lahko se tudi zgodi, da je postopek spremenjen, ker preiskovalci nimajo v danem trenutku na voljo vso potrebno opremo. To se lahko zgodi zaradi rednega servisa opreme, napake na opremi ali pa je oprema zasedena zaradi drugega primera. Lahko se tudi zgodi, da ponekod nimajo vse potrebne opreme za forenzično preiskavo. V teh primerih se katerega od korakov izpusti.

Opisan triažni sistem se poskuša približati dvema nasprotujočima si poloma – varstvo zasebnosti in temeljita preiskava. Celoten sistem je zgrajen na osnovi, ki transparentno pokaže posamezne postopke. To je pomembno zaradi lažjega reševanja problemov na sodiščih.

Če bi se opisani triažni sistem na dolgi rok izkazal za uspešnega je zaradi lažjega vodenja priporočljivo narediti namensko programsko opremo. Pri tem je zelo pomembno, da se vsak posamezni korak zapiše in omogoča kasnejše enostavno branje ali nadaljnje raziskovanje.

## 9. REFERENCES


[1] M. M. Pollitt, *Triage: A practical solution or admission of failure*, Digital Investigation Volume 10, Issue 2, September 2013, Pages 87-88

[2] A. Shaw, A. Browne, *A practical and robust approach to coping with large volumes of data submitted for digital forensic examination*, Digital Investigation Volume 10, Issue 2, September 2013, Pages 116-128

[3] *Slovar slovenskega knjižnega jezika*, Slovenska akademija znanosti in umetnosti, Znanstvenoraziskovalni center Slovenske akademije znanosti in umetnosti, Inštitut za slovenski jezik Frana Ramovša, 2002.

[4] *Zakon o kazenskem postopku*, Uradni list Republike Slovenije, `http://www.uradni-list.si/1/content?id=108445`, na dan 6.5.2014.

[5] *Zakon o varstvu osebnih podatkov*, Uradni list Republike Slovenije, `http://www.uradni-list.si/1/objava.jsp?urlurid=20074690`, na dan 6.5.2014.

[6] CyberCrime@IPA, *Electronic evidence guide*, `http://www.coe.int/t/dghl/cooperation/economiccrime/cybercrime/Documents/Electronic%20Evidence%20Guide/default_en.asp`, na dan 7.5.2014.

[7] K. Rogers, J. Goldman, R. Mislan,T. Wedge, S. Debrota *Computer Forensics Field Triage Process Model*, `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.169.1878`, na dan 10.5.2014.

[8] Framework for a Digital Forensic Investigation, `http://mo.co.za/open/dfframe.pdf`, na dan 10.5.2014