Zbornik

Digitalna forenzika

Seminarske naloge, 2016/2017

Ljubljana, 2017

Zbornik

Digitalna forenzika, Seminarske naloge 2016/2017

Editors: Andrej Brodnik, Nejc Ambrožič, Jošt Lajovec, študenti

Ljubljana: Univerza v Ljubljani, Fakulteta za računalništvo in informatiko, 2017.

Kazalo

Uvod

Digitalna forenzika je veja forenzične znanosti, ki se ukvarja z zbiranjem, shranjevanjem, iskanjem in obnavljanjem digitalnih podatkov. Zajema tudi proces identifikacije, ohranitve, analize digitalnih dokazov in predstavitev le teh v pravnih postopkih na sodišču. Digitalna forenzika se ne omejuje le na računalnike in računalniška omrežja, ampak se v danes vedno spreminjajočem se svetu ukvarja tudi s telefonijo, pametnimi (plačilnimi) karticami, tiskalniki, kamerami, mikrofoni in v resnici vsemi napravami, ki vsebujejo digitalni podatkovni nosilec. S tehnološkim razvojem na različnih področjih se je znanost močno razširila in pokriva vedno večje področje našega življenja, saj vsi uporabljamo vedno več elektronskih (digitalnih) naprav.

V zborniku so zbrane seminarske naloge študentov magistrskega študija na Fakulteti za računalništvo in informatiko Univerze v Ljubljani 2016/2017. V okvru predmeta Digitalna forenzika je vsaka skupina študentov prejela en članek kot izhodišče za seminarsko delo.

Članki so bili izbrani iz 4 glavnih raziskovalnih področij za letošnje leto: metodologija, bitcoin, omrežna forenzika, storitve in internet ter par člankov, ki niso spadali v nobeno od prej omenjenih kategorij.

Pri področju metodologije sta bila predstavljena tako metodološki model za digitalno triažo na terenu, kakor tudi analiza realnega primera.

Najbolj podrobno je predstavljeno področje bitcoina, saj gre za razmeroma novo tehnologijo, ki odpira veliko novih in zanimivi vprašanj. Seminarske naloge dobro predstavijo celovit pregled nad področjem bitcoina, razblinijo določene mite o bitcoinu, analizirajo uporabnike le-tega, predstavijo orodje za vizualizacijo bitcoin transakcij, izvedejo analizo napada s prikrivanjem, predlagajo in analizirajo varnostne izboljšave bitcoina, prikažejo uporabo strojnega učenja za iskanje prevar pri trgovanju z bitcoinom in analizirajo uporabo CrypoLocker odkupnin v bitcoinih.

Na področju omrežne forenzike seminarske naloge prikažejo primerjavo metod za iskanje najboljšega orodja za preiskavo mobilne naprave, forenzično preiskovanje v programsko definiranih omrežjih in iskanje značilnih vzorcev v njih ter predstavijo nov način in ogrodje za sledenje IP paketom.

V sklopu področja storitev in interneta seminarske naloge zajamejo forenzično preiskavo spletnega zalezovanja z uporabo analize vedenjskih karakteristik, forenzično analizo podatkov iz oblačne storitve Google Docs, uporabo analiz internetne zgodovine za lažje izvajanje forenzične preiskave, preverjanje avtorstva dokumentov za različne jezike, žanre in tematike ter posodobitev o digitalni forenziki kot storitev.

V skopu razno pa seminarski nalogi predstavita avtomatizirano generiranje profila za analizo živega Linux pomnilnika in obnovitev močno fragmentiranih JPEG slik.

Ta zbornik združuje vse končne seminarske naloge, ki so bile izdelane v študijskem letu 2016/2017. Namenjen je vsem, ki jih področje digitalne forenzika ali pa zgolj eno ali več predstavljenih področij.

Ljubljana, 2017 Nejc Ambrožič

1 Povzetki

1.1 Večstopenjski forenzični metodološki model za digitalno triažo na terenu

Zaradi finančnih omejitev in zahteve po visokem usposabljanju digitalnih forenzikov primanjkuje po celem svetu. To posledično vodi do tega, da od zajema digitalnih dokazov do prejema forenzičnega poročila preteče veliko časa. V namen skrajšanja tega časa je bilo predlaganih nekaj postopkov za vzpostavitev triaže digitalnih dokazov. Z uporabo triaže je preiskovalcu omogočen hitrejši dostop do informacij, medtem ko čaka na celotno poročilo. V tem delu je opisano izobraževanje osebja, ki izvaja proces triaže na terenu ter tako izključuje potrebo po digitalnem forenziku na kraju zločina. Takšen način dela je že uspešno vzpostavljen v običajni forenziki, tj. preiskavi kraja zločina. Na tem področju je terensko osebje izobraženo za specifične naloge, s katerimi dopolnjujejo izobražene strokovnjake. Takšen koncept digitalne triaže na terenu je možen z razvojem novega procesnega modela, ki zagotavlja smernice terenskemu osebju. Kot dokaz ustreznosti je novi procesni model predstavljen in ocenjen. Rezultati nam prikazujejo, kako se vključevanje specialistov in nespecialistov za delo z digitalnimi dokazi lahko bolje kosa s povečanim številom preiskav, ki vključujejo digitalne dokaze.

1.2 Digital evidence, 'absence' of data and ambiguous patterns of reasoning

In this paper we discuss the use of digital data by the Swiss Federal Criminal Court in case of attempted homicide. This case is example of drawback for the defense, where the presentation of scientific evidences is partial. This paper consists of two parts, first is non-technical presentation of the topic, which means drawing parallels between the court's summing up of the case and flawed patterns of reasoning commonly seen in other forensic disciplines such as gunshut residues. Second part is a formal analysis of the case, where we are using probability and graphical probability models for scientific approach. In that part we will justify the claim that the partial presentation of digital evidence brings a risk of hiding vital information from the defense.

1.3 Bitcoin: Celovit pregled decentraliziranih digitalnih valut

Poleg ustvarjanja miljardo dolarjev vredne ekonomije je Bitcoin revolucioniral področje digitalnih valut in vplival na veliko podobnih področij. To je privabilo tudi veliko znanstvenega interesa. V tem članku začnemo s pregledom Bitcoin protokola in njegovih gradnikov. Nato predstavimo temeljne strukture in vpoglede v jedro Bitcoin protokola in njegovih aplikacij. Opišemo tudi glavne varnostne groženje in kako se Bitcoin z njimi spopada. Na koncu se dotaknemo tudi problematike zasebnosti.

1.4 Bitcoin in mit decentralizacije: Predlogi za ponovno decentralizacijo sistema

Bitcoin je prva od tako imenovanih kripto valut. Z njenim prihodom na sceno leta 2009 se je začelo novo obdobje v zgodovini denarja. Namen Bitcoin-a ni bil nič manj kot spodnesti noge današnjim monetarnim inštitucijam in njihovemu modelu centraliziranega upravljanja s tokom denarja. Zaupanja v te ustanove naj bi bilo zamenjano z zaupanjem v računalniško kodo, pravilno zasnovo algoritmov in decentralizirane sisteme. Kljub tem visokim ciljem pa se je izkazalo da zasnova samega Bitcoin protokola ne spodbuja željene decentralizacije ampak ravno nasprotno. V tem članku si bomo pogledali na kakšen način Bitcoin protokol ne spoštuje prvotno zastavljenih ciljev in predstavili tri kategorije možnih popravkov, popravki na nivoju strojne opreme, programske opreme in električnega omrežja. Te bi lahko pomagale vrniti Bitcoin in podobne kripto valute na pravo pot decentralizacije.

1.5 Raziskava Bitcoin bločne verige: analiza celotnega grafa uporabnikov

Bitcoin je decentralizirana kripto valuta, ki je pred kratkim dobila pozornost širšega občinstva. Zanimiva značilnost tega sistema oziroma valute je ta, da je seznam vseh transakcij ki se shranjujejo od samega nastanka valute, javno dostopen. To omogoča preiskavo gibanja sredstev za odkrivanje zanimivih lastnosti ekonomije te value. V tem članku bomo povzeli opravljene analize omrežja Bitcoin, ki so jih predstavili v izvirnem članku [16]. Analize so narejene na verigi blokov (angleško blockchain), iz decembra 2015, ko je število transakcij eksponentno naraslo po zadnjih dveh letih. Skupek analiz, ki so opredeljene vsebuje med drugim

analizo časovnega razvoja Bitcoin omrežja, preverjanje domneve *Bogato se bogati* in odkrivanje vozlišč, ki so ključnega pomena za povezljivost omrežja valute.

1.6 BitConeView: Vizualizacija pretoka Bitcoin transakcij

Bitcoin je digitalna valuta, katere transakcije so shranjene v javno dostopnih zapisih. Te zapise imenujemo blockchain in si jih lahko predstavljamo kot ogromen usmerjen graf z več kot 70 milijoni vozlišč, kjer vsako vozlišče predstavlja transakcijo in vsaka povezava predstavlja bitcoine, ki se pretakajo med transakcijami. V poročilu je opisano orodje za vizualizacijsko analizo, ki nam pove, kdaj in kako se tok bitcoina meša z drugimi tokovi v transakcijskem grafu. Ta sistem temelji na prispodobah, s katerimi prikažemo velikost in ostale lastnosti transakcij in s tem omogoča visokonivojsko analizo drugače nepreglednih podatkov.

1.7 Bitcoin napad s prikrivanjem blokov : Analiza in ublažitev napada

Bitcoin je prva kriptovaluta, ki še danes prevladuje v popularnosti in količini uporabe. V tem članku je obravnavana varnostna luknja v obstoječi shemi sistema Bitcoin, ki omogoča,izvajanje napadov s prikrivanjem blokov (blockwitholding attack BWA). Ta napad se izvaja nad rudarskimi bazeni (mining pool) in ima lahko velike posledice tako za člane bazena kot tudi za celoten Bitcoin sistem. Avtorji so raziskali nekaj posebnih različic tega napada in poskušali ugotoviti dobiček, ki ga pridobi napadalec. Predlagali so tudi nekaj načinov za preprečitev tega napada, ki se razlikujejo obstoječih predlaganih rešitev. Namesto odkrivanja ali zmanjšanja motivacije za napad so se avtorji odločili za pristop, ki popolnoma izniči zmožnosti izvajanja takšnega napada, s pomočjo kriptografskih in računskih metod.

1.8 Varnostna analiza predlaganih izboljšav Bitcoina

Če pri internetnem prenosu denarja ne želimo zaupati posebnim spletnim posrednikom, se poslužimo decentraliziranih sistemov kriptovalut. Najbolj priljubljena kriptovaluta je trenutno Bitcoin, ki deluje po principu bločne verige, ki omogoča sprotno preverjanje transakcij in hranjenje zgodovine transakcij kar s strani preostalih uporabnikov kriptovalutnega sistema. Napad dvojne porabe in verižni razcep sta trenutno glavni dve težavi v protokolih, ki temeljijo na tem principu. Pri napadu dvojne porabe napadalec uporabi isto enoto bitcoina večkrat. Verižni razcepi pa povzročijo neskladnosti v zgodovini med uporabniki. Z verjetnostno analizo pokažemo, da predlogi za rešitev teh dveh težav, ki so bili nedavno objavljeni, ne delujejo dobro na večjih bločnih verigah.

1.9 Nenadzorovano učenje za namene iskanja prevar pri trgovanju z Bitcoinom

V zadnjem času kripto-valuto Bitcoin prevzema vse več uporabnikov, s tem pa se dogaja tudi vedno več spletnega kriminala, v katerem je Bitcoin vpleten. V primeru, da se želimo obvarovati pred različnimi prevarami, potrebujemo metode strojnega učenja, ki nam na podlagi zaznavanja anomalij v omrežju skrbijo in nas opozarjajo pred možno prevaro. Patrick Monamo in ostali raziskovalci z inštituta CSIR (angl. Council for Scientific and Industrial Research) so v svo jem članku z naslovom Unsupervised Learning for Robust Bitcoin Fraud Detection predstavili novo metodo za odkrivanje prevar v omrežju Bitcoin. Predstavljena metoda temelji na strojnem učenju z uporabo rezanih razvrščanj z voditeljem (angl. trimmed k-means). S to metodo so zmožni hkratnega gručenja objektov in odkrivanja prevar v omrežju. Njihov pristop se izkaže za uspešnega, saj so na podatkovnem viru odkrili več goljufivih transakcij, kot so jih odkrile podobne raziskave na istem podatkovnem viru.

1.10 Behind Closed Doors: Measurement and Analysis of CryptoLocker Ransoms in Bitcoin

This paper analyze ransom payments required by CryptoLocker which were done using Bitcoin. In the Introduction the basic description of cryptocurrency Bitcoin and ransomware CryptoLocker is provided. Firstly the cluster of Bitcoin addresses which belong to Cryptolocker was gathered, then the analysis of incoming transactions of these addresses were made. These incoming transactions was filtered (by amount which was usual in given time) and then several conclusions about the typical targets (and especially their geographical location) of CryptoLocker attacks. Later, we tried to replicate part of original research with lately appeared WannaCry ransomware which affected lot of companies and institutions in Europe.

1.11 A method and a case study for the selection of the best available tool for mobile device forensics using decision analysis

Mobilne naprave v današnjem času najdemo povsod. Uporabljajo se za zabavo, učenje, finančne transakcije, poslovne namene in še bi lahko naštevali. Zaradi velikega števila naprav (več kot 1 mobilna naprava na osebo dandanes ni nič nenavadnega) to privede do velikega digitalnega odtisa vsakega posameznika. Posledica tega pa je zato vedno večja količina ilegalnih dejanj, ki se tičejo tudi mobilnih naprav. Naučiti se moramo, kako ta dejanja najučinkoviteje identificirati in preprečiti. V tem članku sta predstavljeni in ovrednoteni dve orodji, s katerima si pomagamo pri soočanju s to težavo. Orodje za forenzično analizo izberemo v fazi priprave (angl. preparatory phase) v samem postopku digitalnega preiskovanja mobilne naprave. V primeru, da ne izberemo najbolj primernega orodja za izvedbo preiskave, lahko to hitro pripelje do nepopolne in nepravilne analize digitalnega dokaza. Orodji sta ocenjeni z dvema faktorjema in sicer z ustreznostjo tipa dokaza (v smislu koliko pozitivnega doprinese tip digitalnega dokaza k preiskavi) ter z zmogljivostjo orodja (v smislu zmogljivosti posameznega orodja glede na tip digitalnega dokaza). V tem članku sta opisani orodji XRY (alternativa 1, Alt1) in UFED (Universal Forensic Extraction Device,alternativa2,Alt2),pri čemer je dokazano, da je orodje XRY v večini primerov boljše od orodja UEFD.

1.12 Forenzično preiskovanje v programsko definiranih (SDN) omrežjih

Računalniška omrežja se iz leta v leto povečujejo. Ocenjujejo, da se vsako sekundo v internet poveže približno 80 novih naprav, številka pa se z leti še povečuje. Ogromno število novih naprav tako za načrtovalce omrežij predstavlja izziv. Internetno omrežje kot ga poznamo danes temelji na protokolu IP. Glavni sestavni del takšnega omrežja so usmerjevalniki, ki glede na naslov kamor je paket namenjen, le tega usmerjajo po omrežju. Za ta namen uporabljajo različne usmerjevalne protokole. S povečevanjem omrežja se povečuje tudi kompleksnost usmerjanja. Posledično postavitev takšnega omrežja zahteva veliko dela s konfiguracijo omrežnih naprav. Takšno omrežje je tudi zelo težko spreminjati oziroma posodabljati. Kot odgovor na zgornjo problematiko so se pojavila programsko definirana omrežja. V nadaljevanju je najprej predstavljeno nekaj osnov programsko definiranih omrežij. Nato je predstavljeno, kako programsko definirana omrežja vplivajo na omrežne forenzične preiskave ter kako so nam lahko pri tem v pomoč.

1.13 Iskanje značilnih vzorcev v programsko definiranih omrežjih

Programsko definirana omrežja (SDN Software Defined Networks) so pomemben del internetne infrastrukture. Zaradi poenostavitve upravljanja z omrežjem so SDN zanimiva za velike podatkovne centre in ponudnike interneta ter internetnih storitev. Pri le-teh je zelo pomembno nemoteno in stabilno delovanje omrežja, ki ga lahko zmotijo napadi na omrežje. Pomembno je, da potencialni napadalci vedo čim manj o omrežju, saj je s tem oteženo njihovo delovanje. V SDN se omrežne naprave upravlja centralizirano ločeno od delovanja (izvajanja funkcije) naprave. Ločeno upravljanje in delovanje ima za posledico drugačno obravnavo paketov (na kontrolni in podatkovni ravni), ki se pretakajo skozi omrežje. V tem delu bom povzel in opisal članek On the Fingerprinting of Software-Defined Networks avtorjev Heng Cui et. al [7], ki obravnava iskanje značilnih vzorcev (fingerprinting) v delovanju SDN omrežja. Članek predstavi ranljivost SDN omrežja in sicer, da je mogoče razločiti, kdaj omrežje za določeni podatkovni tok uporablja že vzpostavljena pravila in kdaj je potrebna intervencija kontrolne ravni za določitev novih pravil. Pokazano je, kako lahko z veliko verjetnostjo razločimo med tema načinoma delovanja ter kaj lahko s to informacijo naredi napadalec (npr. nad omrežjem izvede napad DoS). Prav tako je predlagan način, kako se možnosti razločevanja zmanjša. V seminarski nalogi bom opisal SDN omrežja ter obravnaval delo in rezultate članka. Obravnaval bom implikacije zbiranja značilnih vzorcev obnašanja omrežja z vidika digitalne forenzike in varnosti.

1.14 Opportunistic Piggyback Marking for IP Traceback

Sledenje IP naslova je rešitev za problem iskanja izvora paketa, pri kibernetskih napadih ali spletnih prevarah in je uporabna pri zbiranju in analizi spletnega prometa. Ena izmed rešitev sledenja IP naslova je metoda, ki sloni na principu sledenja z označevanjem (Marking-based traceback MBT). MBT metoda zelo obeta in je požela veliko pozornosti stroke. Čeprav metoda MBT veliko obeta pa ima tudi pomankljivosti in ena večjih je prenos sporočil namenjenih sledenju paketov. Prenos teh sporočil je ena glavnih funkcionalnosti sledenja paketov. V članku opisujeva rešitev pomankljivosti metode MBT imenovano oportunistično označevanje za sledenje IP paketov (OPM). OPM se od večine metod razlikuje, da ima ločeno vsebino sporočil za sledenje

in funkcijo za dostavljanje sporočil. Poleg tega pa učinkovito doseže hitro in robustno dostavo sporočil z izkoriščanjem označevalnih možnosti. Na podlagi predlagane OPM sheme predstavimo prilagodljivo ogrodje na podlagi označevanja, ima nekaj prednosti pred ostalimi metodami za sledenje IP paketov. Z evalvacijo simulacij prikažemo, da predstavljen sistem učinkovito zmanjša število izgubljenih sporočil ter zmanjša obremenjenost usmerjevalnikov.

1.15 Forenzična raziskava primerov spletnega zalezovanja, rešenih z uporabo analize vedenjskih karakteristik

Analiza vedenjskih karakteristik (angl. behavioural evidence analysis) je postopek, ki pripomore k razumevanju digitalnih dokazov in rekonstrukcije zločina. Kljub pomembnosti analize vedenjskih karakteristik še ne obstaja veliko raziskav o apliciranju tega postopka na kriminalna dejanja. V seminarski nalogi bomo opisali pomen analize vedenjskih karakteristik (v nadaljevanju AVK), pri čemer bomo upoštevali izvlečke iz članka Forensic investigation of cyberstalking cases using Behavioural Evidence Analysis [2]. Izkaže se, da AVK pripomore k sami raziskavi, k razumevanju zločinca in žrtve ter k razbiranju sledi iz digitalnih dokazov. Tako raziskava postane bolj smiselna in natančna.

1.16 Forenzična analiza podatkov iz oblačnih storitev: študija primera Google Docs

za pridobivanje podatkov iz spletnih storitev za shranjevanje in urejanje dokumentov. Na koncu bomo predstavili še lastno implementacijo orodja za iskanje po zgodovini Google Docs dokumentov. Članek [6] se osredotoča na priljubljeno Googlovo storitev Docs, vendar so predstavljeni koncepti in pristopi uporabni tudi za druge oblačne storitve. Poudarek je na pridobivanju in analizi podatkov, ki se nahajajo v samem oblaku (angl. cloud-native artifacts), v nasprotju z analizo sledi, ki jih spletne storitve pustijo na uporabnikovih napravah.

1.17 Uporaba analiz sej internetne zgodovine za lažje izvajanje forenzičnih preiskav večuporabniških računalniških okolij

Raziskava predstavi nov pristop za identifikacijo uporabnika, ki je uporabljal računalnik v času zločina. Pristop se izvaja z agregacijo internetne zgodovine obnovljene naprave v seje. S primerjavo pridobljenih sej se lahko določi, ali je posamezna seja enkraten ali ponovljiv dogodek (npr. uporabnikova navada). Pri tem se osredotoči na dva pristopa za agregacijo sej. Seje nespremenljive (fiksne) dolžine in seje spremenljive dolžine. Predstavi tudi pristop za odkrivanje ponovljivih vzorcev, ekstrakcijo teh vzorcev in njihovo predstavitev v obliki binarnih nizov. Za primerjavo sej se uporabi Jaccardov podobnostni koeficient, s katerim lahko določimo mero podobnosti med sejami in z visoko verjetnostjo identificiramo uporabnika neke seje. Eksperimenti so bili izvedeni na dveh testnih množicah, kjer je več uporabnikov imelo dostop do istega računalnika.

1.18 Preverjanje avtorstva dokumentov za različne jezike, žanre in tematike

V članku je podan pregled področja preverjanja in določanja avtorstva dokumentov. Jedro članka je osredotočeno na opis ključnih komponent izvornega članka Authorship verification for different languages, genres and topics[3], na podlagi katerega smo tudi implementirali predlagano metodo preverjanja avtorstva. Članek navaja rezultate testiranja in ugotovitve avtorjev izvornega članka, podana pa je tudi krajša primerjava uspešnosti algoritma s konkurenčnimi rešitvami. Metoda je bila testirana na 28 različnih korpusih s 16 žanri in različnimi tematikami, ki skupno zajemajo kar 4525 primerov besedil. Rezultati so primerljivi s konkurenčnimi metodami (mediana 75%) in ponekod boljši (v povprečju 5%) od trenutno najboljših metod. Glavni prednosti predstavljene metode so v enostavni razširljivosti z novimi jeziki in nizka računska zahtevnost.

1.19 Digital Forensics as a Service: an update

V današnjih časih, ko količina podatkov strmo narašča, se v računalniški forenziki pojavljajo vedno novi izzivi. Kako implementirati učinkovito centralizirano storitev, katera bi sprostila forenzike ter omogočila, da svojo preiskavo opravijo bolj učinkovito in podrobno. S tem problemom se ukvarjajo avtorji članka, kateri

opisujejo kako na Nizozemskem rešujejo ta problem s sistemom Xiraf ter njegovim naslednikom sistemom Hansken.

1.20 Avtomatizirano generianje profila za analizo živega Linux pomnilnika

Analiza živega pomnilnika na Linux platformi zaradi narave jedra že od zmeraj predstavljala težavo računalničarjem. Zahteva namreč izredno veliko znanja o sami razporeditvi vsebine pomnilnika, ki ga je običajno veliko lažje pridobiti z razhroščevalnimi simboli generiranimi v času prevajanja programa. Jedro Linuxa je običajno brez razhroščevalnega načina, poleg tega pa je izredno konfigurabilno, kar običajno preprečuje, da bi se informacije o razhroščevanju širile med ostale sisteme, ki si jih ne lastijo. Trenutno je kakršnokoli pridobivanje informacij za odzivne aplikacije na varnostne incidente postalo izjemno zapleten in časovno potraten postopek, kar pomeni, da je tovrsten način v praksi neprimeren. Avtorji članka [6] so razvili orodje z imenom Layout Expert, ki omogoča izračun razporeditve vsebine pomnilnika kritičnih struktur jedra med izvajanjem programa brez uporabe dodatnih orodij (npr. prevajalna veriga). Namen njihovega orodja je adaptacija generiranih profilov za Linuxova jedra poljubne verzije, kjer profili označujejo začetne in končne naslove za strukture v pomnilniku. Rezultat je sistemsko specifičen profil z natančno informacijo o postavitvi. Orodje je bilo dodano kot razširitev odprto kodni programski opremi Rekall za analizo pomnilnika v forenzičnih preiskavah. V tem članku predstavimo problem izvajanja analize nad pomnilnikom v Linux sistemov, in pregledamo obstoječe rešitve. Opišemo in demonstriramo tudi orodje za analizo na živem pomnilniku (angl. live memory).

1.21 Obnovitev močno fragmentiranih JPEG datotek

Obnavljanje izbrisanih dokumentov predstavlja pomemben del forenzične raziskave. V tem delu so se avtorji osredotočili na obnavljanje JPEG datotek, s poudarkom na datotekah, ki so močno fragmentirane. Glavna težava pri obnavljaju močno fragmentiranih datotek nastopi pri procesu sestavljaja fragmentov v pravilni vrstni red. Za soočanje s to problematiko, avtorji predlagajo novo metriko CED (Coherence of Euclidean distance), za boljše predvidevanje sosednjosti dveh fragmentov. Razvili so tudi svoj algoritem, ki ima poleg drugačne metrike tudi nekatere druge prednosti pred obstoječimi algoritmi. Učinkovitost novega algoritma so primerjali proti dobro poznanemu orodju APF (Adroit Photo Forensic). Teste so izvajali na slikah iz SD kartice neke digitalne kamere.Rezultati primerjave so pokazali, veliko prednost novega orodja pred obstoječim APF. Avtomatsko so uspeli obnoviti kar 97% fragmentiranih JPEG slik, med tem ko je bil APF uspešen le pri 79% slik.

Del I
Metodologija

Večstopenjski forenzični metodološki model za digitalno triažo na terenu

Primož Lavrič
Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana, Slovenija
pl9506@student.uni-lj.si

Janez Štular
Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana, Slovenija
js2267@student.uni-lj.si

POVZETEK

Zaradi finančnih omejitev in zahteve po visokem usposabljanju digitalnih forenzikov primanjkuje po celem svetu. To posledično vodi do tega, da od zajema digitalnih dokazov do prejema forenzičnega poročila preteče veliko časa. V namen skrajšanja tega časa je bilo predlaganih nekaj postopkov za vzpostavitev triaže digitalnih dokazov. Z uporabo triaže je preiskovalcu omogočen hitrejši dostop do informacij, medtem ko čaka na celotno poročilo.

V tem delu je opisano izobraževanje osebja, ki izvaja proces triaže na terenu ter tako izključuje potrebo po digitalnem forenziku na kraju zločina. Takšen način dela je že uspešno vzpostavljen v običajni forenziki, tj. preiskavi kraja zločina. Na tem področju je terensko osebje izobraženo za specifične naloge, s katerimi dopolnjujejo izobražene strokovnjake. Takšen koncept digitalne triaže na terenu je možen z razvojem novega procesnega modela, ki zagotavlja smernice terenskemu osebju. Kot dokaz ustreznosti je novi procesni model predstavljen in ocenjen. Rezultati nam prikazujejo, kako se vključevanje specialistov in nespecialistov za delo z digitalnimi dokazi lahko bolje kosa s povečanim številom preiskav, ki vključujejo digitalne dokaze.

1. UVOD

V sodobnem policijskem okolju narašča potreba po hitrejšem izvajanju dela. Na področju digitalne forenzike takšen način dela povečuje zamude za obe strani, tako preiskovalce kot sodni sistem, ki čaka na podrobno poročilo digitalnih forenzikov. Problem povečevanja zamud na področju digitalnih dokazov je bil raziskan na številnih policijskih oddelkih in je dobro dokumentiran v čalnkih [7, 3, 6]. Ozko grlo preiskave digitalnih dokazov predstavlja pretečeni čas od zajema digitalnih dokazov, dostave tehnološki enoti za kriminal TCU (angl. Technological Crime Unit) in dodelitvijo dokazov digitalnemu forenziku, ki mora opraviti podrobno analizo in napisati poročilo. Posledično preiskovalci ostanejo brez potencialnih informacij o kaznivem dejanju v času, ko bi jih najbolj potrebovali. Sodobna družba postaja vse bolj povezana z uporabo mobilnih telefonov, tablic in računalnikov, kar se odraža tudi pri uporabi omenjenih tehnoloških naprav v kriminalne namene. Zamude pri preiskavi digitalnih dokazov posledično vplivajo na pravico obtoženca do hitrega sojenja. Kot primer: pravica iz Kanadske listine o pravicah in svoboščinah iz razdelka 11b, ki pravi, da bodo obtoženemu poizkušali soditi v doglednem času.

Na tem področju obstajajo številne metodologije, ki opredeljujo potrebo po analizi digitalnih dokazov na terenu. V tem delu je predstavljen neformalni način ocenjevanja zaostankov digitalnih preiskav, ki lahko trajajo od enega do štirih let, kot je opisano v članku [6]. Problem so omejena finančna sredstva za opravljanje digitalnih preiskav in omejeno število ustrezno usposobljenih digitalnih forenzikov. Posledično so digitalni forenziki poslani na teren z namenom digitalne analize na terenu in ne morejo hkrati opravljati dodeljenega dela v laboratoriju. Preiskave na terenu imajo zanje najvišjo prioriteto ne glede na stopnjo zločina in jim morajo posvetiti polno pozornost. Ta zahteva pripomore k še večjemu povečanju zaostanka pri pregledovanju digitalnih dokazov v laboratoriju.

Predlagan procesni model razširimo s triažo na terenu in vključimo komponento, v kateri je delo digitalnih forenzikov dodeljeno posebej usposobljenemu osebju. Omenjeno osebje bi sodelovalo s preiskovalci in bi bilo deležno osnovnega usposabljanja s področja digitalne forenzike. Njihovo delo mora ohranjati celovitost digitalnih dokazov. Dokazov ne smejo obdelovati, saj so samo preiskovalci z dodatnimi znanji. Takšen procesni model že uporabljajo za nekatere naloge v kanadski policijski enoti Royal Canadian Mounted Police. Kot primer je pobiranje prstnih odtisov naloga posebej usposobljenih preiskovalcev in ne več forenzikov. Od preiskovalca, ki je pobral prstni odtis ni pričakovano, da bo opravil njegovo analizo in identifikacijo njegovega lastnika, temveč je to še vedno delo forenzika, ki pa se posledično ne potrebuje udeleževati vseh krajev zločina.

Predlagani rezultati modela digitalne triaže na terenu so povečanje učinkovitosti preiskovanja in zmanjšanje zaostankov pri čakanju poročil digitalnih forenzikov. Glavni cilj tega dela je zagotoviti okvir za organe pregona, katerih določeno osebje je posebej usposobljeno za digitalno triažo na terenu. Predstavlja izhodišče za digitalno triažo na terenu, ki je lahko nadaljnje prilagojena za različne agencije. Agen-

cije lahko zavzemajo poljubno velikost, vse od mestnih policijskih postaj do državne enote za zločin. Posledično so lahko usposobljeni preiskovalci detektivi v isti stavbi ali pa policisti na oddaljenih lokacijah, njihov pristop k digitalnim dokazom pa bo še vedno ostal konsistenten. To osebje bo lahko pomagalo tudi pri oceni stopnje zločina in s tem stopnje prednosti.

1.1 Prispevek tega dela

Trenutno ni standardiziranega pristopa za neforenzično začetno usposabljanje osebja za delo z digitalnimi dokazi. V tem delu je predstavljen okvir za obravnavanje digitalnih dokazov v začetnih korakih preiskave, ki potekajo izven forenzičnih laboratorijev in jih izvaja osebje, ki ni posebej specializirano za delo z digitalnimi dokazi. Doseči skušamo dva glavna cilja:

1. povečati učinkovitost preiskav z zagotavljanjem pravočasnega zbiranja digitalnih dokazov,
2. zmanjšati zamudo pri obravnavi digitalnih dokazov s strani digitalnih forenzikov v laboratoriju.

Za doseg teh ciljev je predstavljen formalni model, ki vključuje delo nespecializiranega osebja z digitalnimi dokazi, kot tudi pregled realizacije.

2. SORODNA DELA

2.1 Procesni model triaže na terenu s področja računalniške forenzike

Procesni model triaže na terenu s področja računalniške forenzike, predlagan v članku [12], je skupni pregled potrebe po triaži na terenu, ki bi bila del katerekoli forenzične metodologije. Avtorji članka [12], so ugotovili, da posamezniki, ki so radikalni, izobčeni in se ne strinjajo s sistemom ter se vedejo kriminalno, večinoma uporabljajo tehnološke naprave za izboljšanje in širjenje njihovih kriminalnih dejanj. Izveden je bil pregled preiskovalnih modelov, razvitih z namenom pomoči organom pregona pri obdelavi digitalnih dokazov. Vsi modeli skušajo zajeti celoten postopek, povezan z analizo digitalnih dokazov. Ti postopki so za digitalne dokaze časovno potratni in še vedno zahtevajo prenos digitalnih dokazov do glavne lokacije za analizo. Tak procesni model odpove v časovno kritičnih situacijah, kot so ugrabitve in grožnje s terorističnimi napadi. V takšnih situacijah hitro potrebujemo informacije, ki odtehtajo potrebe po podrobni preiskavi vseh potencialnih digitalnih dokazov.

2.2 Procesni model

Procesni model triaže računalniške forenzike CFFTPM (angl. Computer Forensics Field Triage Process Model) so avtorji članka [12] definirali na naslednji način: postopki preiskave, ki so opravljeni v prvih nekaj urah preiskave in zagotovijo uporabne informacije za zasliševanje osumljencev ter nadaljnjo preiskavo. Zaradi potrebe po zajemu teh informacij v razmeroma kratkem času, model običajno vključuje analizo računalniških sistemov na terenu pod vprašajem. Glavni cilji tega modela so:

1. Takoj najti uporabne dokaze.
2. Takoj prepoznati potencialno ogrožene žrtve.
3. Voditi preiskavo v teku.
4. Ugotoviti potencialne obtožbe.
5. Natančno oceniti nevarnost storilca za družbo.

Pri zagotavljanju omenjenih ciljev mora model istočasno tudi zagotavljati celovitost digitalnih dokazov in ohranjati dokaze za nadaljnjo preiskavo in analizo.

Prednost tega modela je zmožnost, da preiskovalni skupini informacije zagotavlja hitro in učinkovito. Pristop tega procesnega modela ni delo s celotnim procesom, ampak razširitev na stališče terena z opredelitvijo splošnih in specifičnih faz preiskave. Teh šest faz predstavlja visoko kategorizacijo, saj ima vsaka faza več podnalog in zahtev, ki so specifične glede na primer, datotečni in operacijski sistem.

2.2.1 Faze

Začetna faza CFFTPM opredeli, da je pravilna priprava in načrtovanje ključni del vsake preiskave. Njen razpon sega od logistike do informiranja o vrsti kriminalnega dejanja in ostale inteligence. Fazi načrtovanja sledi sama triaža, ki je definirana kot:

postopek, v katerem so stvari razvrščene po pomembnosti in prioriteti. Te stvari, dele dokazov ali potencialne vsebovalnike dokazov, ki so najbolj pomembni ali občutljivi, je potrebno obdelati prve, kot je opisano v članku [12].

Triaža je ključnega pomena za CFFTPM in v povezavi s fazo načrtovanja služi kot temelj za vse ostale faze. Preiskovalcem in zasliševalcem, ki delajo direktno z osumljencem ali žrtvijo, je potrebno v tej fazi zagotoviti direktno povezavo z digitalnim forenzikom. To zagotavlja pravilno prednostno razvrstitev in pravilno ustvarjanje domnev, saj se pogosto zgodi, da je digitalni forenzik takrat prvič vključen v preiskavo in tako nima predznanja o primeru.

Naslednje faze zagotavljajo podrobnosti o vrstah artefaktov, ki lahko pomagajo digitalnemu forenziku pri zbiranju dokazov. Faza artefakte loči na splošne artefakte in artefakte, ki so specifični za primer. Splošni artefakti so podobni za vse primere in se tičejo uporabniških računov, časovnice in interneta. Artefaktom, ki so specifični za vsak primer, prilagodimo širino forenzične raziskave, saj so posebej povezani s trenutno preiskavo. V primeru otroške pornografije morajo imeti najvišjo prioriteto datoteke z zvočno in grafično vsebino, ki vsebujejo otroško pornografijo. Čeprav tradicionalna preiskava verjetno vključuje temeljit pregled vseh teh artefaktov tako kot tudi vseh drugih. Urejeno zaporedje CFFTPM zahteva, da preiskovalec razumno presodi morebitne pridobitve pri preiskavi vseh teh artefaktov, pri čemer mora upoštevati dodatno porabo časa.

2.2.2 Diskusija

S pomočjo CFFTPM je bilo ugotovljeno, da je v časovno kritičnih situacijah potrebno artefakte iz digitalnih naprav pridobiti hitreje, kar je najbolje storiti na terenu. Ko digitalni forenzik sledi CFFTPM je pomembno, da ne onemogoči ponovnega podrobnejšega forenzičnega preizkusa računalnika

v laboratoriju. Skozi celotni postopek je potrebno zagotavljati tako celovitost digitalnih dokazov kot tudi verigo skrbništva. CFFTPM je odlična začetna točka, vendar se zanaša na uporabo usposobljenih digitalih forenzikov. Ideje CFFTPM bodo uporabljene kot osnova za model, ki vpeljuje osebje, ki ni specializirano za delo z digitalnimi dokazi, vendar pa izvršuje podobne naloge kot digitalni forenziki.

2.3 Standard ISO 27037

Naslov standarda ISO 27037 je: "Informacijska tehnologija - varnostne tehnike - smernice za identifikacijo, zbiranje, pridobivanje in ohranjanje digitalnih dokazov", podrobneje opisan v priporočilih [2]. To je mednarodni standard, ki določa smernice za specifične dejavnosti, odgovorno osebje za ravnanje z morebitnimi digitalnimi dokazi ter definira postopke v zvezi z digitalnimi dokazi. Dva definirana položaja sta prve odzivne osebe za digitalne dokaze DEFR (angl. Digital Evidence First Responders), ki so odgovorne za identifikacijo digitalnih dokazov, ter specialisti za digitalne dokaze DES (angl. Digital Evidence Specialists), ki so odgovorni za zbiranje digitalnih dokazov.

Razvili bi lahko mnogo metodologij, ki bi jih lahko certificirali in bi sledili standardu. Standard opisuje, kaj mora biti storjeno in ne kako mora biti storjeno. Standard na primer predpisuje, da mora biti forenzična kopija narejena, jasno identificirana v kontekstu in sledljiva, ne predpisuje pa, katero orodje moramo za to uporabiti. Znotraj standarda ISO je za omenjena položaja DEFR in DES predpisano, da morajo njihova dejanja in digitalni dokazi slediti konceptom sledljivost, ponovljivost in obnovljivost.

Standard ISO prinaša koncept triaže tako, da prepozna, da mora rokovanje z digitalnimi dokazi ustvariti ravnotežje med kvaliteto dokazov, časom, potrebnim za analizo, obnovljivostjo in ceno zbiranja digitalnih dokazov. Kakršnakoli razvrstitev tega ravnotežja mora minimizirati tveganje za morebitno ogrozitev celovitosti dokazov in maksimizirati vrednost dokazov iz potencialnega zbiranja digitalnih dokazov. Digitalni dokazi so urejeni s tremi glavnimi načeli:

- Ustreznost - Digitalni dokazi so ustrezni, kadar prispevajo k potrditvi ali ovržbi elementov specifičnega preiskovanega primera.
- Zanesljivost - Za zagotovitev, da so digitalni dokazi to, kar naj bi bili.
- Zadostnost - Dovolj velika zbirka morebitnih digitalnih dokazov, da je primer lahko zadostno preiskan; razumevanje tega koncepta je pomembno zaradi prioritiziranja dela, ko je omejen čas ali denar.

2.4 Zaostanki

Pred uporabo triaže je bil forenzični postopek časovno potraten, kot je definirano v modelu CFFTPM opisanem v članku [12]. Med forenzičnim postopkom je bilo tu ozko grlo predvsem zato, ker je bilo število terenskih preiskovalcev večje od števila razpoložljivih digitalnih forenzikov v tehnološki enoti za kriminal TCU. Medtem ko so forenziki organizirali in analizirali podatke ter pisali poročilo, so prihajale nove preiskave, ki so se nabirale v čakalni vrsti. Ker je vrsta preiskav naraščala, je bil zaostanek pri preiskavah vse bolj očiten.

Zaradi nastalega zaostanka je potreben mehanizem, ki pove, katera preiskava naj bo izvedena prva. Klasičen mehanizem "prvi pride, prvi gre" ni primeren, saj ne upošteva stopnje zločina. Identifikacija ukradene lastniške datoteke ne more imeti prednosti pred preiskavo pedofilije. Standardni mehanizem temelji na stopnji zločina, tako da kaznivim dejanjem zoper osebe dodeli višjo prioriteto. Velik delež preiskav vključuje otroško pornografijo in vedno obstaja možnost storilca. Seveda zločini te vrste pridobijo prioriteto, kar pomeni, da zločini, ki so povezani z goljufijo, pridobijo vedno nižjo prioriteto.

Model CFFTPM prikazuje prednosti pri udeležbi na kraju zločina, vendar vseeno prepozna potrebo po kasnejši podrobnejši analizi digitalnih dokazov. Vprašanje je, ali digitalni forenzik tehnološke enote za kriminal TCU nadaljuje z analizo digitalnih dokazov, ki jih je raziskal na kraju zločina, ali pa morajo dokazi iti v čakalno vrsto. Kakorkoli, čakalna vrsta se ne zmanjšuje in na ta način jo lahko celo podaljšamo.

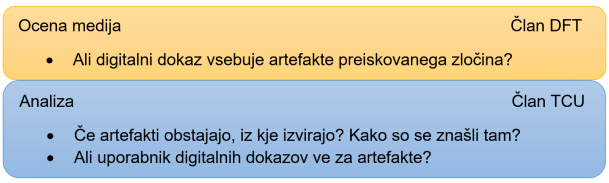
Prizadevanje za zmanjšanje zaostankov in čakalne vrste vključuje zaposlovanje tehnikov, ki le sprejemajo in pridobivajo digitalne dokaze, tako da se lahko digitalni forenziki osredotočijo le na analizo in pisanje poročil. V nekaterih primerih ta način zmanjša zaostanke, vendar jih vseeno ne odstrani, niti jih ne spravi na sprejemljivo raven. Koncept triaže je bil uveden tudi v laboratoriju, kar je zmanjšalo čas za zaključitev analize, saj nerelevantni artefakti niso bili deležni polne analize. Z naraščanjem števila digitalnih naprav, ki so bile zasežene med preiskavo, se mora digitalni forenzik še vedno posvetiti vsaki posredovani napravi, kar je ponovno potrata časa.

3. DIGITALNA TRIAŽA NA TERENU

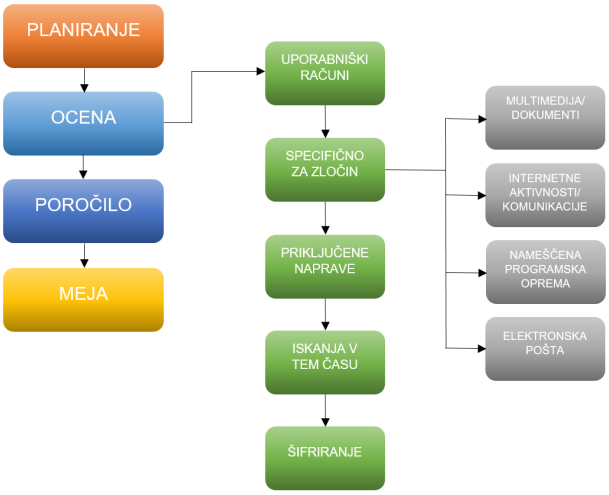
Digitalna triaža na terenu DFT (angl. Digital Field Triage) je zasnovana za zagotavljanje znanja, spretnosti in zmožnosti osebju, ki ni specializirano za delo z digitalnimi dokazi, z namenom izvajanja omejenih forenzičnih dejavnosti, kot je opisano v članku [12]. Da lahko DFT deluje, so postavljeni trije temeljni koncepti:

- DFT ne more delovati izolirano brez nadrejene TCU.
- DFT mora zagotavljati forenzično integriteto podatkov.
- Ocena DFT ne nadomešča analize digitalnega forenzika iz TCU.

Zaradi zagotavljanja standardiziranega pristopa člani ekip DFT prejemajo usposabljanje, stalno podporo in upravljanje s strani nadrejene TCU. Nadrejeni TCU je odgovoren za varovanje programa z upoštevanjem pravil, pregledov in stalnih ocenjevanj. Član ekipe DFT je odgovoren za ugotavljanje in ocenjevanje, kateri digitalni dokazi so povezani s preiskovanim kriminalnim dejanjem. V primeru preiskave otroške pornografije član ekipe DFT oceni, da je potrebna nadaljnja analiza, če se na računalniku nahajajo prepovedane slike. Član ekipe DFT tako lahko navede, da je opazil



Slika 1: Vloga članov digitalne triaže na terenu DFT in tehnološke enote za kriminal TCU.



Slika 2: Model digitalne triaže na terenu

več prepovedanih slik, vendar ni usposobljen za pridobivanje podrobnejših podatkov, kot so viri in lokacija slik. Te podatke lahko pridobi le digitalni forenzik. Slika 1 prikazuje enostaven pregled nalog članov ekip DFT in TCU.

3.1 Model digitalne triaže na terenu

Predlagan model digitalne triaže na terenu je sestavljen iz štirih faz, ki ohlapno temeljijo na modelu CFFTPM opisanem v čalnku [12], vendar je zasnovan s člani ekip DFT in ne le z digitalnimi forenziki, kot prikazuje slika 2.

3.1.1 Načrtovanje - Kako bo preiskava obravnavana?

V začetnih fazah preiskave član ekipe DFT nudi pomoč preiskovalcu pri delu z digitalnimi dokazi. Ko preiskava napreduje proti izvršitvi naloga za preiskavo, član ekipe DFT zagotavlja dodatno pomoč pri podrobnostih iskanja. S tem mora vključevati oceno tveganja s tematikami, kot so:

- Ali gre za časovno kritične naprave, ki ne smejo biti izpostavljene izpadom (ISO 27037,2012)?
- Ali je v pristojnosti DFT (angl. comfort zone)?
- Kakšne so sposobnosti osumljenca?
- Katera vrsta kriminalnega dejanja je preiskovana?

Ker je član ekipe DFT del preiskave že od samega začetka, je posledično seznanjen z vsemi informacijami o primeru.

3.1.2 Ocena - kako bodo alocirani ustrezni artefakti

V povezavi s preiskovalno ekipo, član ekipe DFT identificira digitalne dokaze na kraju zločina in jih ustrezno obdela. Vsak kos digitalnega dokaza priotizira glede na verjetnost, da vsebuje relevantne artefakte. Preprost primer je računalnik, ki pripada znanemu spolnemu prestopniku, bi imel večjo prioriteto kot računalnik sostanovalca brez policijske kartoteke. Podobno kot socialna analiza, kjer naredimo profil osumljenca in oškodovanca, vendar razširimo analizo na vse prebivalce preiskovane lokacije, kot je opisano v članku [5]. Pred nadaljnjo oceno mora član ekipe DFT ponovno opraviti analizo tveganja. Član ekipe DFT mora vedno delovati v svojih okvirih in v obsegu orodja. Ko so digitalni dokazi razvrščeni po prioriteti, jih lahko član ekipe DFT oceni z uporabo orodij, odobrenih s strani TCU.

Običajno se ena izmed prvih ocen poda prav za osebni računalnik, zato lahko orodja, odobrena s strani TCU, ustvarijo seznam priloženih elementov. Te informacije podajajo dodaten seznam iskanih digitalnih dokazov. Pripadajoči elementi potem ravno tako pridobijo visoko prioriteto.

3.1.3 Poročanje - dokument dela in alocirani artefakti

Po zaključku ocene digitalnih dokazov, član ekipe DFT napiše poročilo o opazovanju. Na podlagi orodja, odobrenega s strani TCU, lahko izvlečki artefaktov vsebujejo šum, ki ga predstavljajo številni pridobljeni dokumenti in slike. Član ekipe DFT potem pregleda surove datoteke in preiskovalca usmeri k pomembnejšim artefaktom. V primeru preiskave kraje identitete so ponarejeni osebni dokumenti, fotografije za osebne dokumente in podobni artefakti poudarjeni, vendar imata preiskovalec in član ekipe DFT še vedno pregled nad vsemi fotografijami. Poročilo o opazovanju temelji na preprostem opazovanju pridobljenih artefaktov in ni forenzično poročilo. Član ekipe DFT tako ne podaja mnenja, ki bi temeljilo na artefaktih. Član ekipe DFT priloži tudi vse opombe in iskanja, ki jih je opravil s pomočjo orodja, odobrenega s strani TCU.

3.1.4 Prag - kaj je treba storiti z digitalnimi dokazi?

Član ekipe DFT in preiskovalec ugotovita, ali opazovani artefaktov ustrezajo pragu (angl. treshold) za nadaljnjo analizo s strani TCU. Ta faza temelji na moči orodja, odobrenega s strani TCU, in usposobljenosti člana ekipe DFT za izvedbo natančne ocene.

3.1.5 Prednosti modela DFT

Model DFT zagotavlja povečanje operativne učinkovitosti preiskave z zagotavljanjem informacij preiskovalcem, ko jih najbolj potrebujejo. Te informacije lahko vodijo do identificiranja nadaljnjih možnosti preiskave ali pa predlagajo artefakte, ki so predstavljeni osumljencu med intervjujem.

Z oceno digitalnih dokazov, ki je pridobljena v začetnih fazah preiskave, se tehnološki enoti za kriminal TCU posreduje le digitalne dokaze, ki so relevantni oz. dosegajo prag relevantnosti. Tako zmanjšamo število posredovanih digitalnih dokazov ter posledično zmanjšamo čakalno vrsto in zaostanek. Preiskovalec ima ravno tako možnost deliti poročilo o opazovanju s tožilcem, nato pa je lahko sprejeta odločitev, da je

edini potrebni digitalni dokaz specifičen artefakt, npr. dokument ali elektronska pošta. Digitalni forenzik TCU mora potem pripraviti le poročilo o specifičnem artefaktu. S tem ponovno zmanjšamo delo digitalnega forenzika, povečamo njegovo učinkovitost in zmanjšamo zaostanek.

3.1.6 Tveganje

Vodenje, izobraževanje in orodja, podprta s strani TCU morajo zmanjševati tveganja, povezana z modelom DFT. Prvo tveganje je vedno izključitev digitalnih dokazov, ki so pomembni za preiskavo. O tem je bila narejena raziskava, v kateri so ugotovili, da napreden predogled zmanjša število digitalnih dokazov, ki jih je potrebno v celoti analizirati s strani digitalnih forenzikov, hkrati pa ne izključi nobenega artefakta povezanega s kaznivim dejanjem, kot je opisano v članku [5].

Prav tako obstaja tveganje, kadar ni uporabljen model DFT. Tu se mora digitalni forenzik odločiti o zadostni natačnosti pregleda oziroma ali so zadostili minimalnim zahtevam analize opisanim v članku [8]. Brez predznanja o primeru digitalni forenzik nima jasne slike kaj iskati, kje iskati in kdaj prenehati z iskanjem, kot je opisano v članku [9]. Primer je odločitev o fotografijah, pridobljenih iz digitalne naprave, povezane s preiskavo goljufije. Digitalni forenzik mora pregledati fotografije in odločiti, katere so povezane z goljufijo. Ko jih bo zbral dovolj, ne bo poročal o ostalih družinskih fotografijah, ki jih je našel na napravi. Član ekipe DFT, ki se sooči s pregledovanjem enakih fotografij v začetku postopka, lahko na kakšni fotografiji prepozna osumljenca na srečanju z znanim pralcem denarja. Te informacije prinesejo dodatno možnost preiskave, ki bi lahko bila izgubljena, če bi se analiza zgodila brez prisotnosti preiskovalne ekipe.

Med usposabljanjem je član ekipe DFT opremljen s smernicami, ki ga vodijo v primeru odsotnosti dokazov. Če član ekipe DFT ne najde relevantnih artefaktov, je usposobljen, da pretehta verjetnost šifriranja. Pri tem mu je zagotovljena pomoč z orodjem, odobrenim s strani TCU, ki prepozna popolno šifriranje diska ali nameščenih programov. Član ekipe DFT mora ravno tako opredeliti prioriteto digitalnim dokazom in izvesti socialno analizo. Če se preiskovalcu in članu ekipe DFT zdi, da določen dokaz vsebuje relevantne artefakte, ga nato posredujeta nadrejenemu TCU.

4. IMPLEMENTACIJA

Prvo verzijo modela DFT je pred šestimi leti implementiral nadrejeni TCU, ki ga je sestavljajo 25 članov, od tega 20 digitalnih forenzikov. Nadrejeni TCU je bil odgovoren za približno 8500 zaposlenih v zveznih, provincialnih in občinskih okrajih, ki jih je sestavljalo 127 policijskih postaj, velikih od dveh do 800 članov. Območje, ki so ga pokrivali, je bilo veliko približno 945000 kvadratnih kilometrov.

Model DFT je bil razdeljen v dve področji:

- Digitalna računalniška triaža na terenu DCFT (angl. Digital Computer Field Triage)
- Digitalna mobilna triaža na terenu DMFT (angl. Digital Mobile Field Triage)

4.1 Orodje, odobreno s strani TCU

Začetna strategija je bila, da bi člani ekip DFT na področju DCFT prekinili proces zagona računalnika in podali oceno digitalnih dokazov v forenzično primerno okolje. Zaradi velikega števila članov ekip DFT, ki jih je bilo treba izobraziti, razvoj komercialno dostopnega orodja ni bil izvedljiv zaradi s tem povezanih stroškov. Mnogo orodij, tako komercialnih kot tudi odprtokodnih, je bilo namenjenih digitalnim forenzikom in so bila za osebje, ki ni specializirano za delo z digitalnimi dokazi, prezapletena za uporabo ali pa so ponujala preveč funkcionalnosti. Sprejeta je bila odločitev za razvoj namenskega "boot" diska z operacijskim sistemom Linux, ki bi ga načrtoval, implementiral in podpiral TCU. Ta disk omogoča uporabo preprostega tekstovnega menija za interakcijo s članom ekip DFT in podpira iskanje po različnih vrstah zločina.

Ena izmed prednosti namensko razvitega orodja je možnost prilagajanja in dodajanja funkcionalnosti. Primer tega je bila potreba po novi funkcionalnosti, ki omogoča ekstrakcijo številk kreditnih kartic, preverjanje njihove veljavnosti z Luhnovim algoritmom in razvrščanje po pripadajočih bankah. S pomočjo nove funkcionalnosti je lahko preiskovalec prevare iz digitalnih dokazov pridobil številke potencialno ogroženih kreditnih kartic in o tem obvestil primerne finančne institucije. Analiza, ki bi prej trajala več mesecev, je bila končana v nekaj urah, s tem so bile nadaljnje finančne izgube preprečene.

Program DFT se je s časoma razširil tudi na mobilne naprave, zato je bilo potrebno določiti najboljše orodje za program DMFT. Glavna zahteva za to orodje je bila, da mora podpirati večino mobilnih naprav na tržišču. V ta namen je bilo odobreno komercialno orodje.

Za programa DMFT in DCFT je treba periodično preverjati programsko in strojno opremo, da ta še vedno najbolje zadovolji potrebe DFT.

4.2 Izobraževanje

Za implementacijo modela DFT je izbira kandidatov kritičnega pomena. Izbrani kandidati bodo postali ambasadorji modela DFT, uspeh implementacije pa je odvisen samo od teh kandidatov. Za zagotovitev primernosti kandidatov se preveri njihovo znanje z uporabo vprašalnika o preiskovalnih in računalniških izkušnjah. Merila za kandidatovo računalniško znanje niso stroga, vseeno pa potrebujejo osnovno znanje o delovanju računalnikov.

Ker se na razpis vedno prijavi več kandidatov, kot jih je potrebnih, se naredi tudi drugi izbor, ki temelji na potrebah TCU, kandidatovi geografski lokaciji in razpoložljivosti. Ker je potrebno pokriti velik teren, imajo prednost kandidati, ki lahko pokrivajo čim večja območja, saj želimo da ima vsak preiskovalec na razpolago člana ekipe DFT. Ker se na razpis pogosto prijavi več kandidatov iz iste policijske postaje, je potrebno prioritizirati tiste kandidate, ki so manj zadolženi. To pomeni, da imajo kandidati kot na primer detektivi, ki preiskujejo resnejše zločine in niso v odzivni ekipi ali niso zadolženi za patruljiranje prednost pred kandidati, ki so v odzivni ekipi ali so zadolženi za patruliranje.

4.3 Tečaj digitalne računalniške triaže na terenu (DCFT)

Tečaj traja pet dni in daje poudarek praktičnem znanju, saj želimo, da bi kandidati po končanem tečaju imeli čim več izkušenj z rokovanjem z dokazi.

Prvi dan se kandidatom predstavi njihovo vlogo v programu DFT ter kaj se od njih pričakuje. Predstavi se jim TCU ter njene zmožnosti, rokovanje z dokazi znotraj enote, tehnike preiskovanja in zakone, ki obvezujejo to področje dela. Kandidati morajo nato nastaviti nastavitve prenosnih računalnikov in namestiti forenzično orodje, ki ga je odobrila TCU. Izvedba nastavitev, namestitve in splošna spretnost z računalnikom nudi dodatni vpogled v znanje kandidatov.

Drugi dan se kandidati naučijo kako prekiniti zagonsko zaporedje računalnika in se bolje spoznajo s forenzičnim ogrodjem. Največji poudarek je na prekinitvi zagonskega zaporedja in vsilitvi zagona forenzičnega orodja, saj je ta del za kandidate eden izmed bolj zahtevnih. Nadaljuje se z obrazložitvijo iskalnih funkcionalnosti forenzičnega ogrodja in nato tudi praktično uporabo teh na učnih primerih dokazov, ki so podani na ključku USB. Za vsak učni primer mora kandidat ponovno zagnati računalnik in vaditi celoten proces (prekinjanje zagonskega zaporedja in iskanje dokazov). Ta dan je še vedno voden s strani inštruktorja, hkrati pa kandidati tudi uporabijo pridobljeno znanje na praktičnih primerih. Za konec drugega dneva se kandidatom predstavi poročilo analize in opredeli kaj naj bi poročilo vsebovalo.

Tretji in četrti dan je več poudarka na praktičnem pristopu. Kandidate ne vodi več inštruktor, tako da lahko vsak kandidat dela s svojim tempom. Kandidati dobijo štiri različne scenarije osnovane preiskavah pri katerih bodo morali sodelovati na terenu. Vsak scenarij poda prenosni računalnik in definira platformo ter operacijski sistem z artefakti, ki so nastali pri zločinu. Od kandidatov je pričakovano, da sledijo pravilom in proceduram s katerimi so bili seznanjeni ter da zapišejo ustrezno poročilo analize kateremu priložijo potrebne datoteke in podatke. Kandidatu se po koncu oddana poročila pregleda in poda povratno informacijo o pravilnosti ter skladnosti.

Peti dan se kandidate testira, da se preveri, če so primerni, da postanejo člani ekip DCFT. Kandidatom se poda testni scenarij, ki ga morajo rešiti brez pomoči inštruktorja. Testni scenarij je podoben štirim scenarijem na katerih so kandidati vadili tretji in četrti dan in vsebuje podobne artefakte. Po končanem testnem scenariju morajo kandidati rešiti še pisni izpit, ki vključuje snov obravnavano na tečaju. Inštruktorji nato teste pregledajo in jih ocenijo. Samo kandidati, ki dosežejo dovolj dober rezultat lahko postanejo certificirani člani ekip DCFT.

4.4 Tečaj digitalne mobilne triaža na terenu (DMFT)

Tečaj traja štiri dni in tudi ta daje poudarek na praktičnem znanju.

Tako kot pri tečaju DCFT je prvi dan namenjen predavanjem, ki kandidate seznani z programom DFT. Pri tečaju DMFT kandidati vadijo na komercialnem orodju vendar so vseeno seznanjeni z inštalacijskim postopkom in uporabo orodja.

Drugi in tretji dan vključuje vaje na mobilnih napravah treh glavnih mobilnih platformah (iOS, Android in Blackberry), ki od kandidata zahtevajo, da iz naprav pridobi podatke. Vsak kandidat dobi tudi forenzično orodje odobreno s strani TCU, ki ga uporabi za reševanje vaj. Na forenzičnem ogrodju in njegovo uporabo na pridobljenih podatkih je večji poudarek tretji dan.

Zadnji dan se tako kot pri tečaju DCFT preveri znanje kandidatov. Kandidati dobijo testne scenarij, ki ga morajo rešiti individualno ter test z vprašanji, na katera morajo odgovoriti. Na koncu tečaja inštruktorji glede na rezultate določijo kandidate, ki so primerni, da postanejo člani ekip DMFT.

4.5 Nadaljevanje izobraževanja

Tako kot pri vseh področjih digitalne forenzike, se tehnike in digitalni dokazi konstanto spreminjajo. Zato je potrebno informacije iz TCU posredovati članom ekip DCFT in DMFT. To smo dosegli z forumom, ki je bil postavljen na namenskem strežniku in je dostopen zgolj članom ekipe DFT. Na forumu so naložene zadnje informacije in poučni video posnetki.

Problem je tudi, ker člani ekip DFT s časom pozabijo znanje pridobljeno na tečaju, hkrati pa jim lahko pade samozavest zaradi neuporabe znanja. Za zagotavljanje kvalificiranosti članov ekip DFT skozi čas, morajo ti nekajkrat letno opraviti test znanja, da lahko obdržijo svoj certifikat. Test znanja morajo opraviti iz postopkov, ki jih že nekaj časa niso uporabili za forenzično analizo. Člani ekip DFT test pridobijo s strani pristojne TCU. Test sestoji iz scenarija, ki je osnovan na dejanskem primeru iz prakse in omogoči članom ekip DFT, da scenarij rešijo in s tem osvežijo svoje znanje.

4.6 Vodenje programa

Dobro vodenje programa je ključnega pomena za uspeh modela DFT. Na začetku sta program vodila dva nadrejena analitika, vendar sta bila preobremenjena, ko se je povpraševanje po podpori povečalo. Njuno delo je prevzel drug analitik (koordinator ekipe DFT), ki svoj čas nameni izključno za vodenje programa DFT.

Dolžnost koordinatorja ekipe DFT je, da nadzoruje vse člane ekipe DFT in zagotavlja, da kvaliteta njihovega dela z digitalnimi dokazi zadostuje postavljenim standardom. V praksi mora koordinator ekipe DFT pregledati vsa poročila analiz, da so ta točna in kvalitetna, hkrati pa mora identificirati pomanjkljivosti izobraževanja. Za transparentno vodenje mora koordinator ekipe DFT slediti programu, ki definira, kaj je pričakovano od članov ekipe DFT. Program definira minimalno število izvedenih analiz na leto, definira, da se lahko uporabljajo samo orodja in oprema, odobrena s strani TCU, in določa posledice ob kršenju pravil.

Vloga koordinatorja ekipe DFT je zelo pomembna, saj člani ekipe DFT ne smejo svojega dela opravljati brez nadzora in podpore TCU. Model DFT ni zamenjava za celotno analizo ampak je samo del strategije za rokovanje z digitalnimi dokazi. Zato je komunikacija med TCU in DFT ključnega pomena za uspeh strategije. Prav tako model DFT ne sme

Tabela 1: Lokacije članov ekip DFT v sedežu (S) in okrožjih (O1-O4).

Tip DFT	S	O1	O2	O3	O4	Skupaj
Člani DCFT	15	46	21	22	14	118
- Podprte policijske postaje	4	13	15	22	10	54
Člani DMFT	13	45	11	14	13	96
- Podprte policijske postaje	5	9	5	6	6	31

zamenjati TCU v manjših oddelkih, ki nimajo potrebe po svojem TCU. V takšnih primerih potrebujejo manjši oddelki podporo nadrejene TCU.

Za pomoč pri vodenju programov DFT je bil postavljen namenski strežnik (isti strežnik, kot strežnik namenjen nadaljnjemu izobraževanju) za člane ekipe DFT. Na njem se hranijo tako zadnje različice orodij, ki so bila odobrena s strani TCU kot starejše različice orodij. Poleg omenjenih funkcionalnosti je funkcionalnost strežnika tudi, da članom ekip DFT priskrbi datotečne številke DFT, ki so unikatne za vsako preiskavo. Te številke člani ekipe DFT uporabljajo za analizo medijev. V primeru, ko več članov ekipe DFT analizira isto datoteko mora vsak izmed njih pridobiti svojo datotečno številko DFT. V primeru, ko pa ena datoteka vsebuje več digitalnih dokazov pa član ekipe DFT uporabi samo eno datotečno številko DFT za delo na tej datoteki.

Datotečne številke DFT so nato zabeležene skupaj z imeni članov ekipe DFT. To omogoča koordinatorju ekipe DFT vpogled v število obdelanih datotek na posameznega člana ekipe DFT.

4.7 Rezultati

Glavni cilji:

1. Povečanje učinkovitosti preiskave s časovno učinkovitim pridobivanjem artefaktov iz digitalnih dokazov.
2. Zmanjšati zalogo datotek, ki jih morajo analizirati digitalni forenziki

Ena izmed metrik za merjenje učinkovitosti preiskave je število članov ekipe DFT, ki so na voljo preiskovalcem ter število zadolženih članov ekipe DFT. Vsi statistični podatki v tem podpoglavju so rezultat implementacije programa DFT od leta 2009.

1 prikazuje število usposobljenih članov ekip DFT od leta 2009 razdeljenih na DCFT in DMFT. Okrožje 1 ima največjo koncentracijo prebivalstva in posledično so policijske postaje v tem okrožju največje, kar je opazno na številu članov ekip DFT v tem okrožju. Člani ekip DFT so po regijah konsistentno razporejeni in dostopni preiskovalcem. S povečanjem dostopnosti je potencialno tudi povečana učinkovitost preiskav

Od ustanovitve programa DFT se je povečalo število preiskav kjer je osumljenec priznal krivdo med preiskavo ekipe DFT in posredovanjem dokazov v TCU. Pogovor s preiskovalci je pokazal, da je tožilčev in branilčev vpogled v poročilo analize delno vodil v priznanje krivde. Tožilci in branilci so komentirali, da je vpogled v artefakte povezane s kaznivim dejanjem pomagal pri njihovi odločitvi.

Tabela 2: Procesiranje datotek digitalne triaže

Leto	Dat.	člani DFCT	člani DMFT	dat. TCU
2006	0	0	0	345
2007	0	0	0	435
2008	0	0	0	526
2009	26	9	0	522
2010	73	0	0	480
2011	376	53	0	468
2012	265	81	0	476
2013	260	104	24	422
2014	409	118	84	329
2015	469	118	96	137

Preiskovalci so se nasploh strinjali, da dobivajo relevantne informacije od članov ekip DFT v kratkem času. Pregled datotek DFT posredovanih nadrejeni TCU to tudi potrjuje. Sam cilj bi bil tudi v nasprotnem primer vseeno dosežen, saj so preiskovalci mnenja, da so informacije na voljo, ko jih rabijo in na njih ne čakajo.

Drugi cilj je bil zmanjševanje zaostanka nadrejene TCU, s pomočjo modela DFT. Trenutni zaostaneg nadrejene TCU je 58 datotek od katerih je bilo 30 datotek analiziranih s strani članov ekip DFT. Od 30 datotek, ki so bile analizirane s strani članov ekip DFT se je število podatkov posredovanih TCU zmanjšalo za 75%, kar pomeni, da morajo forenzični analiki obdelati manj podatkov in posledično za eno datoteko porabijo manj časa. To se pozna na večjem številu datotek obdelanih v enem letu, ki posledično zmanjšuje zaostanek.

2 prikazuje korelacijo med povečevanjem števila članov ekip DFT in zmanjševanjem števila datotek posredovanih nadrejeni TCU. Iz tabele lahko razberemo, da je v treh letih pred prihodom modela DFT, število datotek posredovanih v nadrejeno TCU konstantno naraščalo. Leta 2009 se je odvil prvi tečaj, vendar odsotnost naraščaja ne moremo pripisati programu DFT. Razlog za odsotnost naraščaja je, da so bili v nadrejenem TCU spremenjeni postopki izbiranja datotek za analizo in postali bolj selektivni. Drugi razlog ji bil zaostanek. Ta je bil namreč tako velik, da je bil čas za analizo tako dolg, da preiskovalci niso želeli pošiljati digitalnih dokazov v TCU. Datoteke, ki niso bile posredovane v TCU so bile samo del preiskave pri kateri so dokazi iz drugih področij privedli do obsodbe. Leta 2011 se je tečaj DCFT okrepil zaradi povečane ozaveščenosti in dostopnosti programa DFT. Od te točke naprej lahko vidimo velik naraščaj v številu datotek obdelanih s strani članov ekip DFT. Z čedalje večjim številom usposobljenih članov ekip DFT se je opazno zmanjšalo število datotek posredovanih v TCU.

Z zmanjševanjem števila posredovanih datotek v TCU se pomanjšuje tudi zaostanek. Tu ni bil cilj popolnoma izničiti zaostanek ampak ga zmanjšati na obvladljiv nivo.

4.8 Pregled

Dostop do kvalificiranega osebja ekip DFT je povečal količino sredstev, ki so na voljo nadrejeni TCU, kar posledično izboljšuje učinkovitost celotnega programa. Tekom implementacije programa se je obdelava digitalnih dokazov preselila iz kraja zločina v pisarno iz katere deluje član ekipe DFT. Kljub temu je član ekipe DFT pri nekaterih preiskavah še vedno prisoten, da pomaga pri identifikaciji potencialnih dokazov in njihovem zajemu. Prednost obdelave dokazov v pisarni je manjša obremenjenost in časovna stiska, saj član ekipe DFT dela v svojem okolju. Prav tako je obdelava na sceni pogosto težko izvedljiva zaradi prostorske stiske. Kljub temu se še vedno sledi modelu DFT, le lokacija obdelave se je spremenila. Ena izmed pomanjkljivosti tega pristopa je, da člani ekipe DFT ne morejo pomagati preiskovalni ekipi pri identifikaciji potencialnih virov digitalnih dokazov. Smiselno bi bilo, da bi člani ekipe DFT naredili kratek pregled medijev na sceni z namenom, da identificirajo potencialne vire, katere bi nato obdelali na lokalni policijski postaji.

Za zagotavljanje integritete programa je potrebno redno ovrednotiti vse aspekte programa in zagotoviti, da so ti znotraj meja, ki jih opredeljuje program. Koordinator ekipe DFT bi lahko ovrednotil kvaliteto tako, da bi naključno izbral in analiziral preiskavo, ki je uporabljala model DFT. S tem bi posledično tudi bolje ovrednotili orodje TCU in povečali kredibilnost programa. Prav tako je ovrednoten tudi vsak tečaj, da ta ohrani svojo vrednost in kandidatom priskrbi najkakovostnejše izobraževanje. Zelo pomembno je, da kandidati naberejo izkušnje v varnem in kontroliranem okolju s pomočjo scenarijev, hkrati pa je prednost tudi to, da dobijo takojšen odziv. Končen test preiskusi kandidate, njihovo znanje in sposobnosti v bolj stresnem okolju. Ena izmed vpeljanih sprememb je, da sedaj tečaj vodita dva inštruktorja TCU in izkušen član ekipe DFT. Član ekipe DFT poda kandidatom vpogled v delo ekipe DFT ter predstavi zahteve tega položaja. To je zelo pomembno, saj bodo podobno pričakovano tudi od novih kandidatov za delo v ekipi DFT. Poleg tečaja je mogoče izboljšati tudi metriko uporabljeno za izbor kandidatov ali izbrati boljšo metriko, ki bi dajala boljši vpogled v kandidatovo znanje.

Orodje odobreno s strani TCU deluje sprejemljivo za program DCFT in zadostuje postavljenim ciljem, vendar pa je problem orodja, da je namensko, kar pomeni, da je posledično deležno minimalnih nadgradenj in omejene podpore. Da bi se izognili stroškom komercialnega orodja, ki bi omejilo število članov ekip DFT zaradi omejenih virov, je potrebno v program vključiti tudi akademske inštitucije. Z njihovo podporo bi bila lahko orodja deležna zadovoljivega testiranja, ki bilo znanstveno podprto in posledično povečalo kredibilnost in zaupanje v orodje. Preiskovalci in forenzični analitiki imajo obširno znanje o potrebah preiskave in postopkih, ki pripeljejo do obsodbe. Akademske inštitucije pa bi lahko prispevale znanstveno podporo, uveljavljene postopke testiranja in računalniške specialiste.

S povečevanjem ozaveščenosti, čedalje večji izpostavi in širjenjem modela DFT, želijo tudi drugi organi pregona izobraziti kader in ustanoviti svoj program DFT.

4.9 Ocena ustreznosti modela

Model DFT sledi principom procesa digitalne znanstvene raziskave [10]. Posledično moramo pri oceni ustreznosti nasloviti naslednje pogoje:

- Je model konsistenten z ostalimi modeli na področju digitalne forenzike?
- Je model uporaben?
- Ali model narekuje postopke rokovanja z digitalnimi dokazi?

Osnova tega modela je bila zastavljena v članku, ki piše o računalniškem forenzičnem modelu za digitalno triažo na terenu [11]. Poleg tega model DFT ne zamenja procedure izvedene v laboratoriju kot na primer metodologijo zastavljeno s strani ameriškega ministrstva za pravosodje [1]. Glavno načelo, ki se ga moramo držati je da opisane in uporabljane procedure modela DFT ne ogrozijo celovitosti dokazov. S tem zagotovimo, da je model DFT konsistenten z modeli, ki se že uporabljajo v digitalni forenziki.

Model lahko smatramo kot uporabnega, če ga je mogoče uporabiti v realnem scenariju pri katerem zadosti želenim ciljem. Do danes je bilo ustvarjenih 1878 datotek DFT, ki so bile ovrednotene. Na podlagi ovrednotenja se je nato določilo, če je nadaljnja analiza potrebna v TCU. Kljub temu, da niso vse datoteke popolne je iz njih razvidno, da se modelu sledi, da je razumljiv in da daje nespecializiranemu osebju napotke za rokovanje z digitalnimi dokazi. Model poda tudi informacije o postopkih, ki jim je potrebno slediti ter opredeli potencialno kritične dele. To naslovi zadnje dve zahtevi, ki pravita, da mora biti model uporaben in mora narekovati postopke rokovanja z digitalnimi dokazi.

4.10 Zaključek

Cilj tega članka je, da bi razbremenili forenzične analitike in del njihovega dela preložili na nespecializirano forenzično osebje. ISO 27037 to opredeljuje kot DEFR, ampak delo DEFR presega delo, ki ga lahko opravlja ekipa DFT. Glavna razlika je zmožnost zajemanja digitalnih dokazov. Lahko bi dodatno izobrazili člane ekip DFT, da bi bili sposobni tudi zajemati dokaze ampak trenutno po tem ni potrebe. To bi sicer prinašalo prednosti v poslovnem svetu, kjer je večina dokazov digitalnih in ostanejo na kraju preiskave. Pri kriminalistični preiskavi pa je večina digitalnih dokazov del zločina in je zajeta pod pooblastilom preiskovalnega naloga. Član ekipe DFT nato določi kateri dokazi so relevantni za nadaljnjo analizo, preiskovalec pa lahko nato določi, če se dokazi, ki so bili označeni kot ne relevantni vrnejo lastniku. V določenih primerih pa digitalni dokazi ne morejo biti vrnjeni, saj lahko vsebujejo podatke, ki bi bili lahko potencialno koristni za kriminalno združbo. Želja je tudi po tem, da bi člani ekip DFT lahko pridobili podatke iz delovnega pomnilnika (RAM). V zadnjih letih se je namreč povečala vrednost podatkov pridobljenih iz pomnilnika RAM (angl. Random Access Memory), hkrati pa je čedalje več orodij, ki to omogočajo. Pridobivanje podatkov iz pomnilnika RAM poteka na delujoči napravi, zato je pomembno, da lahko člani ekip DFT samozavestno in zanesljivo pridobijo podatke. Tudi to vodi v koncept naprednejšega tečaja za člane ekip DFT.

Začetni cilj je bil vzpostaviti model digitalne triaže na terenu (DFT), ki bi pripomogel k učinkovitosti in zmanjšal zaostanek TCU. Tekom ocene ustreznosti in implementacije modela, se je izkazalo, da model dosega začetne cilje z poudarkom na učinkovitosti preiskav.

Na področju digitalne forenzike je še vedno veliko pomislekov glede prelaganje dela na nespecializirano osebje [4]. Rezultati tega članka postavijo metriko, ki potrjuje, da pomisleki o delovanju modela nimajo trdne podlage. Kljub temu je potrebno to področje podrobneje raziskati, predvsem zanesljivost uporabljenih orodij. V številnih sodnih oblasteh se število zaposlenih zmanjšuje, število digitalnih naprav pa stalno narašča. Potrebno je povečati število usposobljenega osebja z znajem o digitalni forenziki. Prav to pa omogoča večstopenjski odzivni mehanizem.

5. VIRI

[1] forensics_chart.pdf. https://www.justice.gov/sites/default/files/criminal-ccips/legacy/2015/03/26/forensics*chart.pdf*. (Dostopano: 05/10/2017).

[2] A. Author. Information technology e security techniques e guidelines for identification, collection, acquisition and preservation of digital evidence, 2012.

[3] N. L. Casey E, Ferraro M. Investigation delayed is justice denied: proposals for expediting forensic examinations of digital evidence. *J forensic Sci*, 54(6):1353–1364, 2009.

[4] J. I. James and P. Gladyshev. A survey of digital forensic investigator decision processes and measurement of decisions based on enhanced preview. *Digital Investigation*, 10(2):148–157, 2013.

[5] G. P. James JI. A survey of digital forensic investigator decision processes and measurement of decisions based on enhanced preview. *Digit Investig*, 10(2):148–157, 2013.

[6] G. P. James JI. Automated inference of past action instances in digital investigations. *Int J Inf Secur*, 14(3):249–261, 2015.

[7] K. G. Mislan RP, Casey E. The growing need for on-scene triage of mobile devices. *Digit Investig*, 6(3):112–124, 2010.

[8] P. MJ. Computer corner - examination backlogs, the management challenge. *U. S Dep Justice Drug Enforc Agency Microgram Bull*, 36(2):44–45, 2003.

[9] P. MM. Triage: a practical solution or admission of failure. *Digit Investig*, 10(2):87–88, 2013.

[10] K. Peffers, T. Tuunanen, C. E. Gengler, M. Rossi, W. Hui, V. Virtanen, and J. Bragge. The design science research process: a model for producing and presenting information systems research. In *Proceedings of the first international conference on design science research in information systems and technology (DESRIST 2006)*, pages 83–106. ME Sharpe, Inc., 2006.

[11] M. K. Rogers, J. Goldman, R. Mislan, T. Wedge, and S. Debrota. Computer forensics field triage process model. In *Proceedings of the conference on Digital Forensics, Security and Law*, page 27. Association of Digital Forensics, Security and Law, 2006.

[12] M. R. W. T. D. S. Rogers MK, Goldman J. Computer forensics field triage process model. *J Digital Forensics Secur Law*, 1(2):19–38, 2006.

Digital evidence, 'absence' of data and ambiguous patterns of reasoning

Amra Omanović
University of Ljubljana
Faculty of Computer and Information Science
1000 Ljubljana
amraomanovic1@hotmail.com

Amela Špica
University of Ljubljana
Faculty of Computer and Information Science
1000 Ljubljana
amela.spica@gmail.com

ABSTRACT

In this paper we discuss the use of digital data by the Swiss Federal Criminal Court in case of attempted homicide. This case is example of drawback for the defense, where the presentation of scientific evidences is partial. This paper consists of two parts, first is non-technical presentation of the topic, which means drawing parallels between the court's summing up of the case and flawed patterns of reasoning commonly seen in other forensic disciplines such as gunshut residues. Second part is a formal analysis of the case, where we are using probability and graphical probability models for scientific approach. In that part we will justify the claim that the partial presentation of digital evidence brings a risk of hiding vital information from the defense.

Keywords

Forensic interpretation, Digital traces and the law, Case discussion

1. INTRODUCTION AND CASE DESCRIPTION

The case which we are going to present in this paper is about the incident which happened to A.'s estranged wife B. She was walking in the city of Z.(Switzerland) one morning when she was hit by a hand grenade thrown at her and it left her injuries to the abdomen and left hand. B. had not seen who attacked her, but she suspected that A. did that, because he had been following her for some time. The police also seized several fragments of plastic, small metal balls and a lever, all identified as constituting elements of a model of hand grenade used by the former Yugoslavian army. When police asked A. where he was at the time of the crime he said that he was in Bosnia and Herzegovina. However, police found out that A. had crossed the Swiss border about 5 hours after this attack.

There are two scientific evidences in this case: DNA-profile established from the surface of the hand grenade and technical report about analyses of telecommunication and navigation data carried out on A.'s mobile devices.

DNA was found to correspond to the DNA-profile of A. but paper does not deal with this issue. Beside that, second evidence shows that no signals of A.'s mobile devices were detected on the crime scene when this attack happened. [1]

In the remainder of this paper, we will take a closer look at the way in which the digital evidence mentioned above was used by the Swiss Federal Criminal Court to justify prolonging A.'s pretrial detention, based on the judgement of the court as it was published.

In section 2 we will present Case Analysis, in section 3 we will take a look at Principles of Scientific Interpretation and in the section 4 we will show Extended Analysis of the case using Bayesian network[1].

2. CASE ANALYSIS

The Swiss Federal Court summed up the digital evidence in these sentences:

1. "At the time of the crime, no mobile device belonging to A. could be located at the scene."

2. "This investigative result does not necessarily exclude that the complainant [Mr.A.] could have been present at the crime scene at the time when the crime was committed."

This conclusion raises questions: "What - if anything - does the digital evidence say with respect to A.'s case?" and "If digital evidence does not falsify the prosecution's case, does it have any effect on case?". These questions will be approached from general perspective and scientific criteria of evaluation. Attention will be on potentially ambiguous patterns of reasoning and drawing parallels to drawbacks commonly seen in other forensic disciplines. Outcomes will be contrasted in section Principles of scientific interpretation, with a formal analysis of the case using elements of probability and graph theory.

[1]A Bayesian network is a probabilistic graphical model that represents a set of random variables and their conditional dependencies via a directed acyclic graph (DAG).

2.1 Legal considerations

Now, court is examining the question of whether A.'s pretrial detention should be prolonged. At this stage, court considers the following conditions:

1. there is a strong suspicion that the suspect committed a felony
2. there is a serious risk of flight, of reoffending or of tampering with the evidence
3. the duration of pretrial detention is proportionate with regards to the sentence that will likely be imposed by the court in case of conviction

The court held that the digital evidence did not falsify the hypothesis presented by the prosecution, that it is actually neutral. We will see that this perspective is not optimal.

2.2 Consistent with

The court finds the evidence to be 'consistent with' the complainant being on the crime scene ('could have been present at the crime scene'). Sentences with 'could' or 'might' are little informative, since they only express probability different from zero, so this only shows one's belief in the truth, which is different in case of propositions when we know if something is true or false. Besides the confusion between possibilities and propositions, terminology 'consistent with' is defective, and this example illustrates that:
"Buying a ticket is consistent with winning the National Lottery, but it does not make winning very likely. Buying a ticket is also consistent with not winning the National Lottery, and this second outcome is very much more likely than the first, though both are equally 'consistent with' the premiss (buying a ticket)."

2.3 Convenience conclusions

Further, expressing one's belief with respect to only a single proposition is akin to framing conclusions around a preferred proposition, that is at one's convenience, whatever the evidence actually available. In this case it means if signal of mobile device was detected, it would have meant that owner of mobile device was at crime scene. On the other hand, if there was no detected signal, one would still conclude that such a "negative finding" doesn't rule out the proposition according to which the owner of the mobile device was on the crime scene. Yet, one could also retain conclusions with respect to proposition of not being on the crime scene, whatever the evidence being found.

2.4 Parallels to conclusion patterns seen with other forensic evidence types

We can draw parallel with example of gunshot residue (GSR) particles. When a catridge is fired by a firearm, discharge residues are produced and their analysis is of interest from a forensic point of view because they deposit in the vicinity of the discharging firearm, in particular hands, face and clothing of shooters and bystanders. Considering absence of detected particles scientists can state:
"The absence of gunshot residue on a person's hands does not eliminate that individual from having discharger firearm."
They may also argue that:
"If a subject has washed or cleaned the hands in some manner, the likelihood of detecting GSR decreases. The hands may have been covered with gloves or some other covering. For these reasons and many others, negative results obtained from an examination are not necessarily exculpatory."

2.5 Clarifying the nature of the report

Scientific and technical evidence is used in different ways and for distinct purposes at the various stages in the legal process, which is why it is important to clarify at which stage and in which process they operate. It is equally important to take into account the nature of the reports issued by scientists because the scope and purpose may vary according to different report categories. Guideline by the European Network of Forensic Science Institutes (ENFSI) distinguishes between three main categories.
First category regards technical reporting and it amounts reporting on observations in a descriptive way without elaborating any propositions.
Descriptive account is preliminary for *second category* of report: investigative reports. These reports explore explanations for particular observations and are useful at an early stage of investigation
Descriptive accounts are also a preliminary for a *third category* of reports: evaluative reports. They are used in more advanced stages of process, when potential source is available and comparative examinations with trace material have been conducted.
In this case, the digital evidence takes the form of what is referred to as a "technical evaluation of telecommunication and navigation data" and is seen as an "investigative result". While the investigative value of that information is clear, a trial is not about exploring potential explanations for the findings in an informal and deliberate way, but an instance where two sides oppose competing scenarios, and the purpose is to weigh the evidence against those two positions.
A conclusion that refers to only one proposition and that is limited to an expression of the kind 'cannot be excluded' is both incomplete and unbalanced.

3. PRINCIPLES OF SCIENTIFIC INTERPRETATION

Statistical evidence and probabilistic reasoning today play an important and expanding role in criminal investigations, prosecutions and trials, not least in relation to forensic scientific evidence (including DNA) produced by expert witnesses. There is a long history and ample recent experience of misunderstandings relating to statistical information and probabilities which have contributed towards serious miscarriages of justice. [2]
To overcome obscurity about how the transition between the observation and the statement is made we introduce question triad:

1. what is the probability of the evidence given the prosecution's case and the case circumstances?

2. what is the probability of the evidence given the defense's case and the case circumstances?

3. under which proposition are the findings probable, under the first or the second proposition?

With this we emphasize three principles of forensic interpretation:

- it clarifies that evaluation is conditioned by a framework of circumstances
- forensic results ought to be looked at from at least two competing viewpoints
- the scientist will avoid interfering with the role of the judicial decision maker

We are also using pre-assessment to add credibility to the scientist's evaluation with avoiding:

1. specifying potential results prior to performing any analyses
2. assessing the probative value for each potential finding
3. assigning probabilities with which the various results may be obtained under each of the competing propositions

The case assessment and interpretation model (CAI) for forensic case rests on certain foundational notions and fundamental principles, which were clarified and refined as the model was developed, taking appropriate account of criminal practitioners input and feedback.[3]
Specifically, CAI:

- clarifies the role of forensic expertise in criminal investigations, highlighting a vital distinction between investigative advice and evaluative opinions
- identifies the different forms of logical reasoning characteristic of expert assistance in its investigative and evaluative modes
- provides an illuminating taxonomy of the formulations currently routinely employed in forensic practice to report scientific findings, covering a spectrum ranging from hard scientific facts to evaluative expert opinions
- rests on a rigorous logical method for evaluating the results of forensic examinations probabilistically
- explains how the form in which evaluative opinion is expressed maps onto a hierarchy of issues, such that the probative value of the evidence may change according to the issue addressed
- enables discrete evaluations of particular scientific inquiries to be amalgamated into a single evaluative opinion, addressed to issues at activity level (the level which may, at least sometimes and in certain circumstances, provide greatest assistance to criminal investigators and fact-finders)

3.1 Application to the present case

We have two propositions of interest:

1. A. is the person who threw the hand grenade at the victim
2. A. was nowhere near the scene of the crime when the grenade was thrown

Let's use the following notation: H_p is the first proposition and H_d is the second proposition, I presents time and location of the crime, E means "signals of A.'s mobile devices were detected in the crime scene area during the time interval when the explosion occurred" and E' means "no such signals were detected". Pr denotes probability.
Information I is important and we need to take into account facts like where A. lives , who had access to his mobile device at the time of the attack and so on.
We are interested in finding these three questions:

- how probable is the finding E if A. was on the crime scene $Pr(E|H_p, I)$?
- how probable is the finding E if A. was nowhere near the crime scene $Pr(E|H_d, I)$?
- is the finding E more probable given H_p or given H_d?

Qualitative probability allows us to make distinctions:

1. if $Pr(E|H_p, I) > Pr(E|H_d, I)$ then evidence supports H_p over H_d
2. if $Pr(E|H_d, I) > Pr(E|H_p, I)$ then evidence supports H_d over H_p
3. if $Pr(E|H_p, I) = Pr(E|H_d, I)$ then evidence is neutral

Values of $Pr(E|H_p, I)$ and $Pr(E|H_d, I)$ determine values of $Pr(E'|H_p, I)$ and $Pr(E'|H_d, I)$ because they sum to 1. In the first case likelihood ratio[2] is greater than 1, in the second case it is smaller than 1 and in the third case it is equal to 1.
Now, let's go back to the court's summing up of the case. They are silent about the effect of observing no signal E'. If the court accepts that detecting no signals is more compatible with H_d than with H_p then it holds:
$Pr(E'|H_d, I)) > Pr(E'|H_p, I)$, which is in favour of H_d.
We should also be aware that A.'s device was present but not turned on and in that case the value of $Pr(E|H_p, I)$ is very low and this analysis works.

4. EXTENDED ANALYSIS OF THE CASE

In this section we are going to formulate and implement probabilistic reasoning schemes at more advanced levels of complexity using graphical probabilistic models such as Bayesian networks. It is based on Bayes' Theorem which calculates a posterior probability for the issue, conditioned

[2]ratio of probabilities $Pr(E|H_p, I)$ and $Pr(E|H_d, I))$

on the combined value of the prior probability and the likelihood ratio for the evidence. This posterior probability can then be treated as a new prior probability to which a further additional piece of evidence can be added, and a new posterior probability calculated (now taking account of the original prior probability and the likelihood ratios for both pieces of evidence). The process can be repeated over and over, finally resulting in a posterior probability conditioned on the entire corpus of evidence in the case. We need to render additional considerations in digital evidence interpretation explicit and to incorporate them into the reasoning process.

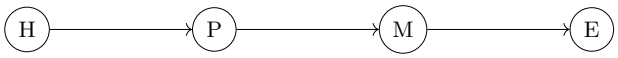


Figure 1: Bayesian network

Nodes in this Bayesian network on Figure 1. have the following meanings:

1. $H \longrightarrow$ A. is the offender
2. $P \longrightarrow$ A. was in relevant area at the relevant time
3. $M \longrightarrow$ A.'s mobile devices were in the relevant area at the relevant time
4. $E \longrightarrow$ Signals of A.'s mobile devices have been detected at the crime scene

Analysis of these nodes is:

- Node P: it is obvious that $Pr(P|H_p, I) = 1$, the offender must be present on the crime scene to commit this crime. For probability $Pr(P|H_d, I)$ we will assume the value of 0.001.
- Node M: let's say that $Pr(M|P, I) = 0.99$ and $Pr(M|P', I) = 0.001$ where P means that A. was present at the crime scene and P' means that A. was not present at the crime scene.
- Node E: $Pr(E|M, I) = 0.9$ and $Pr(E|M', I) = 0.0001$. We put low value for M' because it is implausible event to detect signal of a device that is not present.

Outcomes P,M and E are more probable given the conditioning events H_p, P and M rather than the events H_d, P' and M'.
A Bayesian network captures, respectively, structural and numeric probabilistic relationships among random variables in terms of a directed acyclic graph and conditional probability tables. Given observations about some random variables, we evaluate the Bayesian network to obtain the conditional probability distributions of random variables of interest. The evaluation process is also known as inference in Bayesian networks, and this active research field has seen a wide variety of approaches for computing exact and approximate probability distributions.[4]

We can formulate two conclusions:

1. we can compute the sign of qualitative influence between the variable E and H based on the signs of the qualitative influences that are associated with each arc of the trail connecting these two variables
2. we can compute the effect that observing E' has on the variable H

Let's see the second case: in the beginning all links have sign + and all nodes are initially set to 0. We put $-$ sign in the node E and compute with the product operator what to send to the node M. So, we are sending to the node M sign $-\bigotimes + = -$. Now, we need to update the value of M. We are doing that with the addition operator and we get a value of $0 \bigoplus - = -$ in M.
Qualitative belief propagation continues to the node representing the main propositions H where H_d is supported over H_p.
With this network we can see how intermediate variables are affected during inference about the main proposition H.

5. CONCLUSION

In this paper, we have discussed importance and feasibility of using formal methods of logical reasoning under uncertainty, in particular graphical probability models to help deal with digital evidence at trial. In the case study, the evidence reveals a subtle breach.
On the one hand, the formal analysis shows that the evidence regarding the absence of A.'s mobile devices signals ought to decrease the evaluator's belief in the prosecution's story.
On the other hand, the formal analysis shows us that this is just one side of the assessment. When decreasing one's belief in one proposition, then one must redistribute probability among the remaining alternatives. In this case, there was only one alternative, the one from defense. Considering scientific evidence only from the prosecution's position hinders vital evidence for the defense to be brought to the decision maker's attention.
The evaluation that the Court did of the digital evidence can be considered as incomplete, but it is entirely legitimate at this stage of the proceedings since standard of proof is closer to preponderance of evidence than proof beyond a reasonable doubt. Evaluating the evidence in this way at the stage of adjudication would support the rights of the defense and be damaging to the search for the truth.

6. REFERENCES

[1] Alex Biedermann, Joelle Vuille. *Digital evidence, absence of data and ambiguous patterns of reasoning.* The proceedings of the Third Annual DFRWS 2016 Europe.
[2] Aitken CGG, Roberts P, Jackson G. *Fundamentals of probability and statistical evidence in criminal proceedings (Practitioner guide No. 1), guidance for judges, lawyers, forensic scientists and expert witnesses.* Royal Statistical Society's Working Group on Statistics and the Law; 2010.
[3] Jackson G, Aitken CGG, Roberts P. *Case assessment*

and interpretation of expert evidence (Practitioner guide No. 4), guidance for judges, lawyers, forensic scientists and expert witnesses. Royal Statistical Society's Working Group on Statistics and the Law; 2013.
[4] Chaon-Lin Liu, Michael P. Wellman. *Bounding probabilistic relationship in Bayesian networks using qualitative influences: methods and applications*; 2003.

Del II
Bitcoin

Bitcoin: Celovit pregled decentralizirane digitalne valute

Rok Novosel
Univerza v Ljubljani
Fakulteta za Računalništvo in Informatiko
1000 Ljubljana
rn0450@student.uni-lj.si

Pavlin Poličar
Univerza v Ljubljani
Fakulteta za Računalništvo in Informatiko
1000 Ljubljana
pp1876@student.uni-lj.si

Benjamin Novak
Univerza v Ljubljani
Fakulteta za Računalništvo in Informatiko
1000 Ljubljana
bn5567@student.uni-lj.si

POVZETEK

Poleg ustvarjanja miljardo dolarjev vredne ekonomije je Bitcoin revolucioniral področje digitalnih valut in vplival na veliko podobnih področij. To je privabilo tudi veliko znanstvenega interesa. V tem članku začnemo s pregledom Bitcoin protokola in njegovih gradnikov. Nato predstavimo temeljne strukture in vpoglede v jedro Bitcoin protokola in njegovih aplikacij. Opišemo tudi glavne varnostne groženje in kako se Bitcoin z njimi spopada. Na koncu se dotaknemo tudi problematike zasebnosti.

Ključne besede

Bitcoin, digitalne valute, veriga blokov, transakcije, dvojno zapravljanje

1. UVOD

V prejšnjih desetletjih je bil Internet priča pojavu mnogih aplikacij, ki rešujejo probleme na porazdeljen (angl. *distributed*) način. Med primere takih aplikacij sodi anonimna komunikacija [1], PGP [2], Hashcash [3] in BitTorrent [4]. Praktično uporabne aplikacije so pogostokrat postale na voljo takoj, ko se je ideja za njih prvič pojavila. Izjema tega pravila pa je digitalni denar: že v 80ih letih prejšnjega stoletja je obstajala vizija digitalnega denarja, ampak je minilo več kot 25 let preden se je pojavila prava porazdeljena rešitev.

Zgodnji poskusi za izgradnjo digitalne valute, kot opisani v [5, 6], potrebujejo centralno avtoriteto - banko. Pristopi kot so b-money, RPOW in bit gold kasneje pridejo na idejo, da interpretirajo rešitev kriptografske uganke kot dokaz o opravljenem delu in smatrajo to kot nekaj vrednega. To primerjajo s kosom zlata ali kovanjem kovancev. Na tak način bi lahko vsak postal rudar digitalnega zlata, ampak še vedno potrebuje centralno avtoriteto, ki ohranja evidenco o lastništvu.

Da bi popolnoma eliminirali banko, mora biti evidenca o lastništvu kovancev prav tako porazdeljena. Osnovno tveganje pri digitalnih porazdeljenih valutah je možnost, da lahko dvakrat zapravimo iste kovance. Ker je izdelava digitalnih kopij trivialna, lahko nekdo izda dve paralelni transkaciji različnim prejemnikom. V centraliziranem scenariju banka lahko to zazna in prepreči poskus dvojnega zapravljanja. Preprečevanje tega v porazdeljenem okolju je daleč od trivialnega. Porazdelitev informacij in problem medsebojnega dogovora o konsistentnosti stanja je velik izziv, sploh, če so prisotni sebični in zlonamerni udeleženci. V osnovi je to problem Bizantinskih generalov [7]. Ta vpogled je omogočil, da se vpelje ideja kvoruma. Kvorum kot opisan v [8], dovoli možnost lažne informacije in zlonamernih udeležencev v porazdeljenem okolju. Vpeljejo tudi koncept glasovanja. Dokler je večina katerekoli podmnožice udeležencev pristnih (kvorum), bo izglasovana pravilna izbira. Vendar je tak pristop lahko podvržen Sybil [9] napadu: zlonamerni udeleženec vzpostavi večino ostalih udeležencev, ki vnašajo lažne informacije in spodkopajo glasovanje.

Te težave so bile premagane z Bitcoin protokolom [10], katerega je Satoshi Nakomoto objavil konec leta 2008. Do sedaj dejanska identiteta Nakomota ostaja neznana in obstajajo domneve, če je ime sploh resnično, psevdonim ali pa predstavlja skupino ljudi. Vemo pa, da Bitcoin pametno združuje desetletja raziskav in rešuje temeljne probleme na napreden in praktično izvedljiv način. Uporablja dokaz dela, da omeji število glasov na udeleženca in tako omogoča praktično porazdeljenost valute.

Bitcoin rudarji zbirajo transakcije v bloke in iščejo rešitve za dano uganko. Blok, ki vsebuje rešitev in transkacije je prenesen vsem ostalim udeležencem in porazdeljena evidenca (veriga blokov, angl. *block chain*) je posodobljena. Lastništvo kovancev lahko ugotovimo tako da preiščemo verigo blokov dokler ne najdemo željenega kovanca. Razcepi (angl. *fork*) verige blokov zaradi zlonamerne manipulacije ali zamud pri prenosih so rešeni tako, da se upošteva najdaljši trenutni razcep kot soglasje. Sybil napad in do neke mere tudi napad z dvojnim zapravljanjem kovancev preprečimo s tem, da dodajanje v verigo blokov povežemo z dokazom o delu. Dokaz o delu povzroči tudi stalen pritok novih kovancev kot nagrada in spodbuda za rudarje. Za vse to ne potrebujemo centralne avtoritete in tako demonstriramo praktično izvedljivost porazdeljene digitalne valute.

V tem članku bomo predstavili osnove Bitcoin protokola in predstavili nekaj splošnih konceptov kot sta dokaz o delu

in dvojno zapravljanje. Opišemo tudi varnostne posledice uporabe Bitcoina in zasnovo Bitcoin omrežja. Nato obravnavamo še razmerje med zasebnostjo in povezano transparentnostjo ter pristope, ki povečajo zasebnost uporabnikov.

2. BITCOIN PROTOKOL

V tem poglavju bomo predstavili jedro Bitcoin protokola kot je bil prvotno predstavljen v [10]. To bo nudilo podlago za poglobljeno diskusijo v nadaljnih poglavjih. Začeli bomo z abstraktnim in precej preprostim pogledom na digitalno valuto, ki jo nato iterativno izboljšujemo. Naše diskusije so usmerjene proti temeljem Bitcoin protokola in njegovi osrednji ideji: uporaba dokaza o delu za odstranitev banke ter porazdelitev in zavarovanje evidence o trenutnem stanju.

2.1 Centralizirane digitalne valute

Recimo, da bi Ana rada poslala kovanec Branku. Če hoče to storiti mora generirati digitalno podpisano pogodbo, ki pravi "Prenašam en konvanec Branku". V Bitcoin terminologiji bi se taki pogodbi reklo transakcija. Smatramo jo lahko kot podpisano pogodbo, katero lahko preverimo z Aninim javnim ključem. Ni pa popolnoma varna pred ponarejanjem, ker lahko isto transakcijo večkrat ponovimo. Če se pojavi duplikat pogodbe ne vemo ali Ana želi prevarati Branka ali pa če hoče popolnoma dobronamerno poslati še en kovanec.

Če želimo rešiti tak problem potrebujemo unikatno razpoznavne kovance. To dosežemo s serijskimi številkami, ki jih izdaja centralna banka. Banka tudi skrbi za vzdrževanje evidence o lastništvu ter preslikava med uporabniškimi računi in serijskimi številkami.

Pred prenosom kovanca Ana podpiše in naznani pogodbo v kateri piše: "Prenašam kovanec #1210 Branku". Branko preveri lastništvo kovanca #1210 pri banki. Če je transakcija veljavna in Branko potrdi transakcijo, banka posodobi svojo evidenco o lastništvu kovanca.

Ta preprosta in centralizirana valuta ponazarja osnovni načrt bančniškega modela. Bitcoin pa cilja na višje zastavljeno rešitev, kjer se odstrani centralna banka. V ta namen so potrebni mehanizmi, ki ustvarjajo kovance v porazdeljenem okolju in vzdržujejo evidenco o lastništvu na porazdeljen način. Ključni problem je doseganje konsenza na obstoječih kovancih glede njihovega lastništva brez centralne avtoritete in brez medsebojnega zaupanja med sodelujočimi.

2.2 Dokaz dela: porazdeljevanje valute

Bitcoin odpravi banko na pragmatičen način tako, da vsi udeleženci postanejo banka. To pomeni, da ima vsak udeleženec shranjeno evidenco o lastništvu, ki bi bila ponavadi shranjena v banki. Vlogo evidence v Bitcoinu prevzame veriga blokov.

Porazdeljeno hranjenje večih kopij verige blokov pa prinaša nove načine s katerimi lahko Ana goljufa. Ana lahko pošlje dve ločeni transakciji dvema ločenima prejemnikoma (recimo Branku in Cenetu), kjer prenese isti kovanec. Temu pravimo dvojno zapravljanje. Če bi Branko in Cene ločeno potrdila transakcijo, bi veriga blokov pristala v neskladju. Če pa Branko potrdi transakcijo pred Cenetom, lahko Cene prepozna poskus dvojnega zapravljanja.

Bitcoin rešuje ta problem tako, da pusti celotnemu omrežju preverjati legitimnost transakcij v upanju, da bodo ostali udeleženci prepoznali poskus dvojnega zapravljanja. Če in samo če se večina udeležencev strinja na obstoj in legitimnost transakcije jo lahko Branko sprejme. Vendar Ana lahko uprizori Sybil napad, kjer vzpostavi veliko število lažnih udeležencev, ki bi potrdile transakcijo (in tako postala večina), čeprav je bil kovanec dvakrat zapravljen.

Bitcoin protocol uporablja dokaz o delu, da prepreči Sybil napad. Preden lahko udeleženci potrdijo transakcijo in razširijo novico o njej morajo opraviti nekaj dela, da dokažejo svojo pristnost. Delo se sestoji iz kriptografske uganke, ki umetno poveča računsko ceno za potrditev transakcij. S tem je zmožnost potrjevanja transakcij odvisna od računske moči in ne od števila (potencialno lažnih) udeležencev. Osnovna predpostavka je, da je težje obvladovati večino računske moči kot pa večino udeležencev.

Nove Bitcoin transakcije so posredovane vsem udeležencem v omrežju. Če so veljavne, se jih doda v blok. Uganka uporabljena kot dokaz dela je izračun zgoščevalne funkcije (angl. *hash function*) novo narejenega bloka in nastaviti nonce[1] na tak način, da bo izhodna vrednost manjša od določene ciljne vrednosti. Ko eden izmed udeležencev najde pravilni nonce, bo blok poslan v omrežje in ostali udeleženci si bodo lahko posodobili svojo lokalno kopijo verige blokov.

Bitcoin uporablja SHA-256 zgoščevalno funkcijo [11]. Če imamo nezlomljeno zgoščevalno funkcijo, lahko pravilni nonce najdemo samo s poskušanjem različnih vrednosti.

Zaradi stabilnosti in razumnih čakalnih časov za potrditev transakcije, je ciljna vrednost prilagojena vsakih 2016 blokov. Nato je ponovno izbrana, da ustreza potrditveni hitrosti enega bloka na 10 minut. V povprečju je to na vsaka dva tedna ($= 2016 * 10min$). Nov cilj se izračuna kot

$$T = T_{prev} \cdot \frac{t_{actual}}{2016 \cdot 10min} \quad (1)$$

kjer je T_{prev} stara ciljna vrednost in t_{actual} je čas porabljen za generiranje prejšnjih 2016 blokov.

Naj povzamemo to v kontekstu primera, kjer Ana želi poslati kovance Branku. Ko Ana pošlje svojo transakcijo, jo prejmejo Branko, Cene in ostali udeleženci. Vsi potrdijo njeno legitimnost glede na njihovo lokalno kopijo verige blokov in jo dodajo k ostalim transakcijam (dodajo jo v trenutni blok). Če Cene želi širiti njegovo mnenje o tem, da je njegova zbirka transakcij v bloku pravilna mora rešiti uganko. Predpostavimo, da Cene prvi reši uganko tako, da najde pravilni nonce, za katerega zgoščena vrednost ustreza ciljni vrednosti. Nato pošlje blok transakcij in izračunan nonce ostalim udeležencem, kateri lahko potrdijo, da je to veljavna rešitev in dodajo nov blok v svojo verigo. Sedaj lahko smatramo, da je Anina transakcija veljavna in Branko je novi lastnik kovancev.

Bitcoin spodbuja potrjevanje transakcij tako, da nagradi prvega, ki uspešno dostavi dokaz o delu. Obstajata dva izvora take nagrade: provizija pri transakcijah in rudarjenje. Rudarjenje je proces dodajanja novih blokov v verigo, ker s tem

[1] 32-bitno polje znotraj bloka

ustvarjamo nove kovance. Kot žlahtna kovina in zbirateljski predmeti, so tudi bloki redek pojav, ker so parametri izbrani tako, da imamo eno uspešno rešitev uganke vsakih 10 minut. To pomanjkanje ustvari vrednost, ki je podprto z realnimi računskimi viri, ki so potrebni za rudarjenje. Provizije pri transakcijah pa so prostovoljni prispevek tistih, ki bi radi izvršili transakcijo. Večja kot je provizija večja spodbuda je rudarjem, da dodajo to transakcijo v naslednji blok, ki na koncu tudi poberejo zbrane provizije.

V Bitcoinu je bila začetna nagrada za blok nastavljena na 50 kovancev (50 BTC). Vsakih 210000 blokov, kar približno znaša 4 leta ($= 210000 \cdot 10min$), se nagrada prepolovi. To se je prvič zgodilo novembra 2012, ko se je zmanjšala na 25 BTC. To prepolavljanje se bo nadaljevalo, dokler nagrada ne pade pod 10^{-8} BTC. To je minimalna enota Bitcoina znana kot *satoshi*.

2.3 Veriga blokov

Zaenkrat smo podali samo abstraktno razlago verige blokov in jo opisali kot porazdeljeno evidenco o lastništvu. Sedaj bomo bližje pogledali strukturo verige blokov in videli kako Bitcoin ohranje bloke v pravem vrstnem redu in skrbi za konsenz celotnega omrežja.

Da lahko ugotovimo lastništvo vsakega kovanca, more biti vzpostavljena popolna urejenost blokov in transakcij. Zaradi tega razloga bloki vsebujejo kazalec na prejšnji potrjeni blok v verigi. Kazalec je implementiran kot zgoščena vrednost prejšnjega bloka. To je prikazano na Sliki 1.

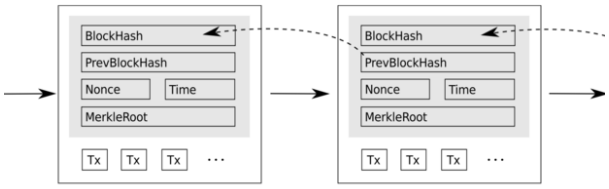


Figure 1: Poenostavljen prikaz verige blokov

Zaradi stalnega rudarjenja veriga blokov stalno raste. Zaradi priljubljenosti Bitcoina in iger na srečo kot je SatoshiDice je število transakcij ogromno naraslo. Stave na SatoshiDice proizvedejo dve transakciji: ena za vložek in ena za izplačilo. Junija 2012 so imeli približno 62000 dnevnih transakcij. Kot posledica tega se je povečala velikost verige blokov, ki sedaj meri desetine gigabajtov. Veliko število transakcij tudi otežuje proces potrjevanja. Da bi ohranili majhno velikost in manjši računski napor, Bitcoin ponuja preprosto preverjanje plačila (angl. *simple payment verification*, SPV) [10] osnovano na Merkle drevesih [12]. Vzame transkacije kot liste drevesa in na njih zgradi zgoščeno drevo. Koren drevesa je zgoščena vrednost, ki vsebuje informacije vseh transakcij in ga vključimo v blok. Zgoščeno drevo nam omogoča, da preverimo vse transakcije brez potrebe za celotno lokalno kopijo vseh transakcij.

Ker so potrditve blokov izračunane na porazdeljen način skozi rudarjenje, lahko pride do racepov. V primeru razcepa imamo dve (ali več) različici verige, ki lahko vsebujejo drugačne množice vključenih transakcij. Različni udeleženci v sistemu se ne bodo strinjali glede strukture verige blokov in ne dosežemo splošnega konsenza. Bitcoin to reši na preprost ampak učinkovit način: rudarjenje se nadaljuje na najdaljši lokalno znani veji (tisti, ki vsebuje največ računskega napora). V neki točki bodo rudarji ene veje razcepa razširili svojo potrditev pred drugimi. Ta veja bo prehitela ostale in postala najdaljša veja tudi za vse ostale udeležence. Ostale veje razcepa pa postanejo sirote (angl. *orphaned*).

2.4 Transakcije

Do sedaj nismo natačno definirali kaj so kovanci v Bitcoin protokolu. Dejansko kovanci kot taki ne obstajajo. Obstajajo samo transakcije, ki dodeljujejo lastniške pravice. Najbližje kovancu v Bitcoinu je zaporedje transakcij.

Preden lahko Branko dobi kovance si mora ustvariti virtualno denarnico, ki vsebuje vsaj javni in zasebni ključ. Brankov Bitcoin naslov je sestavljen iz javnega ključa, ki ga zgostimo s SHA-256 zgoščevalno funkcijo in nato še z RIPEMD-160. Naslovi so base58 kodirani, da se odstranijo dvoumni znaki. Z Bitcoin naslovi skrajšamo in prikrijemo javne ključe.

Predpostavimo, da Branko pošlje svoj Bitcoin naslov Ani. Ana uporabi svojo denarnico, da izvede transakcijo z Brankovim naslovom kot cilj. Slika 2 shematično prikazuje to transakcijo. Ključni elementi transakcije so zgoščena vrednost kot identifikator transakcije in seznam vhodov (angl. *inputs*) in izhodov (angl. *outputs*). Ana uporabi vhode, da uporabi do sedaj neuporabljene izhode prejšnjih transakcij. Podrobneje, prevTx je zgoščena vrednost, ki identificira prejšnjo transakcijo in index je indeks izhoda v prejšnji transakciji. Vsak izhod transakcije je lahko uporabljen kot vhod nove transakcije le enkrat v celotni verigi blokov. Uporaba istega izhoda dvakrat se smatra kot dvojno zapravljanje in je kot tako prepovedano.

Za izvršitev transakcije mora Ana za vsak izhod navesti koliko kovancev bo prenešeno (value) in komu poslati te kovance (scriptKeyPub), kjer lahko navede Brankov javni Bitcoin naslov. Pomembno dejstvo je, da transakcijski vhodi ne določajo koliko kovancev bo poslanih. Zato ker je lahko vsak izhod uporabljen le enkrat, morajo vhodi nujno uporabiti celoten izhod. Ker imajo transakcije lahko več vhodov in izhodov, to ne omejuje Bitcoina. Ana lahko uporabi dodaten izhod v transakciji in pošlje del kovancev nazaj na njen naslov. Na tak način Bitcoin implementira idejo *drobiža*. Vsota vseh vhodov v standardni transakciji mora biti vsaj toliko kot vsota vseh izhodov. Ne rabi biti točno enaka, ker se v nasprotnem primeru razlika v vsotah uporabi kot provizija pri transakciji.

Ker so vse transakcije povezane v verigo lahko sledimo zgodovini vsake transakcije. Vsaka zgodovina se konča pri *genesis* bloku ali pri *coinbase transakciji*. Oboje vsebuje posebne transakcije, ki vsebujejo samo izhode brez vhodov. Genesis blok je začetek celotne verige blokov in predstavlja začetnih 50 BTC v omrežju. Coinbase transakcija je bolj pogosti izvor transakcij. To je transakcija, ki nagradi rudarja za potrjevanje bloka in s tem uvede nove kovance v sistem. Zaradi razcepov v verigi blokov in dejstva, da bodo določene veje osirotele, so coinbase transakcije zaklenjene za 100 blo-

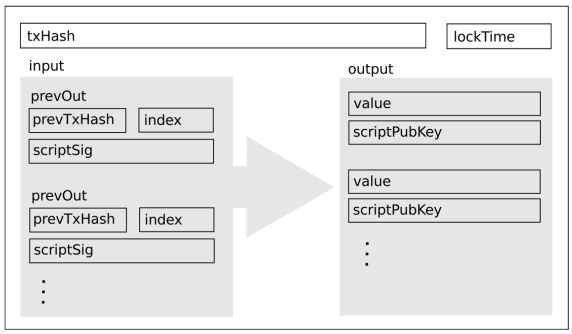


Figure 2: Shematični prikaz Bitcoin transakcije

kov (ni jih možno uporabiti dokler 100 naslednjih blokov ni potrjeno).

2.5 Skripte

S skriptami lahko razširjamo Bitcoinov transakcijski sistem in s tem dosežemo več fleksibilnosti kot pa samo preprost prenos kovancev. Skriptiranje je implementirano kot preprost skladovno orientiran (angl. *stack-based* jezik. Namenoma je načrtovan tako, da ni Turing popoln (angl. *Turing complete*), ker je tako lažje obvladljiv in nima nenačrtovanih stranskih učinkov.

Vsak vhod v Bitcoin transakciji se poveže z danim izhodom prejšnje transakcije. Ukazi, ki se bodo izvedli skupaj s konstantami sestavljajo scriptPubKey. Skripta pričakuje določeno število argumentov imenovanih scriptSig. Vhod, ki povezuje izhod mora zagotoviti scriptSig za dano skripto. Povezava je veljavna takrat, ko se izhodna skripta ovrednoti v true za podani scriptSig.

Najbolj ključna in najbolj pogosto uporabljena skripta je "Pay-to-PubKeyHash" (P2PKH). Transakcija, ki uporablja P2PKH prenese kovance iz ene ali več izvornih naslovov na enega ciljnega. Ideja je imeti skripto na izhodu, ki preveri ali so vsi povezani vhodi podpisani z javnim ključem, ki ima v lasti kovance na izhodu. Skripta 1 vsebuje predlogo za skripto P2PKH. V scriptSig P2PKH potrebuje javni ključ (pubKey) katerega pretvorimo z zgoščevalno funkcijo v naveden Bitcoin naslov in podpis (sig), ki dokazuje last danega zasebnega ključa.

```
scriptPubKey: OP_DUP OP_HASH160 <pubKeyHash>
  OP_EQUALVERIFY OP_CHECKSIG

scriptSig: <sig> <pubKey>
```

Skripta 1: Predloga za skripto "Pay-to-PubKeyHash" (P2PKH)

Obravnavajmo Anino transakcijo Branku. Ana zamenja pubKeyHash v Skripti 1 z Brankovim Bitcoin naslovom. Nato bi dodala to skripto kot eno izmed izhodnih skript povezano z vrednostjo, ki bi jo rada poslala. Če bi Branko rad porabil te kovance, mora priskrbeti njegov javni ključ in podpis v povezan vhod v scriptSig.

Tabela 1 prikazuje izvajanje skripte in stanje na skladu v vsakem koraku. Tukaj pubKeyBranko označuje Brankov javni ključ, pubKeyBrankoHash označuje zgoščeno vrednost javnega ključa in sigBranko označuje Brankov podpis. Na začetku so scriptSig (argumenti skripte) iz vhoda in scriptPubKey (ukazi skripte) iz izhoda združeni. Prva žetona (angl. *tokens*) v združenem rezultatu sta sigBranko in pubKeyBranko. To sta konstanti in vse konstante so preprosto dodane na vrh sklada. Ukaz OP_DUP podvoji najvišji vnos na skladu. OP_HASH160 izračuna zgoščeno vrednost dvakrat (SHA-256 in RIPEMD-160) in doda rezultat pubKeyBrankoHash na vrh sklada. V naslednjem koraku se pubKeyBrankoHash (katerega je dodala Ana v skripto) doda na vrh sklada. OP_EQUALVERIFY preveri, če sta dve vrhnji vrednosti na skladu enaki in sproži napako če nista. Ta korak predstavlja prvi pomembni korak, ki preveri, če je bil vstavljen pravilni javni ključ. Zadnja operacija OP_CHECKSIG preveri podpis z javnim ključem in doda true na vrh sklada, če se ujemata. Če je zadnja preverba uspešna, skripta omogoči Branku, da porabi kovance. Transakcija je veljavna, če vmes ni napake in po zaključku skripte na vrhu sklada ostane vrednost true.

Poleg Bitcoina obstaja še tako imenovana druga generacija kriptovalut kot so Mastercoin, Counterparty in Ethereum. Vse implementirajo novo sintakso transakcij s Turing popolnim skriptnim jezikom. Tako je možno zgraditi pametne pogodbe [13] kot so delnice z avtomatičnim izplačilom dividend ali trgovanje s fizično lastnino kot so avti.

3. OMREŽJE BITCOIN

V tem sestavku bomo predstavili organizacijo porazdeljenega omrežja Bitcoin. Bližje si bomo ogledali, kako poteka propagacija informacij po omrežju, saj je to ključen element v varnosti protokola Bitcoin. Nato naredimo še kratek pregled drugih aplikacij, ki so izvirajo iz tega koncepta.

Omrežje Bitcoin je zasnovano na nestukturiranem omrežju (*ang. unstructured overlay network*), ki temelji na omrežnem protokolu TCP. Tovrstna omrežja so v splošnem zelo robustna proti fluktuaciji povezanih vozlišč. Iz preteklih raziskav podobnih omrežij npr. omrežje Gnutella za porazdeljeno deljenje datotek (*ang. Gnutella peer-to-peer file sharing network*) pa vemo, da imajo taka omrežja težave z povečevanjem obsega delovanja[14]. Vsak zahtevek je preposlan po celotnem omrežju, kar povzroči veliko število podvojenih zahtevkov. Tako obremenitev vsakega vozlišča narašča linearno s številom povezanih vozlišč. Da bi se omilili ta učinek, na vsak zahtevek pripnemo *max_hops*, ki omeji število kopij, vendar to predstavlja novo težavo - tedaj nimamo jamstva, da bo vsak zahtevek dosegel vsako vozlišče v omrežju.

Cilj omrežja Bitcoin je bistveno drugačen od omrežja Gnutella, kjer je popolna distribucija vseh podatkov na najhitrejši možen način ključna za delovanje in varnost protokola Bitcoin. Zato mora Bitcoin tovrstne težave rešiti na drugačen način.

3.1 Priključitev v omrežje

Vsako vozlišče aktivno vzdržuje vsaj 8 odprtih povezav do drugih vozlišč v omrežju. Če katero od teh vozlišč postane nedostopno mora vzpostaviti povezavo z drugimi vozlišči. Če vozlišče sprejema vhodne povezave, se to število lahko bistveno poveča, toda ponavadi to število omejimo navzgor

Sklad	Skripta
	`sigBranko pubKeyBranko OP_DUP OP_HASH160 pubKeyBrankoHash OP_EQUALVERIFY OP_CHECKSIG`
`sigBranko pubKeyBranko`	`OP_DUP OP_HASH160 pubKeyBrankoHash OP_EQUALVERIFY OP_CHECKSIG`
`sigBranko pubKeyBranko pubKeyBranko`	`OP_HASH160 pubKeyBrankoHash OP_EQUALVERIFY OP_CHECKSIG`
`sigBranko pubKeyBranko pubKeyBrankoHash`	`pubKeyBrankoHash OP_EQUALVERIFY OP_CHECKSIG`
`sigBranko pubKeyBranko pubKeyBrankoHash pubKeyBrankoHash`	`OP_EQUALVERIFY OP_CHECKSIG`
`sigBranko pubKeyBranko`	`OP_CHECKSIG`
`true`	

Table 1: Primer izvajanja P2PKH skripte

s parametrom *maxconnections*, ki je tipično okoli 125. Privzeta vrata za vhodne povezave so 8333.

Če neko vozlišče zapusti omrežje Bitcoin, tega ne oznani svojim sosedom, vendar ti to odkrijejo sčasoma. Vsako vozlišče hrani naslov IP svojih direktnih sosdeov, hkrati pa časovni žig zadnje aktivnosti tega soseda. Če je od zadnje aktivnosti nekega vozlišča minilo več kot 90 minut, se smatra, da je vozlišče zapustilo omrežje. Zato vozlišča morajo sama poskrbeti, da obveščajo svoje sosede o svoji prisotnosti s sporočili srčnega utripa (*heartbeat messages*), ki jih pošljejo vsakih 30 minut. Vsako vozlišče hrani tudi seznam naslovo IP vseh nedavno aktivnih vozlišč v omrežju. Vsako vozlišče oznani svoj naslov IP celotnemu omrežju vsake 24 ur s sporočilom tipa *addr*.

Če neko vozlišče želi poiskati dodatne sosede (torej je število njegovih direktnih sosedov padlo pod 8) lahko to stori na več načinov. Prvi način smo že omenili, in sicer pogleda v svoj seznam nedavno aktivnih vozlišč, ki se prenašajo preko sporočil tipa *addr*. Drug način je, da pošlje vozliščem zahtevek tipa *getaddr*. Ob prejetju takega sporočila, vozlišče odgovori z naključnim izborom 25% vozlišč v svojem seznamu aktivnih vozlišč (število vozlišč v odgovoru je navzgor omejeno s 1000). Opazimo, da ta izbor ne vsebuje samo direktnih sosedov tega vozlišča, vendar člane celotnega omrežja. Na tak način skrijemo strukturo omrežja. To je en ključnih arhitekurnih ciljev omrežja Bitcoin, saj bi lahko zlikovec s tem znanjem uporabil celo vrsto napadov na posamezna vozlišča.

Ob prejemu sporočila tip *addr* vozlišče prepošlje sporočilo največ 10 svojim sosedom. To število je določeno z dostopnostjo naslova IP (torej ali je naslov IP javen ali zaseben) v sporočilu *addr*. To je smiselno, saj na tak način promoviramo javne IP naslove in ne naslovov IP, skritih za NAT omrežji.

Ob prvi povezavi v omrežje Bitcoin, mora vozlišče najti sosede. To lahko stori na tri načine: DNS, IRC in izpraševanje sosedov. Od Bitcoin verzije 0.6 je privzet način delovanja iskanje sosedov z DNS. Naslovi DNS so predloženi z vsako distribucijo programske opreme Bitcoin.

3.2 Širjenje transakcij in verige blokov

Širjenje informacij znotraj omrežja Bitcoin je konceptualno zelo preprosto - sporočila so razposlana po celotnem omrežju.

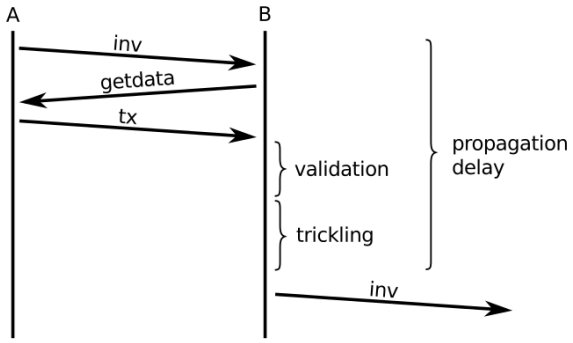


Figure 3: Postopek širjenja transakcij po omrežju Bitcoin.

Vendar da bi omejili količino podatkov znotraj vsakega sporočila, se uporabi poseben postopek za širjenje transakcij. Denimo, da Ana želi opraviti transakcijo in bi rada o tej transakciji obvestita preostalo omrežje. Predenj razpošlje podatke o sami transakciji, pošlje v omrežje sporočilo tipa *inv*, ki vsebuje zgoščene vrednosti transakcij, za katere Ana ve. Anini sosedi prejmejo to sporočilo in ga pregledajo, ali obstaja kakšna zgoščena vrednost transakcije, ki je morda še niso videli. Če tako transakcijo najdejo, Ani odgovorijo s sporočilom tipa *getdata*, nakar Ana odgovori s samo transakcijo. Točen postopek je prikazan an Sliki 3.

Vendar pa ni vse tako preprosto. Ob pošiljanjiu sporočila tip *inv* se dejansko ne pošljejo vse transakcije, za katere je Ana že slišala, vendar se naključno izbere nekaj transakcij z $p = 0.25$. Le eno samo vozlišče, ki ga imenujemo trickling node dobi podatke o vseh transakcijah. Na tak način zmanjšamo Anino obremenitev, saj bodo vozlišča, ki so prejeli transakcije od Ane, potem, ko transakcije preverijo, tudi ona poslala sporočila tipa *inv* svojim sosedom. To sicer zmanjša obremenitev na omrežje, vendar pa upočasni širjenje informacij po celotnem omrežju. Zaradi tega mehanizma pa se včasih lahko zgodi, da kakšni transakciji ne uspe priti v nov blok. Za to mora poskrbeti izvornik transakcije, in v primeru, da se to zgodi, mora znova obvestiti omrežje o opravljeni transakciji. V praksi se to le redko zgodi (okrog 3%).

Na podoben način se širijo na novo prozivedeni bloki, vendar je tukaj bolj pomembno, da se bloki po omrežju razširijo čim hitreje, zato se koncept *trickling node* tukaj izpusti. Počasno širjenje blokov bi posledično pomenilo več razcepov, ki jih protokol Bitcoin sicer zna razrešiti, vendar kljub temu pa izgubimo veliko računskega dela.

V delu Deckerja[15] so raziskali širjenje informacij znotraj omrežja Bitcoin. Povezali so se na veliko število vozlišč kot opazovalec t.j. niso posredovali sporočil ostalim vozliščem. Beležili so časovne zamike zahtevkov tipa *inv*. Rezultati kažejo na eksponentno porazdelitev časovnih zamikov z mediano okoli 6.5 sekund in povprečjem okoli 12.6 sekund. Porazdelitev je pokazala, da 5% vozlišč še ni prejela informacij o novem bloku tudi 40 sekund po prvem obvestilu. Avtorji se osredotočijo tudi na verjetnosti razcepov verige blokov z večanjem časovnega zamika. To je intuitivno lahko razumeti, saj če je čas širjenja nekega bloka večji, bo tudi verjetnost razcepa večja. Tako zaključijo, da se z večanjem premera omrežja (z dodajanjem novih vozlišč), hkrati težave povečujejo.

Decker in Watenhofer[15] sta tudi poiskusila zmanjšati premer omrežja, tako da sta se povezati na veliko število vozlišč in tako bistveno zmanjšala premer omrežja (na skoraj 2). Posledično se je verjetnost razcepa zmanjšala iz 1.69% na 0.78%. Za to sta potrebovala povezavo 100MB/s, kar nakazuje na hujšo težavo, s katero se Bitcoin sooča danes - s skalabilnostjo.

3.3 Skalabilnost

Glavni cilj omrežja Bitcoin je čim hitrejša distribucija informacij o verigi blokov. V splošnem so nekonsistentna stanja t.j. razcepi v verigi blokov nezaželjena, saj olajšajo napad dvojnega zapravljanja. Omrežje Bitcoin se sooča z težavo skalabilnosti zaradi velikosti verige blokov in zaradi časovnih zamikov širjenja informacij, ki je opisan v prejšnem poglavju.

Velikost enega bloka v trenutni specifikaciji protokola Bitcoin je 1MiB, kar omeji število transakcij, ki jih lahko spravimo v en blok. Meja je postavljena z namenom, da bi preprečili prehitro večanje verigo blokov. Glede na postavljeno težavnost za uresničitev bloka, je realna zmogljivost protokola Bitcoin okoli 4 transakcije na sekundo. Če bi zmanjšali čas za "izkop"novega bloka, bi povečali verjetnost za razcep v verigi blokov, kar je zopen nezaželjeno.

V nasprotnem primeru, če bi povečali velikost blokov na 512MiB, bi lahko teoretično dosegli hitrosti 2000 transakcij na sekundo, vendar bi posledično potrebovali internetno povezavo s hitrostjo 1MiB/s. Če bi Bitcoin res kdaj želel nadomestiti centralizirane bančne sisteme, bi se moral najprej spopasti s takimi omejitvami.

Predlagan pristop, ki delno rešuje te težave je razdelitev vozlišč v tri skupine: polna vozlišča, tanka vozlišča in denarnice. Polna vozlišča bi hranila celotno verigo blokov in preverjala vse bloke do prvega. V nasprotju bi tanka vozlišča hranila samo glave blokov, s katerimi bi lahko preverjal veljavnost transakcij, saj vsaka glava bloko vsebuje zgoščeno vrednost Merklovega drevesa vseh transakcij celotnega bloka. V primeru, da bi tanko vozlišče potrebovalo podatke o posamezni trasnakciji, bi lahko le te zahtevale od polnih vozlišč. Tukaj nastopi nova težava zasebnosti, saj bi lahko na podlagi zahtev o posameznih transakcijah polna vozlišča lahko povezala naslove IP s posameznimi transakcijami (v večini primerov lahko sklepamo, da je izvor nove transakcije tudi njen lastnik). Temu bi se izognili tako, da za poizvedbe uporabimo *Bloom filter*, ki od polnega vozlišča zahteva vse transakcije, ki se ujemajo z zgoščeno vrednostjo. Na tak način otežimo identifikacijo posameznih uporabnikov.

Kljub temu, pa je tak pristop protisloven s celotnim konceptom Bitcoina, ki želi vzpostaviti decentraliziran bančni sistem. S polnimi vozlišči na nek način vpeljemo centraliziranost v sicer decentraliziran sistem.

3.4 Anonimnost

Iz sledenja toku informacij po omrežju Bitcoin lahko izvemo veliko o samih uporabnikih. Kaminsky[16] je pokazal, da lahko z nadzorom nad visoko povezanem vozličem v omrežju lahko izvemo izvorne IP naslove velikega števila transakcij.

Temu se lahko delno izognemo z uporaba anonimizacijske storitve kot je npr. Tor. Ta nam omogoča, da skrijemo izvorni IP naslov transakcije, in jo zamenjamo z t.i *exit node*. Vendar pa je bilo pokazano, da lahko izstopna vozlišča omrežja Tor napademo z *denial-of-service* napadom, kjer onemogočimo uporabnikom omrežja Tor dostop do omrežja Bitcoin[17]. Za to lahko izkoristimo mehanizem v protokolu Bitcoin, ki služi kot zaščita pred napadi tipa *denial of service*. Za vsako vozlišče se hrani ocena zlonamernosti, ki se poveča ob neveljanem sporočilu. Če ta vrednost preseže neko mejo, temu vozlišču onemogočimo dostop do preostalega omrežja Bitcoin za 24 ur. Tako bi lahko zlikovec pošiljal neveljavna sporočila iz izstopnih vozlišč omrežja Tor, tako da bi jim bil dostop do omrežja Bitcoin onemogočen.

Toda kljub uporabi storitve Tor nam anonimnost ni garantirana. Biryukov[17] je pokazal, da lahko sosedje posameznih vozlišč služijo kot enoličen podpis posameznega vozlišča. Ko se vozlišče poveže v omrežje se namreč poveže na 8 sosedov, preko katerih oznanja svoj naslov IP preostnku omrežja. Ker vsako vozlišče aktivno vzdržuje povezave do teh 8 sosedov, lahko predpostavljamo, da je ta podpis stabilen in se bo le redkokdaj spremenil. S takim pristopom jim je uspelo identificirati 11% identitet lastnikov transakcij.

3.5 Botneti

Botneti so množica računalnikov, ki jih nadzoruje ena sama entiteta. Največkrat imamo opravka z ilegalnimi botneti, kjer proces za upravljanja računalnika teče v ozadju brez vedenja lastnika. Upravljatelj botneta ima dostop do datotek in do računskih virov. Upravljanje botnetov poteka preko kanala CC (command and control channel), ki največkrat teče skozi IRC, Tor, ipd. Kot zanimivost omenimo, da lahko navodila CC kanala zakodiramo v transakcije v verigi blokov v protokolu Bitcoin.

Botneti so največkrat uporabljeni za ilegalno storitve, kot so npr. spam, DDoS (*ang. distributed denial of service*). Povsem naravno je, da so botneti postali uporabni za dobiček z dokazom dela. Najprej botneti preverijo računsko moč okuženih računalnikov, in če je le ta dovoljšna, izkoristijo sistem za reševanje kriptografskega izziva. Tukaj obstaja

več pristopov. Prvi je najbolj preprost - okuženi računalniki nakazujejo svoje zaslužke na eno entiteto. Tak pristop imenujemo pooled mining s posrednikom. Drug pristop je tak, da se vse računalniki povežejo na javne mining poole. Tovrstne botnete lahko ti pooli zelo hitro zaznajo, saj opazijo veliko število računalnikov z razmeroma slabo računsko močjo pod enim samim lastnikom. Tretji pristop je tak, da preko CC kanala vzpostavimo lasten pool, na katerega objavljajo rešitve okuženi računalniki.

Ker so dobički z uporabo botnetov kar znatni, sta Ragan in Salazar[18] preizkusila bolj legalno rešitev. Izkoristila sta javne oblačne storitve. Dobički so bili precejšni in sicer sta tedensko zaslužila več tisoč dolarjev.

4. VARNOST

Odkar je Bitcoin digitalna valuta z opazno tržno vrednostjo, je postal zanimiv za izkoriščanje njegovih slabosti oz. pomanjkljivosti. Najpogostejša varnostna težava Bitcoina niso vezane na tehnologijo, ampak na porabnika in njegovo denarnico. Ta hrani ključe za dostop do Bitcoinov, ki pa jo uporabnik lahko izbriše, jo izgubi ali pa mu jo ukradejo. Vendar pa so se skupaj z težavami razvijale nove tehnologije, ki skušajo rešiti problem varovanja denarnice uporabnika. Tako so nastale šifrirne denarnice, denarnice brez povezave in strojne denarnice, ki vsaka na svoj način preprečuje napade. Vendar pa ostaja na lastniku odgovornost do digitalne lastnine, podobno kot to velja v nedigitalnem svetu. To lahko zagotovi z varnostnimi kopijami, z uporabo večjega števila varnih lokacij za shranjevanje, šifriranjem, uporabo močnih gesel, itd.

Poleg omenjenih varnostnih težav uporabnika so bile v preteklosti najdene tudi tehnološke varnostne grožnje med katerimi najbolj izstopa t. i. 51% napad, ki ga bomo predstavili v tem poglavju.

4.1 Napad z dvojnim zapravljanjem

V povezanem načinu s centralizirano banko in s kovanci, ki se razlikujejo po serijskih številkah, je poskus dvojnega zapravljanja enostavno odkriti. Vendar pa digitalne valute pogosto omogočajo nepovezaven način, ki ne dovoljuje vspostavitve povezave z namenom avtorizacije transakcije. To je pomemben problem, ki ga lahko izkorišča dvojno zapravljanje, kljub obstoju centralne avtoritete. Čeprav Bitcoin omogoča tudi delovanje brez povezave[19], pa je njegova značilnost trgovanje s povezavo. Kljub temu, da Bitcoin ni centraliziran, pa porazdeljena evidenca o lastništvu odpira možnosti za dvojno zapravljanje.

Dva splošna načina soočenja z dvojnim zapravljanjem sta: (i) zaznati ga po tem, ko se je prevara dejansko zgodila in prepoznati storilca za pregon, ali (ii) poskušati preprečiti že pred samo izvedbo. Kot smo že omenili, se Bitcoin zaščiti pred dvojnim zapravljanjem preko pravila, da so samo do sedaj ne porabljeni transakcijski izhodi lahko uporabljeni kot vhodi v naslednjo transakcijo. Pravilo se uveljavlja pri dodajanju novih transakcij in pri rudarjenju, kjer je zaporedje transakcij določeno z njihovim zaporedjem v verigi blokov. Tukaj izvira potreba po porazdeljenem časovnem žigu in algoritmu soglasja oz. konsenza. Izraz, ki se ga lahko razume kot problem Bizantinskih generalov.

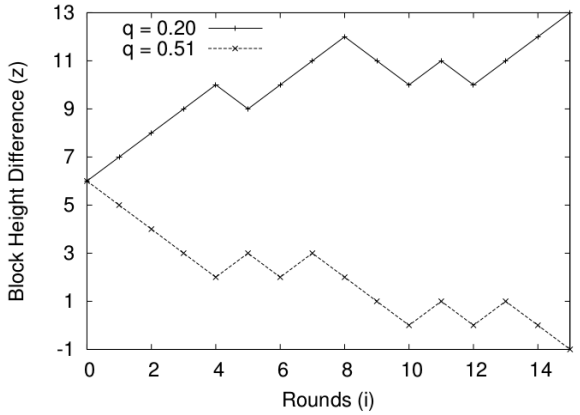


Figure 4: Prikaz naključnega pohoda z rezultatom v > 50% napadu (označeno črtkano)[21].

Avtorji članka [20] so modelirali protokol Bitcoin po vzoru algoritma Bizantinskega soglasja. Pokazali so, da protokol dejansko doseže soglasje. V splošnem, pri predpostavki sinhrone komunikacije, Bitcoin doseže najvišjo vzdržljivost oz. odpornost na grožnje, t. i. $2f + 1$ vzdržljivost, kljub prisotnosti zlikovcev. To pomeni, da je sistem varen tako dolgo, dokler pošteni uporabnik (n poštenih) prevladajo nad goljufivimi (f goljufivih uporabnikov). To razmerje lahko predstavimo z enačbo:

$$n > 2f + 1$$

V splošnem načrt za izvedbo napada dvojnega zapravljanja vsebuje naslednje korake: (i) izvedi pravilno transakcijo (npr. plačaj izdelek z Bitcoin denarjem), (ii) na skrivaj rudari na veji, ki je zgrajena na zadnjem bloku in vsebuje vsebuje konfliktno transakcijo (vsebuje enake vhode kot transakcija v točki (i), vendar namesto prodajalcu plačuje napadalcu), (iii) počakaj dokler prodajelec ni samozavesten - dobi dovolj potrdil rudarjev in izroči izdelek, (iv) ko postane skrivna veja daljša od verige blokov, razpošlji njene bloke. Ker skrita veja postane daljša, jo omrežje sprejme kot veljavno glavno verigo blokov. Pravilne transakcije postanejo neveljavne in ne morejo biti vključene v verigo.

Če je verjetnost, da pošteno vozlišče poišče naslednji blok v verigi manjše, kot je verjetnost, da to v lažni verigi blokov stori napadalec, potem bo napadalec v svoji nameri uspel. Tako bo pridobil več kot polovico računske moči, ki mu omogoča, da izvede napad. Tako tekmovanje med pravo in goljufivo verigo blokov v Bitcoin terminologiji imenujejo naključni pohod (*angl. random walk*). Slika 4 tako prikazuje dva različna razpleta takšnega pohoda, kjer je na začetku pravilna veriga za 6 blokov daljša od goljufive ($z_0 = 6$). Verjetnost, da napadalec poišče naslednji blok v skriti veji je v enem primeru 20% v drugem pa 51% ($q = 0.51$). Zaradi želje po pridobitvi več kot polovice virov, se ta posebna oblika napada dvojnega zapravljanja imenuje > 50% ali včasih tudi 51% napad.

5. ZASEBNOST

V prvotnem Bitcoin dokumentu je na kratko opisana problematika zasebnosti: za razliko od klasičnega bančništva, ki uporablja model zaupanja vredne avtoritete, ki omejuje dostop do informacij o trgovanju, Bitcoin javno razkrije verigo blokov oz. vse podatke o transakcijah. Javni naslovi v verigi blokov poskušajo uporabnikom zagotoviti anonimnost, saj so transakcije zabeležene le z vzdevki. Vendar pa ta odprtost zgodovine transakcij samo po sebi ne pomeni razpoznavnost oz. identifikacije uporabnikov. V podporo te funkciji, je potrebno uporabiti nov par ključev (in s tem nov naslov) za vsako transakcijo. Bitcoin uporabniki tako prevzeto uporabljajo tako imenovane spremenjene naslove (*angl. change addresses*), ki so generirani za vsako transakcijo posebej.

Do sedaj je bralec v predhodnih poglavjih že lahko opazil, da v Bitcoin transakcija ne vsebuje atributa od kje izhaja. Transakcije kažejo le na pretekle izhode. Če sledimo tem naslovom, lahko sklepamo o cilju in izvoru transakcije. Kljub podpori vzdevkov pa smo se že iz zasebnosti na socialnih omrežjih naučili, da se skrivanje za množico psevdonimov oz. vzevkov lahko poveže med seboj. Tako je moč pogosto ugotoviti podatek o identiteti.

5.1 Omogočanje zasebnosti

Čeprav zasebnost ni prirojena lastnost Bitcoina, pa je močno povezana z njim. Zasebnost se pogosto uporablja za namene, ko pošiljatelj in/ali prejemnik želi ostati anonimen. Obstaja velika želja po ustvarjenju popolno anonimne digitalne valute, ki pa do sedaj še ni bila udejanjena.

Za zaščito svoje zasebnosti v Bitcoin omrežju lahko poskrbi vsak uporabnik sam. Ta mora poskrbeti, da za vsako nakazilo, v katerem prejme novce, uporabi nov naslov. Poleg tega lahko za različne namene uporabljate različne denarnice in si s tem zagotovi, da nakazila brez dodatnih informacij ni mogoče povezati med seboj. S tem si zagotovi, da ljudje, s katerimi sodeluje v transakcijah, ne vedo, katere druge naslove ima in kakšna plačila z njimi izvaja.

Z namen preprečitvanja uspešnega analiziranja verige blokov, je potrebno ločiti podatke o pošiljatelju in sprejemniku. Obstajajo internetne storitve, ki se imenovanujejo mešalne storitve. Te mešajo novce uporabnikov tako, da se sled preteklih transakcij izgubi. Uporabnik tako lahko pošlje svoje novce v mešalni sistem, ki mu vrne nazaj novce enake vrednosti, ki pa niso enaki. Sam postopek ni priznan oz. zakonit v vseh državah. Poleg tega je potrebno zaupanje uporabnika ponudniku storitve, da mu bo res vrnil vloženi denar in si ne bo zapolnil, kako je novce premešal. Ta pristop lahko zabriše sledljivost za manjše zneske, višji pa kot so zneski, težje je sledljivost zabrisati.

V prejšnem poglavju smo že omenili, da je v omrežju Bitcoin mogoče prepoznavati IP naslove posrednikov nakazil. Prav tako odjemalci posredujejo tuja nakazila na enak način kot svoje. To se odraža v oteženem iskanju izvora posameznega nakazila in vir se lahko napačno interpretira. Nekateri uporabniki, ki jih je beleženje IP-naslovov motilo, so uporabili program Tor za maskiranje IP-naslova in s tem zaščito identitete med izvanjem transakcij. Vendar uporaba takih programov ni priporočljiva, saj odpira nova varnostna tveganja.

V zadnjem času se je bilo na področju zasebnosti narejenih veliko raziskav, zato lahko v prihodnosti pričakujemo dodatne izboljšave v Bitcoinu. V programskem vmesniku za opravljanje transakcij se tako poskuša preprečiti plačevanja z več različnih naslovov, saj jih ta način javno poveže kot naslove istega lastnika. Za namen lažjega zagotavljanja zasebnosti se je začel razvoj grafičnih uporabniških vmesnikov denarnic, ki omogočajo uporabniku prijaznejše izvajanje transakcij oz. plačil in skušajo preprečiti večkratno uporabo enakih naslovov. Veliko raziskav se ukvarja tudi z razvojem razširjenih funkcionalnosti za zagotavljanje večje zasebnosti. Ena takšnih je možnost združevanja nakazil nepovezanih uporabnikov, ki v omrežju prepričijo sledljivost transakcij.

6. ZAKLJUČEK

V tem članku smo naredili pregled širokega področja Bitcoina, njegove značilnosti in sorodne koncepte. Pri tem smo raziskovali temelje Bitcoin protokola, vključno z vlogo dokazila o delu, in njihovega odnosa do varnosti in omrežnih vidikov. S tem smo zagotovili celovit tehnični pogled na porazdeljenimi valutami. Opozorili smo, da ostajajo odprte raziskovalne možnosti na različnih področjih.

7. LITERATURA

[1] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM*, vol. 24, no. 2, pp. 84–90, 1981.

[2] P. R. Zimmermann, *The official PGP user's guide*. 1995.

[3] A. Back *et al.*, "Hashcash-a denial of service counter-measure," 2002.

[4] B. Cohen, "Incentives build robustness in bittorrent," 2003.

[5] D. Chaum, "Blind signatures for untraceable payments," in *Advances in cryptology*, pp. 199–203, Springer, 1983.

[6] L. Law, S. Sabett, and J. Solinas, "How to make a mint: the cryptography of anonymous electronic cash," *Am. UL Rev.*, vol. 46, p. 1131, 1996.

[7] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 4, no. 3, pp. 382–401, 1982.

[8] D. Malkhi and M. Reiter, "Byzantine quorum systems," *Distributed computing*, vol. 11, no. 4, pp. 203–213, 1998.

[9] J. R. Douceur, "The sybil attack," in *International Workshop on Peer-to-Peer Systems*, pp. 251–260, Springer, 2002.

[10] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system,"

[11] T. Hansen, "Us secure hash algorithms (sha and sha-based hmac and hkdf)," 2011.

[12] R. C. Merkle, "A digital signature based on a conventional encryption function," in *A Conference on the Theory and Applications of Cryptographic Techniques on Advances in Cryptology*, pp. 369–378, Springer-Verlag, 1987.

[13] N. Szabo, "The idea of smart contracts," *Nick Szabo's Papers and Concise Tutorials*, 1997.

[14] P. Kirk, "The annotated gnutella protocol specification v0. 4," in *The Gnutella Developer Forum (GDF)*, 2003.

[15] C. Decker and R. Wattenhofer, "Information propagation in the bitcoin network," in *Peer-to-Peer Computing (P2P), 2013 IEEE Thirteenth International Conference on*, pp. 1–10, IEEE, 2013.

[16] D. Kaminsky, ""black ops of tcp/ip," *Black Hat USA*, 2011.

[17] A. Biryukov, D. Khovratovich, and I. Pustogarov, "Deanonymisation of clients in bitcoin p2p network," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pp. 15–29, ACM, 2014.

[18] R. Ragan and O. Salazar, "Cloudbots: Harvesting crypto coins like a botnet farmer," *BlackHat USA*, 2014.

[19] A. Dmitrienko, D. Noack, A.-R. Sadeghi, and M. Yung, "On offline payments with bitcoin (poster abstract)," in *International Conference on Financial Cryptography and Data Security*, pp. 159–160, Springer, 2014.

[20] A. Miller and J. J. LaViola Jr, "Anonymous byzantine consensus from moderately-hard puzzles: A model for bitcoin," *Available on line: http://nakamotoinstitute.org/research/anonymous-byzantine-consensus*, 2014.

[21] F. Tschorsch and B. Scheuermann, "Bitcoin and beyond: A technical survey on decentralized digital currencies," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 3, pp. 2084–2123, 2015.

Bitcoin in mit decentralizacije

Predlogi za ponovno decentralizacijo sistema

Povzetek članka*

Dominik Sedmak
Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
ds8283@student.uni-lj.com

Jernej Poles
Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
jp8046@student.uni-lj.com

POVZETEK

Bitcoin je prva od tako imenovanih kripto valut. Z njenim prihodom na sceno leta 2009 se je začelo novo obdobje v zgodovini denarja. Namen Bitcoin-a ni bil nič manj kot spodnesti noge današnjim monetarnim inštitucijam in njihovemu modelu centraliziranega upravljanja z tokom denarja. Zaupanje v te ustanove naj bi bilo zamenjano z zaupanjem v računalniško kodo, pravilno zasnovo algoritmov in decentralizirane sisteme. Kljub tem visokim ciljem pa se je izkazalo da zasnova samega Bitcoin protokola ne spodbuja željene decentralizacije ampak ravno nasprotno. V tem članku si bomo pogledali na kakšen način Bitcoin protokol ne spoštuje prvotno zastavljenih ciljev in predstavili tri kategorije možnih popravkov, popravki na nivoju strojne opreme, programske opreme in električnega omrežja. Te bi lahko pomagale vrniti Bitcoin in podobne kripto valute na pravo pot decentralizacije.

KLJUČNE BESEDE

Bitcoin, decentralizacija, omrežje, popravki, P2P, denar, kripto valuta

1. UVOD

Zgodovinsko je bil nadzor nad tokom denarja v rokah bogatih posameznikov in kasneje finančnih ustanov. Za tem smo dobili centralne banke kar je do neke mere preneslo nadzor v roke prebivalstva. Centralne banke so uradno od vlade neodvisne ustanove prav tako pa velja, da je večina denarja ustvarjenega v samih zasebnih bankah in ima centralna banka le omejeno moč nadzora. V vsakem primeru je bilo za pravilno delovanje plačilnega in posojilnega sistema potrebno zaupanje v takšne ustanove. Navsezadnje svojega denarja ne bi prepustili nekomu za katerega ne veste ali boste lahko dobili cel znesek nazaj. Te finančne ustanove so svoj položaj izkoristile za proizvodnjo dobička z zaračunavanjem raznih stroškov in dandanes tudi prodajo osebnih podatkov.

Leta 2009 je zato neznanec pod psevdonimom Satoshi Nakamoto objavil odprtokodni protokol Bitcoin [1]. Namen tega protokola je bil stvaritev prve moderne kripto valute in s tem zamenjava zaupanja v finančne ustanove z preverljivo računalniško kodo in zaupanja vrednimi algoritmi. Velik poudarek je bil na decentraliziranosti celotnega sistema kar je bilo doseženo z uporabo protokola za pridobivanje soglasja večine sodelujočih. Ta protokol se uporablja za to da lahko celotno decentralizirano omrežje pride do soglasja o tem katere spremembe oz. transakcije so se zgodile v času od prejšnjega soglasnega stanja. Sodelujoči sistemu prostovoljno darujejo procesorsko moč in ga tako držijo pri življenju. Ta proces se imenuje rudarjenje in v zameno rudarji dobijo Bitcoin-e kot plačilo.

Za validacijo celotnega procesa rudarjenja in vzpostavljanja soglasja Bitcoin uporablja t.i. Proof-of-Work sistem [2]. Rudarji proizvajajo te dokaze tako da računajo SHA-256 vrednosti nad podatki o transakcijah, ki so ob uspešni stvaritvi takšnega dokaza vključeni v glavni tok transakcij. Ta proces računanja dokazov dela je tudi glavni krivec za centralizacijo, ki še vedno poteka v Bitcoin sistemu. Sčasoma ko je Bitcoin postajal vse bolj popularen so rudarji in njihovi sistemi postali vse bolj zapleteni in dragi ter so tako bolj ali manj izrinili manjše rudarje in tako monopolizirali nadzor nad delovanjem sistema.

V članku predstavljamo več načinov na katere bi lahko spremenili smer transformacije Bitcoin sistema. Za vsak način si pogledamo kako pomemben oziroma učinkovit je pri ponovni decentralizaciji sistema, kako težaven je za implementacijo, kakšna je verjetnost za usvojitev in kakšen je vpliv tržnih sil.

2. PREGLED PODROČJA

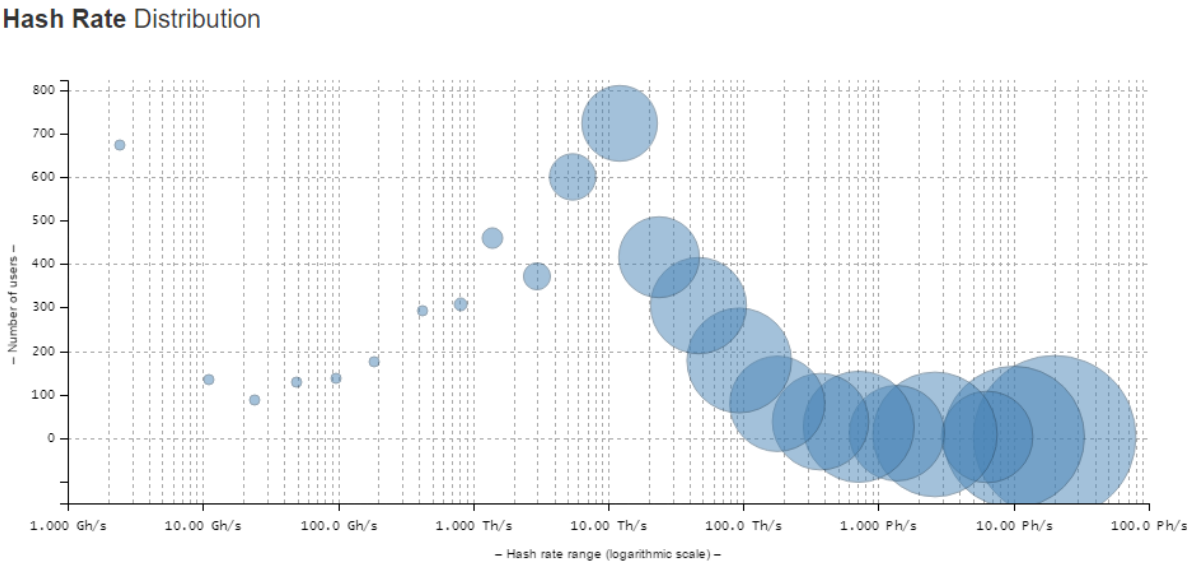
Nekaj avtorjev se je že lotilo analize postopne centralizacije Bitcoin-a in tudi vpliva, ki ga imajo današnje finančne ustanove.

V članku “When telcos become banks: sociotechnical control in mobile money.” je Rachel O'Dwyer govorila o nadzoru, ki ga imajo telekomunikacijska podjetja nad podatki posameznikov in tudi nad njihovim dostopom do osnovnih storitev [3]. Predstavila je primer kako več mobilnih operaterjev v Veliki Britaniji spremlja uporabnike in uporablja njihove podatke za oglase.

Andrew Poelstra je v spletni publikaciji govoril o vplivu ASIC vezij na centralizacijo sistemov kripto valut in pomeni nizke cene električne energije [4].
Ittay Eyal in Emin Gun Sirer sta v članku "Majority is not Enough: Bitcoin Mining is Vulnerable" pokazala možnost uporabe velikih bazenov za zlonamerne namene in predlagala rešitev, ki prepreči t.i sebično rudarjenje ima pa tudi velik vpliv na centralizacijo sistema [5].

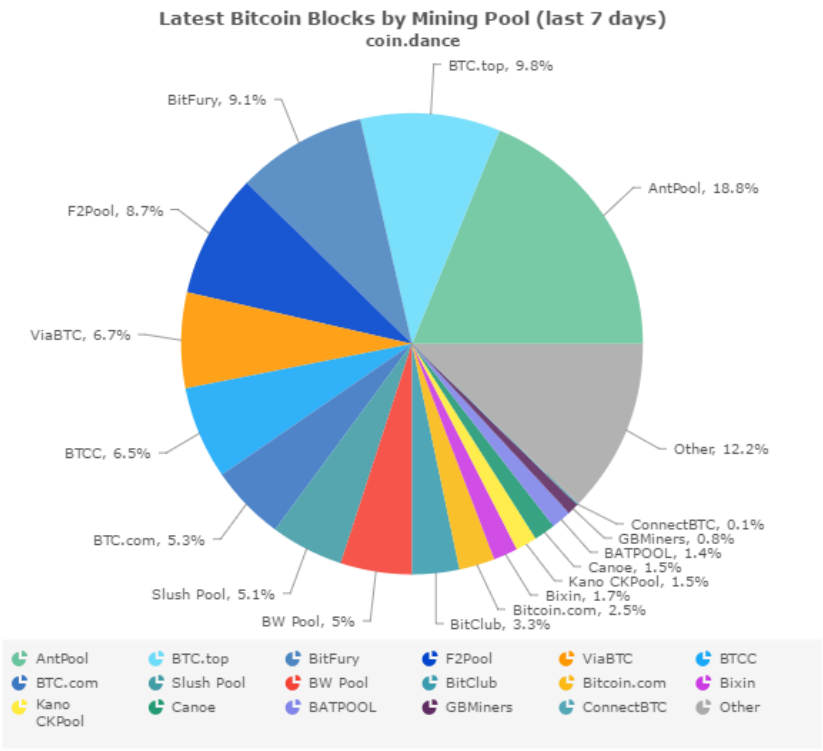
3. NAPAKE V ZASNOVI

Bitcoin je na začetku predvideval, da bo glavnina rudarjenja potekala na običajnih CPE-jih. Na žalost zasnova Bitcoin sistema pomeni, da lahko posamezniki ali skupine z več denarja z lahkoto dramatično povečajo svojo računsko moč.



Slika 1 Porazdelitev procesne moči 5.2017.[6]

Ker je rudarjenje zelo računsko zahtevno je največji strošek za rudarja običajno napajanje z elektriko. Zaradi tega sistem rudarje potiska proti bolj energetsko učinkovitim rešitvam kot je običajen CPE. Na tak način si lahko zvečajo pričakovan dobiček.
Energetsko učinkovitejše rešitve lahko razdelimo na GPE-je FPGA-je (ang. Field Programmable Gate Array) in ASIC-e (Application Specific Integrated Circuit). FPGA je bolj učinkovit kot GPE ampak dražji in isto velja za ASIC v primerjavi z FPGA. Zaradi cene razvoja in proizvodnje takšnih vezij je število enot omejeno in lastijo si jih večinoma večja rudarska podjetja ali pa jih ta celo sama razvijajo za svoje potrebe.
Ker je kot rečeno največji dolgoročni strošek rudarjenja elektrika, sistem kot tak daje prednost rudarjem ali skupinam, ki si lahko z svojo velikostjo pogajajo za nižjo ceno elektrike in pa tistim, ki se nahajajo na območjih z nižjo ceno električne energije.
Ob začetku delovanja Bitcoin-a je bila večina rudarjev posameznikov z navadnimi računalniki. S tem ko je Bitcoin postajal bolj popularen in pritegnil pogled ljudi in podjetji, ki so v rudarjenju videle zaslužek, so se takšni posamezniki bili primorani združevati v t.i. bazene (ang. pool). Tako je vsak bazen imel večjo možnost zaslužka, ki je bil potem razdeljen med člane. Dandanes je malih neodvisnih rudarjev zelo malo in tvorijo zanemarljiv delež procesne moči kar pomeni, da imajo tej bazeni veliko moči v sistemu kot je vidno na Slika 2.
Poleg prednosti, ki jih prinaša sama količina procesne moči pod kontrolo rudarja, obstajajo tudi druge lastnosti sistema, ki favorizirajo velike rudarje in podjetja. Ena od takšnih je način na katerega omenjeni bazeni upravljajo in komunicirajo z rudarji. Izkušnje kažejo, da rudarje ne zanima preveč kaj dela in kako deluje bazen katerega del so.
Pridružijo se zaradi večje verjetnosti dobička. Zaradi tega je komunikacija z vodstvom šibka kar lahko privede to tega, da se procesna moč rudarjev uporabi v zlonamerne namene. Nekateri bazeni so že presegli polovico procesne moči celotnega sistema kar je dovolj da nadzorujejo tok novih transakcij.
Bitcoin ima v protokolu predviden način glasovanja kjer rudarji v svoje dokaze dela priložijo svoj glas. To glasovanje se uporablja na primer pri izbiri nadgradenj protokola. Tu spet obstaja možnost, da vodje večjih bazenov glasujejo na način, ki ne predstavlja pravih želj članov.
Zaradi omenjenih pritiskov, ki so vgrajeni v samo strukturo sistema Bitcoin, se je velika večina procesne in glasovalne moči združila v majhnem številu rudarjev oziroma skupin rudarjev (Slika 1). Prednost imajo podjetja, ki se lahko premaknejo kamorkoli po svetu, da izkoristijo nihanja v ceni elektrike in uporabijo svojo velikost za znižanje njene cene.



Slika 2 Porazdelitev procesne moči med bazeni uporabnikov 5.2017 [7]

Ti pritiski silijo k centralizaciji kar je ravno nasprotno od začetnih ciljev projekta.

4. PREDLOGI ZA SPREMEMBE

Tukaj so predstavljeni predlogi za spremembe Bitcoin sistema in tudi spremembe izven samega sistema, ki pa na njegovo delovanje vseeno vplivajo. Rešitve so razdeljene na strojne, programske in poslovne.
Za vsak način si pogledamo kako pomemben oziroma učinkovit je pri ponovni decentralizaciji sistema, kako težaven je za implementacijo, kakšna je verjetnost za usvojitev in kakšen je vpliv tržnih sil.

4.1 Spremembe strojne opreme

4.1.1 ASIC odpornost.

V tradicionalni implementaciji Bitcoin-a se za dokazovanje dela uporablja razpršilna funkcija SHA-256. Ta funkcija je relativno enostavna za implementacijo na vezjih ASIC kar je privedlo do hitrega razvoja na tem področju po začetku delovanja kripto valut. Ker imajo do ASIC vezij dostop samo dovolj zagreti in predani rudarji je to pomenilo, da so občasni neodvisni rudarji hitro postali manjšina procesne moči.

Iz tega razloga so nekatere kripto valute poskušale razviti nov način dokazovanja dela, ki se zanaša na algoritem katerega ASIC implementacija je zelo zapletena in posledično zelo draga. S tem bi popravili neuravnoteženost moči strojne opreme. Na žalost se je ta sprememba izkazala za precej težko izvedljivo saj so obstoječi rudarji do sedaj vložili že veliko denarja v strojno opremo optimizirano za SHA-256. Ti rudarji imajo danes velik del procesne moči in s tem velik del glasovalne moči kar pomeni, da lahko blokirajo predlagane spremembe v algoritmu. Poleg tega pa zaradi glasovanja po modelu direktne demokracije večina odločitev traja zelo dolgo kar se samo še poslabša ko je govora o tako temeljni spremembi kot je zamenjava razpršilne funkcije.

4.1.2 Odprtokodna strojna oprema

Druga možnost izboljšanja decentralizacije omrežja je ta, da vsem omogočimo dostop do ASIC čipov. To lahko dosežemo tako, da načrte za takšne čipe objavimo na spletu pod kako permisivno licenco. Na tak način si lahko vsak naroči svoj čip za bolj zmerno ceno kot če bi ga kupili pri kakem večjem podjetju.
Razvoj odprtokodnih vezij ASIC bi lahko izvajali znalci sami z podporo ostalih neodvisnih rudarjev preko katere od prosto dostopnih spletnih strani za množično financiranje (*ang. crowdfunding*).
Še ena prednost tega pristopa je, da ne zahteva nikakršnih sprememb v delovanju Bitcoin sistema ali temeljnih algoritmov.

4.2 Spremembe programske opreme

4.2.1 Dokazovanje dela brez zunanjih virov

Algoritem za dokazovanje dela je možno prilagoditi tako, da ne uporablja zunanjih virov (je *ang. non-outsourceable*). Takšna sestava dokaza dela omogoča rudarjem, da anonimno ukradejo del zaslužka celotnega bazena. S tem dosežemo to, da združevanje rudarjev v bazene ni več zaželeno kar bi bolj enakomerno razporedilo procesorsko

moč. Bazeni sestavljeni iz ljudi, ki si zaupajo bi lahko še vedno obstajali ker kraja ne bi bila problem.
Problem tega pristopa je, da se zanaša na temeljne spremembe v Bitcoin sistemu kar bi s težavo prišlo skozi proces odločanja poleg tega pa bi to pomenilo da strojna oprema prirejena trenutno dominantni različici sistema ne deluje pravilno.
Izdelava takšne metode za dokazovanje dela se tudi zanaša na še ne čisto zrel seznam orodij kar bi otežilo implementacijo.

4.2.2 Sprememba komunikacije znotraj bazenov

Komunikacija med vodstvom bazena in člani je pogosto zelo osnovne narave. Samo nekateri zagrizeni člani sodelujejo v vseh aktivnostih bazena, spremljajo novice na spletni strani in so naročeni na e-pošto bazena.
Boljši model komunikacije bi spodbudil tekmovanje med bazeni za člane saj bi ti lahko, oboroženi z novimi informacijami o delovanju in politiki bazena lahko izbirali med vsemi, ki so na voljo in našli pravega. Tekmovanje bi povzročilo zmanjšanje velikih bazenov saj je prav tam največja možnost, da se nek član ne strinja z vodstvom in bi morda če bi imel več informacij želel preiti na drug bazen.
Za ta izboljšan model komunikacije bi lahko uporabili sistem potisnih sporočil (*ang. push notifications*). Tak sistem bi lahko integrirali v popularne rudarske aplikacije kot je CGMiner in bi tako večina članov dobivala redna obvestila o odločitvah bazena.
Koristne informacije, ki jih člani večinoma ne vidijo bi bile spremembe pri načinu računanja deležev, prenehanje dodajanja novih članov, spremembe v vodstvu, posodobitve protokola in drugo.

4.2.3 Samonadzor bazena

Dokazano je bilo, da ni potrebno imeti več kot polovico procesne moči, da dobimo nepravično stopnjo nadzora nad delovanjem sistema. Ittay Eyal in Emin Gun Sirer sta pokazala, da je dovolj le 25% moči, da lahko izrabimo sistem. Takemu obnašanju pravimo sebično rudarjenje (*ang. selfish mining*).
Kot možen popravek za ta problem je izpostavljena možnost spremembe programske opreme za upravljanje bazenov tako, da ta ne dovoljuje dodajanja novih uporabnikov potem, ko bazen doseže 25% moči sistema. S tem pristopom elegantno poskrbimo da sistem ostane popolnoma decentraliziran in da noben akter nima prevelike moči.
Lahko bi zahtevali, da bazeni pokažejo svojo kodo in tako preverimo, da imajo dejansko implementirano omejevanje uporabnikov ali pa bi se zanašali na prostovoljni pristop, kar pa seveda ni optimalno.

4.3 Spremembe električnega omrežja

4.3.1 Spremenljiva cena elektrike

Ena od možnih sprememb je prehod na zaračunavanje elektrike v odvisnosti od porabe. Na ta način lahko pomagamo vsem rudarjem, malim in velikim hkrati. Ne zahteva nobene večje spremembe pri rudarjih, je pa potrebno narediti spremembo pri podjetju, ki upravlja z lokalnim električnim omrežjem. Neodvisni rudarji lahko potem prilagodijo kdaj poteka rudarjenje in si tako zmanjšajo ceno procesa.
Odvisno od strukture elektrarn v omrežju bi lahko operaterji rudarjem celo plačali za to da izravnajo porabo v 24 urnem ciklu. Ta je precej predvidljiv, z vrhom enkrat popoldne, ko ljudje pridejo domov in se zmrači. Ponoči in zjutraj pa bi lahko rudarji pognali svoje naprave. Ta izravnava potrebe lahko izboljša učinkovitost delovanja določenih tipov elektrarn kot so nuklearne in premogovne ker se te le s težavo prilagajajo spremembam v potrebi po elektriki v omrežju. Če bi bila potreba bolj izravnana bi se lahko odpovedali dragim majhnim plinskim elektrarnam, ki stojijo pripravljene za takšne premike v potrebi.
Rudarjenje lahko tudi uporabi odvečno energijo iz sistemov, ki generirajo elektriko iz obnovljivih virov pa recimo niso priklopljeni na električno omrežje. Ko pride do prekomerne proizvodnje elektrike lahko višek porabimo.
Spremenljiva cena energije se hitro uveljavlja v Evropski uniji in drugje po svetu. S tem pristopom bi lahko poskrbeli, da so različne lokacije po svetu enako privlačne za rudarjenje in nobena dežela nima prevelikega deleža procesne moči. Danes recimo večina procesne moči za rudarjenje izvira iz Kitajske.

5. ZAKLJUČEK

Bitcoin sistem je bil zasnovan z namenom demokratizacije moči, ki je do sedaj bila v rokah vlad in finančnih ustanov. Kmalu pa so se realnosti sveta vtihotapile v delovanje sistema in povzročile prekomerno centralizacijo. Za ta problem obstajajo rešitve, ki pomaknejo Bitcoin nazaj proti decentralizaciji in osnovni viziji avtorjev.
Izmed predlaganih rešitev je daleč najbolj enostavna razširitev odprtokodne strojne opreme. Ta rešitev ima največjo verjetnost implementacije saj ne spremeni temeljnih pravil obstoječega sistema in ne škoduje tistim, ki so investirani v obstoječi sistem. Za to izboljšavo izgleda, da ima največjo možnost implementacije reforma cenjenja električne energije saj se ta proces izvaja v več državah iz popolnoma drugih razlogov. Spremembe, ki zahtevajo da kdo od obstoječih rudarjev ali bazenov odstopi del svoje moči v sistemu zelo verjetno ne bodo sprejete saj taki igralci upravljajo z večino procesne moči v sistemu in s tem večino glasov.
V prihodnosti bo potrebno vložiti še več truda v iskanje izboljšav v sistemu Bitcoin in podobnimi, saj se bo njihova pomembnost še povečala. Pravilno delovanje teh sistemov lahko zagotovimo samo, če ohranimo decentraliziranost za katero so zasnovani.

REFERENCES

[1] S. Nakamoto. “Bitcoin: A peer-to-peer electronic cash system” (2009).
[2] (2017) Proof of Work. Dostopno na: https://en.bitcoin.it/wiki/Proof_of_work
Dostopano: Maj 13, 2017.
[3] R. O’Dwyer. “When telcos become banks: sociotechnical control in mobile money.” Proceedings of ISIS Summit Vienna 2015—The Information Society at the Crossroads (2015).
[4] (2017) A. Poelstra. “ASICs and decentralization FAQ.” Apr. 8, 2015. . Dostopno na: https://download.wpsoftware.net/bitcoin/asic-faq.pdf
Dostopano: Maj 13, 2017.
[5] I. Eyal, E. G. Sirer. “Majority is not enough: Bitcoin mining is vulnerable.” Financial Cryptography and Data Security Lecture Notes in Computer Science (2014).
[6] “Pool statistics.” Slush Pool. . Dostopno na: https://slushpool.com/stats/
Dostopano: Maj 13, 2017.
[7] “Latest Bitcoin Blocks by Mining Pool” Coin dance Dostopno na: https://coin.dance/blocks/thisweek
Dostopano: Maj 13, 2017.

Raziskava Bitcoin bločne verige: analiza celotnega grafa uporabnikov

Jure Prevc
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana 1000
jure.prevc@gmail.com

Marko Novak Hindel
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana 1000
mn4006@student.uni-lj.si

Jernej Koželj
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana 1000
jk0108@student.uni-lj.si

POVZETEK

Bitcoin je decentralizirana kripto valuta, ki je pred kratkim dobila pozornost širšega občinstva. Zanimiva značilnost tega sistema oziroma valute je ta, da je seznam vseh transakcij ki se shranjujejo od samega nastanka valute, javno dostopen. To omogoča preiskavo gibanja sredstev za odkrivanje zanimivih lastnosti ekonomije te valute. V tem članku bomo povzeli opravljene analize omrežja Bitcoin, ki so jih predstavili v izvirnem članku [16]. Analize so narejene na verigi blokov (angleško blockchain), iz decembra 2015, ko je število transakcij eksponentno naraslo po zadnjih dveh letih. Skupek analiz, ki so opredeljene vsebuje med drugim analizo časovnega razvoja Bitcoin omrežja, preverjanje domneve *Bogato se bogati* in odkrivanje vozlišč, ki so ključnega pomena za povezljivost omrežja valute.

1 UVOD

Zaradi spletne revolucije je sedaj možno, da se katerikoli posameznik na svetu poveže s komurkoli drugim in si z njim izmenja podatke. Kljub temu ni obstajala možnost za neposredno plačilo. Iz tega razloga je bilo v zadnjem času predstavljenih več predlogov digitalnih valut.

Prva prava digitalna valuta, Bitcoin, je izšla 3. januarja 2009. Zaradi velike priljubljenosti med širšo, tudi manj strokovno javnostjo in večih pomembnih dogodkov (nihanja vrednosti, odpoved večje izmenjevalnice), lahko govorimo o pravem gospodarskem sistemu, ki ga je vredno analizirati.

Analizirani članek se nanaša na definicijo in analizo grafa transakcij v valuti Bitcoin. Sama valuta po zasnovi hrani celotno zgodovino transakcij.

V članku so predstavljeni:

- Definicija skalabilnih algoritmov gručenja za potrebe krčenja grafa transakcij.
- Pregled časovnega razvoja omrežja Bitcoin.
- Prepoznanje najbolj osrednjih vozlišč grafa.
- Potrditev domneve "bogatejši bogatijo".
- Meritev razvoja gostote bogastva.
- Analiza podgrafov, pridobljenih s filtriranjem vrednosti transakcij, večjih od meje, za naraščajoče meje.

2 PREGLED PODROČJA

V zadnjih nekaj letih se je zvrstilo kar nekaj analiz Bitcoin omrežja. Večina jih kot vhod dobi uporabniški graf, ki je pridobljen z grafa transakcij s pomočjo uveljavljene hevristike. To hevristično pravilo, ki je bilo že predstavljeno v članku [20] in podrobno razloženo v članku [25] pravi, da vsi vhodni naslovi večje transakcije pripadajo istemu uporabniku. Pravilo temelji na ugotovitvi, da je treba vsak vnos več vhodne transakcije podpisati s pravim zasebnim ključem kar pomeni, da tisti, ki podpiše pozna vse zasebne ključe transakcij in je zato lastnik vseh vhodnih naslovov. Kot rezultat dobimo graf, ki se približa pravemu uporabniškemu grafu, ker lahko hevristika podcenuje ali precenjuje skupno lastništvo nekaterih naslovov. Medtem ko se hevristično podcenjevanje pojavi zaradi tega, ker se naslovi istega lastnika niso zabeležili v isti transakciji, lahko do precenjevanja pride, ker lahko nabor uporabnikov skupaj podpiše isto transakcijo. To hevristično pravilo je bilo nato uporabljeno v večini analiz, kakor so analize v člankih [14, 15, 17, 22, 26, 27]. Izjemi sta deli [17, 27], v katerih se pojavi tudi precej bolj kompleksna hevristika na podlagi menjave naslovov, to je mehanizem, ki je uporabljen za vračilo denarja uporabniku, ki je izvršil transakcijo. V najpomembnejših analizah do leta 2012 so odkrili, da Bitcoin omrežje vsebuje ogromno manjših transakcij, kot tudi manjšo množico transakcij, pri katerih so vsote zelo velike. Analiza se je nato osredotočila na velike transakcije, da bi odkrili, kako so se takšne vsote akumulirale in kasneje razpršile. Analiza v članku [14] ne uporablja hevristike pač pa direktno analizira transakcijski graf, ki je pridobljen iz bločne verige do maja 2013. Avtorji so prepoznali začetno fazo rasti Bitcoin omrežja, označena z velikim nihanjem karakteristik omrežja in fazo trgovanja označeno s stabilnejšimi ukrepi omrežja. Ugotovili so, da prednostna navezanost povečuje rast omrežja.

Glavni cilj analize predstavljene v članku [17] je poudariti vrzel med potencialom in dejansko anonimnostjo omrežja Bitcoin. Avtorji uporabljajo prej omenjeni hevristiki na bločni verigi kot v aprilu 2013 za krčenje grafa transakcij. Kot v večini prejšnjih del, se v delu [15] obravnava stanje bločne verige kot v aprilu 2013 in izkorišča le prvo hevristiko, ki je bila prej omenjena za krčenje grafa transakcij. Avtorji kategorizirajo transakcije na podlagi pridobljenih označb (ang. tag) vsakega naslova. Prav tako pa so predstavili analizo geografske porazdelitve Bitcoin transakcij.

3 PODATKI: OMREŽJE BITCOIN

3.1 Bitcoin protokol: osnove

Uporabniki sodelujejo v Bitcoin ekonomiji s pomočjo naslova. Naslov je dvojna zgoščena vrednost (najprej algoritma SHA-256 nato pa še Ripemd-160) javnega ključa pridobljenega iz ECDSA para ključev. Naslov (in s tem javni ključ), se uporabi za uporabnikovo pošiljanje in prejemanje plačil, medtem ko se zasebni ključ uporablja za dokazilo o lastništvu. Ustvarjanje novih ECDSA parov (in

tako naslova) ni drago, in tako lahko vsak uporabnik ustvari in uporabi več naslovov. To vodi k uporabi psevdonimov, kar pomeni, da je vsak naslov alias uporabnika brez kakršne koli informacije o povezavi do uporabnika samega ali pa do naslovov, ki jih je ta uporabnik ustvaril. Ti psevdonimi so edina zaščita anonimnosti pri Bitcoin omrežju, pa še ta je šibka. Da povečamo varnost transakcije, je priporočljivo da generiramo nov naslov za vsako novo plačilo. Čeprav to ni drago v smislu računalniške zahtevnosti, pa je zahtevno nenehno generiranje novih naslovov in samim ravnanjem s takšno količino naslovov. Za izmenjavo sredstev med naslovi so ustvarili transakcije. Transakcije so lahko več vhodne in več izhodne, kar pomeni, da lahko ima transakcija več kot en vhod (naslov s katerega so prejeta sredstva) in več kot en izhod (naslov na katerega se sredstva naložijo). Transakcija v celoti prenese sredstva z vhoda na izhod. Transakcije so edini način za upravljanje s sredstvi, sredstva se lahko razdeli ali pridobi le tako, da se jih porablja. To je mogoče zato, ker transakcija vključuje naslove in ne uporabnikov. Vsak uporabnik ima lahko različne naslove, tako da lahko uporabnik uporabi transakcijo za razdeljevanje, združevanje ali prenašanje sredstev med svojimi naslovi. Transakcija lahko določi tudi prostovoljni prispevek za kritje stroškov postopka potrjevanja (to je pojasnjeno kasneje). Če je vsota vhodnih vrednosti višja kot vsota izhodnih vrednosti, se ta presežek šteje kot prostovoljni prispevek za potrjevalca. Pri transakciji je vsak izhod predstavljen kot dvojica (znesek, naslov prejemnika). Sredstva predstavljajo verige transakcij, ki prikazujejo prehod vrednosti (razdeljevanje in združevanje) med naslovi, potrjenimi na vsakem koraku s podpisom ključa prejšnega lastnika. V Bitcoin omrežju transakcije predstavljajo celotno stanje sistema.

Nove transakcije, ki jih generira katerikoli uporabnik so prikazane kot sporočilo na P2P Bitcoin omrežju. Opaziti je tudi da obstaja posebna vrsta transakcij imenovanih coinbase, ki omogočajo ustvarjanje nove vrednosti. Te posebne transakcije nimajo vhodnih naslovov, temveč samo izhodne naslove, na katere se nakaže skovana vrednost. Pri vsaki transakciji je vsak vhod podpisan s strani uporabnika in privatnega ključa naslova, ki bo sredstva porabil. Ta digitalni podpis garantira, da lahko le pravi lastnik porablja sredstva ampak ne preprečuje, da bi se uporabil več kot enkrat v različnih transakcijah. To je tako imenovan problem dvakratne porabe. Rešitev tega je, da se spominjamo celotne zgodovine vseh transakcij, da določimo dejanskega lastnika sklada v vsakem danem trenutku. Zgodovina se vzdržuje v porazdeljeni bazi podatkov, imenovani bločna veriga, ker so transakcije razvrščene v blokih, povezanih v verigo, samo povezovanje med bloki pa se doseže s shranjevanjem zgoščene vrednosti zaglavja prejšnjega bloka v zaglavju naslednjega bloka. Da je vsako zaglavje bloka odvisno od vseh transakcij iz tega bloka, je v zaglavju tudi koren Merklovega drevesa [18]. Potrebno je doseči potrebno porazdeljeno soglasje za izbiro bloka za dodajanje k verigi, saj bi lahko prišlo do nezdružljive transakcije povročene s problemom dvakratne porabe. Protokol porazdeljenega soglasja uporabljen v Bitcoin se imenuje Nakamoto soglasje in se opira na HashCash Proof-of-Works [8]. To Nakamotovo soglasje je en izmed najbolj zanimivih vidikov vendar ga ne opisujejo bolj podrobno, saj to presega obseg tega članka.

3.2 Izgradnja grafa

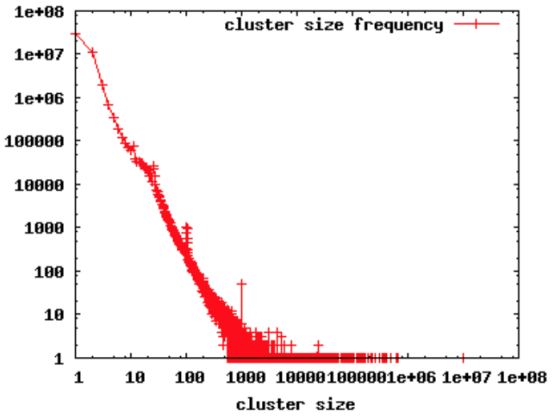
Bitcoin nabor podatkov lahko formalno zmodeliramo z uteženim usmerjenim hipergrafom H=(A,T), pri čemer je A množica vseh naslovov, T je množica transakcij, ki se lahko oblikuje kot sklop urejenih parov (A1, A2) z A1, A2 podmnožici A, kar pomeni, da naslovi v A1 plačujejo naslovom v A2. Še več, vsaki transakciji s=(A1,A2), ki je element T povežemo:

- Časovni žig, ki pove, kdaj je transakcija potekala.
- Porazdelitev zneskov med vozlišči v A2 označimo kot bs. Bolj formalno, bs je funkcija, ki na vsak a, ki je element A2 poveže večkratno množico v R. Tako se obvesti, da lahko pride do večkratnih transakcij.
- Znesek prostovoljnega prispevka za potrjevanje transakcije.

Kot je razvidno v prejšnjem poglavju v Bitcoin omrežju vsak uporabnik nadzira različne psevdonimne naslove. Da bi prepoznali uporabnike, želimo združiti vse naslove, ki jih isti uporabnik uporablja.

3.3 Statistika gručenja

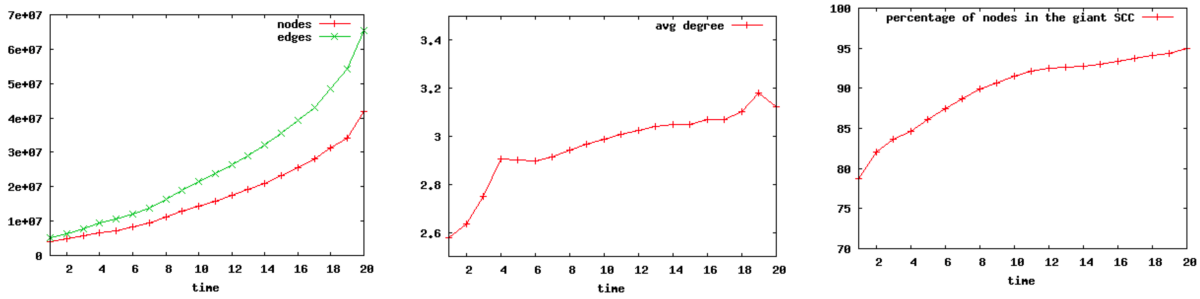
Na sliki 2 vidimo porazdelitev velikosti gruč. Vidimo, da porazdelitev sledi potenčnemu zakonu.



Slika 2: Porazdelitev velikosti gruč

V tabeli 1 je prikazano nekaj osnovne statistike podatkov, kot sta skupno število naslovov in skupno število transakcij. Prikazana je tudi statistika postopka gručenja.

V spodnjem delu tabele 1 je seznam največjih najdenih gruč. Opazimo, da je velikost teh gruč za nekaj velikostnih redov večja od povprečne velikosti gruč, zaradi česar rep porazdelitve velikosti gruč močno oddebeljen (slika 2). Videli bomo, da je več gruč iz tega seznama največjih deset tudi v seznamu največjih deset glede na druge topološke mere središčnosti.



(a) Št. vozlišč in povezav skozi čas

(b) Povprečna izhodna stopnja skozi čas

(c) Odstotek vozlišč v največji močno povezani komponenti

Slika 1: Evolucija omrežja G^t

Število naslovov, t.j. $\|A\|$	113 221 083
Število transkakcij, t. j. $\|T\|$	99 602 440
Število gruč, t.j. vozlišč G	46 144 246
Število usmerjenih povezav G	294 705 549

10 največjih gruč		
Id gruče	identiteta	velikost
66 482	Mt. Gox	10 216 380
2 899 325	LocalBitcoins.com	676 402
26 784 111	GoCoin.com	611 885
11 032 019	AgoraMarket	497 995
12 388 597	EvolutionMarket	420 632
2 477 299	N/A	392 589
2 547 597	SilkRoadMarketplace	372 753
10 072 646	SilkRoad2Market	349 874
1 175 285	BTC-e.com1	348 438
11 828 673	999Dice.com	301 990

Tabela 1: Statistika gručenja

3.4 Pridobitev podatkov

Za pridobitev verige blokov je dovolj, da vzpostavimo Bitcoin vozlišče in pričnemo zahtevati bloke ostalih vozlišč. Ker je ta postopek lahko zelo dolgotrajen, je bila uporabljena posneta veriga blokov, shranjena v formatu Protocol Buffers[3]. Veriga blokov vseh prvih 389 800 blokov, od začetnega bloka (Genesis), do bloka višine 389 799, torej vsebuje vse transakcije od 2009-01- 03 18:15:05 GMT do 2015-12-23 09:40:52 GMT.

Protokol Bitcoin uporablja skladovni skriptni jezik. Te skripte so (večinoma) uporabljene za opis pogojev, pod katerimi lahko odkupimo sredstva te transakcije. Najpogostejši primer takega pogoja je podpis. Ko se transakcija preverja, se vhodne skripte združijo z izhodnimi in preverijo, da se lahko transakcija izvrši, se morajo uspešno preveriti vse skripte. Skripte so lahko poljubno kompleskne, a se v praksi uprablja le nekaj standardiziranih skript, imenovanih *standardne* skripte. Še pomembneje je, da vozlišča sprejmejo ne-*standardne* skripte, a jih ne posredujejo naprej, torej je verjetnost, da se ne-*standardna* skripta znajde v verigi blokov zelo majhna. Najpogosteje uporabljene *standardne* skripte so Pay to PubKey Hash (p2pkh), Pay to PubKey (p2pk), Pay to Script Hash (p2sh) in Pay to Multisig (p2ms). V izvornem članku so obdelali le skripte tipov p2pkh, p2pk, p2sh, izhode ostalih transakcijskih skript pa so zavrgli. S tem so poenostavili razčlenjevalnik. Na koncu je bilo uspešno obdelanih 295 144 677 skript in neuspešno 1 489 903 skript, s čimer je bila pokritost 99, 4977%.

Iz zgoraj opisane razčlenitve verige blokov je bila sestavljena zbirka podatkov o transakcijah, nad katero se je izvajal algoritem gručenja in analiza. Podatki o naslovih so bili pridobljeni iz javnih zbrik naslovov [2, 4].

4 METODE

4.1 Analiza povezanosti skozi čas

Avtorji izvirnega članka so za potrebe analize povezanosti omrežji G^t predstavili kot enostaven graf brez večkratnih povezav.

4.1.1 Evolucija omrežja. Gre za pristop kjer opazujemo kako se omrežje razvija skozi čas. Za vsako časovno enoto t lahko opazujemo:

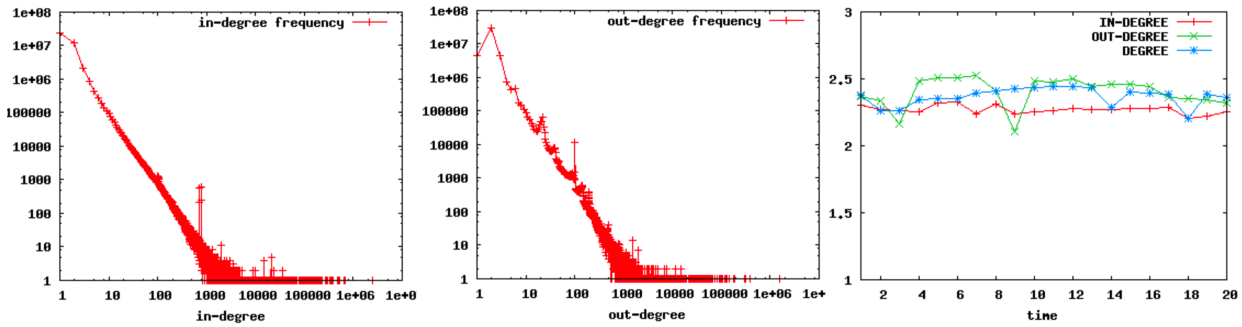
- Število vozlišč in povezav
- Povprečna vhodna ali izhodnja stopnja vozlišč. V dotičnem primeru vhodna stopnja predstavlja število prejetih nakazil, izhodna stopnja pa število izdanih nakazil.
- Odstotek vozlišč v največji krepko povezani komponenti. Krepko povezano komponento predstavljajo vozlišča iz katerih so dosegljiva vsa vozlišča znotraj komponente.

4.1.2 Fenomen malega sveta. Čeprav gre pri analizi omrežij običajno za velika omrežja (t.j veliko število vozlišč) lahko za večino rečemo, da gre za omrežja malega sveta. Zanimiv eksperiment [29] sega v leto 1967, kjer so pokazali, da je v povprečju potrebnih le 5 ljudi, da vsak posameznik doseže kogarkoli v Združenih državah Amerike. Fenomen malega sveta lahko dokažemo s povprečno razdaljo med posameznimi vozlišči in nakopičenostjo [5, 30, 31].

Povprečna razdalje med vozlišči je definirana kot:

$$\bar{d} = \frac{1}{n(n-1)} \sum_{i \neq j} d_{ij} \tag{1}$$

kjer je n število vozlišč in d_{ij} najkrajša razdalja med vozliščema i in j.



(a) Porazdelitev vhodne stopnje G^t pri $t = 20$ **(b) Porazdelitev izhodne stopnje G^t pri $t = 20$** **(c) Eksponent potenčnega zakona skozi čas**

Slika 3: Porazdelitev stopenj vozlišč

Nakopičenost za posamezno vozlišče je definirana kot:

$$C_i = \frac{2t_i}{k_i(k_i - 1)} \tag{2}$$

kjer je t_i število trikotnikov, ki jih sklene vozlišče i, in pa k_i kar predstavlja stopnjo vozlišča. Iz nakopičenosti za posamezno vozlišče trivialno izračunamo povprečno nakopičenost:

$$\bar{C} = \frac{1}{n}\sum_i C_i \tag{3}$$

Avtorji izvirnega članka so poleg povrečne razdalje izračunali tudi premer omrežja, ki ga lahko zapišemo kot:

$$d_{max} = \max(d_{ij}) \tag{4}$$

Na tem mestu bi izpostavili, da v izvirnem članku nismo zasledili izračuna povprečne nakopičenosti. Za potrditev hipoteze, da gre za omrežje malega sveta je potrebno izračunati tako povprečno razdaljo kot tudi nakopičenost.

4.1.3 Porazdelitev stopenj vozlišč. Pri tej vrsti analize želimo opazovati kako se porazdeljujejo stopnje posameznih vozlišč. Kot stopnje lahko obravnavamo vhodno, izhodno ali skupno stopnjo vozlišč. Običajno opazujemo ali porazdelitev sledi potenčnemu zakonu [23] t.j. veliko vozlišč z nizko stopnjo in nekaj žarišč z zelo visoko stopnjo (žarišča) oziroma sledi binomski porazdelitvi, kjer so stopnje vozlišč blizu povprečja.

4.2 Analiza središčnosti.

Pri analizi središčnosti opazujemo pomembnost posameznih vozlišč oziroma odgovarjamo na vprašanje "Katero vozlišče je najbolj pomembno". Posameznih mer za središčnost je več, ki jih lahko razdelimo med 5 skupin:

(1) Središčnost glede na koeficiente gručenja [7, 28, 31]
(2) Središčnost glede na razdalje [9, 10, 21]
(3) Spektralna analiza Središčnosti [6, 11, 13]
(4) Središčnost glede na fragmente [12, 19, 24]
(5) Središčnost glede na stopnjo

V izvirnem članku so se osredotočili na naslednje mere za analizo središčnosti:

- Harmonična središčnost (2. kategorija)

$$h_i = \sum_j \frac{1}{d_{ij}} \tag{5}$$

- Središčnost glede na stopnjo (skupna, vhodna in izhodna stopnja) (5. kategorija)

4.3 Bogato se bogati

Definicija bogatosti v izvirnem članku je definirana kot: uporabnik (vozlišče) je bogat, če je njegovo stanje oziroma število prejetih transakcij visoko glede na ostale uporabnike. Avtorji so preverjali naslednje lastnosti:

(1) Najbogatejši uporabniki v času t so bogatejši kot najbogatejši uporabniki v času $t^{'} < t$.
(2) Najbogatejši uporabniki v času t strmijo k ohranitvi bogastva v času $t^{'} > t$.
(3) Bogastvo se koncentrira skozi čas.

Za podan utežen multigraf $G^t = (V^t, E^t, \omega^t)$ in ϕ^t so za vsako vozlišče $u \in V^t$ njegovo bogastvo definirali kot $b^t(u)$ oziroma število prejetih transakcij $d_t(u)$ po naslednjih formulah:

$$b^t(u) = \sum_{(v,u)\in E^t} \omega(v,u) - \sum_{(v,u)\in E^t} \omega(u,v) - \phi^t(u) + \beta^t(u). \tag{6}$$

$$d^t(u) = |\{(v,u) : (v,u) \in E^t\}| \tag{7}$$

V enačbi 6 upoštevajo tudi $\phi^t(u)$, katero plača uporabnik u in pa $\beta^t(u)$, ki je vrednost iz transakcij, ki nastopijo kot nagrada pri odkritju novega bloka.

Za celoštevilski k so definirali množici D_k^t in B_k^t, v katerih je vsebovanih k vozlišč z najvišjim d^t oziroma b^t.

Zgoraj opisane lastnosti so preverjali po enačbah:

(1)

$$r_b^t = \frac{\sum_{u\in B_k^t} b^t(u)/k}{\sum_{u\in V^t} b^t(u)/|V^t|} \tag{8}$$

$$r_d^t = \frac{\sum_{u\in D_k^t} d^t(u)/k}{\sum_{u\in V^t} d^t(u)/|V^t|} \tag{9}$$

kjer je r_b^t (oziroma vzporedno r_d^t) razmerje med povprečnim stanjem na računih (oziroma številom prejetih transakcij) najvišjih k vozlišč in povprečjem ostalih vozlišč v omrežju.

(2)

$$u_b^t = |\cup_{i=1}^t B_k^i| \tag{10}$$

$$u_d^t = |\cup_{i=1}^t D_k^i| \tag{11}$$

(3)

$$h_b^t = min\left\{k : \frac{\sum_{u\in B_k^t} b^t(u)}{\sum_{u\in V_k^t} b^t(u)} > r\right\}/|V^t| \tag{12}$$

$$h_d^t = min\left\{k : \frac{\sum_{u\in B_k^t} d^t(u)}{\sum_{u\in V_k^t} d^t(u)} > r\right\}/|V^t| \tag{13}$$

kjer je h_b^t najmanjši k pri katerem B_k^t poseduje $r * 100\%$ bogastva celotnega omrežja. Vzporedno enako velja tudi za h_d^t v obliki prejetih transakcij. Majhna h_b^t in h_d^t pomenita, da je bogastvo skoncentrirano le na nekaj vozlišč (oziroma uporabnikov).

5 REZULTATI

V nadaljevanju so predstavljeni rezultati izvirnega članka. Zaradi razsežnosti problema oziroma velikosti podatkov vseh transakcij (135.47 GB dne 10.5.2017 [1]) in časovne omejitve se za ponovitev analize nismo odločili.

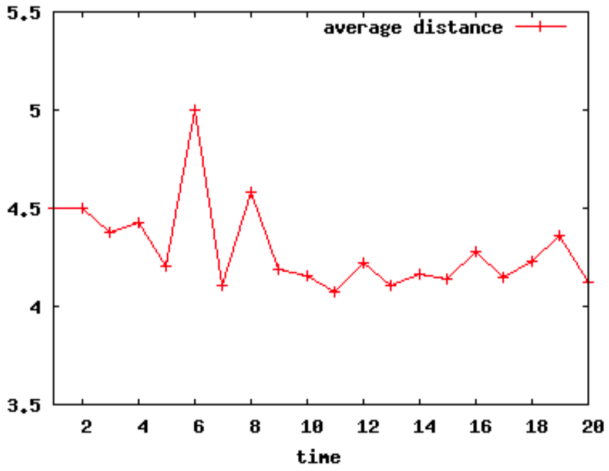
5.1 Povezanost skozi čas

5.1.1 Evolucija omrežja. Evolucija omrežja je prikazana na sliki 1. Graf 1a prikazuje rast vozlišč in povezav. Posnetki omrežja pri danem t so enakomerno porazdeljeni zato lahko na 1a opazujemo tudi, da vozlišča in povezave rastejo hitreje kot linearno, kar pomeni, da iz dneva v dan Bitcoin uporablja več uporabnikov. Graf 1b prikazuje rast izhodne stopnje, kar izpostavi dejstvo, da hitreje nastajajo nove povezave kot nova vozlišča. Iz grafa 1c lahko opazujemo, da je omrežje vedno bolj povezano in iz tega zaključimo, da postaja vedno bolj robustno.

5.1.2 Fenomen malega sveta. Avtorji navajajo visok premer omrežja (t.j 2050). Povprečne razdalje so pričakovano kratke. Na grafu 4 lahko vidimo spreminjanje povprečne razdalje skozi čas.

Kot smo že omenili v 4.1.2, bi bilo potrebno za potrditev fenomena malega sveta izračunati še povprečno nakopičenost, zato se v tem delu zaradi primankljaja rezultatov ne moremo strinjati z izvirnimi avtorji. Pri fenomenu malega sveta so povprečne razdalje kratke, povprečna nakopičenost pa $\bar{C} \gg 0$.

5.1.3 Porazdelitev stopenj vozlišč. Avtorji ugotavljajo, da porazdelitev stopenj sledi eksponentnemu zakonu. Porazdelitve vhodnih in izhodnih stopenj pri času $t = 20$ sta na slikah 3a in 3b. Tako kot izvirni avtorji opazimo določene anomalije (npr. špica pri vhodni stopnji 1000) v porazdelitvi katerih si ne znamo razlagati. Eksponent potenčnega zakona (graf 3c) skozi čas ostaja približno konstanten, kar pomeni, da se tudi porazdelitev vozlišč ostaja podobna skozi čas.



Slika 4: Spreminjanje povprečne razdalje skozi čas

5.2 Središčnosti

V tabeli 2 so zbrani rezultati vozlišč z najvišjimi stopnjami. Na najvišjem mestu prevladuje menjalnica Mt.Gox, ki je do propada leta 2014 veljala za največjo menjalnico value Bitocin. Pričakovano glede na visoko skupnjo stopnjo prevladuje tako na prejetih in izhodnih transakcijah (vhodna in izhodna stopnja). Med zanimivimi vozlišči v tej tabeli lahko vidimo tudi SilkRoadMarketplace, ki je širši javnosti znan kot tržnica z zakonom prepovedanimi (večinoma) dobrinami.

Vozlišča, ki so uvrščena najvišje glede na harmonično središčnost lahko gledamo kot vozlišča preko katerih poteka največ povezav med ostalimi vozlišči. Rezultati Harmonične središčnosti so v tabeli 3. Ponovno prevladuje Mt.Gox. Ker so podatki do trenutka pisanja tega članka stari slabi 2 leti, sama valuta pa pridobiva na popularnosti, bi se ob ponovni analizi rezultati spremenili.

Vozlišča, ki so označena s številkami v času analize ni bilo moč identificirati.

5.3 Bogato se bogati

Na grafu 5 so prikazani rezultati preverjanja hipotez opisanih v poglavju 4.3.

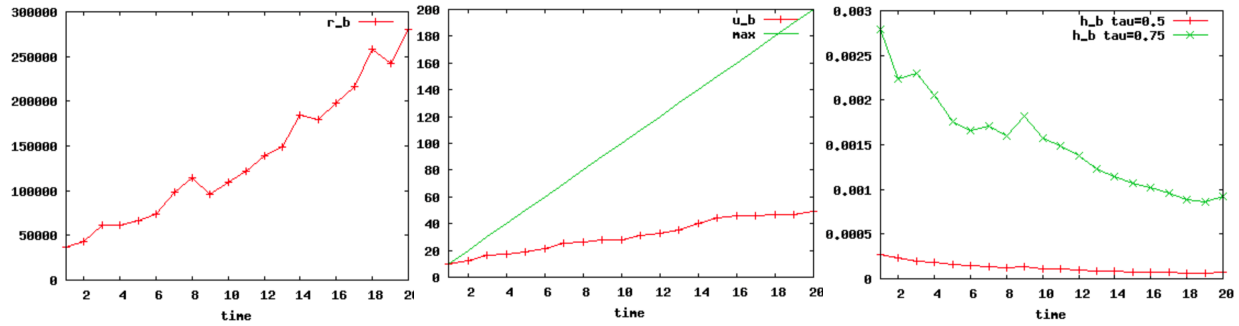
Podgrafa 5a in 5d kažeta na rast količin r_b^t in r_d^t, kar nakazuje na potrditev hipoteze 1 iz 4.3.

Podgrafa 5b in 5e potrjujeta hipotezo 2 iz 4.3. Zelena premica nakazuje, kakšno bi bilo najvišje gibanje vrednosti u_b^t in u_d^t, če bi prihajalo do sprememb med najbogatejšimi uporabniki.

Pri potrjevanju hipoteze 3 iz 4.3 lahko na grafu 5c opazujemo padanje vrednosti h_b^t tako pri $\tau = 0.5$ in $\tau = 0.75$. To pomeni, da s časom število vozlišč, ki vsebujejo $\tau * 100\%$ bogastva pada. Obratno pa za prejete transakcije iz grafa 5f tega ne opazujemo. Zdi se, kot da vrednost h_d^t skozi čas narašča. Izvirni avtorji so podali razlog v prehitrem naraščanju števila povezav, ki bi jih lahko enak odstotek vozlišč prejel.

Tabela 2: Središčnosti glede na stopnje vozlišč

	Vozlišče	Stopnja	Vozlišče	Vhodna stopnja	Vozlišče	Izhodna stopnja
1	Mt. Gox	3 386 581	Mt. Gox	2 452 049	Mt. Gox	1 591 319
2	LocalBitcoins.com	902 151	BTC-e.com1	683 875	2477299	381 426
3	2477299	848 176	LocalBitcoins.com	650 269	FaucetBOX.com	317 742
4	BTC-e.com1	740 402	AgoraMarket	636 969	LocalBitcoins.com	301 692
5	AgoraMarket	722 331	SilkRoadMarketplace	527 718	14782788	191 867
6	SilkRoadMarketplace	577 124	2477299	511 239	MoonBit.co.in	180 161
7	BitPay.com1	500 990	BitPay.com1	493 067	3454364	178 349
8	BTC-e.com2	492 219	BTC-e.com2	479 452	26638073	176 508
9	Cryptsy.com	461 111	BitPay.com2	394 447	Cryptsy.com	148 015
10	BitPay.com2	401 254	Cryptsy.com	361 298	23144512	146 624



(a) r_b^t skozi čas

(b) u_b^t skozi čas in teoretičen maksimum

(c) h_b^t skozi čas; $\tau = 0.5$ in $\tau = 0.75$

(d) r_d^t skozi čas

(e) u_d^t skozi čas in teoretičen maksimum

(f) h_d^t skozi čas; $\tau = 0.5$ in $\tau = 0.75$

Slika 5: Preverjanje lastnosti navedene v odstavku 4.3

Tabela 3: Harmonična središčnost

	Vozlišče	Harmonična središčnost
1	Mt. Gox	11 798 171
2	2477299	10 447 302
3	LocalBitcoins.com	10 320 862
4	Cex.io	10 144 968
5	FaucetBOX.com	10 136 604
6	26638073	10 071 881
7	MoonBit.co.in	10 065 853
8	19860816	10 025 701
9	Poloniex.com	9 976 766
10	Bittrex.com	9 926 321

6 ZAKLJUČEK IN NADALJNJE DELO

V tem članku je predstavljen nabor analiz uporabniškega grafa Bitcoin omrežja. Avtorji so analize predstavili na bločni verigi iz decembra 2015, ki vsebuje okoli 100 milijonov transakcij, ki imajo več izhodnih in več vhodnih naslovov. Za podporo gradnji uporabniškega grafa iz tako ogromne količine podatkov, so definirali razširljiv algoritem na podlagi gručenja. Analize kažejo vrsto zanimivih lastnosti Bitcoin omrežja, tako kot lastnost *bogato se bogati* in obstoj osrednjih vozlišč, ki delujejo kot privilegirani mostovi med različnimi deli omrežja. Ugotovili so tudi, da je velikost oziroma obseg Bitcoin omrežja veliko večji od omrežja socialnih omrežij, ter da so vidni vrhovi pri porazdelitvi nekaterih posebnih vrednostih. V prihodnjem delu načrtujejo raziskati vedenje uporabnikov, ki

privede do takšnih značilnosti. Načrtujejo tudi razširiti analizo, da bi izpostavili odnos med strukturo uporabnikov grafa in drugimi zanimivimi lastnostmi ekonomije Bitcoin, na primer, kako se zazna špekulante z analizo grafa uporabnikov.

LITERATURA

[1] *Bitcoin, Litecoin, Namecoin, Dogecoin, Peercoin, Ethereum stats.* https://bitinfocharts.com/.

[2] *Block chain info tags.* https://blockchain.info/tags.

[3] *Protocolbuffers.* https://developers.google.com/protocol-buffers/.

[4] *Wallet Explorer.* https://www.walletexplorer.com/.

[5] Alex Bavelas. 1950. Communication patterns in task-oriented groups. *The Journal of the Acoustical Society of America* 22, 6 (1950), 725–730.

[6] Phillip Bonacich. 1987. Power and centrality: A family of measures. *American journal of sociology* 92, 5 (1987), 1170–1182.

[7] Wouter De Nooy, Andrej Mrvar, and Vladimir Batagelj. 2011. *Exploratory social network analysis with Pajek*. Vol. 27. Cambridge University Press.

[8] Cynthia Dwork and Moni Naor. 1992. Pricing via processing or combatting junk mail. In *Annual International Cryptology Conference*. Springer, 139–147.

[9] Linton C Freeman. 1977. A set of measures of centrality based on betweenness. *Sociometry* (1977), 35–41.

[10] Linton C Freeman, Stephen P Borgatti, and Douglas R White. 1991. Centrality in valued graphs: A measure of betweenness based on network flow. *Social networks* 13, 2 (1991), 141–154.

[11] S Grin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual Web search engine. *Computer networks and ISDN systems* 30, 1-7 (1998), 107–117.

[12] Pablo Jensen, Matteo Morini, Márton Karsai, Tommaso Venturini, Alessandro Vespignani, Mathieu Jacomy, Jean-Philippe Cointet, Pierre Mercklé, and Eric Fleury. 2015. Detecting global bridges in networks. *Journal of Complex Networks* (2015), cnv022.

[13] Leo Katz. 1953. A new status index derived from sociometric analysis. *Psychometrika* 18, 1 (1953), 39–43.

[14] Dániel Kondor, Márton Pósfai, István Csabai, and Gábor Vattay. 2014. Do the rich get richer? An empirical analysis of the Bitcoin transaction network. *PloS one* 9, 2 (2014), e86197.

[15] Matthias Lischke and Benjamin Fabian. 2016. Analyzing the bitcoin network: The first four years. *Future Internet* 8, 1 (2016), 7.

[16] Damiano Di Francesco Maesa, Andrea Marino, and Laura Ricci. 2016. Uncovering the Bitcoin Blockchain: An Analysis of the Full Users Graph. In *Data Science and Advanced Analytics (DSAA), 2016 IEEE International Conference on*. IEEE, 537–546.

[17] Sarah Meiklejohn, Marjori Pomarole, Grant Jordan, Kirill Levchenko, Damon McCoy, Geoffrey M Voelker, and Stefan Savage. 2013. A fistful of bitcoins: characterizing payments among men with no names. In *Proceedings of the 2013 conference on Internet measurement conference*. ACM, 127–140.

[18] Ralph Merkle. 2006. A digital signature based on a conventional encryption function. In *Advances in Cryptology—CRYPTO'87*. Springer, 369–378.

[19] Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. 2002. Network motifs: simple building blocks of complex networks. *Science* 298, 5594 (2002), 824–827.

[20] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system. (2008).

[21] Mark EJ Newman. 2005. A measure of betweenness centrality based on random walks. *Social networks* 27, 1 (2005), 39–54.

[22] Micha Ober, Stefan Katzenbeisser, and Kay Hamacher. 2013. Structure and anonymity of the bitcoin transaction graph. *Future internet* 5, 2 (2013), 237–250.

[23] de Solla D. J. Price. 1965. *Networks of scientific papers*. Vol. 194. Science. 510–515 pages.

[24] Nataša Pržulj. 2007. Biological network comparison using graphlet degree distribution. *Bioinformatics* 23, 2 (2007), e177–e183.

[25] Fergal Reid and Martin Harrigan. 2013. An analysis of anonymity in the bitcoin system. In *Security and privacy in social networks*. Springer, 197–223.

[26] Dorit Ron and Adi Shamir. 2013. Quantitative analysis of the full bitcoin transaction graph. In *International Conference on Financial Cryptography and Data Security*. Springer, 6–24.

[27] Ahmad-Reza Sadeghi. 2013. *Financial Cryptography and Data Security: 17th International Conference, FC 2013, Okinawa, Japan, April 1-5, 2013, Revised Selected Papers*. Vol. 7859. Springer.

[28] Sara Nadiv Soffer and Alexei Vazquez. 2005. Network clustering coefficient without degree-correlation biases. *Physical Review E* 71, 5 (2005), 057101.

[29] Jeffrey Travers and Stanley Milgram. 1967. The small world problem. *Phychology Today* 1 (1967), 61–67.

[30] Duncan J Watts. 1999. *Small worlds: the dynamics of networks between order and randomness*. Princeton university press.

[31] Duncan J Watts and Steven H Strogatz. 1998. Collective dynamics of 'small-world'networks. *nature* 393, 6684 (1998), 440–442.

BitConeView: Vizualizacija pretoka Bitcoin transakcij

Jaka Klančar
Univerza v Ljubljani
Fakulteta za Računalništvo in Informatiko
1000 Ljubljana
jk7808@student.uni-lj.si

Jakob Košir
Univerza v Ljubljani
Fakulteta za Računalništvo in Informatiko
1000 Ljubljana
jk6818@student.uni-lj.si

Matic Novak
Univerza v Ljubljani
Fakulteta za Računalništvo in Informatiko
1000 Ljubljana
mn9499@student.uni-lj.si

POVZETEK

Bitcoin je digitalna valuta, katere transakcije so shranjene v javno dostopnih zapisih. Te zapise imenujemo blockchain in si jih lahko predstavljamo kot ogromen usmerjen graf z več kot 70 milijoni vozlišč, kjer vsako vozlišče predstavlja transakcijo in vsaka povezava predstavlja bitcoine, ki se pretakajo med transakcijami. V poročilu je opisano orodje za vizualizacijsko analizo, ki nam pove, kdaj in kako se tok bitcoina meša z drugimi tokovi v transakcijskem grafu. Ta sistem temelji na prispodobah, s katerimi prikažemo velikost in ostale lastnosti transakcij in s tem omogoča visokonivojsko analizo drugače nepreglednih podatkov.

Ključne besede

Bitcoin, kriptovaluta, vizualizacija toka, vizualna analiza, detekcija prevar

1. UVOD

Zadnja leta smo priča strmi rasti bitcoina, novega tipa digitalne valute. Gre za virtualno valuto, ki je shranjena in se izmenjuje izključno v digitalni obliki. Ključno se od vseh prejšnjih e-valut razlikuje v tem, da je neodvisen od zaupanja vrednih institucij, kot so banke oziroma vlada. Namesto tega je zasnovan na odprto socialnem modelu zaupanja, ki vzpodbuja sodelovanje.

Po njegovi prvi fazi, ko so ga poznali zgolj nekateri entuziasti, je bitcoin pridobil ogromno na njegovi popularnosti, saj ga je moč uporabljati kjer koli, kjer se nahaja internetna povezava. Veliko število trgovcev danes podpira bitcoin kot plačilno sredstvo in različne menjalnice ponujajo menjavo le-tega za tradicionalne valute. Trenutna vrednost vseh bitcoinov v obtoku je čez US $3B.

Bitcoin je privaten in hkrati transparenten. Evidentiranje vseh transakcij v javni knjigi zagotavlja transparentnost. Integriteta transakcij je zagotovljena s široko sprejetimi kriptografskimi metodami. Zasebnost pa se zagotavlja z "neprosojnimi" kriptografskimi identifikatorji, ki prikrijejo dejansko identiteto, ki stoji za njimi. Anonimnost, ki jo ponuja bitcoin, zadovoljuje potrebe ljudi, ki so ozaveščeni o pomembnosti zasebnosti. Po drugi strani pa lahko bitcoin izkoristimo za plačilo nelegalnih dobrin in storitev.

Bitcoinova javna knjiga, imenovana "blockchain", je velik, povezan, acikličen graf, ki shranjuje vse transakcije, ki so se kadarkoli zgodile. Vsako vozlišče predstavlja transakcijo, ki je nekakšen vsebnik bitcoina. Povezava od transakcije t do transakcije t' je označena s številom bitcoinov, ki so bili preneseni od t do t'. Blockchain ponuja širok nabor informacij o uporabi bitcoina. Lahko ga uporabimo za sledenje toka bitcoinov skozi čas, ali pa za iskanje vzorcev v transakcijah, kot so prisotnost verig, dreves, ali več stopenjskih vozlišč. Z nekaj spretnosti ga lahko uporabimo za razkritje identitete pošiljatelja sumljive transakcije. S tem ciljamo predvsem na transakcije, katerih namen je plačilo ilegalnih storitev. Iz istega razloga so se pojavile spletne storitve, ki sumljive transakcije pomešajo s "čistimi" bitcoini in tako zakrijejo sledi.

V članku predstavljamo BitConeView, ki je orodje za vizualizacijo toka bitcoinov. BitConeView omogoča grafično sledenje porabi bitcoinov z določenega vira (vhodi transakcije) skozi čas in kako so, po možnosti, shranjeni v več "koritih" (neporabljeni izhodi transakcije). BitConeView je prvo grafično orodje za analizo tokov v blockchainu. Lahko ga uporabimo za iskanje vzorcev v tokovih oziroma poteh. Hitro in enostavno lahko opazimo akumulacije, distribucije in mešanje bitnih kovancev.

BitConeView je bil narejen z namenom zadovoljitve določenih potreb. Za primer vzemimo, da je začetna transakcija sestavljena iz M bitcoinov. Uporabnik bi moral imeti v vsakem trenutku možnost vizualnega pogleda na svojo transakcijo. Bolj natančno:

- R1:
  - sledenje M-tim kovancem in njihovi porabi skozi čas
  - razumevanje, kako je M kovancev zmešanih skupaj z drugimi bitcoini
  - vpogled v še neporabljene dele M v določenem trenutku
- R2: pridobivanje jasne informacije o procentu mešanja M kovancev z drugimi bitcoini in kako ta odstotek

varira skozi čas

- R3: pregledovanje možnih menjav bitcoinov med entitetami v grafu transakcij
- R4: podpora odkrivanju anomalij v vzorcih mešanja M kovancev

Na kratko predstavimo strukturo članka. Osnovni koncepti bitcoin transakcij in grafi le-teh so predstavljeni v drugem poglavju. V tretjem poglavju opisujemo principe vizualizacij, grafičnega vmesnika, uporabniških scenarijev in sistemske arhitekture BitConeViewa. Sledijo še povezana dela in zaključek.

2. BITCOIN VALUTA

V tem poglavju predstavljamo poenostavljen opis bitcoin protokola z namenom definiranja grafa transakcij. Bitcoin je valuta, ki se shranjuje in izmenjuje zgolj v digitalni obliki. Definirana je z odprtimi standardi in vzdrževana z omrežjem "peer-to-peer", imenovanim "omrežje bitcoin". Bitcoin se prenaša med uporabniki v obliki transakcij. Vse transakcije so shranjene v javni knjigi, imenovani blockchain. Kopije le-te so shranjene v vsakem vozlišču grafa. Vsaka nova transakcija se propagira skozi omrežje, potrjena je s strani vsakega vozlišča in sčasoma je dodana v blockchain.

Transakcija (v nadaljevanju tx) t je sestavljena iz množice vhodov $i_t{}^1$, ..., $i_t{}^h$ in množice izhodov $o_t{}^1$, ..., $o_t{}^k$. Vsak je povezan s svojim kriptografskim identifikatorjem, imenovanim "naslov", in s številom bitcoinev. Te zneske označujemo z $A(i_t{}^1)$, ..., $A(i_t{}^h)$ in $A(o_t{}^1)$, ..., $A(o_t{}^k)$ zaporedoma. Tx t prenese bitcoine od vhodov do izhodov. Vsota vhodnih zneskov je večja ali enaka vsoti izhodnih zneskov. V primeru neenakosti, je razlika, imenovana strošek transakcije, implicitno prenesena k vzdrževalcem bitcoin omrežja. Ker so stroški transakcije izjemno nizki, v nadaljevanju članka predpostavimo, da sta vsoti vhodov in izhodov enaki.

Izhode iz tx označimo txo. V določenem trenutku T, je lahko vsak txo transakcije tx t neporabljen ($utxo$) ali porabljen ($stxo$). Edini način, da porabimo $utxo$ o_t t-ja je, da ga uporabimo kot vhod $i_{t'}$ neke druge transakcije tx t' (in ne iste). Zneska $A(i_{t'})$ in $A(o_t)$ naj bi bila enaka. Ko je $uxto$ porabljen, postane $stxo$. Na množico vseh $utxo - ov$ vseh transakcij tx v trenutku T lahko pogledamo kot na množico bitcoinev v obtoku v času T.

Definiramo usmerjen graf, imenovan "graf bitcoin transakcij (tx graf)" v določenem trenutku. Vozlišča so $tx - i$. Vsako vozlišče je opremljeno s svojimi vhodi in izhodi. Vozlišči t in t' sta povezani z usmerjeno povezavo (t, t'), če je en izhod o_t $t - ja$ uporabljen kot vhod $i_{t'}$ $t' - ja$. Z $A(t, t')$ označimo število bitcoinev v o_t in $i_{t'}$. Bolj natančno, tx graf je multigraf, ker lahko več izhodov $t - ja$ ustreza vhodom t'. Zaradi enostavnosti se bomo na tovrsten graf sklicevali brez predpone "multi".

Blockchain je razdeljen na strani, imenovane "bloki". Vsak blok v grobem vsebuje $tx - e$, izdane v časovnem intervalu dolžine 10 minut. Višina bloka predstavlja sekvenčno številko bloka. Na blok se torej sklicujemo na podlagi njegove višine.

Četudi so vsi $tx - i$ javni, identitete ljudi, ki uporabijo izhode $tx - a$, niso. V bistvu je naslov kriptografska zgoščena vrednost javnega ključa, zato lastnik pripadajočega zasebnega ključa ostane anonimen. Za boljše zagotavljanje anonimnosti, uporabnik ustvari enega ali več novih naslovov za vsak nov tx.

Junija 2015 je bil blockchain sestavljen iz približno 360.000 blokov in je vseboval 83 milijonov $tx - ov$. To je število vozlišč v grafu tx. V povprečju ima vsak tx dva vhoda, vendar graf vsebuje razvejane podgrafe. V grobem je v blockchain vsak dan dodanih približno 100.000 $tx - ov$.

3. VIZUALIZACIJA PRETOKA

V tem poglavju so opisane osnove orodja **BitConeView**, uporabniški vmesnik skupaj s primeri uporabe in tehnične podrobnosti arhitekture.

3.1 Osnove BitConeView orodja

BitConeView omogoča pregleden prikaz grafa tx, pri čemer so nekateri manj pomembni deli skriti, na zahtevo pa jih lahko prikažemo.

BitCone je osnovna enota za prikaz transakcij in prenosa bitcoinov, ki jo uporablja **BitConeView**. Ta prikazuje kako se bitcoini transakcije s mešajo z ostalimi bitcoini skozi čas. Vsak **BitCone** se povečuje, ko se bitcoini mešajo z že vsebovanimi. Gledano z vidika grafa tx, je **BitCone** transakcije s podgraf, ki je dosegljiv iz s v nekem časovnem intervalu. Prikaz omejimo na časovni interval, saj ni smiselno prikazovati celotne zgodovine.

Čas je prikazan diskretno, BitConeView ga prikaže kot višino bloka neke transakcije. Oblika vsakega bloka je odvisna od začetne transakcije s in časa T, ki je enak časovnemu intervalu, za katerega vizualiziramo podatke. Pravimo, da blok pripada **BitConu**, če vsebuje vsaj eno od v njem vsebovanih transakcij.

Utxi so izhodni kovanci iz **BitCona**, ki ob opazovanem času T še niso bili porabljeni. Gledano s perspektive grafa tx (grafa transakciji) so to povezave v njem, kjer ob času T še ni prišlo do transakcij z njimi, ali pa niso del grafa.

Vhodi. BitCone prav tako prikazuje vnos novonastalih kovancev. Ti so prikazani kot vhodne povezave v **BitCone.**

3.2 BitConeView analitika

Orodje podpira več različnih vrst analiz toka kovancev v grafu transakcij.

Analiza proračuna. Za določen **BitCone** določimo proračun in skupno vsoto kovancev, ki vanj vstopijo. Orodje nam potem prikaže te vhode, torej kovance, ki se mešajo v opazovan BitCone v času t.

Analiza Utxov. Za vsak blok b, ki pripada nekemu **BitConu**, določimo $utxos(b)$ kot vsoto kovancev, ki pripadajo transakcijam znotraj bloka b. Ti kovanci ustrezajo izhodnim transakcijam, ki so se zgodile po času T.

Analiza čistosti. Čistost je mera mešanja kovancev v bloku transakcij z zunanjimi kovanci, natančneje jo bomo

definirali v poglavju 3.4. Za prikaz količine opazovanih kovancev vhodnih transakcij, ki so bili mešani z ostalimi, je prikaz **BitCona** obogaten s prikazom čistosti skozi čas.

Analiza prenosov. Za vsak par vhodne (s) in izhodne (u) transakcije lahko prikažemo maksimalno količino kovancev iz s, ki jih lahko prenesemo v u. To izračunamo kot maksimalen pretok v grafu transakcij, z izvorom v s in ponorom v u, kjer je pretok vsake povezave e enak $A(e)$.

3.3 Uporabniški vmesnik

V vmesniku je en **BitCone** prikazan kot na Sliki 1, kjer je s izvorna transakcija. Čas teče od vrha proti dnu. Vsak blok (tj. transakcija) je označen kot pravokotnik, katerega vrh prikazuje višino bloka. Njegova širina in postavitev kažeta izvor kovancev, ki so v tej transakciji vstopili v **BitCone** in njihovo količino. V vsakem bloku je zgornji del pol temno sivih pravokotnikov (ti prikazujejo vsiljivce) ali barvnimi pravokotniki, ki kažejo izvor kovancev, ki vstopajo v BitCone. Spodnji del kaže izhodne transakcije. Temno sivi pravokotniki označujejo kovance, porabljene do časa T, črni pa neporabljene.

Možno je prikazati tudi BitCone, katerega kovanci izvirajo iz več izvornih transakcij. V tem primeru so bloki prvega BitCona (tj. tistega, ki mu pripada časovno gledano prva transakcija) lahko tudi BitConi, glej Sliko 2.

3.4 Definicija čistosti

V tem poglavju formalno definiramo čistost, koncept, ki se v bitcoin skupnosti pogosto uporablja za ločevanje kovancev dveh izvornih transakcij.

Ko skupina bitcoinov vstopi v transakcijo, ni natančno definirano, kako se kovanci iz različnih izvorov porazdelijo med izhodi transakcije. Na Sliki 3 recimo ne vemo, kateri od kovancev v o_t je prišel iz i_1 in kateri iz i_2.

Zaradi tega predvidimo enakomerno razporeditev, torej v Sliki 3 predpostavimo, da se je iz vsake od zgornjih vej polovica kovancev razporedila v spodnjo levo, polovica pa v spodnjo desno. Iz tega sledi definicija čistosti kot količina kovancev, ki vstopi v transakcijo iz izvora s, kot delež vseh kovancev, ki vstopajo v transakcijo.

3.5 Primeri uporabe

Pogledali si bomo dva primera uporabe. V prvem bo Alenka poglobila svoje razumevanje toka bitcoin kovancev, v drugem pa bo Brane opravil eksperiment, povezan z odkrivanjem pranja denarja.

Alenka se poglobi. Alenko zanima tok kovancev, ki izvirajo iz neke transakcije s. V bitcoin svetu ima vsaka transakcija svojo zgoščeno vrednost (hash), mnogo orodij omogoča prikaz teh transakcij. Ampak Alenko zanima, kaj se je dogajalo s kovanci te transakcije v nekem časovnem obdobju T, recimo en teden po transakciji. Alenka zato v orodju BitConeView izbere transakcijo glede na *hash* vrednosti in nastavi čas T.

Alenka takoj opazi, da je bil velik delež kovancev del drugih transakcij. Po širini BitCona lahko sklepa, da ob nekem času t_1 v BitCone vstopi veliko zunanjih kovancev. Alenko zanima, če je opazila aktivnost, ki je sumljivo podobna pranju denarja. Da preveri to hipotezo, si ogleda krivuljo čistosti, ki pa pokaže, da se le majhen delež novih kovancev meša s kovanci v BitConu.

Ampak Alenki se tok iz s_1 še vedno zdi zelo sumljiv. Dejstvo, da ni našla, kar je iskala, je ne ustavi. Zato nastavi čas T na dva tedna. Tako najde čas t_2, v katerem v BitCone spet vstopi veliko novih kovancev, takrat pa se tudi čistost zaradi mešanja kovancev močno zmanjša.

Iz tretje roke izve tudi, da se kovanci iz transakcije s_3 mešajo s temi v s_1, zato izbere prikaz obeh transakcij. V tem prikazu čistost vseskozi ostaja visoka. Alenka zaključi, da je tok med transakcijama pričakovan in ni prišlo do ničesar sumljivega.

Brane opravi eksperiment. Brane bo ocenil učinkovitost orodij za pranje denarja, *BitLaundry* in *BitcoinFog*. Najprej v obe storitvi vstavi nekaj svojih kovancev. Po nekem času jih dvigne in poskuša iz izvornih transakcij ugotoviti začetno. Brane si najprej ogleda *BitLaundry* in v orodju BitConeView prikaže transakcijo, s katero je v storitev vstavil kovance. Opazi, da so bili v roku približno 10 ur kovanci mešani z veliko količino ostalih kovancev, kar se seveda odraža v velikem upadu čistosti. V tem primeru bi bilo težko odkriti začetno transakcijo.

Brane si nato ogleda še *BitcoinFog* in odkrije zanimiv vzorec. Kovanci, ki jih storitev izplača, so pogosto del verig zelo velikih transakcij. Brane zato v BitConeViewu prikaže te transakcije. V začetku teh transakcij pride do združevanja 5 velikih vhodnih vej, v skupni vrednosti 6000 BTC z 279 ostalimi vhodi, skupaj 50,000 BTC. Vsi kovanci iz teh transakcij se v celotnem toku ne mešajo z ostalimi.

3.6 Sistemska arhitektura in prototip orodja

Podatke orodje pridobi iz grafa transakcij, ki je vsebovan v blockchainu in zato dostopen vsem. Dodatno uporablja še podatke iz *Chain.com*, ki omogoča sprehajanje po grafu transakcij naprej.

BitConeView je spletna aplikacija. Na strežniški strani uporablja Python in Flask, da pridobi podatke in poganja algoritem za izračun maksimalnih pretokov. Za sam prikaz rezultata pa skrbi uporabnikov spletni brskalnik. Ta temelji na grafični knjižnici D3.js. Prototip aplikacije je že javno objavljen.

4. POVEZANA DELA

Kljub temu, da je transakcijski graf idealen za vizualizacijo in analizo mrež transakcij, pa obstaja le malo programov, ki omogočajo vizualen pregled bitcoin transakcij.

CoinAnalytics JARVIS [6] je mišljen kot orodje za temeljito raziskavo po blockchainu. Njegov grafični uporabniški vmesnik nam ponuja pregled in analizo razmerij ter pregled transakcijskega grafa, ki je obogaten z dodatnimi podatki (naslovi). Ne ponuja pa nobene funkcionalnosti za pomoč raziskovalcu pri preiskovanju pranja denarja s pomočjo bitcoina (pošiljanje denarja preko tretje osebe).

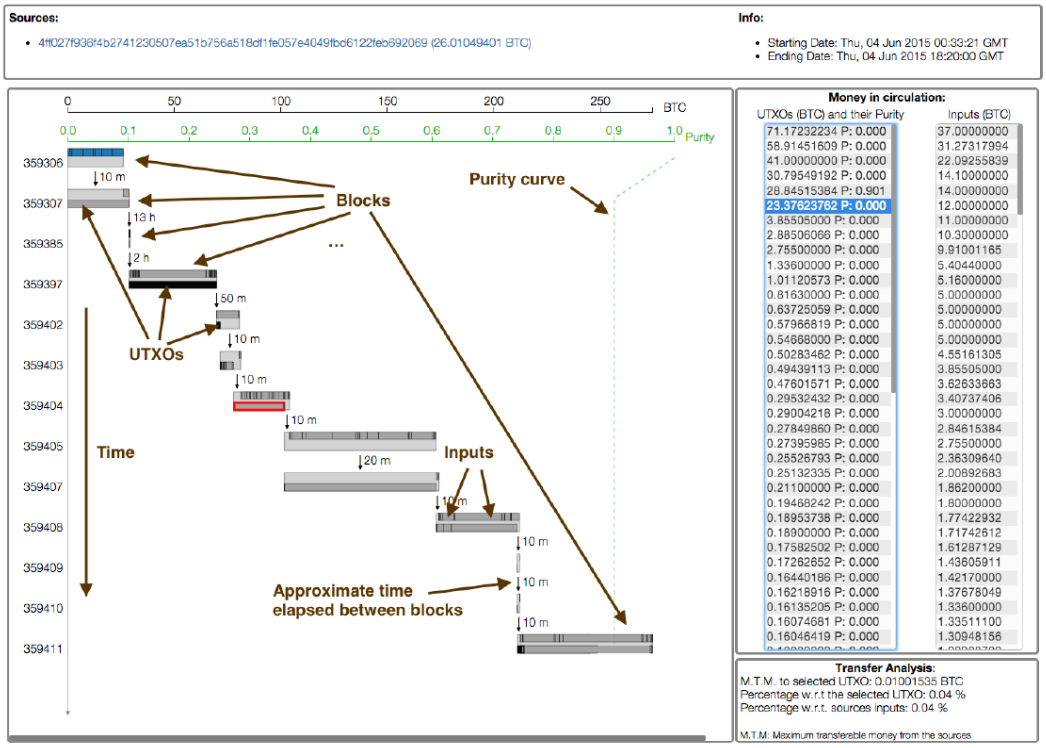


Figure 1: Prikaz enega BitCona

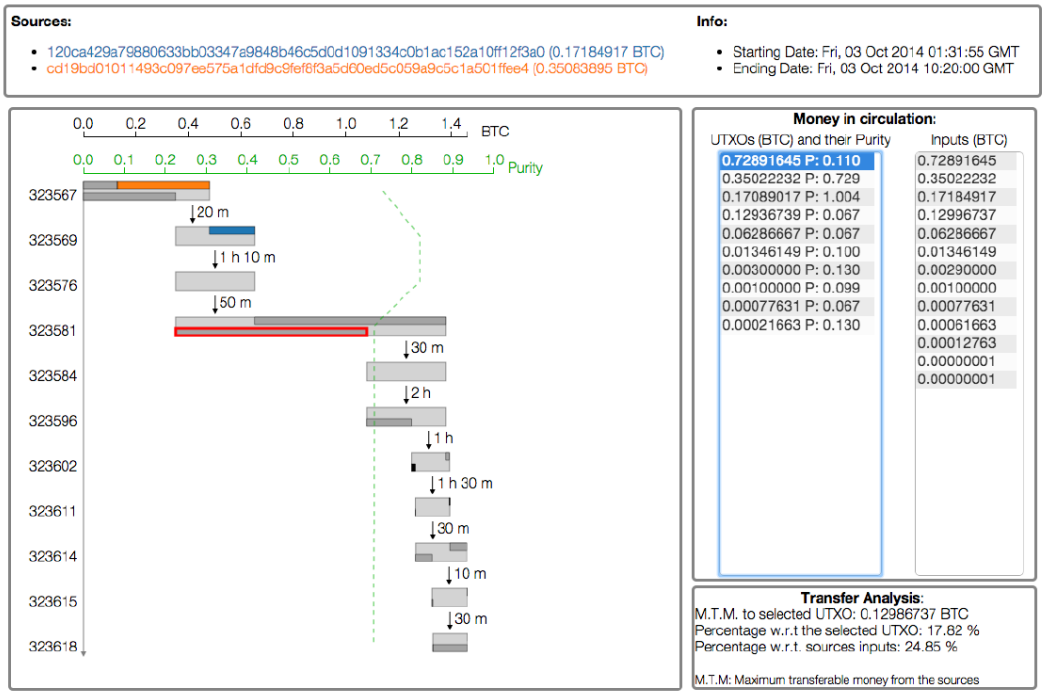


Figure 2: Prikaz več BitConov

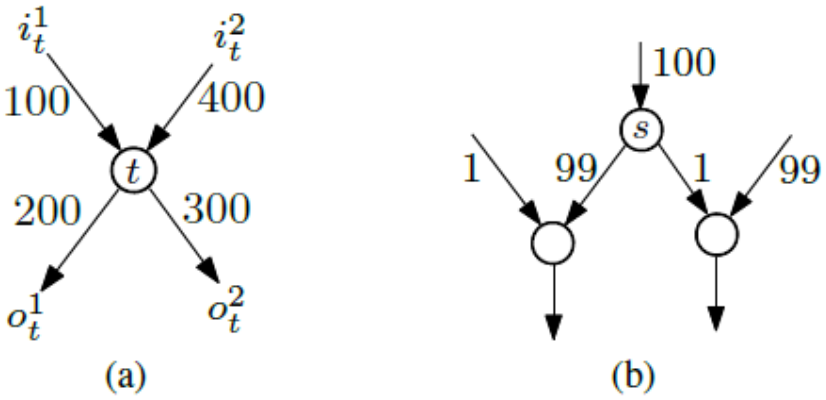


Figure 3: Prikaz transakcij, ni mogoče določiti točne razporeditve kovancev.

CoinViz [1] je finančni vizualizator za transakcije bitcoina. Omogoča prikaz transakciji v realnem času. Za vsako transakcijo lahko vidimo naslov pošiljatelja in prejemnika. To orodje žal ne ponuja nič uporabnega za analizo pretoka bitcoina in je namenjeno izključeno v izobraževalne namene. Zelo podobno orodje je **bitcoin-tx-graph-visualizer** [7].

Druga orodja, na primer **Blockchain.info** [3] in **blockr.io** [4], ponujajo uporabniku možnost sprehoda skozi graf transakcij in možnost prikaza različnih grafov iz finančnih podatkov bitcoina.

Več znanstvenih člankov [8, 9, 10] analizira graf transakcij. Vključene so slike, kjer so našli zanimive oziroma sumljive podgrafe. V literaturi preiskovanja finančnih prevar predvsem izstopa članek [5], kjer opisujejo sistem za vizualno analizo, ki pomaga pri odkrivanju sumljivih bančnih transakcij. Uporabniku omogoča številna vizualna orodja.

5. ZAKLJUČEK

Predstavili smo BitConeView, sistem za vizualno analizo pretoka bitcoina. Sistem vsebuje orodje, s katerim lahko opazujemo čistost bitcoinov, s čimer hitro opazimo, kdaj je prišlo do sumljivega mešanja. Uporabnost sistema BitConeView je bila prikazana skozi nekaj eksperimentov na temo pranja denarja. Sistem je bil analiziran tudi s študijo, kjer je devet ljudi, med katerimi sta bila dva detektiva za finančne zločine, podalo povratno informacijo glede uporabe sistema. BitConeView se še vedno razvija, zato lahko v prihodnosti pričakujemo še vrsto uporabnih dodatnih orodij.

6. LITERATURA

[1] Coin viz. `http://people.ischool.berkeley.edu/~shaun/infoviz/bitcoin/index.html`. Dostopano: 12.5.2017.

[2] G. D. Battista, V. D. Donato, M. Patrignani, M. Pizzonia, V. Roselli, and R. Tamassia. Bitconeview: visualization of flows in the bitcoin transaction graph. In *2015 IEEE Symposium on Visualization for Cyber Security (VizSec)*, pages 1–8, Oct 2015.

[3] Blockchain. Blockchain.info. `https://blockchain.info/`. Dostopano: 12.5.2017.

[4] blockr. blockr.io. `http://blockr.io/`. Dostopano: 12.5.2017.

[5] R. Chang, M. Ghoniem, R. Kosara, W. Ribarsky, J. Yang, E. Suma, C. Ziemkiewicz, D. Kern, and A. Sudjianto. Wirevis: Visualization of categorical, time-varying data from financial transactions. In *2007 IEEE Symposium on Visual Analytics Science and Technology*, pages 155–162, Oct 2007.

[6] Coinalytics. Jarvis. `http://coinalytics.co/jarvis.html`. Dostopano: 12.4.2017.

[7] W. Lu. bitcoin-tx-graph-visualizer. `https://www.npmjs.com/package/bitcoin-tx-graph-visualizer`. Dostopano: 12.5.2017.

[8] S. Meiklejohn, M. Pomarole, G. Jordan, K. Levchenko, D. McCoy, G. M. Voelker, and S. Savage. A fistful of bitcoins: Characterizing payments among men with no names. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, IMC '13, pages 127–140, New York, NY, USA, 2013. ACM.

[9] F. Reid and M. Harrigan. *An Analysis of Anonymity in the Bitcoin System*, pages 197–223. Springer New York, New York, NY, 2013.

[10] D. Ron and A. Shamir. *Quantitative Analysis of the Full Bitcoin Transaction Graph*, pages 6–24. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

Bitcoin napad s prikrivanjem blokov: Analiza in ublažitev napada

Povzetek članka

Anej Budihna
Univerza v Ljubljani, Fakulteta za računalništvo in informatiko

Luka Golinar
Univerza v Ljubljani, Fakulteta za računalništvo in informatiko

Matjaž Glumac
Univerza v Ljubljani, Fakulteta za računalništvo in informatiko

POVZETEK

Bitcoin je prva kriptovaluta, ki še danes prevladuje v popularnosti in količini uporabe. V tem članku je obravnavana varnostna luknja v obstoječi shemi sistema Bitcoin, ki omogoča, izvajanje napadov s prikrivanjem blokov (block witholding attack - BWA). Ta napad se izvaja nad rudarskimi bazeni (mining pool) in ima lahko velike posledice tako za člane bazena kot tudi za celoten Bitcoin sistem. Avtorji so raziskali nekaj posebnih različic tega napada in poskušali ugotoviti dobiček, ki ga pridobi napadalec. Predlagali so tudi nekaj načinov za preprečitev tega napada, ki se razlikujejo obstoječih predlaganih rešitev. Namesto odkrivanja ali zmanjšanja motivacije za napad so se avtorji odločili za pristop, ki popolnoma izniči zmožnosti izvajanja takšnega napada, s pomočjo kriptografskih in računskih metod.

Ključne besede

Bitcoin rudarjenje, napad z bločnim prikrivanjem, sebičen rudar, rudarske skupine, zavezujoče sheme

1. UVOD

Bitcoin je popularna kriptovaluta, ki jo je prvi predlagal Satoshi Nakamoto leta 2008 [1]. Vse izvedene transakcije se hranijo v verigi blokov (blockchain). Veriga blokov je javno dostopna in preverljiva knjiga nakazil na kateri temelji celotno omrežje Bitcoin. Veriga zagotavlja, da je mogoče trošiti samo bitcoine, ki so dejansko v lasti plačnika in omogoča preverjanje stanj denarnic. Vsak blok je sestavljen iz več nakazil, nakazilo pa predstavlja prenos bitcoinov iz ene denarnice v drugo. Vsako nakazilo mora biti podpisano s skrivnim ključem, ki ga hrani denarnica. Polek avtentikacije podpis zagotavlja da nakazila, ki so vedno javna, ni mogoče kasneje spreminjati. Informacija o novem nakazilu oziroma transakciji je vedno poslana vsem zainteresiranim članom omrežja.

Zato, da je veriga blokov vredna zaupanja je potrebno vsako transakcijo overiti s potrditvijo omrežja, temu procesu pravimo rudarjenje (mining). Rudarjenje je razpršen sistem doseganja soglasja, ki ga izvajajo člani omrežja. Rudarji nakazila zberejo in jih zapakirajo v bloke, ki morajo zadoščati zelo strogim pravilom šifriranja, ki ji preverja omrežje. Bloke se overi tako, da se izračuna njihovo hash vrednost, zato da rudar ustreže zahtevam omrežja mora bloku transakcij dodati žeton (nonce). Iskanje te vrednosti predstavlja glavnino dela, ki jo opravlja rudar. Ko rudar najde pravo vrednost objavi dokaz in ga skupaj z blokom doda v verigo. Za najdbo pravega žetona je nagrajen z 25 novo ustvarjenimi bitcoini. Ta nagrada služi kot motivacija za izvajanje overjanja, z regulacijo pravil šifriranja pa se uravnava dotok novih bitcoinov v omrežje. Rudar v blok transakcij vključi transakcijo nagrade v svojo denarnico, kar onemogoča možnost da bi kdo ukradel njegovo opravljeno delo. Pri tem je treba poudariti, da je rudarjenje "tekmovalna loterija", kar pomeni, da rudarji tekmujejo med seboj, saj nagrado dobi samo tisti, ki prvi reši uganko, in da je izid tekmovanja močno odvisen od sreče.

Ker rudarjenje temelji na metodi grobe sile in je zelo računsko zahtevno opravilo se rudarji pogosto združujejo v rudarske bazene (mining pool). V takšnih bazenih administratorji porazdeljujejo delo tako, da rudarjem pošljejo podpisan blok transakcij z razponom žetonov, ki jih morajo raziskati. Rudarji pa izvajajo izračune in periodično pošiljajo preverjene žetone ter izračunane hash vrednosti, ki predstavljajo delni dokaz o opravljenem delu (partial proof of work). Izračun teh delnih dokazov je veliko lažji od iskanja polnega dokaza, ki zagotavlja nagrado. Vsakič, ko rudar poskuša najti delni dokaz ima možnost najdbe polnega dokaza, ki ga lahko posreduje administratorju. Administrator nato vknjiži blok in pobere nagrado, ki jo nato porazdeli med sodelujoče rudarje. Delež s katerim je nagrajen vsak rudar je odvisen od količine delnih dokazov, ki jih pošlje administratorju med rudarjenjem. Administratorjev podpis v bloku zagotavlja, da noben izmed rudarjev ne more nagrado pobrati zase.

Toda, ko eden izmed članov bazena najde poln dokaz ima tudi opcijo, da dokaza ne posreduje administratorju. Rudar lahko pošilja vse delne dokaze in zakrije polne dokaze, čeprav nagrade ne more zahtevati zase, lahko na ta način od bazena dobiva deleže nagrade na račun drugih članov, kljub temu, da omrežju ne doprinaša nobenih koristi. Takšen napad je poimenovan napad s prikrivanjem blokov(block witholding attack)[5],[6]. Trenutno je takšen napad nemogoče zaznati ali preprečiti.

1.1 Napad s prikrivanjem blokov

Napad s prikrivanjem blokov je bil v prvo predlagan leta 2011 [5], ko so raziskovalci hkrati dokazali, da lahko napadalec na ta način služi tako da zlorablja računsko moč poštenih rudarjev za lasten dobiček. Napadalčev dobiček izvira tudi iz zmanjšanja verjetnosti zmage bazena, ki je žrtev napada. Na ta način se namreč poveča verjetnost, da bo napadalec našel poln dokaz na bloku transakcij s svojim podpisom. Podobno se lahko več članov bazena združi in skupaj napade konkurenčen bazen. Eden izmed največjih takšnih napadov je bil zabeležen junija 2014, zaradi katerega so pošteni člani bazena Eligius zabeležili 300 bitcoinov izgube.

V tem članku so avtorji raziskali podoben scenarij, s tem da so se osredotočili na dve variaciji napada. V prvem primeru so raziskali količino dobička, ki jo ustvari napadalec, ki izvede napad s prikrivanjem blokov (BWH) na izbran bazen in je za napad nagrajen s strani drugega bazena. V drugem primeru je model spremenjen tako, da napadalec porazdeli svojo računsko moč in napade oba bazena. Napadalec lahko celo prejme nagrado s strani obeh dveh bazenov. Takšne napade avtorji poimenujejo "sponzorirani napad s prikrivanjem blokov". Na ta način napadalec z deležem svojih računskih moči poveča možnost zmage za naročnika napada in zase, če ima dodatna sredstva za rudarjenje. Ta ranljivost Bitcoin sistema močno vpliva na njegovo delovanje, zato avtorji v članku predlagajo nekaj rešitev, ki so jih sami zasnovali.

1.2 Pregled področja

Obstaja že več člankov na to temo. En izmed takih je bil posvečen analizi primera, kjer se dva identična bazena napadata med seboj [7]. Izkazalo se je, da v takem primeru oba bazena zaslužita manj kakor bi če napada ne bi izvedla. V drugem članku so avtorji preverili več scenarijev in dokazali, da ta napad prepričljivo poveča dobiček napadalca [8]. V prvem članku [7] pa je obravnavan primer, kjer določen bazen pošlje svoje člane, da napade konkurenčen bazen. V tem primeru napadalci delijo svoj dobiček z ostalimi člani bazena.

V tem delu avtorji predstavijo nov napad poimenovan "sponzorirani napad s prikrivanjem blokov". V tem napadu napadalec poveča možnosti zmage vseh drugih rudarjev in bazenov. Zato lahko vzpostavi dogovor z drugim bazenom kjer je nagrajen za zmanjšanje verjetnosti zmage bazena, ki je tarča napada. Dogovor veli, da je napadalec poplačan sorazmerno z dobičkom, ki ga pomaga ustvariti bazenu s katerim sodeluje. Posebnost tega napada v primerjavi tistimi opisanimi v drugih člankih je ta, da tukaj napadalec deluje samostojno, torej ni član drugih bazenov in ves dobiček obdrži zase.

Motivacija za izdelavo tega članka izvira iz tega povečanega dobička. Članek je razdeljen v dva dela. V prvem avtorji analizirajo dobiček napadalca in analizirajo metode za maksimiranje tega dobička. Pokažejo tudi, da je ob določenih pogojih izvajanje napada bolj dobičkonosno od poštenega rudarjenja. Na koncu avtorji predlagajo še nekaj sprememb in izboljšav trenutnega sistema rudarjenja s katerimi bi popolnoma izkoreninili to ranljivost sistema.

2. ANALIZA NAPADA

Varnostna shema (Commitment scheme)

Varnostno shemo C sestavljajo tri verjetnostni algoritmi, ki se izvedejo v polinomskem času:

1. $C.Setup()$: Za varnostni parameter k vrne javni parameter CK.
2. $C.Commit()$: Sprejme bitni niz x in vrne par $(com, decom)$.
3. $C.Open()$: Sprejme com in $decom$, vrne pa vrednost x ali napako.

Pri obravnavi izbranih primerov sponzoriranih napadov, so avtorji postavili nekaj predpostavk. Predpostavili so, da je v bazenih samo en administrator, ki je zadolžen z koordinacijo in računanjem svojih delnih dokazov o delu. Administrator določi nabor transakcij in druge parametre, člani bazena pa poskušajo ustvariti delni dokaz o delu glede na te parametre. Administrator določi tudi zahtevnostni nivo delih dokazov, ki mu jih rudarji morajo periodično pošiljati za overjanje. Nivo zahtevnosti delnih dokazov mora bit dovolj nizek, zato da ne preobremeni administratorja. Administrator preverja vse dokaze dokler ne najde ustreznega s katerim lahko zahteva nagrado. Predpostavili so tudi, da je administrator pošten in ga ni mogoče podkupiti.

Napad je opisan na sledeči način. Avtorji enačijo računsko moč celotnega Bitcoin sistema z 1. Obravnavajo dva bazena P in P', z računskimi močmi p in p', ter napadalca $\mathcal{A}$ z računsko močjo α. Napadalec porazdeli svojo računsko moč na dva dela, prvi del uporabi za zasebno samostojno rudarjenje, drugi del pa za izvajanje napada na bazen. Napadalec se pridruži bazenu P, ki je tarča napada, kjer se pretvarja da je pošten rudar in administratorju periodično pošilja delne dokaze o opravljenem delu. V primeru, da napadalec najde poln dokaz, ta dokaz zadrži in ga ne posreduje bazenu ali Bitcoin sistemu. Napadalec poln dokaz posreduje bazenu P', ki je konkurent bazenu P. Bazen P' ima skrivni dogovor z napadalcem $\mathcal{A}$. Vsakič ko napadalec zadrži poln dokaz mu bazen P' ponudi nagrado. Bazen P' lahko zlahka preveri, če je blok, ki ga je posredoval napadalec pristen, če prevzamemo da P' pozna nabor transakcij v bloku. Ker avtorji predpostavijo, da je P javno dostopen bazen je povsem smiselno, da lahko bazen P' dostopa do teh podatkov. Nagrada, ki jo napadalec dobi od bazena P' je odvisna od pričakovanega dodatnega dobička, ki ga ustvari bazen P'. Na ta način napadalec in bazen P' oba imata dodaten dobiček na račun poštenih rudarjev bazena P.

3. UBLAŽITEV NAPADA

Napad s prikrivanjem blokov je lahko poguben za vse javno dostopne bazene rudarjev, ki omogočajo včlanitev nezanesljivih rudarjev. V času nastanka članka ni obstajal nobena dobra zaščita pred tem tipom napadov. V sorodnem delu je bila je opisana metoda za odkrivanje takih napadov. Administrator bi ustvaril opravilo za člana bazena, za katero ve da je rešitev poln dokaz o opravljenem delu [5]. V primeru, da rudar ne posreduje rešitve se tako ujame v past. Slabost tega pristopa je ta, da od administratorja in poštenih rudarjev zahteva trošenje sredstev, ki bi jih lahko namenili rudarjenju. V drugem sorodnem delu, pa je kot rešitev predlagana dodatna nagrada za rudarja, ki najde poln dokaz

[12]. Avtorji v članku predlagajo enostavno spremembo obstoječe sheme rudarjenja, ki bi preprečila napad tako s strani članov bazena kot tudi administratorja bazena.

3.1 Predlagana shema

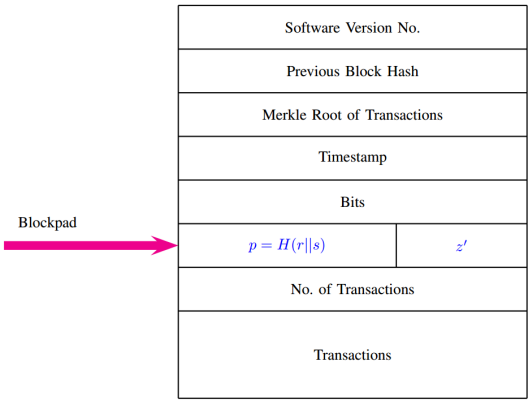
V obstoječi shemi je poln dokaz hash vrednost, bloka in žetona, ki ima na najbolj pomembnih bitih z ničel. Delni dokazi o opravljenem delu pa so nizi, ki imajo z' začetnih ničelnih bitov, kjer je $z' < z$. Delne dokaze rudarji pošiljajo administratorju, ki jih zbira in preverja. Administrator je odgovoren za določitev vrednosti z', ki mora biti taka, da lahko rudarji redno najdejo dokaz in hkrati ne presežejo računskih sposobnosti zaradi preverjanja.

Avtorji članka predlagajo sledečo spremembo Bitcoin sheme: Administrator izbere vrednost z'. Bazen poleg tega določi še naključen niz bitov s in vrednost zadnjih $z \check{} z'$ najmanj pomembnih bitov hash vrednosti. Ta niz bitov poimenujemo r. Bazen iz parametrov ustvari varnostno shemo *com* z ključem *decom*. Administrator pošlje shemo *com* vsem rudarjem, ključ pa hrani pri sebi in skrbi za to, da ostane tajen. Ko rudarji sestavijo blok vanj vključijo tudi shemo *com* in dolžino ciljne vrednosti delnega dokaza o opravljenem delu (z'). Rudarji administratorju posredujejo vse bloke, ki imajo hash vrednosti z z' začetnimi ničlami. Administrator prejme bloke in med preverjanjem išče takega, ki ustreza začetnim kriterijem. V primeru da najde takšen blok, ga posreduje Bitcoin sistemu skupaj z ključem. Ko vsa vozlišča v sistemu prejmejo blok in ključ, shemo odklenejo in preverijo, če ustreza kriterijem. Blok je sprejet, če je prvih z bitov hash vrednosti sestavljenih iz z' ničel in niza r.

V tej shemi rudarji ne morejo razlikovati med delnim in polnim dokazom dela. Ko se zamenja obdelovani blok mora administrator razkriti ključ s katerim rudarji preverijo svoje izračunane dokaze. Na ta način rudarjem odvzamemo sposobnost izvajanja dokazov, od administratorja pa zahtevamo dokazovanje poštenosti. Avtorji članka so dokazali, da ta shema ne vpliva na težavnost rudarjenja. Avtorji so predlagali tudi alternativno shemo, kjer kjer se namesto varnostne sheme uporablja hash funkcijo vsi ostali koraki in delovanje pa ostaneta enaka. Obe shemi sta varni dokler sta vrednosti s in r tajni.

4. ZAKLJUČEK

Avtorji so v tem članku pokazali kako lahko sebičen rudar pridobi dodaten zaslužek za izvajanje napada s prikrivanjem blokov. Ta dobiček dobi od drugih bazenov, ki si prav tako želijo povečati zaslužek potom tega napada. Izmerili so pričakovan dobiček napadalca in razkrili nekaj zanimivih strategij s katerimi si napadalec lahko poveča dobiček. Predstavili so tudi ukrep s katerimi bi lahko izničili zmožnosti takšnega napada s pomočjo preprostih kriptografskih metod. Ukrep temelji na slepljenju rudarjev in odvzemanju sposobnosti napada ter zavezovanju administratorja v sistem, kjer rudarji lahko preverijo njegovo poštenost. Ukrep je mogoče zlahka implementirati in to brez spreminjanja količine dela, ki je potrebna za izračun delnega dokaza o opravljenem delu.



Slika 1: Nova shema za rudarjenje

5. LITERATURA

[1] S. Nakamoto, *Bitcoin: A peer-to-peer electronic cash system*, 2008.

[2] A. Miller, A. Juels, E. Shi, B. Parno, and J. Katz, *Permacoin: Repurposing bitcoin work for data preservation*, in 2014 IEEE Symposium on Security and Privacy, SP 2014, Berkeley, CA, USA, May 18-21, 2014. IEEE Computer Society, 2014, pp. 475–490.

[3] B. Sengupta, S. Bag, S. Ruj, and K. Sakurai, *Retricoin: Bitcoin based on compact proofs of retrievability*, in Proceedings of the 17th International Conference on Distributed Computing and Net- working, ser. ICDCN '16, 2016, pp. 14:1–14:10.

[4] J. A. Kroll, I. C. Davey, and E. W. Felten, *The economics of bitcoin mining, or bitcoin in the presence of adversaries,*Proceedings of WEIS, vol. 2013, 2013.

[5] M. Rosenfeld, *Analysis of bitcoin pooled mining reward systems*, CoRR, vol. abs/1112.4980, 2011. [Online]. Available: http://arxiv.org/abs/1112.4980

[6] A. Laszka, B. Johnson, and J. Grossklags, *Financial Cryptography and Data Security: FC 2015 International Workshops, BITCOIN, WAHC, and Wearable, San Juan, Puerto Rico, January 30, 2015, Revised Selected Papers*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, ch. When Bitcoin Mining Pools Run Dry, pp. 63–77.

[7] . Eyal, *The miner's dilemma*, in 2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015. IEEE Computer Society, 2015, pp. 89–103.

[8] L. Luu, R. Saha, I. Parameshwaran, P. Saxena, and A. Hobor, *On power splitting games in distributed computation: The case of bit- coin pooled mining*, in IEEE 28th Computer Security Foundations Symposium, CSF 2015, Verona, Italy, 13-17 July, 2015, C. Fournet, M. W. Hicks, and L. Vigano', Eds. IEEE Computer Society, 2015, pp. 397–411.

[9] N. T. Courtois and L. Bahack, *On subversive miner strategies and block withholding attack in bitcoin digital currency*, arXiv preprint arXiv:1402.1718, 2014.

[10] I. Damgárd, *Lectures on Data Security: Modern Cryptology in The- ory and Practice*. Berlin,

Heidelberg: Springer Berlin Heidelberg, 1999, ch. Commitment Schemes and Zero-Knowledge Protocols, pp. 63–86.

[11] D. B. Okke Schrijvers, Joseph Bonneau and T. Roughgarde, *In- centive compatibility of bitcoin mining pool reward functions*, in Financial Cryptography and Data Security: FC 2016 International Workshops.

[12] S. Bag and K. Sakurai, *Yet another note on block withholding attack on bitcoin mining pools*, in Information Security - 19th International Conference, ISC 2016, Honolulu, HI, USA, September 3-6, 2016, Proceedings, ser. Lecture Notes in Computer Science, M. Bishop and A. C. A. Nascimento, Eds., vol. 9866. 2016, pp. 167–180.

[13] Samiran Bag, Sushmita Ruj and Kouichi Sakurai, *Bitcoin Block Withholding Attack : Analysis and Mitigation* IEEE Transactions on Information Forensics and Security

Varnostna analiza predlaganih izboljšav Bitcoina

Matej Vitek
Fakulteta za računalništvo in informatiko
Večna pot 113
Univerza v Ljubljani

Andrej Levičnik
Fakulteta za računalništvo in informatiko
Večna pot 113
Univerza v Ljubljani

POVZETEK

Če pri internetnem prenosu denarja ne želimo zaupati posebnim spletnim posrednikom, se poslužimo decentraliziranih sistemov kriptovalut. Najbolj priljubljena kriptovaluta je trenutno Bitcoin, ki deluje po principu bločne verige, ki omogoča sprotno preverjanje transakcij in hranjenje zgodovine transakcij kar s strani preostalih uporabnikov kriptovalutnega sistema. Napad dvojne porabe in verižni razcep sta trenutno glavni dve težavi v protokolih, ki temeljijo na tem principu. Pri napadu dvojne porabe napadalec uporabi isto enoto bitcoina večkrat. Verižni razcepi pa povzročijo neskladnosti v zgodovini med uporabniki. Z verjetnostno analizo pokažemo, da predlogi za rešitev teh dveh težav, ki so bili nedavno objavljeni, ne delujejo dobro na večjih bločnih verigah.

Ključne besede

Bitcoin, sistemi peer-to-peer, varnost, kriptovaluta, analiza uspešnosti delovanja

1. UVOD

Denarne transakcije preko spleta predstavljajo problem, saj moramo pri vsaki zaupati neki centralni avtoriteti, ki poskrbi za zanesljiv in varen prenos denarja. Decentralizirani sistemi kriptovalut odpravijo to težavo in omogočajo šifriran prenos denarja brez posrednikov.

V takih sistemih lahko pride do napadov *dvojne porabe*, pri katerih napadalec uporabi isto enoto valute za več različnih prenosov. V klasičnih centraliziranih sistemih take napade prepreči centralna avtoriteta (npr. spletna banka), medtem ko moramo v porazdeljenih, decentraliziranih sistemih za preprečitev takih napadov poskrbeti na drugačen način.

Leta 2008 je Satoshi Nakamoto predstavil Bitcoin [15]. Bitcoin je decentraliziran sistem, ki uporabnikom omogoča kupovanje in prodajanje z virtualno kriptovaluto. Temelji na javni zgodovini transakcij, ki je v obliki t. i. *bločne verige*. Bločno verigo gradijo uporabniki sistema, ki jim rečemo *rudarji*, in sicer tako, da ji dodajajo nove bloke. Vsak posamezen blok vsebuje zgodovino najnovejših transakcij. Ker gre za javno informacijo, je bločna veriga zanesljiva tudi v primeru, da imamo v sistemu zlonamerne rudarje.

Rudarji med sabo tekmujejo, saj je nagrajen tisti, ki verigi prvi doda nov veljaven blok (pri tem mora priložiti t. i. dokaz opravljenega dela, izračun katerega je računsko zahteven proces). Da lahko vsak uporabnik sistema brez težav preveri, da ni prišlo do dvojne porabe, morajo biti transakcije v blokih urejene, da je blok veljaven. Sistem mora paziti tudi na to, da ne pride do *verižnega razcepa*, ki se zgodi, če več rudarjev bločni verigi hkrati doda neskladne bloke. Verižni razcep zaradi neskladnosti sočasnih blokov omogoča napade dvojne porabe s strani udeležencev neskladnih transakcij v teh blokih.

Trije nedavni poskusi razrešitve teh dveh težav so Bitcoin-NG [3], PeerCensus [2] in BizCoin [9], ki preložijo velik del odgovornosti na rudarje z uporabo protokolov konsenza, ki temeljijo na algoritmih za odpornost na bizantinske napake [11], in z dodeljevanjem vodstva posameznim rudarjem.

V tej seminarski nalogi analiziramo učinkovitost teh treh pristopov glede na velikost bločne verige (številom transakcij) in njihovo odpornost na bizantinske (zlonamerne) rudarje. Pred tem podrobno predstavimo sistem Bitcoin in njegove lastnosti, ki so pomembne za analizo. Dobljeni rezultati so mešani. Bitcoin-NG se zanaša na enega rudarja na vsakem koraku in tako ne izboljša varnosti Bitcoina, niti ni primeren za večje bločne verige. Kljub uporabi algoritmov za odpornost na bizantinske napake, PeerCensus ni odporen na 1/3 bizantinskih rudarjev, kar je običajen prag pri taki analizi. BizCoin pa je teoretično precej učinkovit. Združuje prejšnja dva pristopa (in uporablja protokol CoSi [17]) in je varen pri deležih bizantinskih rudarjev do 1/3. Vendar pa še ni bil implementiran v sistemih z več kot 148 rudarji, saj je protokol CoSi prezahteven za take sisteme.

2. OMREŽJE BITCOIN

Sistem Bitcoin [15] je peer-to-peer (P2P) sistem za plačevanje, ki z uporabo porazdeljenih algoritmov in šifriranja uporabnikom omogoči anonimno kupovanje in prodajanje z enotami virtualne valute, ki se imenujejo bitcoini (bitni kovanci).

Temelji na treh glavnih sestavnih delih:

- **Transakcije** so posamezni prenosi, ki jih sprožijo kupci.
- **Bločna veriga** je urejeno zaporedje blokov, v kateri vsak blok vsebuje množico izpeljanih transakcij. Za gradnjo in preverjanje veljavnosti bločne verige skrbijo uporabniki omrežja.
- **Bazen transakcij** vsebuje transakcije v teku, ki čakajo na potrditev in vnos v bločno verigo. Vsak uporabnik hrani svojega.

Uporabniki omrežja so lahko v katerikoli od sledečih vlog:

- **Udeleženec** pošilja ali prejema bitcoine v transakciji.
- **Vrstnik** prepošilja transakcijo in preverja veljavnost bločne verige.
- **Rudar** določi, katere transakcije se dodajo v bločno verigo in njihov vrstni red.

2.1 Protokol Bitcoin

Za ponazoritev delovanja Bitcoina vzemimo hipotetične uporabnike Ana, Boris in Cene. Ana ima bitcoine in želi z njimi nekaj kupiti od Borisa in Ceneta. Bitcoini so virtualni in so dostopni preko uporabniških računov v omrežju Bitcoin, katerih varnost temelji na asimetričnih šifrirnih algoritmih.

Ana ustvari nove uporabniške račune (s pripadajočimi javnimi in zasebnimi ključi). Na vsak uporabniški račun prejme dobi število bitcoinov v t. i. *bazni transakciji* (to so transakcije, s katerimi Bitcoin ustvarja virtualno valuto kot nagrado rudarjem za uspešno ustvarjanje blokov), ali pa bitcoine prejme v transakciji od kakšnega drugega uporabnika. Nato bitcoine v natanko eni transakciji pošlje na uporabniške računa Borisa in Ceneta.

Če ima pri transakciji na svojih uporabniških računih več bitcoinov, kot jih za transakcijo potrebuje, mora odpreti nove račune, na katere potem prenese preostanek. Vsak račun namreč lahko prejme in pošlje le eno samo transakcijo. Ob transakciji lahko Ana prostovoljno plača še transakcijski honorar, ki ga nato kot nagrado prejme rudar, ki jo prvi uspešno uvrsti v bločno verigo. Vsako transakcijo Ana tudi digitalno podpiše.

Transakcija je *dobro osnovana*, če je število bitcoinov na vhodnih (Aninih) računih zadostno za pokritje želenih prenosov na izhodne (Borisove in Cenetove) račune. V nadaljevanju predpostavimo, da imamo opravka z dobro osnovanimi transakcijami.

Ko Boris in Cene prejmeta podpisano transakcijo, jo pošljeta nekemu vrstniku v pregled veljavnosti. Transakcija je *lokalno veljavna*, če je ta vrstnik prejel vse transakcije, ki so prenesle bitcoine na vhodne (Anine) račune in še ni prejel nobene transakcije, pri kateri bi bili ti računi porabljeni.

Če vrstnik odloči, da je transakcija veljavna, to sporoči Borisu in Cenetu ter transakcijo pošlje naprej po omrežju. Če vrstnik za nek vhodni račun še ni prejel transakcije, ki bi računu dodelila bitcoine, transakcijo doda v svoj bazen transakcij in jo pošlje naprej po omrežju. Ko prejme potrebne transakcije, nato transakcijo označi za veljavno in to sporoči Borisu in Cenetu. Če je vrstnik za nek vhodni račun že dobil izhodno transakcijo, je račun v stanju dvojne porabe.

Za transakcijo pravimo, da je *brez konfliktov*, če ni noben vhodni račun v stanju dvojne porabe in so vse transakcije, ki so vhodnim računom dodelile bitcoine brez konfliktov. Taka induktivna konstrukcija je primerna, saj v končnem številu korakov pridemo do bazne transakcije, ki je gotovo brez konfliktov.

Največji izziv v sistemu Bitcoin je pravočasno zaznavanje stanj dvojne porabe in zavrnitev takih transakcij, s čimer preprečimo napade dvojne porabe. Bitcoin uspešnost takih napadov omejuje z uporabo blokov, ki jih v urejenem vrstnem redu rudarji dodajajo v bločno verigo. Zaradi kompleksnosti izračuna zahtevanega dokaza opravljenega delu je povprečni čas izdelave bloka 10 minut. Ko nek rudar blok uspešno doda bločni verigi, ga pošlje v omrežje, nakar ga vsak vrstnik doda v svojo kopijo bločne verige. Za lokalno veljavno transakcijo pravimo, da je *lokalno potrjena*, ko vrstnik prejme blok, ki to transakcijo vsebuje. *Nivo potrditve* nam pove, koliko blokov je bilo dodanih v bločno verigo od potrditve transakcije (vključno z blokom, ki je transakcijo potrdil).

2.2 Verižni razcep

Ker so rudarji nagrajeni za čim hitrejše ustvarjanje blokov, lahko v bločni verigi pride do t. i. sočasnih blokov, če dva rudarja poskusita hkrati dodati vsak svoj blok v bločno verigo. V tem primeru bločna veriga v sistemu dobi drevesno strukturo (namesto običajne linearne). Temu pojavu pravimo verižni razcep.

Verižni razcep lahko zlonamerni rudarji zlorabijo za uspešne napade dvojne porabe, če veja, ki po razrešitvi razcepa ostane, vsebuje neveljavno transakcijo napadalca. Poleg tega se ob večjem številu vej v bločni verigi poveča kompleksnost dodajanja novih blokov. Zato morajo biti verižni razcepi razrešeni kar se da hitro.

Verižni razcep je razrešen, ko rudar uspešno izračuna dokaz opravljenega dela in ga pošlje po omrežju dovolj hitro, da ga vsi ostali rudarji prejmejo preden izračunajo svoj dokaz opravljenega dela.

Nakamoto [15] in kasnejša dela [4, 7, 14] analizirajo tekmovanje med zlonamernimi in poštenimi rudarji. Nakamoto je pokazal, da se transakcija izkaže za neveljavno z verjetnostjo manj kot 0,1%, če je potrjena z nivojem potrditve 5 s strani nekega vrstnika, v primeru, ko imamo 10% zlonamernih rudarjev. Če imamo 15% zlonamernih rudarjev, se potreben nivo potrditve zviša na 8, pri 25% pa na 15. Transakcijam s takim nivojem potrditve pravimo *globoko potrjene* transakcije.

2.3 Lastnosti Bitcoina

Za sistem Bitcoin veljata lastnosti *živosti* in *varnosti*.

Živost pomeni, da bo vsaka transakcija brez konfliktov prej ali slej globoko potrjena v bločni verigi poštenega vrstnika.

Varnost pomeni, da bo vsaka transakcija brez konfliktov, ki

jo je globoko potrdil nek pošten vrstnik, prej ali slej globoko potrjena v bločnih verigah vseh poštenih vrstnikov, in sicer z enakim nivojem potrditve.

Ti dve lastnosti nam zagotavljata uspešno preverjanje veljavnosti transakcij, ki jih pošljejo pošteni uporabniki, ne zagotavljata pa, da bodo pošteni prejemniki transakcij s konflikti dejansko prejeli pripadajoče bitcoine.

Bitcoin torej omogoča napade dvojne porabe proti prejemnikom, ki pričakujejo, da bodo vse veljavne ali celo lokalno potrjene transakcije prej ali slej globoko potrjene. Veljavne (ali lokalno potrjene) transakcije se namreč lahko izkažejo za konfliktne, če so njihovi predhodniki neveljavni. Težava je, da globoka potrditev ponavadi traja več kot eno uro. Nedavno razviti pristopi, ki jih analiziramo, se zato osredotočajo na izboljšanje zagotovljenih lastnosti Bitcoina.

3. PRELAGANJE ODGOVORNOSTI NA RUDARJE

Nedavno predstavljeni algoritmi Bitcoin-NG [3], PeerCensus [2] in BizCoin [9] se za potrjevanje veljavnosti zanašajo izključno na rudarje. Zagotoviti poskušajo atomsko doslednost transakcij. Le-ta pomeni, da so vse posodobitve bločne verige v enakem vrstnem redu z vidika vseh uporabnikov sistema.

V vseh treh je čas diskretiziran v posamezne dobe. Doba se konča, ko nek rudar uspešno ustvari nov blok. Ta rudar nato postane *vodja* do konca naslednje dobe.

Bitcoin-NG se za atomsko doslednost pri preverjanju veljavnosti zanaša le na nazadnje izbranega vodjo. PeerCensus uporablja protokole konsenza, odporne na bizantinske napake [1, 5, 10] na množici vseh vodij. BizCoin deluje podobno, vendar se omeji na množico zadnjih w vodij.

V vseh treh vodje odločajo o veljavnosti transakcij in jih potrjujejo ter jih nato razpošiljajo po omrežju. V preostanku poglavja pokažemo, da ne glede na uporabljen algoritem, zanašanje izključno na rudarje pri preverjanju veljavnosti ne preprečuje napadov dvojne porabe. Najprej pa predstavimo model, ki ga uporabimo za analizo varnosti.

3.1 Model

Predpostavimo prisotnost zlonamernega uporabnika, ki kontrolira delež $\mu \in (0,1)$ vseh rudarjev. Namen zlonamernega uporabnika je izkoriščanje protokola za izvajanje napada dvojne porabe. Rudarji katere tak uporabnik upravlja in bloke, ki jih generirajo označimo s predpono zlonamerni.

Na drugi strani imamo rudarje, kateri niso pod kontrolo, označimo jih s predpono iskreni. Delež teh je $(1-\mu)$ vseh rudarjev.

Pravtako predpostavimo da je:

- računska moč vsakega rudarja (bodisi iskrenega bodisi zlonamernega) enaka,
- kreacija posameznega bloka konstantna.

Naj $B_k = (h, m)$ označuje stanje bločne verige v časovni dobi k, število iskrenih oz. zlonamernih blokov predstavljata h in m.

Predpostavimo, da je Nakamoto - ustvarjalec Bitcoina iskren, iz tega sledi $B_0 = (1, 0)$.

Proces $B = \{B_k \mid k \geq 0\}$ opisuje kompozicijo bločne verige čez časovne dobe.

Iz stanja $B_k = (h, m)$ lahko veriga preide v dve stanji. Blok lahko zgenerira:

- iskreni rudar z verjetnostjo $(\mu - 1)$, v tem primeru dobimo $B_{k+1} = (h + 1, m)$,
- zlonamerni rudar z verjetnostjo u v tem primeru dobimo $B_{k+1} = (h, m + 1)$.

Proces B je diskretno-časovna Markova veriga čez diskretni prostor stanj $\mathbb{N}^* \times \mathbb{N}$. Neničelne vrednosti prehodov so podane za vse $(h, m) \in \mathbb{N}^* \times \mathbb{N}$ z

$\mathbb{P}\{B_{k+1} = (h + 1, m)|B_k = (h, m)\} = 1 - \mu$ in
$\mathbb{P}\{B_{k+1} = (h, m + 1)|B_k = (h, m)\} = \mu$.

3.2 Analiza varnosti Bitcoin-NG

Kot je opisano v prejšnjem razdelku, v Bitcoin-NG vsako časovno dobo vodi posamezen rudar, ki validira množico transakcij. Po prejetju transakcije, vodja preveri, da je transakcija lokalno legitimna po definiciji 1. V primeru, da je, jo podpiše in razpošlje. Samo podpisane transakcije so legalne in vstavljene v blok.

Če je vodja zlonameren, lahko ustvari transkacije dvojne porabe, jih podpiše brez, da bi upošteval ostale transakcije pri katerih so prejemniki iskreni. Iz predpostavke, da je $\mu \in (0, 1)$ rudarjev zlonamernih sledi, da je prav tako μ blokov zlonamernih.

Napredek oz. evolucijo bločne verige lahko vzamemo kot naključen sprehod po $\mathbb{N}^* \times \mathbb{N}$.

Glede na $k \geq 0$, $h \geq 1$ in $m \geq 0$ ter $B_0 = (1, 0)$ lahko izpeljemo:

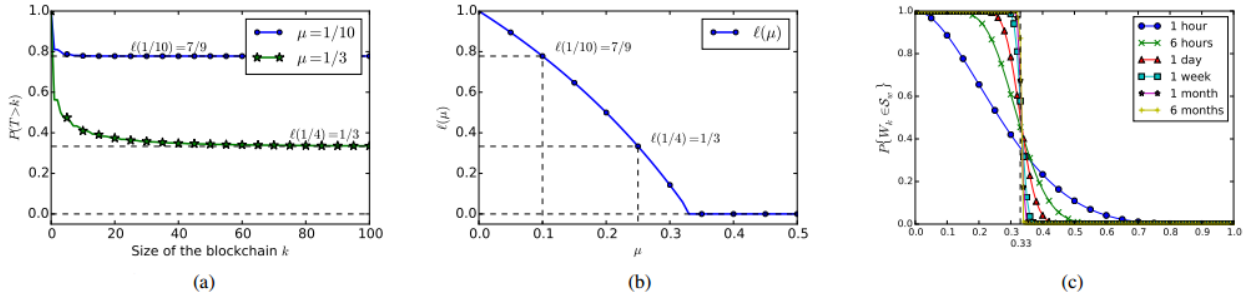
$$\mathbb{P}\{B_k = (h, m)\} = (1 - \mu)^{h-1} \mu^m 1_{\{k=h+m-1\}}$$

Verjetnost, da veriga pri časovni dobi k ne vsebuje nobenih zlonamernih blokov je enaka $\mathbb{P}\{B_k = (h, 0)\} = (1-\mu)^{k-1}$ če drži $h = k-1$. Trenutna Bitcoin veriga šteje več kot 420000 blokov, kar pomeni da je ta verjetnost blizu 0.

Poleg tega je vredno omeniti tudi probleme z razširjanjem takega protokola. Trenutno se na Bitcoin mreži izvede približno 1500 transakcij na eno časovno dobo. Pri trenutni popularizaciji Bitcoina ta številka vztrajno raste.

3.3 Analiza varnosti PeerCensus Safety

V primerjavi z Bitcoin-NG, PeerCensus predlaga, da so pri validaciji vpleteni vsi, do takrat uspešni rudarji ε_∞. Rudarji



(a) (b) (c)

Slika 1 Analiza varnosti PeerCensus in BizCoin algoritmov

bi sledili sistemu konsenza, ki je odporen na bizantinske napake kot je naprimer PBFT [2].

Še preden pogledamo varnost protokola, lahko opazimo problem razširljivosti. Kompleknost sporočila v takih sistemih je ponovadi $O(k^3)$, algoritmi za odpornost na bizantinske napake so pri $k > 10$ zelo požrešni. Bitcoin ima trenutno 420000 blokov ($k >= 420000$).

Članstvo ε_∞ pri k-temu doseganju konsenza je odvisna od odločitve, pridobljene pri soglasju $(k-1)$, iz tega sledi, da je velika možnost stalne onesnaženosti ε_∞. Onesnaženost definiramo kot stanje množice v katerem je prisotnih več kot 1/3 bizantinskih oz. zlonamernih rudarjev.

Po navedbah [15], člani ne morejo priti do konsenza, če je prisotnih $(n-1)/3$ bizantinskih članov. Trdimo, da je stanje $B_k = (h, m)$ od ε_∞ pri časovni dobi k onesnaženo, če je število m bizantinskih rudarjev, ki pripadajo ε_∞ večje ali enako $(k-1)/3$. Prav tako velja, da je stanje, ki ni onesnaženo, varno.

Prostor stanj $\mathbb{N}^* \times \mathbb{N}$ razdelimo na varna in onesnažena stanja

$S_\infty = \{(h, m) \in \mathbb{N}^* \times \mathbb{N} | h \geq 2m + 1\}$ in
$P_\infty = \{(h, m) \in \mathbb{N}^* \times \mathbb{N} | h \leq 2m\}$.

Verjetnost, da je ε_∞ varen, pri dani časovni dobi k lahko izračunamo:
$\mathbb{P}\{B_k \in S_\infty\} = \sum_{h=1,3h\geq 2k+3}^{k+1} \binom{k}{h-1}(1-\mu)^{h-1}\mu^{k-h+1}$

Z uporabo centralnega limitnega izreka dobimo:

$$\lim_{k\to\infty} \mathbb{P}\{B_k \in S_\infty\} = \begin{cases} 0 pri \mu > 1/3 \\ 1/2 pri \mu = 1/3 \\ 1 pri \mu < 1/3 \end{cases}$$

Vseeno pa ne moremo trditi, da je bil korak, ki je privedel do stanja B_k varen. To je zelo pomembno, kajti, ko je enkrat ε_∞ onesnažen, lahko zlonamernik po lastni volji odloča katere transakcije bodo potrjene in katere bloki bodo vključeni v verigo.

Definirajmo verjetnost, da je bilo k zaporednih soglasij varnih. Naj bo T število časovnih dob, ki so se zgodile v stanju S_∞ preden prvič dosežemo stanje P_∞. T definiramo kot $T = min\{k \geq 0 | B_k \in P_\infty\}$ in dobimo $\mathbb{P}\{T \geq k\} = \mathbb{P}\{B_0 \in S_\infty, B_1 \in S_\infty, ..., B_k \in S_\infty\}$

S spodnjo enačbo izračunamo verjetnost da smo v stanju $B_k = (h, m) \in S_\infty$ preden se onesnaži.

Za vse vse $(h, m) \in S_\infty$ to so $h \geq 1, m \geq 0, h \geq 2m + 1$ in $k = m + h + 1$ dobimo izrek:
$\mathbb{P}\{T \geq k, B_k = (h, m)\} = \left[\binom{k+1}{h} - 3\binom{k}{h}\right](1-\mu)^{h-1}\mu^m 1_{\{k=m+h-1\}}$

Dokaz, definiramo $f(h, m) = \mathbb{P}\{T \geq k, B_k = (h, m)\}$ za $(h, m) \in S_\infty$ za $(k = m+h+1)$ in $f(h, m) = 0$ v nasprotnem primeru.

Pri začetnem stanju (1,0), imamo f(1,0) = 1. Če uporabimo lastnost Markove verige:

$$f(h, m) = (1-\mu)f(h-1, m)1_{\{h\geq 2m+2\}} + \mu f(h, m-1).$$

Enačba drži za $m = 0$, vstavimo v zgornjo enačbo:

$$f(h, 0) = (1-\mu)f(h-1, 0)1_{\{h\geq 2\}},$$

to je $f(h, 0) = (1-\mu)^{h-1}$ za vsak $h \geq 1$. Uporabimo rekurenčnost na dveh nivojih. Predvidevamo, da zgornji izrek drži za cela števila $m - 1$. Za $h = 2m + 1$ imamo:

$$\begin{aligned} f(2m+1, m) &= \mu f(2m+1, m-1) \\ &= \left[\binom{3m}{2m+1} - 3\binom{3m-1}{2m+1}\right](1-\mu)^{2m}\mu^m \\ &= \frac{(3m)!}{(2m+1)!m!}(1-\mu)^{2m}\mu^m, \end{aligned}$$

Če predvidevamo, da izrek drži za cela števila $h - 1$ in m kjer $h \geq 2m + 2$. Z rekurenčno hipotezo dobimo:

$$\begin{aligned} f(h, m) &= (1-\mu)f(h-1, m) + \mu f(h, m-1) \\ &= \left[\binom{m+h-1}{h-1} - 3\binom{m+h-2}{h-1}\right. \\ &\quad \left. + \binom{m+h-1}{h} - 3\binom{m+h-2}{h}\right](1-\mu)^{h-1}\mu^m. \end{aligned}$$

Združimo prvi in tretji izraz, in drugi in četri izraz, ker je $k = m - h + 1$ dobimo

$$f(h, m) = \left[\binom{k+1}{h} - 3\binom{k}{h}\right](1-\mu)^{h-1}\mu^m,$$

ki zaključuje dokaz.

S spodnjim izrekom dobimo distribucijo prve pojavitve T pri onesnaženju ε_∞. Za vsak $\mu \in (0,1)$ in $k \geq 0$ imamo

$$\mathbb{P}\{T > k\} = \frac{1}{1-\mu} \sum_{h=\lceil 2k/3 \rceil + 1}^{k+1} \binom{k+1}{h} (1-\mu)^h \mu^{k+1-h} - \frac{3\mu}{1-\mu} \sum_{h=\lceil 2k/3 \rceil + 1}^{k} \binom{k}{h} (1-\mu)^h \mu^{k-h}.$$

Limita $l(\mu)$ je podana z:

$$l(\mu) = \begin{cases} 0 & \text{če } \mu > 1/3 \\ 1 - \frac{2\mu}{1-\mu} & \text{če } \mu \leq 1/3. \end{cases}$$

Dokaz, izrek velja za vse $k \geq 0$,

$$\begin{aligned} \mathbb{P}\{T > k\} &= \sum_{(h,m) \in \mathcal{S}} \left[\binom{k+1}{h} - 3\binom{k}{h} \right] \\ &\quad \times (1-\mu)^{h-1} \mu^m 1_{\{k=m+h-1\}} \\ &= \sum_{h=1, 3h \geq 2k+3}^{k+1} \binom{k+1}{h} (1-\mu)^{h-1} \mu^{k-h+1} \\ &\quad - \sum_{h=1, 3h \geq 2k+3}^{k} \binom{k}{h} (1-\mu)^{h-1} \mu^{k-h+1} \\ &= \frac{1}{1-\mu} \sum_{h=\lceil 2k/3 \rceil + 1}^{k+1} \binom{k+1}{h} (1-\mu)^h \mu^{k+1-h} \\ &\quad - \frac{3\mu}{1-\mu} \sum_{h=\lceil 2k/3 \rceil + 1}^{k} \binom{k}{h} (1-\mu)^h \mu^{k-h}. \end{aligned}$$

Drugi rezultat je pridobljen z izrekom centralne limite.

V sliki 1a opazimo hitro konvergenco T do limite $l(\mu)$, medtem ko slika 1b pokaže, da ko $0 < \mu \leq 1/3$ je verjetnost, da bi imeli serijo varnih soglasij striktno manj kot 1. Na primer, za $\mu = 1/4 < 1/3$, dobimo $l(\mu) = 1/3$, kar pomeni, da je od vseh možnih poti k le $1/3$ varnih. Ta rezultat močno nakazuje limitacije PeerCensus pristopa.

3.4 BizCoin

BizCoin [9] je kombinacija idej predstavljenih v Bitcoin-NG in PeerCensus: BizCoin vlogo vodje trenutne časovne dobe dodeli zadnjemu uspešnemu rudarju, s to razliko, da za validacijo skrbi tolerantni bizantinski konsenz, ki je implementiran s kriptografsko podpisovalno shemo [17]. Prav tako je razlika pri številu množice, ki sestavlja konsenz. PeerCensus konsenz sestavljajo vsi uspešni rudarji ε_∞, BizCoin pa to število omeji na ε_ω, ki vsebuje trenutno vodjo in zadnjih $(\omega - 1)$ rudarjev.

Nadaljujmo z analizo varnosti protokola BizCoin.

Definirajmo spremenljivko M, ki je odvisna od tipa (zlonameren/iskren) trenutne vodje. Po navedbah enačb 4 in 5; v časovni dobi k je vodja iskren ($M = 0$) z verjetnostjo $\mathbb{P}\{M = 0\} = 1 - \mu$ in zlonameren ($M = 1$) z verjetnostjo $\mathbb{P}\{M = 1\} = \mu$. Definirajmo zaporedje zadnjih ω voditeljev v časovni dobi k - $M_{0,k}, \ldots, M_{w-1,k}$ Vektor $W_k = (M_{0,k}, \ldots, M_{w-1,k}) \in \{0,1\}^w$ predstavlja stanje ε_ω. Proces $W = \{W_k, k \geq 0\}$ se razvija sledeče:

$$\forall k \geq 1, \forall 1 \leq i \leq w-1, M_{i,k} = M_{i-1,k-1}$$

kjer je $(M_{0,k})_{k \geq 1}$ zaporedje neodvisnih in identično razporejenih Bernoulli naključnih spremenljivk z $\mathbb{P}\{M_{0,k} = 0\} = 1 - \mu$ in $\mathbb{P}\{M_{0,k} = 0\} = \mu$.

Proces $W = \{W_k, k \geq 0\}$ $\{0,1\}^w$ je zato homogena diskretno časovna Markova veriga v prostoru $\{0,1\}^\omega$, ki predstavlja razvoj kompozicije ε_ω skozi časovne dobe.

Tako kot pri PeerCensus je stanje onesnaženo, če je v ε_ω več kot ena tretjina bizantinskih rudarjev. Če temu ni tako je stanje varno. Zanima nas število bizantinskih rudarjev v ε_ω v časovni dobi k. Definiramo spremenljivko $N_k = \sum_{i=0}^{w-1} M_{i,k}$.

Razdelimo prostor stanj $\{0,1\}^\omega$ v dve podmnožici (varno in onesnaženo):

$$\mathcal{S}_w = \{(m_0, \ldots, m_{w-1}) \in \{0,1\}^w | \sum_{i=0}^{w-1} m_i \leq (w-1)/3\},$$

$$\mathcal{P}_w = \{(m_0, \ldots, m_{w-1}) \in \{0,1\}^w | \sum_{i=0}^{w-1} m_i > (w-1)/3\}.$$

Spodnji izrek nam pove verjetnost, da je ε_ω v varnem in stabilnem stanju in pokaže, da pridemo do stabilnega stanja pri časovni dobi $k = \omega$.

$$\mathbb{P}\{W_k \in \mathcal{S}_w\} = \sum_{l=0}^{(w-1)/3} \binom{w}{l} \mu^l (1-\mu)^{w-l}.$$

Dokaz. Z zgornje enačbe lahko za vsak $k \geq 1$ dobimo $N_k = N_{k-1} + M_{0,k} - M_{w-1,k-1}$

To lahko razširimo na:

$$N_k = N_0 + \sum_{l=1}^{k} M_{0,l} - \sum_{l=0}^{k-1} M_{w-1,l}.$$

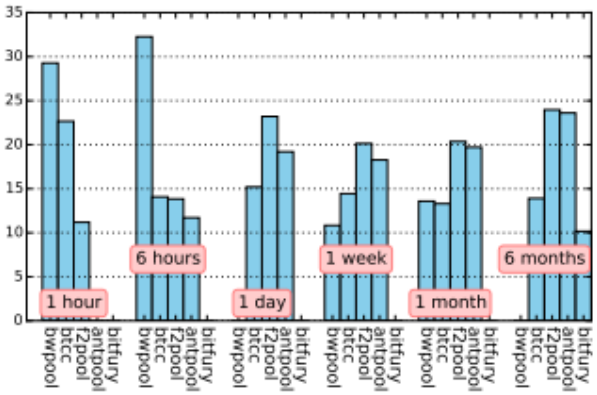
Če pogledamo:

$$\sum_{l=0}^{k-1} M_{w-1,l} = \begin{cases} \sum_{l=0}^{k-1} M_{w-1-l,0} & \text{if } k \leq w \\ \sum_{l=0}^{w-1} M_{w-1-l,0} + \sum_{l=w}^{k-1} M_{0,l-w+1} & \text{if } k > w \end{cases}$$

in vstavimo v prejšno enačbo dobimo naslednji rezultat:

$$N_k = \begin{cases} \sum_{l=1}^{k} M_{0,l} + \sum_{l=0}^{w-1-k} M_{l,0} & \text{if } k \leq w-1 \\ \sum_{l=k-w+1}^{k} M_{0,l} & \text{if } k \geq w. \end{cases}$$

Za vsak $k \geq \omega$, je N_k vsota ω neodvisno identično porazdeljenih Bernoulli spremenljivk s parametrom μ, torej dobimo:



Slika 2 Delež rešenih blokov, katere so podpisala največji rudarski bazeni

$$\mathbb{P}\{W_k \in \mathcal{S}_w\} = \mathbb{P}\{N_k \leq \lfloor (w-1)/3 \rfloor\} = \sum_{l=0}^{(w-1)/3} \binom{w}{l} \mu^l (1-\mu)^{w-l}.$$

Rezultat, ki ga dobimo z zgornjim izrekom je konsistenten z limitnim izrekom, ki smo ga definirali v razdelku Analiza varnosti PeerCensus Safety, ko gre ω v neskončnost. Slika 2c prikazuje delež varnih izvedb BizCoin kot funkcije ω in μ. Trenutno je v Bitcoin bločni verigi približno 422000 blokov. Izračunamo za $k \geq \omega$ vrednosti $\mathbb{P}\{W_k \in \mathcal{S}_w\}$ za časovno okno ene ure ($\omega = 6$), šestih ur ($\omega = 36$), enega dneva ($\omega = 144$), enega tedna ($\omega = 1008$), enega meseca ($\omega = 4320$) in šestih mesecev ($\omega = 25920$). Dva trenda slonita na μ: v primeru, da je zlonamernik šibek ($\mu \leq 1/3$) je sistem varnejši z večjim časovnim oknom (ω), in obratno, v primeru da je zlonamernik močan ($\mu \geq 1/3$) je sistem varnejši z manjšim časovnim oknom (ω). Velikost časovnega okna je obratno sorazmerna z varianco. Manjša varianca onemogoča šibkemu zlonamerniku, da bi naključno prišel do moči, medtem ko večja varianca omogoča iskrenim rudarjem 'krajo' možnosti zlonamerniku, da pride do moči.

Preglejmo še efektivno moč, ki jo imajo posamezniki na dejanski Bitcoin bločni verigi. Zadnje leto, so bili skoraj vsi bloki zgenerirani prek rudarskih bazenov. Rudarski bazeni združujejo rudarje oz. njihovo računsko moč v namen, da bi povečali verjenost odkritja rešitve bloka. Če je blok uspešno vstavljen v bločno verigo, je nagrada razdeljene med rudarje. Slika 2 prikazuje delež blokov, ki so ga zgenerirali najbolj pomembni rudarski bazeni, BWPool, BTCC, F2Pool, AntPool in BitFury, čez različna časovna obdobja ω. Lahko opazimo, da pri vsakem ω noben rudarski bazen ni presegel $\omega/3$ blokov. Če bi naštete bazene upravljal zlonamerni posameznik, bi imel v lasti 60% rudarjev, kar očitno ogroža protokol BizCoin.

Če povzamemo, opisali smo tri predloge, ki izboljšajo Bitcoin. Glavna težava izhaja iz dejstva, da so vse pomembne odločitve prepuščene kvorumu rudarjev. Prav tako o članstvu v kvorumu odločajo člani kvoruma. Kar še poveča moč zlonamernih rudarjev.

4. SORODNA DELA

Bitcoin smatramo kot pionirja kriptovalutnih sistemov. Od časa njegove ustanovitve se je pojavilo mnogo alternativ. Večina razlik med njimi se skriva v implementaciji, uporaba podatkovne baze [13], čas ustvarjanja bloka [18], algoritem zgoščevanja [12], neomejeno število kovancev [8], itd. Protokol GHOST [16] predlaga drugačno pravilo reševanja bločnih razcepov. Ta temelji na številu blokov v posamezni podveji (če do te pride). Protokola CoinJoin in CoinShuffle predlagata mešane transakcije, te omogočajo težje sledenje izvoru in ponoru transakcij.

Avtorji članka [14] dokažejo, da protokol Bitcoin z veliko verjetnostjo doseže soglasje. Članka [6, 7] opisujeta izvedljivosti napada dvojne porabe in detekciji takšnega napada. Medtem ko članki [3, 2, 9] opisujejo metode za izboljšanje varnosti. Ravno te smo podrobno pogledali.

5. ZAKLJUčEK

V seminarski nalogi smo predstavili ključne principe delovanja Bitcoina. Navedli smo lastnosti, ki so posledica teh principov. Pregledali smo tri pristope za zagotavljanje močne doslednosti v Bitcoinu. Povzeli smo zakaj ti pristopi niso varni.

6. LITERATURA

[1] M. Castro and B. Liskov. Practical byzantine fault tolerance. In *Proceedings of the Symposium on Operating Systems Design and Implementation*. OSDI, 1999.

[2] C. Decker, J. Seidel, and R. Wattenhofer. Bitcoin meets strong consistency. In *Proceedings of the Annual International Conference on Distributed Computing and Networking*. ICDCN, 2016.

[3] I. Eyal, A. E. Gencer, E. G. Sirer, and R. V. Renesse. Bitcoin-ng: A scalable blockchain protocol. In *Proceedings of the USENIX Symposium on Networked Systems Design and Implementation*. NSDI, 2016.

[4] J. A. Garay, A. Kiayias, and N. Leonardos. The bitcoin backbone protocl: Analysis and applications. In *Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic*

Techniques – Advances in Cryptology. EUROCRYPT, 2015.

[5] R. Guerraoui, N. Knežević, V. Quéma, and M. Vukolić. The next 700 bft protocols. In *Proceedings of the European Conference on Computer Systems*. EuroSys, 2010.

[6] G. O. Karame, E. Androulaki, and S. Capkun. Double-spending fast payments in bitcoin. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security*. CCS, 2012.

[7] G. O. Karame, E. Androulaki, M. Rzeschlin, A. Gervais, and S. Capkun. Misbehavior in bitcoin: A study of double-spending and accountability. *ACM Transactions on Information and System Security*, 2015.

[8] S. N. S. King. Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. 2012.

[9] E. Kokoris-Kogias, P. Jovanovic, N. Gailly, I. Khoffi, L. Gasser, and B. Ford. Enhancing bitcoin security and performance with strong consistency via collective signing. In *Proceedings of the USENIX Security Symposium*. USENIX Security, 2016.

[10] R. Kotla, L. Alvisi, M. Dahlin, A. Clement, and E. Wong. Zyzzyva: Speculative byzantine fault tolerance. In *Proceedings of the Symposium on Operating Systems Principles*. SOSP, 2007.

[11] L. Lamport, R. Shostak, and M. Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 1982.

[12] Litecoin. Global decentralized currency based on blockchain technology. 2011. https://litecoin.org.

[13] A. Loibl. Namecoin. 2014. http://namecoin.info/.

[14] A. Miller and J. L. Jr. Anonymous byzantine consensus from moderately-hard puzzles: A model for bitcoin. 2014. http://nakamotoinstitute.org/research/anonymous-byzantine-consensus/.

[15] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008. https://bitcoin.org/bitcoin.pdf.

[16] Y. Sompolinsky and A. Zohar. Accelerating bitcoin's transaction processing. fast money grows on trees, not chains. *IACR Cryptology ePrint Archive*, 2013.

[17] E. Syta, I. Tamas, D. Visher, D. Wolinsky, L. Gasser, N. Gailly, and B. Ford. Keeping authorities "honest or bust" with decentralized witness cosigning. In *Proceedings of the IEEE Symposium on Security and Privacy*. S&P, 2016.

[18] G. Wood. Ethereum: A secure decentralised generalised transaction ledger. http://gavwood.com/Paper.pdf.

Nenadzorovano učenje za namene iskanja prevar pri trgovanju z Bitcoinom

Analiza članka pri predmetu Računalniška forenzika *

Jure Malovrh
Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
jm3315@student.uni-lj.si

Juš Lozej
Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
jl3231@student.uni-lj.si

Žiga Pušnik
Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
zp8826@student.uni-lj.si

POVZETEK

V zadnjem času kripto-valuto Bitcoin prevzema vse več uporabnikov, s tem pa se dogaja tudi vedno več spletnega kriminala, v katerem je Bitcoin vpleten. V primeru, da se želimo obvarovati pred različnimi prevarami, potrebujemo metode strojnega učenja, ki nam na podlagi zaznavanja anomalij v omrežju skrbijo in nas opozarjajo pred možno prevaro.

Patrick Monamo in ostali raziskovalci z inštituta CSIR (angl. *Council for Scientific and Industrial Research*) so v svojem članku z naslovom *Unsupervised Learning for Robust Bitcoin Fraud Detection* predstavili novo metodo za odkrivanje prevar v omrežju Bitcoin. Predstavljena metoda temelji na strojnem učenju z uporabo rezanih razvrščanj z voditeljem (angl. *trimmed k-means*). S to metodo so zmožni hkratnega gručenja objektov in odkrivanja prevar v omrežju.

Njihov pristop se izkaže za uspešnega, saj so na podatkovnem viru odkrili več goljufivih transakcij, kot so jih odkrile podobne raziskave na istem podatkovnem viru.

Ključne besede

Prevare, anomalije, razvrščanje z voditelji (k-means clustering), Bitcoin

*Šolsko leto 2016/2017, izvajalca predmeta: dr. Andrej Brodnik, dr. Gašper Fele Žorž

1. UVOD

Z napredkom na tehnološkem področju, je napredovalo tudi finančno področje. Zadnje tehnološke izboljšave so prinesle izdelavo prvega plačilnega sistema, ki temelji na internetu. Ta sistem se imenuje Bitcoin peer-to-peer omrežje. To je sistem, pri katerem tiskanje valute ni odvisno od zaupanja vredne finančne institucije. To, da za njim ni zaupanja vredne finančne institucije; da je precejšnja novost na trgu; ter skepsa glede njegove globalne uporabe, naredi Bitcoin ranljiv za različne prevare in goljufije.

Prevare, ki jih obravnavajo v članku so natančneje opisane na spletni strani Bitcointalk [1], ki predstavlja nekakšno glavno točko Bitcoina, saj se tam odvijajo vsi pomembni dogodki v povezavi z Bitcoinom. Težava, ki jo lahko pri opisanih prevarah in goljufijah opazimo je predvsem to, da so to precej različne prevare. pod prevaro tako spada kraja 25000 kovancev, zaseg kovancev s strani FBI-ja ali pa hekerski vdor. Goljufive transakcije so tako podvržene subjektivni oceni, zato so morda tudi nenatančne. To predstavlja težavo pri odkrivanju.

1.1 Bitcoin

Bitcoin je decentraliziran elektronski denarni sistem [1], ki ga je razvil Satoshi Nakamoto (za ime se ne ve, ali je pravo ali ne). Razvoj se je začel leta 2008, ko je Satoshi oktobra objavil članek [5] o razvoju kripto-valute. Prva transakcija se je zgodila januarja 2009.

Sistem je v obliki peer-to-peer, vse transakcije pa se zgodijo direktno med uporabniki, torej brez vmesnih posrednikov. Te transakcije so nato potrjene s strani omrežnih vozlišč in zapisane v javno distribuirano knjigo (angl. *ledger*), ki se imenuje tudi veriga blokov (angl. *blockchain*). Ker ni nobenega centralnega repozitorija ali administratorja, je Bitcoin imenovan tudi prva decentralizirana digitalna valuta.

Ker ni nobenega sistema, ki bi skrbel za kreacijo in distribucijo bitcoinov, je potrebno bitcoine rudariti. Rudarjenje je postopek, ki skrbi, da je veriga blokov konsistentna, popolna in nespremenljiva. Med samim rudarjenjem, se zbira različne transakcije, ki so se zgodile na omrežju. Te transakcije se združijo v t.i. bloke. Vsak blok nato vsebuje varnostno vsoto (angl. *hash*) prejšnjega bloka. Iz te povezave je veriga blokov tudi dobila svoje ime.

[1] https://bitcointalk.org/index.php?topic=576337

Da bi bil blok sprejet v celotnem omrežju, mora vsebovati tudi t.i. dokaz dela (angl. *proof-of-work*). Dokaz dela od rudarjev zahteva, da izračunajo številko, ki se imenuje žeton (angl. *nonce*). Žeton mora imeti tako vrednost, da je vrednost varnostne vsote SHA-256 nad celotnim blokov z žetonom manjša od trenutne zahtevnosti rudarjenja. Ta postopek je precej enostaven za vozlišče, vendar precej potraten za rudarja, saj mora preizkusiti veliko različnih vrednosti, preden dobi pravilno vrednost.

Glavne prednosti, ki jih ponuja Bitcoin v primerjavi s trenutno uveljavljenimi valutami so: odstranitev vmesnih posrednikov in posledično znižanje stroškov transakcij; druga pomembna prednost je odprtost sistema, celotna koda je na voljo vsem, vsak se lahko pridruži sistemu brez omejitev, transakcije se lahko zgodijo v katerem koli delu dneva; prednost je tudi anonimnost sistema, kar pa prinese tudi slabosti, saj se Bitcoin lahko uporabi za spletni kriminal.

1.2 Nenadzorovano učenje

Nenadzorovano učenje je področje strojnega učenja, kjer učni podatki niso predhodno označeni oziroma klasificirani v razrede. Cilj nenadzorovanega učenja je zaznati strukture in vzorce, ki se pojavljajo v podatkih. Ker učni podatki niso predhodno označeni v razrede, ni možno evaluirati uspešnosti učenja. Nenadzorovanega učenja se poslužujemo takrat, ko nimamo označene učne množice, oziroma označena učna množica ni dovolj velika za potrebe nadzorovanega strojnega učenja. Mnogokrat pa nenadzorovano učenje uporabimo v kombinaciji z nadzorovanim učenjem. Glavni primer takšnega pristopa so globoke nevronske mreže.

Najpreprostejši algoritmi nenadzorovanega učenja delujejo na principu gručenja, kjer osebke s podobnimi značilkami dodelimo v posamezno gručo. Primer takšnega algoritma sta gručenje k-means in hierarhično gručenje (angl. *hierarchical clustering*). Oba algoritma sta dokaj podobna. Algoritem k-means je podrobneje opisan v razdelku 2.1. Glavna razlika je ta, da imamo pri gručenju k-means konstantno število gruč, medtem ko je pri hierarhičnim gručenju gruča definirana rekurzivno. Na začetku predstavlja vsak osebek svojo gručo, nato pa gruče glede na predoločene metrike (Evklidska razdalja, Manhattanska razdalja, maksimalna razdalja, minimalna razdalja, ...) združujemo skupaj dokler ne ostane samo ena gruča. Tako združene podatke lahko predstavimo z dendogramom.

Nevronsko mrežo, ki se uči ne neoznačenih podatkih imenujemo avtoenkoder. Avtoenkoder je dokaj plitka nevronska mreža z majhnim številom skritih nivojev. Cilj avtoenkoderja se je z tehniko vzvratnega širjenja napake naučiti funkcijo $Y(\vec{X}) \approx \vec{X}$, pri čemer je $\vec{X} = [x_1, x_2, x_3, ..., x_n]$ učni primer oziroma osebek z n značilkami. Na prvi pogled je takšna funkcija trivialna, vendar če upoštevamo omejitev, da je število nevronov v skritem nivoju manjše od števila nevronov vhodnega nivoja, naloga le ni tako enostavna. Če imamo na vhodu samo šum, kjer značilke niso korelirane je takšna naloga nemogoča, če pa imajo podatki določeno strukturo s koreliranimi učnimi primeri, vzorci in koreliranimi značilkami, se avtoenkoder hitro priuči takšne strukture. Prednost takšnega pristopa je, da lahko nevronsko mrežo najprej nenadzorovano učimo na manjšem številu skritih nivojev, nato pa dodamo še dodatnih l nivojev in celotno globoko nevronsko mrežo učimo nadzorovano. Pri globokih nevronskih mrežah je problem ta, da se z večanjem števila nivojev nevronska mreža težje uči na skritih nivojih, ki so blizu vhodnemu nivoju. Posledično potrebujemo veliko število učnih primerov in veliko časovno zahtevnost. Če pa najprej prvih nekaj skritih nivojev naučimo nenadzorovano in potem dodamo še l skritih nivojev, se nevronska mreža z nadzorovanim učenjem uči hitreje in učinkoviteje na manjšem številu učnih primerov. Posledično so lahko nevronske mreže globoke tudi po več sto skritih nivojev.

V naslednjo skupino tehnik nenadzorovanega učenja uvrščamo metode, ki izločajo glavne značilke iz podatkov, ki vsebujejo veliko šuma. Tipičen primer takšnih učnih primerov so slike. Če imamo recimo sliko dimenzije 100×100 in upoštevamo vse kombinacije slikovnih točk, je zelo majhen delež takšnih slik, ki imajo neko vsebino. V to skupino uvrščamo algoritme kot sta metoda glavnih komponent (PCA) in razcep singularnih vrednosti (SVD). Pri metodi glavnih komponent najprej izračunamo korelacijsko matriko $C = X * X^T$, kjer je matrika X sestavljena iz učnih primerov. Lastni vektorji matrike C predstavljajo glavne komponente, lastne vrednosti pa moč glavnih komponent. Prednost takšnega pristopa je, da lahko učne primere uspešno rekonstruiramo že z prvimi nekaj glavnimi komponentami. Podobnega pristopa se poslužujemo pri razcepu singularnih vrednosti, kjer ustrezno skrčimo matriko singularnih vrednosti.

Področje strojnega učenja je zaradi univerzalne uporabnosti zelo razširjeno. Zato obstaja še mnogo različnih in naprednejših metod nenadzorovanega učenja. Vsak algoritem pa ima tudi svoje izpeljanke. Nenadzorovanega strojnega učenja se poslužujemo kadar nimamo označenih podatkov, ali pa želimo izluščiti strukturo in vzorce v podatkih. Ker je algoritmov veliko, se za algoritem nenadzorovanega učenja odločimo glede na vrsto problema in cilje, ki jih želimo doseči.

2. PREGLED PODROČJA

Večina študij zaznavanja anomalij uporablja označene učne množice, kjer so učni primeri klasificirani v enega izmed razredov *goljufiv* in *legitimen* [2, 3, 6, 8]. Prednost označevanja je ta, da se lahko poslužujemo algoritmov nadzorovanega učenja, hkrati pa lahko z metrikami uspešnosti klasifikacije ocenimo uspešnost modelov. Ker pa je omrežje Bitcoin dokaj novo, je le malo vseh transakcij, ki so bile zaznane kot prevara. Zaradi pomanjkanja učnih primerov je zato pristop nadzorovanega strojnega učenja neizvedljiv.

V študiji [7] uporabijo gručenje k-means, Mahalanobisovo razdaljo in nenadzorovano metodo podpornih vektorjev (SVM) za detekcijo goljufivih transakcij v Bitcoin omrežju. Za podatkovno množico uporabijo 100,000 Bitcoin transakcij. Podatke predstavijo kot graf, kjer so vozlišča grafa Bitcoin uporabniki, povezave pa predstavljajo transakcije med uporabniki. S to študijo uspešno klasificirajo 3 goljufive transakcije izmed znanih 30. Razširjena študija [7] je z metodami kot so Faktor lokalnih osamelcev (angl *Local Outlier Factor - LOF*) in k-means gručenja pridobila podobne rezultate. Prav tako so v študiji [9] z gručenjem pridobili 2 gruči, kjer je v prvi gruči večina Bitcoin uporabnikov. Ti so navadni uporabniki in žrtve goljufij, medtem ko so v drugi gruči zaznani uporab-

niki z goljufivimi transakcijami. V študiji [9] so prav tako rekosntruirali generičnega goljufivega uporabnika na podlagi vseh znanih goljufivih uporabnikov. S pomočjo umetno generiranih podatkov modeli v [9] dosegajo 76.5% uspešnost detekcije goljufij.

2.1 Gručenje K-means

Gručenje K-means, v nadaljevanju imenovano k-means, je najbolj popularna in razširjena metoda vektorske kvantizacije oz. gručenja. Namen metode je razdeliti nabor podatkov v k razdelkov tako, da so najbolj podobni podatki vstavljeni v iste razdelke. Vsak razdelek imenovan gruča (angl *cluster*) je predstavljen s točko v prostoru podatkov. Pripadnost podatkov gruči je določena z razdaljo od teh točk. Torej vsaka točka pripada gruči svoje najbližje točke.

K-means to doseže tako, da najprej naključno inicializira centre gruč znotraj prostora podatkov. Algoritem dodeli vsaki točki gručo glede na minimalno razdaljo do centra gruče. Na podlagi točk, ki pripadajo vsaki gruči, nato izračuna nove centre. Torej izračuna povprečje vseh točk, ki pripadajo posamezni gruči in tako dobi nove centre. Ponovno dodeli vsem točkam gruče ter izračuna povprečja. To ponavlja dokler se center ne neha spreminjati. Potek algoritma je razviden na sliki 1.

Največja težava k-means je, da konvergira proti lokalnemu ekstremu relativno na inicializacijo, kar je tudi razvidno na sliki 1. Dobimo lahko različne končne gruče ali pa se bodo oznake posameznih gruč spremenile. To težavo ponavadi rešimo tako, da algoritem večkrat poženemo in na koncu vzamemo najbolj pogosto rešitev, oznake pa lahko indeksiramo na koncu na lasten način, na primer od najmanjše do največje vrednosti prve koordinate. Druga težava pristopa je, da ponavadi ne vemo koliko gruč bomo potrebovali, da pravilno razdelimo prostor. Zato testiramo več možnosti. Algoritem je hkrati tudi zelo občutljiv na zunaj ležeče točke oziroma osamelce (angl. *outliers*). Ti povzročijo, da se centri prestavijo izven pravega mesta, saj ima vsak podatek enak vpliv na izračunani center. Algoritem predpostavi tudi, da so podatki znotraj posameznih gruč porazdeljeni po normalni porazdelitvi, kar ni vedno res.

2.2 Porezano gručenje K-means

Porezano gručenje k-means rešuje problem zunaj ležečih točk oziroma osamelcev. To doseže z dodatno kazensko funkcijo (angl. *penalty function*) s parametrom α. Na podlagi parametra α in razdalje posameznih točk do njihovi centrov gruč kazenska funkcija pri računanju novih centrov izloči točke, ki so preveč oddaljene od njihovih centrov. Rezultat te spremembe je bolj robustno delovanje algoritma ter bolj stabilni rezultati.

3. POVZETEK ČLANKA

Za iskanje anomalij so avtorji [4] uporabljali spremenjeno verzijo gručenja k-means, opisanega v poglavju 2.1, imenovanega porezano gručenje k-means (angl. *trimed k-means clustering*). Za razliko od klasičnega gručenja k-means ta verzija ni občutljiva na osamelce. Pri izgradnji gruč na podlagi parametra α izloči točke, ki so dovolj oddaljene od centrov gruč. Rezultat tega so bolj stabilni centri gruč. Algoritem je bolj natančno opisan v poglavju 2.2.

Točke, ki so bile izločene iz gruč se definira kot anomalije in označi kot možne prevare.

3.1 Podatkovna množica

Za podatkovno množico so avtorji [4] uporabili bazo laboratorija za računsko biologijo Univerze v Illinois. Podatkovna množica vsebuje podatke 6336769 uporabnikov. Med uporabniki je pa bilo 37450461 povezav, ki predstavljajo transakcije. 30 teh transakcij je bilo zlonamernih. V glavnini podatki niso bili označeni ter se zato za izgradnjo niso uporabili.

3.2 Uporabljene značilke

Glede na zbrane podatke o transakcijah znotraj bitcoin omrežja, so zgradili 14 značilk, ki bi pomagale pri izgradnji gruč. Značilke so okvirno razdeljene na 3 skupine:

- Značilke vezane na denarno vsoto: skupna poslana vsota, skupna prejeta vsota, povprečna poslana vsota, povprečna prejeta vsota, standardni odklon poslane vsote, standardni odklon prejete vsote.
- Omrežne značilke: vhodna stopnja, izstopna stopnja, koeficient nakopičenosti, število omrežnih trikotnikov.
- Značilke vezane na povprečno soseščino. To so značilke, ki opisujejo transakcije med njimi glede na smer transakcij. Torej transakcije izven soseščine, znotraj soseščine, iz soseščine in v soseščino.

3.3 Predprocesiranje

Znotraj podatkov so bili uporabniki, ki so samo pošiljali oz. prejemali transakcije. To je povzročilo manjkajoče podatke v končni bazi. Manjkajoče podatke so zato zamenjali z ničlami, kar je bilo identično kot če bi prejeli 0 BTC.

Podatke so tudi normalizirali tako, da je njihova povprečna vrednost nič in varianca ena.

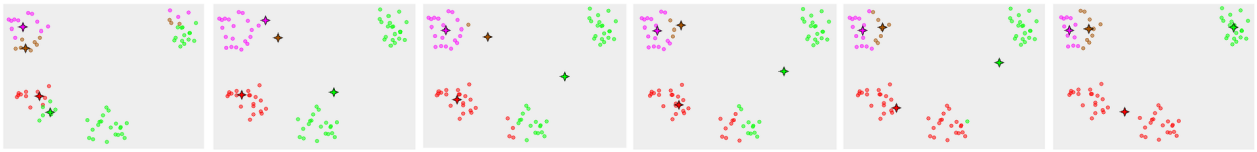
3.4 Rezultati

3.4.1 Gručenje k-means

Zaradi pomankanja znanja glede strukture gručenja Bitcoin omrežja, je bil prvi korak študije [4] ugotoviti optimalno število gruč k. Kot metrika je bila uporabljena vsota kvadratov znotraj gruč (angl. *within sum of squares*). Algoritem k-means je bil tako pognan za različne vrednosti k, pri čemer je za vsak k, zaradi nedeterminističnega obnašanja, algoritem pognan petkrat. Število gruč so v študiji omejili na število 15. V praksi je lahko število gruč tudi večje, vendar pa so se v študiji omejili in tako zmanjšali časovno kompleksnost.

Slika 2 prikazuje, da je optimalno število gruč zagotovljeno pri $k = 8$. Čeprav bi bila dobra izbira tudi $k = 4$, je vse do osmih gruč veliko padcev, krivulja pa se nato poravna. Ti rezultati se skladajo tudi z [7].

Slika 3 prikazuje porazdelitev posameznih primerov znotraj gruč. Porazdelitev prikazuje, da skoraj 60% primerov pripada gruči 8. Slika 4 prikazuje grafično porazdelitev gruč



Slika 1: Tipičen potek algoritma k-means.

Table 1: Izbrani atributi centrov k-means gruč

Gruča	Povprečna poslana	Povprečna prejeta	Koeficient nakopičenosti	Izhodna stopnja
1	2.99	2.99	0.50	9.24
2	1.92	1.97	0.61	4.77
3	0.26	0.24	0.70	2.21
4	99.63	107.44	0.23	5.56
5	87.27	64.98	0.31	7.38
6	41.00	36.44	0.12	16.00
7	98.51	67.48	0.00	532534
8	9.50	17.90	0.00	477035

na prvih dveh glavnih komponentah. Zaradi neobčutljivosti k-means algoritma na osamelce, sta dva skrajna primera združena v eno samo gručo.

Tabela 1 prikazuje centre k-means gruč za naslednje atribute: povprečna poslana vsota, povprečna prejeta vsota, koeficient nakopičenosti in izhodna stopnja. Opazimo, da pri prvih šestih gručah nobeden parameter ne dominira. Gruči 7, 8 imata visoki izhodni stopnji in nizek koeficient nakopičenosti. Kar pomeni, da imata v povprečju ti dve gruči ogromno izhodnih transakcij in slabo povezano soseščino. Opazimo lahko, da imajo gruče z višjim koeficientom nakopičenosti v povprečju tudi nižje transakcije.

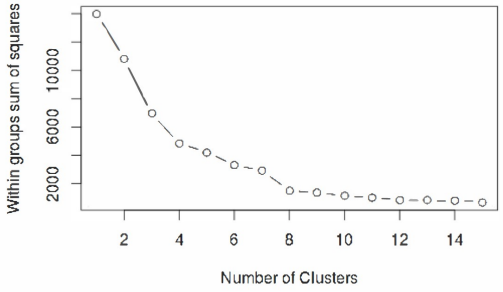
3.4.2 Porezano gručenje k-means

Tudi tukaj je prvi korak ugotoviti optimalno število gruč in parameter α. Čeprav je dejansko število osamelcev neznano je parameter α postavljen na $\alpha = 0.01$. Taka vrednost parametra je pogosta v mnogih študijah o finančnih goljufijah. Optimalno število gruč je pri $k = 8$. Osamelce predstavimo s posamezno gručo 9.

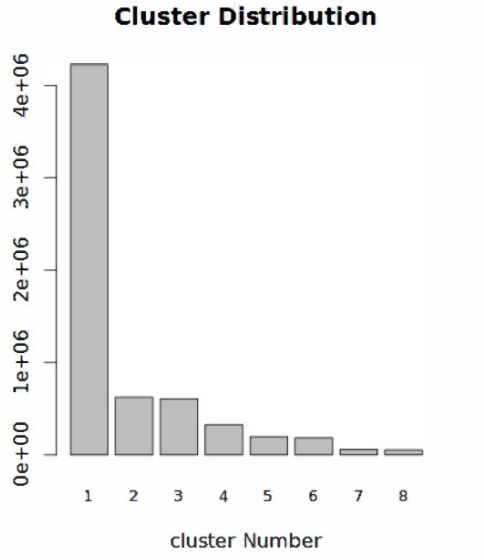
Distribucija posameznih primerov znotraj gruč prikazuje slika 5 z prikazano deveto gručo osamelcev. Čeprav močno prevladuje osma gruča, lahko opazimo, da so objekti glede na navaden algoritem k-means enakomerneje porazdeljeni po posameznih gručah zaradi porezanih osamelcev.

Delež osamelcev je označen na sliki 6 z oznako "o". Iz slike je razvidno, da so osamelci porazdeljeni med večini gručami. Medtem ko slika 7 prikazuje grafično ponazoritev gruč na prvih dveh komponentah brez osamelcev.

Iz tabele 2 lahko opazimo, da imajo osamelci visoke transakcije in veliko število izhodnih povezav.



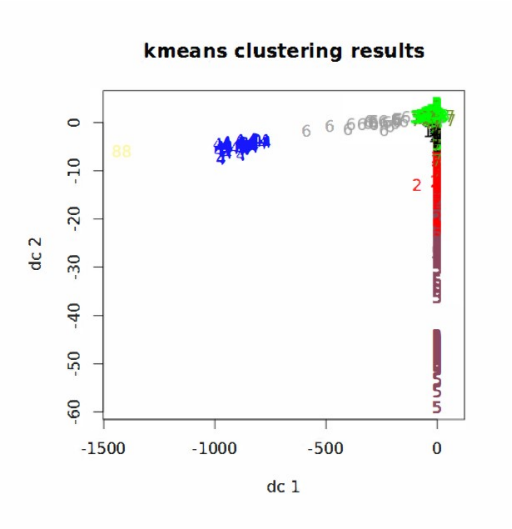
Slika 2: Graf prikazuje optimalno število gruč pri $k = 8$.



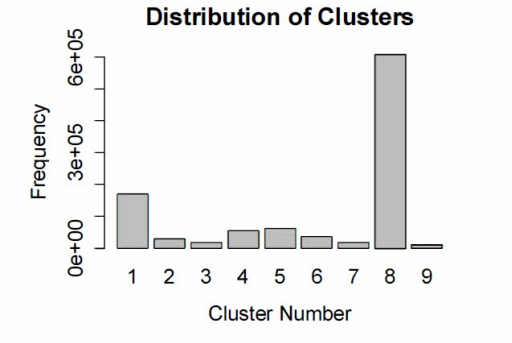
Slika 3: Porazdelitev primerov znotraj posameznih gruč pridobljenih z algoritmom k-means.

Table 2: Izbrani atributi centrov porezanih k-means gruč

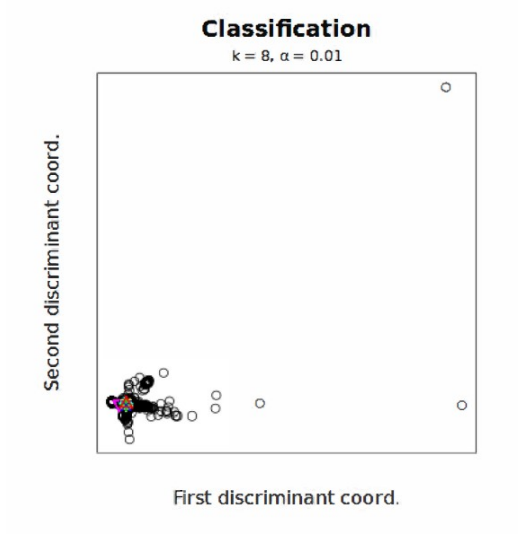
Gruča	Povprečna poslana	Povprečna prejeta	Koeficient nakopičenosti	Izhodna stopnja
1	30.01	29.14	0.26	6.10
2	0.04	0.03	0.70	2.00
3	1.70	2.29	0.55	8.19
4	39.84	35.24	0.14	8.63
5	30.51	42.54	0.60	4.92
6	77.04	51.59	0.98	3.81
7	1.02	1.05	0.61	4.51
8	25.86	24.73	0.01	10.45
9	2217.36	1886.70	0.49	1508.89



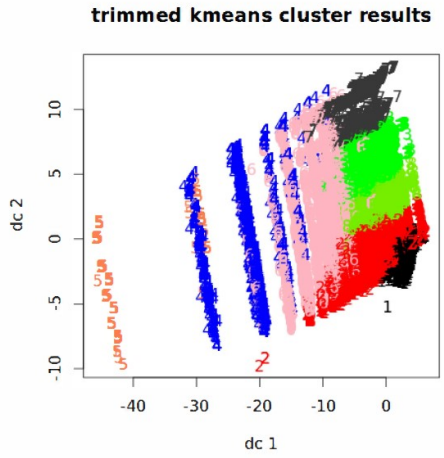
Slika 4: Grafična ponazoritev gruč pridobljenih z k-means.



Slika 5: Porazdelitev primerov znotraj posameznih gruč pridobljenih z algoritmom porezanega gručenja k-means.



Slika 6: Grafična ponazoritev gruč pridobljenih z algoritmom porezanega gručenja k-means z osamelci.



Slika 7: Grafična ponazoritev gruč pridobljenih z algoritmom porezanega gručenja k-means brez osamelcev.

4. ZAKLJUČEK

Rezultati nam pokažejo razliko med dvema algoritmoma za gručenje, v odvisnosti od osamelcev. K-means se zaradi velike občutljivosti na anomalije občasno obnaša nepričakovano. V primeru članka se je to videlo tako, da je klasični K-means našel gruče velikosti 2, kar ni bilo optimalno. Z uporabo porezanega K-means gručenja, so se takšne gruče porazgubile in rezultati so se izboljšali.

Algoritem je bil testiran na znanih podatkih, kjer so imeli 30 že znanih prevar. Iz podatkov so tako izluščili značilke, ki so vsebovale vse atribute potrebne za delovanje algoritma in so bile omenjene v prejšnjih poglavjih. Izluščili so tudi 76 transakcij, ki so nastopale v primeru prevar. Te transakcije so bile primerjane z ostalimi (37000000).

Za odkrivanje goljufij je bilo potrebno podatke še primerjati. Primerjali so podatke, ki so jih pridobili iz 30 znanih prevar in podatke pridobljene iz transakcij. Algoritem je uspešno ugotovil 5 do 30 primerov. Ti primeri so: Mt Gox, Linode Hack, Stone Man Loss, Allinvain in 50 BTC Theft.

Rezultati te raziskave kažejo na to, da se rezultate podobnih člankov da izboljšati. Poleg tega, članek poudari tudi to, da se z razvojem tehnologije, povečuje varnost udeležencev, ki opravljajo transakcije v peer-to-peer omrežjih. Na podlagi te izboljšane varnosti pa se povečuje tudi zaupanje v finančno omrežje Bitcoin in je korak v pravo smer k vsesplošni uporabi.

4.1 Predlogi izboljšav

V tem kratkem podpoglavju opišemo predloge izboljšav, ki so se nam porodile med branjem članka.

- Selekcija značilk: v članku natančneje opišejo, kako so zgenerirali določene značilke, natančneje pa ne opišejo ali so preizkusili tudi izbiro različnih značilk. Morda je ena njihovih zgeneriranih značilk prinesla več škode kot koristi in bi z njeno odstranitvijo rezultate še izboljšali.
- Izbor drugačnih algoritmov za gručenje: obstajajo tudi drugi algoritmi, kot je k-means, ki dosegajo boljše rezultate. Smiselno bi bilo preizkusiti npr. *Affinity Propagation*, ki ne potrebuje števila gruč, temveč jih določi sam ali pa Expectation–maximization, ki bolje modelira distribucije.

5. REFERENCE

[1] S. Barber, X. Boyen, E. Shi, and E. Uzun. Bitter to better—how to make bitcoin a better currency. In *International Conference on Financial Cryptography and Data Security*, pages 399–414. Springer, 2012.

[2] S. Bhattacharyya, S. Jha, K. Tharakunnel, and J. C. Westland. Data mining for credit card fraud: A comparative study. *Decision Support Systems*, 50(3):602–613, 2011.

[3] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15, 2009.

[4] P. Monamo, V. Marivate, and B. Twala. Unsupervised learning for robust bitcoin fraud detection. In *Information Security for South Africa (ISSA), 2016*, pages 129–134. IEEE, 2016.

[5] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[6] E. Ngai, Y. Hu, Y. Wong, Y. Chen, and X. Sun. The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature. *Decision Support Systems*, 50(3):559–569, 2011.

[7] T. Pham and S. Lee. Anomaly detection in bitcoin network using unsupervised learning methods. *arXiv preprint arXiv:1611.03941*, 2016.

[8] M. A. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko. A review of novelty detection. *Signal Processing*, 99:215–249, 2014.

[9] D. Zambre and A. Shah. Analysis of bitcoin network dataset for fraud. *Unpublished Report*, 2013.

Behind Closed Doors: Measurement and Analysis of CryptoLocker Ransoms in Bitcoin

Petr Barta
Faculty of Information Technology,
Czech Technical University in Prague

Nemanja Soldo
Faculty of computer and information science,
University of Ljubljana

***Abstract* — This paper analyze ransom payments required by CryptoLocker which were done using Bitcoin. In the Introduction the basic description of cryptocurrency Bitcoin and ransomware CryptoLocker is provided. Firstly the cluster of Bitcoin addresses which belong to Cryptolocker was gathered, then the analysis of incoming transactions of these addresses were made. These incoming transactions was filtered (by amount which was usual in given time) and then several conclusions about the typical targets (and especially their geographical location) of CryptoLocker attacks. Later, we tried to replicate part of original research with lately appeared WannaCry ransomware which affected lot of companies and institutions in Europe.**

***Index Terms* — Bitcoin, CryptoLocker, ransomware**

I. INTRODUCTION

During the years of rapid development in IT industry, there were notable notion of tries to create universal digital currency that could change the way how people are using money, and most notably, how whole economy of country could be changed, under influence of new paradigm of global trends and changes, which generally started with bigger consumerism of technology and with wider expansion of information technologies.

However, biggest turn over happened in 2008, when anonymous programmer famous under pseudonym Satoshi Nakamoto proposed new solution, called Bitcoin (1). Bitcoin actually was not some great invention of new crypto systems and encryptions, it was just smart way of using existing systems differently, so he introduced distributed public ledger that serializes a record of all confirmed transactions, called the blockhain. Blockchain enabled whole Bitcoin network to operate under a decentralized peer-to-peer system, where users are identifiable by public keys, or more commonly referred to as Bitcoin addresses.

That way, this system was almost untraceable, which provided privacy and anonymousness for end users. In one hand, that is a good thing for normal, regular users, but in other hand, users which are using Bitcoin as a part of criminal activities and fraud are hard to reveal and trace, because their Bitcoin addresses are not connected to them in real life.

As a result of intractability, many financial frauds and lot of paying for illegal stuff was conceived through Bitcoin[1] network for couple of years. Examples are many deep web online illegal marketplaces like Silkroad, other drug selling and illegal activity funding sites. Result was massive rise in popularity and scale in Bitcoin usage and also value, which attracted attention of authorities and public, to examine it in more detail, to distinguish between legal and illicit use of Bitcoin as a paying instrument.

Although some efforts were made to make job of trace revealing to authorities harder, like Bitcoin mixers (Bitcoin Fog (2)), some privacy-enhancing overlays like Coinjoin(3), for couple of years now, Bitcoin is not considered full untraceable anymore, because there are some traces that could connect virtual addresses to real user, to its forum usernames, online marketplaces, Bitcoin exchanges and services.

From a forensics perspective, this is a major relief, considering that a number of crimes which included bitcoin transfers were in constant rise for some time. One of most famous, and even now pretty popular crime activity involving Bitcoin as a mean of payment, is deployment of ransomware CryptoLocker, which is using strong encryption to encrypt and lock files on victim's system and demands a ransom to be paid over Bitcoin network, in order to get key to decrypt data. This papers uses that example as a base to tackle mechanisms in the backend part of ransom paying, and to describe money laundering schemes and methods involved.

II. BACKGROUND

As a good technical introduction to this thematic, in next part of the paper, we will say something about Bitcoin and Bitcoin transactions and CryptoLocker in more detail.

A. Bitcoin and Bitcoin transactions

Back in 2008, anonymous creator, called Satoshi Nakamoto introduced Bitcoin in his paper, and this became the most serious attempt to include cryptocurrency into our everyday lives. Idea of digital money existed for couple of decades, but there were no as nearly popular or widely used cryptocurrencies as Bitcoin became. Nakamoto's idea was to

[1] When it is referred as "Bitcoin", that is pointing on the Bitcoin network and whole system in general, and when it is written "bitcoin" it is unit of currency (or shortly "BTC")

create decentralized, open-source digital currency, which will provide high security and privacy, without big transaction fees which exists while using real currencies. It uses peer-to-peer technology, so no centralized servers, authorities, whole service is managed and maintained by series of computers on network, so nobody owns or controls it.

Bitcoins are created as a reward for payment processing work in which users who "lend" their computers as a tool for doing that processing and recording payments into public ledgers get rewarded (this process is called mining). Miners are digitally signing records of transaction by using specialized software which uses their computer processing capacity to solve complex algorithms used in transaction processing.

Except mining, bitcoins could be bought (exchanged) for other currencies, products and services.

Bitcoin transactions represent transfer of value between Bitcoin wallets that gets included in the block chain. Wallet keep a private key, or seed, which is used to sign transactions, providing a mathematical proof that they have come from the owner of the wallet, ensuring security. Transactions are public, they are broadcasted between users, and it usually took community (miners) about 10 minutes to verify and confirm them, and include them in the block chain. Block chain means that inputs to a new transaction must reference the exact value of outputs of previous transaction, thus forming a chain of transactions. Validity of transaction is dependent on each signature in the transaction chain, so verifying of transaction history is simple, but tampering with confirmed transaction which are deeply embedded in the blockchain is hard.

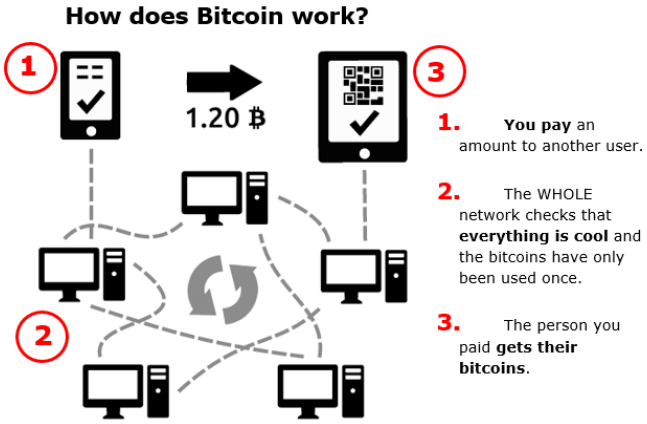


Figure 1. Simple diagram showing how Bitcoin work

B. *CryptoLocker Ransomware*

Cybercriminals are constantly looking for ways to evolve their malware. Evolution is the key for survival because antivirus research, analysis, countermeasures, and public awareness thwart the efficacy of malware and its spread. Starting back in 2013, CryptoLocker(4) was invented as new type of extortion. It is pervasive and it's attacking on victim's biggest fear: loosing data. Unlike previous Ransomwares, which locked operating system and left data untouched and easily recoverable, Cryptolocker is using strong encryption algorithms to lock even complete storage, and make it almost impossible to retrieve files without attacker's private key. If attack succeed, attacker is demanding some payment (ransom) in order to send password and allow victim to decrypt its files. Most often, attackers set deadline after which encryption key is lost.

Usually, victims were infected by receiving and opening spam mails. That is the most common form of infection by CryptoLocker. Malicious executable files are attached in ZIP archives and if opened they would encrypt all files and show victim on-screen warning that he is infected and how, when and where to pay ransom in order to get key for decryption. Most of ransom demands are to be paid in Bitcoin or MoneyPak.

Figure 2. Example of screen with ransom demand after infection

Second way of infecting, introduced with newer versions of CryptoLocker, was sending ransomware over Gameover ZeuS, a peer-to-peer botnet that used Cutwail spam botnet to send a lot of fake spam mail impersonating famous online resellers or financial institutions, with various invoices, order confirmations and warnings attached that should make victim to open them.

III. MEASUREMENT METHODOLOGY

Since we are interested in damages caused by CryptoLocker (and its authors), we first need to know Bitcoin addresses they are using. The authors of the original paper firstly found two CryptoLocker Bitcoin addresses on Reddit and they tried to find as much other addresses as possible. To do so, they've tried two methods:

- Multi-input transactions and
- Change addresses.

These methods are based on the essentials of Bitcoin transactions:

As can be seen in Transaction C on Figure 3, there are two inputs. Both these inputs are coming from the same user, so the input addresses can be easily paired. Also there is an output change address. This is because sum of inputs is 2.5 BTC but the transferred amount should be just 2.0 BTC, so 0.5 BTC is returned to the user who initiated the payment – and again, this means output change address belongs to the same user.

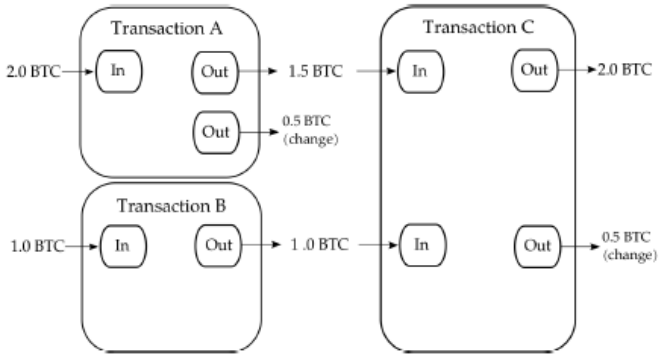


Figure 3. Anatomy of Bitcoin transactions

Because there is a chance these addresses were not used just for the purpose of accepting ransom money, the authors filtered those payments. For this filtering they introduced Ransom Identification Framework.

It needs to be said that ransom amounts are not picked randomly, but (however it may sounds funny) it is business as any other, so "the law of supply and demand" applies here. Authors of CryptoLocker are obviously trying to select the best price such that they make as much money as possible. Because of the the exchange rate BTC/USD is changing rapidly (usually in behalf of BTC), they had to decrease the price of the decryption key such that victims would be willing to pay for it. As the consequence of this, we know the amount which was required as a ransom in given time period (Table 1). In addition to these "regular" ransoms, they also introduced "CryptoLocker Decryption Service" for victims who failed to pay ransoms within the given time frame (usually 3 days) – up to 10 BTC.

Table 1. Required ransom in given time period

Time period	Ransom amount
5. 9. – 7. 11. 2013	2 BTC
8. 11. – 9. 11. 2013	1 BTC
10. 11. – 23. 11. 2013	0.5 BTC
24. 11. – 31. 12. 2013	0.3 BTC

So the aforementioned Ransom Identification Framework is using data from Table 1 to search for the relevant transactions only.

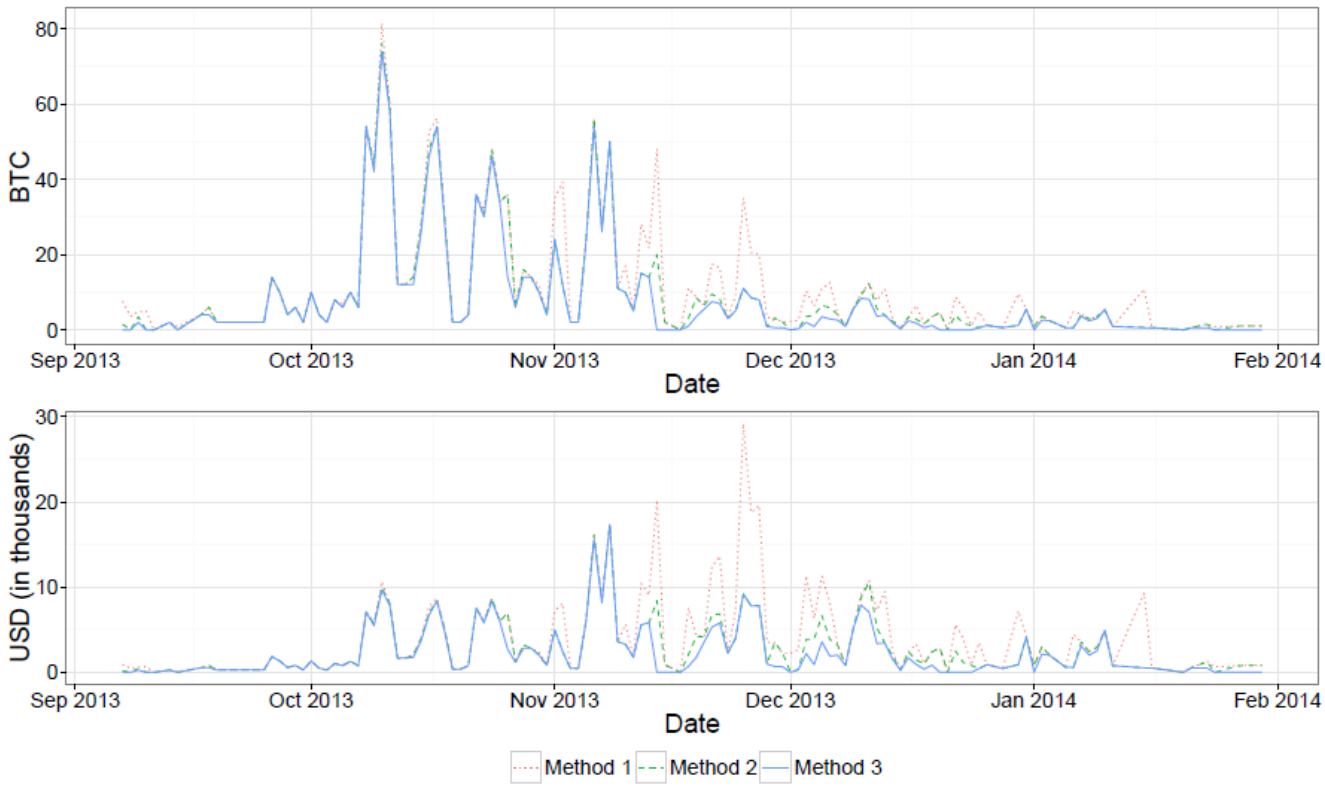
IV. DATA OVERVIEW

Let's look a bit on data gained in previous part. For this purpose we will compare three summarizations of the transactions incoming to CryptoLocker Bitcoin addresses. The first method simply sums up all incoming transactions (regardless of their date or amount). The second method considers just the transactions with amount equal to either 2.0, 1.0, 0.5 or 0.3 BTC (but regardless of their date). The last method uses aforementioned Ransom Identification Framework (i.e. transactions whose amount fits time period in Table 1).

Table 2. Transactions summaries

Method	Transactions	BTC	USD
1	1071	1541	539 080
2	933	1257	373 934
3	795	1128	310 472

As can be seen from the figure, the graph of Method 2 is pretty much the same as the Graph of Method 3. This means the time



periods in Table 1 are reasonable. Because of the varying BTC/USD exchange rate, whose peak in given time period was at the end of November 2013 (1332 $ per 1 BTC), we estimate that the peak valuation of the CryptoLocker economy occurred on 29th of November 2013. They had collected a total of 1044 BTC worth approximately 1.18 million USD.

Because of limited period of time in which the demanded ransom can be paid (after files encrypting procedure is done), we can easily estimate the waves of attack (assuming the ratio of victims who paid the ransom to all victims is more or less constant). The number of ransoms paid each day in given time period can be seen in Figure 4.

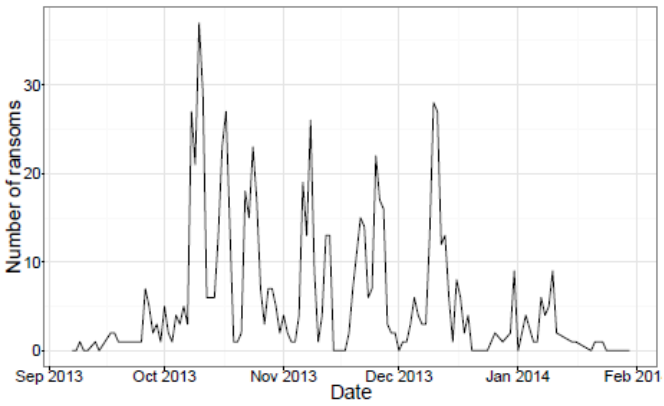


Figure 4. Number of ransoms paid per day

V. DATA ANALYSIS

Goal of this paper is to try to gain insight on CryptoLocker's targets and changes in targets throughout its operations by using statistical methods to determine distributions in the times of day that different ransom types were paid to $S_{CL.}$ For this purposes Kolmogorov-Smirnov tests are used on ransom timestamps. For this study, two-sample test(5)(6) are used to determine whether or not samples from different ransom types come from different populations at the 99.5% confidence level. Sample populations included 2BTC, 1BTC, 0.5BTC, 0.3BTC and 0.6BTC ransoms.

Analysis of obtained Timestamp Data

Obtained statistical results are presented in graphs in Figure 4. Assumption is that CryptoLocker mostly targeted business professionals, so it is expected that ransom payments should be transacted during typical working hours (9:00AM to 5:00PM e.g.). In Pacific Time (PT, UTC-08:00), 9:00 a.m. to 5:00 p.m. would correspond to 17:00 UTC to 1:00 UTC on the following day. In Eastern Time (ET, UTC-05:00), 9:00 a.m. to 5:00 p.m. would correspond to 14:00 UTC to 22:00 UTC. Kolmogorov-Smirnov tests showed that timestamps samples of 2BTC ransoms were different from the 1BTC and 0.3BTC ransoms. Also, 1BTC ransom differs from 0.3BTC ransoms, thus we know that all those samples come from different populations.

In Great Britain for example, by CTU researcher's study (4), in observed period from October 22, 2013 to November 1, 2013, they registered about 1700 CryptoLocker attacks, which is most after the USA. If we use same working hours like in USA, but adjusted time zone, it is expected for ransom payments to be concentrated somewhere between 09:00-17:00 UTC, which is correct with the first and third quartiles for the 2BTC ransom sample (Table 3), but further investigation is needed to make stronger claims.

For 1BTC sample, from table we could see that median timestamp is 17:07 UTC, with interquartile range of approx. 7 hours, and with unimodal distribution, so expectation is that majority of 1BTC ransoms were paid from the USA.

For 0.3BTC sample, because of its bimodal probability distribution which showed that ransoms came from 2 different sources it is a bit trickier to predict origin countries. One of sources had a bit similar distribution as for 2BTC sample, so we could connect those payments to Great Britain, but for second one, considering dispersion of attacks at the end of 2013, and with adjusting time zone, researchers came out with a great possibility that large portion of 0.3BTC payments origin from Australia (Australia was 3rd most attacked country in observed period and time interval of 0.3BTC ransom payments fit in Australia working hours, by adjusting time zone of obtained statistical interval to UTC +10:00, which corresponds to the Australian Easter Time Zone).

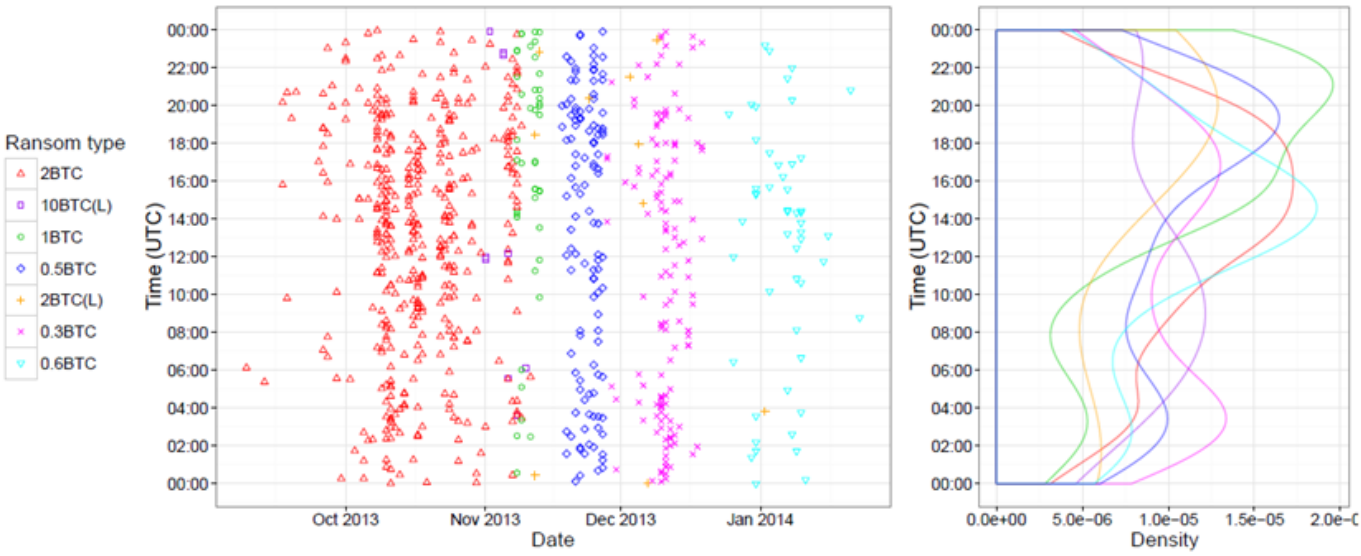


Figure 5. Plots showing trends in which parts of day ransoms were usually paid

Table 3. Statistics of timestamps

Sample	Min.	Q_1	Median	Q_3	Max.	Mean
2 BTC	00:01:20	08:55:36	13:55:06	18:01:00	23:58:33	13:12:42
1 BTC	00:34:31	14:11:09	17:07:49	21:11:29	23:54:33	16:28:47
0.5 BTC	00:07:27	05:51:07	15:16:48	19:23:24	23:55:19	13:14:47
0.3 BTC	00:06:59	04:15:56	11:33:30	17:29:14	23:54:45	11:16:19
0.6 BTC	00:00:01	08:27:37	13:53:14	16:54:00	23:11:33	12:36:46

VI. OUR RESEARCH

A. *Introduction*

Inspired by the work made by authors of the original paper, we've decided to perform similar research on lately released ransomware – WannaCry.

B. *WannaCry*

As all the other ransomware viruses, also WannaCry, once it infiltrates victims computer, the encryption of files begin. WannaCry is deciding which files to encrypt according to theirs file extensions. Basically all the documents (e.g. pdf, docx, odt), audio files (e.g. mp3, flac, ogg), video files (e.g. mp4, avi), databases, emails etc. When the encryption is finished, the window with information what happened and what to do decrypt files appear. Victims - referred as "customers" - are supposed to pay ransom ($ 300 in Bitcoins) in three days. If they won't pay in these three days, the ransom amount doubles. If they won't pay in one week, the files are gone "forever" (or at least till some point in future when some decryption tool will be available).

The spreading of the virus started on Friday May 12th, 2017. It was propagating using EternalBlue exploit, which uses vulnerability of Server Message Block protocol. Computer security expert known as "MalwareTech" make the virus harmless, because he unintentionally discovered the "kill switch". Early the new version of WannaCry (without the "kill switch") was released, though.

C. *Gathering the BTC addresses*

We have got one BTC address from the aforementioned instructions screen, but we cannot rely on the fact it is the only address used by WannaCry authors. Therefore we've tried to extract printable strings out of the virus *exe* file using linux command *strings*, while we are interested just in strings which 26 to 35 characters long, what is length of valid BTC address. Using this command we have extracted three BTC addresses:

115p7UMMngoj1pMvkpHijcRdfJNXj6LrLn
12t9YDPgwueZ9NyMgw519p7AA8isjr6SMw
13AM4VW2dhxYgXeQepoHkHSQuy6NgaEb94

Since we cannot be sure these addresses are the only ones used (the virus can for example request for the BTC address using internet), so we've made some research on the internet, but every addresses we found was one of those three aforementioned. So however it should not be considered as a proof, we can assume those are only used addresses.

D. *Transaction history*

Using the website Blockchain we have exported transaction history of these BTC addresses. By performing some scripting we have eliminated all the transactions under the 0.08 BTC (because it was lowest amount of BTCs equivalent to $ 300 in monitored time period) and we have made a histogram of total incoming payments by the hour of the day.

In total, there were 337 transactions and 50.459 BTC revenue.

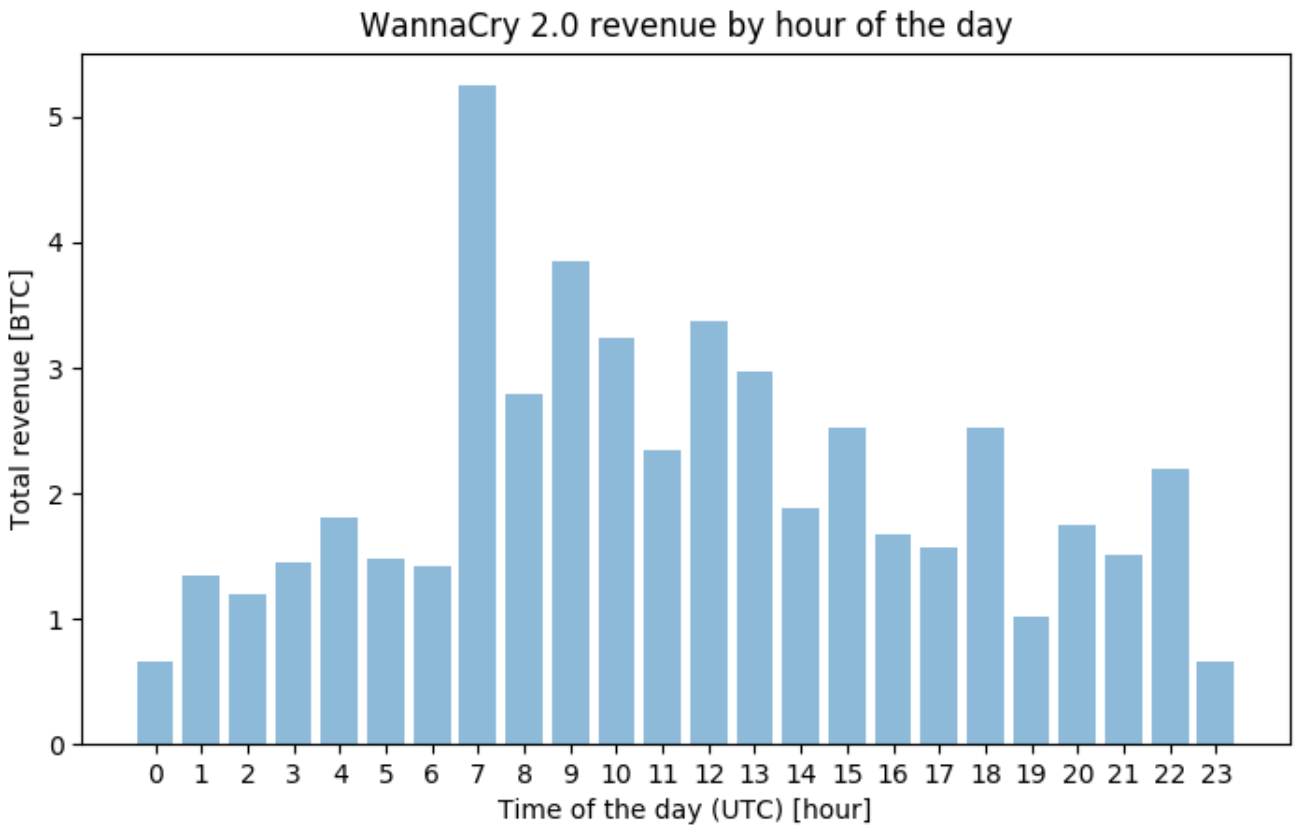


Figure 6. Ransom revenue per hour

E. Data analysis

As can be seen in histogram the peak of payments was between 7 AM and 2 PM UTC what is approximately equal to European day working hours. Day working hours in the America are approximately between 1 PM and 11 PM and as can be seen it is not really peak of payments. Hence we can assume most affected area was Europe. According to (7) the far most affected country was Russian federation, which is in correlation with our results.

VII. CONCLUSION

This paper is a presentation of both Bitcoin and CryptoLocker, with background behind both and with explanation of mechanisms behind CryptoLocker attacks. Through analysis of ransomware attacks and blockchain analysis, we could get idea what is happening with ransom payments, how they could be traced, is there notable possibility to deanonymize attackers by following Bitcoin trails and such. This paper also presented analysis on CryptoLocker's economy and financial infrastructure, analysis of ransom timestamps by statistical methods to form conjectures on regions where CryptoLocker attacks were prevalent in observed periods and made some predictions about relationship of distinct payments to gather as much evidences as possible trying to extract some valuable information about attackers themselves. Later, we did similar research with lately appeared WannaCry ransomware to try and repeat a bit of original research and obtained some results which are in correlation with part of original research.

However, Bitcoin structure and hard traceability are making forensics job for specialist harder, but recent studies showed that there are going to be more and more methods to deanonymize Bitcoin, and that some data about origins of payments could be extracted and traced. Considering that ransomware attacks are one of hot topics in domain of IT security (which we again saw with lately carried WannaCry attack), with increase in registered successful attacks, more efforts should be put in enhancing security of organizations and individuals and doing prevention work to lower chance of getting your data encrypted, or if attack succeed find proper mechanisms and ways to track attackers.

REFERENCES

(1) S. Nakamoto. (2012) Bitcoin: A peer-to-peer electronic cash system. [Online]. Available: http://www.bitcoin.org/bitcoin.pdf
(2) M. Moser, R. Bohme, and D. Breuker, "An inquiry into money laundering tools in the bitcoin ecosystem," in eCrime Researchers Summit (eCRS), 2013. IEEE, 2013, pp. 1–14.
(3) S. Meiklejohn and C. Orlandi, "Privacy-enhancing overlays in bitcoin," in Financial Cryptography and Data Security. Springer, 2015, pp. 127–141.
(4) K. Jarvis. (2014) Cryptolocker ransomware, 2013. [Online]. Available: http://www.secureworks.com/cyber-threat-intelligence/threats/cryptolocker-ransomware/
(5) F. J. Massey Jr, "The kolmogorov-smirnov test for goodness of fit," Journal of the American statistical Association, vol. 46, no. 253, pp. 68–78, 1951.
(6) I. T. Young, "Proof without prejudice: use of the kolmogorov-smirnov test for the analysis of histograms from flow systems and other sources." Journal of Histochemistry & Cytochemistry, vol. 25, no. 7, pp. 935–941, 1977.
(7): https://securelist.com/78351/wannacry-ransomware-used-in-widespread-attacks-all-over-the-world/

Del III
Omrežna forenzika

Preučevanje metod in primerov z uporabo odločitvene analize z namenom izbire najboljšega orodja za mobilno forenziko [9] *

Veronika Blažič
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana, Slovenija

Matej Dolenc
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana, Slovenija

POVZETEK

Mobilne naprave v današnjem času najdemo povsod. Uporabljajo se za zabavo, učenje, finančne transakcije, poslovne namene in še bi lahko naštevali. Zaradi velikega števila naprav (več kot 1 mobilna naprava na osebo dandanes ni nič nenavadnega) to privede do velikega digitalnega odtisa vsakega posameznika. Posledica tega pa je zato vedno večja količina ilegalnih dejanj, ki se tičejo tudi mobilnih naprav.

Naučiti se moramo, kako ta dejanja najučinkoviteje identificirati in preprečiti. V tem članku sta predstavljeni in ovrednoteni dve orodji, s katerima si pomagamo pri soočanju s to težavo. Orodje za forenzično analizo izberemo v fazi priprave (*angl. preparatory phase*) v samem postopku digitalnega preiskovanja mobilne naprave. V primeru, da ne izberemo najbolj primernega orodja za izvedbo preiskave, lahko to hitro pripelje do nepopolne in nepravilne analize digitalnega dokaza.

Orodji sta ocenjeni z dvema faktorjema in sicer z *ustreznostjo tipa dokaza* (v smislu koliko pozitivnega doprinese tip digitalnega dokaza k preiskavi) ter z *zmogljivostjo orodja* (v smislu zmogljivosti posameznega orodja glede na tip digitalnega dokaza). V tem članku sta opisani orodji *XRY* (alternativa 1, Alt1) in *UFED* (*Universal Forensic Extraction Device*, alternativa 2, Alt2), pri čemer je dokazano, da je orodje XRY v večini primerov boljše od orodja UEFD.

Ključne besede

forenzika mobilnih naprav, zloraba, digitalni odtis, orodja za forenzično preiskavo

*Celoten članek je na voljo na spletni strani: `http://digitalcommons.newhaven.edu/electricalcomputerengineering-facpubs/52/`

1. UVOD

Cilj tega članka [9] je izbira najbolj primernega orodja za digitalno analizo. Izbira temelji na zmogljivosti orodja glede na tip digitalnega dokaza in ustreznost tipa digitalnega dokaza pri sami preiskavi. V kolikor pravilno izberemo orodje, bo to pozitivno vplivalo na verjetnost pravičnega sojenja. V tej fazi se že vidi pomembnost, ki jo igrajo digitalni forenziki pri sami preiskavi. Če dokazi niso pravilno analizirani, to lahko privede do nepravičnega sojenja.

Poglejmo si nekaj dejstev. Mobilne naprave so vseprisotne, imajo velik vpliv na človekovo življenje in posledično predstavljajo digitalni odtis posameznika, ki se veča z vsako interakcijo uporabnika z mobilno napravo. Gre pravzaprav za posameznikov digitalni arhiv. To je zelo pomemben podatek, saj so v modernem času digitalni dokazi prisotni v več kot 80% sodnih primerov [6][2].

Problem je v tem, da obstaja veliko orodij za digitalno analizo, ki pa si med seboj niso enakovredna. Primer: eno orodje bolje obnavlja SMS sporočila, medtem ko neko drugo orodje bolje obnavlja posamezne datoteke. Možna rešitev bi bila uporaba večjega števila orodij, pri čemer bi vsako orodje uporabili za določeno podnalogo z maksimalno učinkovitostjo. Večina izkušenih forenzikov se tega drži in rezultate, ki jih dobijo z uporabo nekega orodja preverijo z uporabo še nekega drugega orodja. Resnica je žal ta, da to običajno pripelje do veliko daljših preiskav, kot bi sicer. Skleniti moramo kompromis. Želimo skrajšati čas preiskave, obenem pa uporabljati orodje, ki dobro opravi vse naloge, ki se ob samem postopku preiskave pojavijo. Kot vidimo, je izbira orodja v fazi priprave zelo pomemben korak. Neuspeh pri izbiri lahko posledično vodi do nepravilne ekstrakcije podatkov, do neintegritete podatkov in na koncu do same nekredibilnosti podatkov pri preiskavi.

Na tem mestu bi se ozrli nazaj v zgodovino, kjer si bomo na kratko ogledali postopek razvoja vrednotenja forenzičnih orodij za digitalne preiskave. NIST (National Institue of Standards and Technology) je bila prva organizacija, ki je evaluirala forenzična orodja kot tretja oseba (*angl. third-party*) [5] [4]. Izdali so t.i. Smart Phone Specification Tool in Smart Phone Tool Assertions and Test Plan. Nato so te specifikacije in načrte za testiranje uporabili pri vrednotenju forenzičnih orodij. Ta vrednotenja so se skozi zgodovino nadgrajevala in izboljševala. V članku [7] so prišli do ugotovitve, da različni tipi digitalnih dokazov različno

vplivajo na reševanje posameznega primera, kar je vodilo do MCD pristopa (*angl. Multi Criteria Decision*), pri katerem se orodja ocenjujejo z upoštevanjem dveh kritičnih faktorjev (prej omenjena faktorja ustreznosti tipa dokaza in zmogljivosti orodja). Predstavili so 7 različnih tipov digitalne preiskave [1], pri čemer je ustreznost merjena linearno od 0 do 10 točk in nam pove pomembnost dokaza za posamezno preiskavo (npr. SMS so lahko v nekem primeru bolj pomembni kot pa zgodovina klicev).

2. TEORETIČNO OZADJE RAZISKAVE

V tem odseku je predstavljeno teoretično ozadje same raziskave. S pomočjo sledečih izračunov so v članku prišli do rezultatov, ki so na koncu predstavljeni kot pričakovani grafi uporabnosti.

2.1 Teorija verjetnosti

Na začetku so forenziki izbirali orodja s pomočjo hevristike (glede svoje izkušnje). V članku so sklepali, da izbira orodja včasih ni sledila formalnim metodam zmogljivosti orodja in ustreznosti tipa digitalnega dokaza. V članku so to formalizirali in opredelili, da je zmogljivost orodja merjena kot verjetnost ekstrakcije določenga tipa digitalnega dokaza s pomočjo forenzičnega orodja, določena z

$$P(s) = p_s, p_s \in [0,1], p_s = \frac{x}{n}, \tag{1}$$

pri čemer sta

- x = št. uspešno izluščenih objektov tega tipa,
- n = št. vseh objektov tega tipa.

2.2 Teorija uporabnosti

Na uporabnost lahko gledamo kot na mero stopnje zadovoljstva. V enem od prejšnji člankov [7], so avtorji zajeli stopnjo zadovoljstva za ustreznost vseh tipov digitalnih dokazov.

2.3 MCD analiza

MCD ali Multi Criteria Decision Analysis (analiza na podlagi več kriterijev) je odločitveni model, katerega naloga je ovrednotiti in uravnotežiti več med seboj konfliktnih kriterijev in maksimirati dobiček/donos (angl. gain/output) pri končni izbiri, se pravi čim več pridobiti z izbiro orodja in maksimirati njegove rezultate. Problem, s katerim se ukvarjajo v tem članku (izbira orodja za izvedbo digitalne preiskave mobilne naprave), je lahko modeliran po modelu MCD zato, ker je ustrezen v smislu zahtev, ki jih model MCD predvideva:

- Imamo različne kriterije (tipi digitalnega dokaza).
- Alternative (forenzična orodja), ki jih je potrebno ovrednotiti in se na koncu odločiti za najboljšega v specifičnem primeru.

Za vsak kriterij (tip digitalnega dokaza) se nato določi njegov nivo uporabnosti (*angl. utility*) po enačbi

$$u_i(x) = \frac{x - x_i^-}{x_i^+ - x_i^-}, \tag{2}$$

kjer $u_i(x)$ pomeni uporabnost x v kriteriju i. x je izmerjena vrednost in je rezultat testiranja hipoteze z njeno z-oceno (podrobnosti v članku [8]). Posamezna uporabnost se uporabi za relacijsko povezavo uspešnosti forenzičnih orodij z enačbo

$$Alt1 > Alt2 + z_i / \sum_{i=1}^{n} |z_i|, \tag{3}$$

kjer je z_i z-ocena izračunana za alternativo v času hipotetičnega testiranja za določen kriterij (podrobnosti v članku [8]).

Ko imamo izračunane vse $u_i(x)$, lahko izračunamo globalni $U(x)$ po enačbi

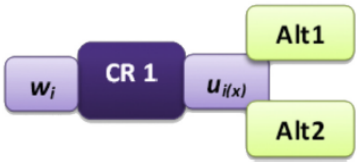
$$U(x) = \sum_{i=1}^{n} w_i u_i(x), \tag{4}$$

pri čemer w_i predstavlja utež, ki jo ima posamezen kriterij. Ko imamo za vse kriterije določeno uporabnost $u_i(x)$ in uteži w_i, lahko zmodeliramo MC model (*angl. Multi Criteria Model*).

2.4 MC model

MC model je predstavljen kot drevo, pri čemer so vsi kriteriji listi drevesa. Vsak list predstavlja en tip digitalnega dokaza (kriterij), ki mora biti ovrednoten glede na zmogljivost pri posameznem orodju in primernost.

Vsak list je sestavljen iz uporabnosti $U_i(x)$, ki meri zmogljivost posameznega forenzičnega orodja. Prav tako ima tudi utež W_i, ki pove, koliko posamezen kriterij doprinese h končni oceni, oziroma kako ustrezen je.



Slika 1: Primer lista v drevesu za prvi kriterij in obe alternativi [9].

Primer lista je viden na sliki 1, pri čemer je

- CR1 - kriterij,
- w_i - teža, določena kriteriju,
- Alt1, Alt2 - forenzični orodji (alternativi)
- $u_i(x)$ - uporabnost, ki veže orodja na ta kriterij.

3. ZMOGLJIVOST IN USTREZNOST

Po [3] je identificiranih 19 tipov digitalnih dokazov, zato je v članku uporabljenih 19 kriterijev.

3.1 Zmogljivost

Zmogljivost lahko merimo iz historičnih podatkov (kako se je neko orodje obneslo v preteklih preiskavah) ali pa iz točno določeno izvedenih eksperimentov (npr. iz strani tretje osebe). V članku gredo še korak dlje. Rezultati iz predhodnih analiz so normalizirani, nato pa za vsak kriterij izračunajo še njegovo uporabnost $u_i(x)$ glede na vsako alternativo (forenzično orodje). Rezultati so vidni v tabelah 1 in 2.

ID	Kriterij	Povezava (enačba 3)
1	Imenik/Kontakti	Alt1 = Alt2
2	Vnosi v koledar	Alt1 > Alt2 + 0.12472
3	Beležke/Zapiski	Alt2 > Alt1 + 0.01156
4	Naloge/Seznami opravil	Alt1 > Alt2 + 0.11483
5	SMS	Alt1 > Alt2 + 0.02105
6	EMS	Alt1 > Alt2 + 0.03867
7	MMS	Alt1 > Alt2 + 0.04998
8	Zvočni klici	Alt1 > Alt2 + 0.09732
9	Video klici	Alt1 = Alt2
10	Email	Alt1 > Alt2 + 0.23780
11	Obiskani URL-ji	Alt1 = Alt2
12	Zaznamki/Priljubljene vsebine	Alt1 > Alt2 + 0.12480
13	Zvok	Alt1 = Alt2
14	Video	Alt2 > Alt1 + 0.09048
15	Slike	Alt2 > Alt1 + 0.08895
16	Word	Alt1 = Alt2
17	Excel	Alt1 = Alt2
18	PowerPoint	Alt1 = Alt2
19	PDF	Alt1 = Alt2

Tabela 1: Zmogljivost mobilne naprave Xperia X1 glede na obe alternativi (forenzični orodji) [9].

3.2 Ustreznost

Vsak kriterij ima določen nivo pomembnosti pri reševanju digitalnega primera. Obravnavani tipi digitalnih primerov, ki imajo povezavo z mobilnimi napravami so:

- preprodaja drog (DT),
- umor (MD),
- zloraba kreditnih kartic (CC),
- prisluškovanje (EE),
- posilstvo (RP),
- nadlegovanje (HMT),
- otroška pornografija (CP).

Ustreznost je merjena na linearni lestvici od 0 do 10. Vrednost 0 pomeni, da kriterij nima nobene vrednosti pri digitalni preiskavi, vrednost 10 pa, da je to najbolj pomemben kriterij pri digitalni preiskavi. Podatki o ustreznosti so bili pridobljeni s pomočjo anketiranja strokonjakov digitalne forenzike. Tabela 3 predstavlja uteži za obravnavane primere pridobljene na podlagi ankete.

ID	Kriterij	Povezava (enačba 3)
1	Imenik/Kontakti	Alt1 > Alt2 + 0.00646
2	Vnosi v koledar	Alt1 > Alt2 + 0.07240
3	Beležke/Zapiski	Alt2 > Alt1 + 0.05686
4	Naloge/Seznami opravil	Alt1 > Alt2 + 0.06519
5	SMS	Alt1 = Alt2
6	EMS	Alt1 = Alt2
7	MMS	Alt1 > Alt2 + 0.08072
8	Zvočni klici	Alt1 > Alt2 + 0.10663
9	Video klici	Alt1 > Alt2 + 0.04106
10	Email	Alt1 > Alt2 + 0.13805
11	Obiskani URL-ji	Alt1 = Alt2
12	Zaznamki/Priljubljene vsebine	Alt1 = Alt2
13	Zvok	Alt1 > Alt2 + 0.14452
14	Video	Alt1 = Alt2
15	Slike	Alt1 = Alt2
16	Word	Alt1 > Alt2 + 0.08313
17	Excel	Alt1 > Alt2 + 0.04004
18	PowerPoint	Alt1 > Alt2 + 0.07919
19	PDF	Alt1 > Alt2 + 0.08573

Tabela 2: Zmogljivost mobilne naprave Nokia 5800 glede na obe alternativi (forenzični orodji) [9].

Criteria (ID)	DT	RP	MD	CC	HMT	EE	CP
1	9.56	9.08	9.64	8.55	9.51	9.32	8.82
2	6.30	6.13	8.48	6.88	7.11	7.51	6.08
3	6.31	4.93	7.92	7.23	6.85	7.79	5.98
4	5.83	4.44	7.03	6.85	6.41	7.49	5.31
5	9.68	9.33	9.68	8.84	9.84	9.16	9.05
6	8.83	9.03	9.17	8.21	9.59	8.58	9.03
7	7.62	7.51	8.20	7.26	8.59	7.80	8.16
8	9.09	8.77	9.23	8.03	9.50	9.37	7.95
9	6.36	6.84	6.82	5.92	7.97	7.47	7.38
10	8.65	7.46	8.87	8.82	9.38	9.13	9.08
11	6.20	5.47	7.36	8.44	6.84	8.39	9.28
12	5.30	4.38	6.18	8.03	6.11	7.55	9.18
13	5.42	5.87	6.00	5.69	7.41	8.67	6.08
14	7.04	7.65	7.13	5.92	8.00	8.50	9.61
15	8.77	9.11	8.56	7.36	9.00	8.79	9.83
16	4.35	3.58	5.38	7.29	5.11	7.92	5.95
17	4.98	2.90	4.93	7.64	3.00	7.63	5.03
18	3.11	2.27	4.35	5.05	3.45	7.11	5.64
19	3.57	2.66	5.00	6.00	3.21	7.58	4.97

Tabela 3: Ustreznost oz. teža posameznega digitalnega primera pridobljena z anketiranjem strokovnjakov s področja digitalne forenzike [9].

4. OVREDNOTENJE

Proces ovrednotenja obeh alternativ (forenzičnih orodij) je predstavljen s pomočjo grafa pričakovane uporabnosti (*angl. expected utility graph*). Preden lahko narišemo graf, potrebujejo še nekaj podatkov (Opomba: vse podatke so izračunali s pomočjo lastnega orodja DecideIT), ki so jih izračunali po enačbi

$$mid(\delta_{12}) = \frac{max(\delta_{12}) + min(\delta_{12})}{2} \quad (5)$$

Vrednost δ_{12} predstavlja razliko med pričakovanima uporabnostima obeh alternativ (*angl. EU - Expected Utility*).

$$\delta_{12} = EU(A1) - EU(A2) \quad (6)$$

$max(\delta_{12})$ predstavlja razliko med pričakovano uporabnostjo alternative 1 in alternative 2 (razlika med obema forenzičnima orodjema), pri čemer je alternativa 1 kar najboljša glede na alternativo 2. To pomeni, da so pri računanju maksimizirali pričakovane uporabnosti alternative 1. $min(\delta_{12})$ podobno, le da je alternativa 1 najslabša glede na alternativo 2 (maksimizirajo pričakovane uporabnosti alternative 2). $mid(\delta_{12})$ predstavlja srednjo vrednost $max(\delta_{12})$ in $min(\delta_{12})$.

$EU(A_i)$ je nadaljnje definiran kot:

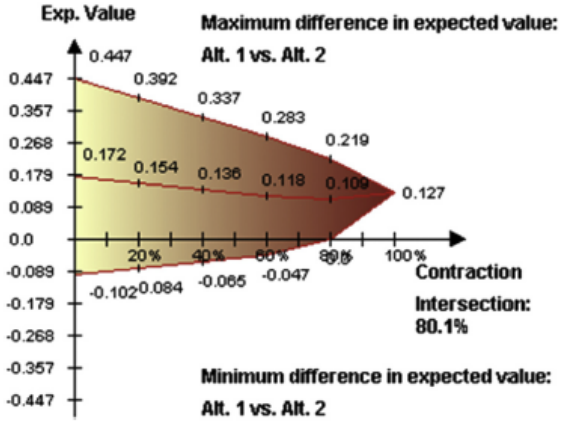
$$EU(A_i) = p_{i1}v_{i1} + p_{i2}v_{i2} + ... + p_{in}v_{in}. \quad (7)$$

p_{in} v tem primeru predstavljajo pričakovane uporabnosti alternative glede na kriterij, v_{in} pa ustreznost posameznega kriterija.

Moč posamezne alternative interpretiramo s t.i. *dominanco*. To si lahko najbolj preprosto razložimo kar s pomočjo grafa, ki je prikazan na sliki 2.

- Če je $min(\delta_{12}) > 0$, alternativa 1 močno dominira alternativo 2.
- Če je $mid(\delta_{12}) > 0$, alternativa 1 delno dominira alternativo 2.
- Če je $max(\delta_{12}) > 0$, alternativa 1 šibko dominira alternativo 2.

Pri vrednotenju so uporabili 20% korake pri računanju vrednosti δ. S tem, ko so povečevali krčenje (krčili interval, ki se uporablja pri računanju δ), so dobivali različne rezultate pri vsakem koraku.



Slika 2: Pričakovana uporabnost za naparavo Xperia X1 za preiskovanje zlorabe kreditnih kartic [9].

5. REZULTATI

Vrednotenje je bilo izvedeno na vseh 14 modelih MCD (7 tipov digitalne preiskave in 2 različni forenzični orodji). Za mobilno napravo so izbrali telefona Nokia 5800 in Xperia X1. Vsi rezultati so predstavljeni na sledečih grafih uporabnosti. V smislu zmogljivosti je alternativa 1 boljša od alternative 2 v večini primerov (Alt1 > Alt2), kar se vidi iz tabel 1 in 2. Iz tega sledi, da bo alternativa 1 običajno prva izbira pri izvedbi digitalnih preiskav.

Iz grafov na slikah 2, 3 in 4 je za oba telefona razvidno sledeče:

- Nokia 5800: Alternativa 1 močno dominira alternativo 2, saj po celotnem grafu velja $min(\delta_{12}) > 0$.
- Xperia X1: Alternativa 1 delno dominira alternativo 2, saj na celotnem grafu velja $mid(\delta_{12}) > 0$. Pri Xperii X1 imamo tudi točko preseka (nivo krčenja pri 80%). Od tam naprej velja, da alternativa 1 močno dominira alternativo 2.

6. ZAKLJUČEK

Iz rezultatov raziskave je zelo dobro razvidno, da je alternativa 1 boljša v večini primerov od alternative 2. Za telefon Nokia 5800 velja, da alternativa 1 močno dominira alternativo 2, za telefon Xperia X1 pa, da alternativa 1 delno dominira alternativo 2, od točke preseka (nivo krčenja 80%) pa močno domirnira alternativo 2.

6.1 Možne razširitve

Cilj raziskave je bil razviti tehniko vrednotenja forenzičnih orodij, ki bo temeljila na odločitvenih teorijah, verjetnosti in uporabnosti. Ena izmed možnih razširitev, ki je predlagana, je ta, da se testiranje razširi na širši spekter naprav, tipov dokazov ter forenzičnih orodij in iz tega izdelali referenčni zbornik, s pomočjo katerega bi bilo možno izbrati najboljše orodje za forenzično preiskavo glede na tip naprave in tip

digitalne preiskave. Ker bi bil zbornik izdelan s strani neodvisne skupine (tretje osebe), bi forenziki v to najbrž vlili več zaupanja kot sicer.

7. REFERENCES

[1] M. Anobah. Testing framework for mobile forensic investigation tools. *Stockholm University*, 2013.

[2] I. M. Baggili, R. Mislan, and M. Rogers. Mobile phone forensics tool testing: A database driven approach. *International Journal of Digital Evidence*, 6(2):168–178, 2007.

[3] A. K. Kubi, S. Saleem, and O. Popov. Evaluation of some tools for extracting e-evidence from mobile devices. In *Application of Information and Communication Technologies (AICT), 2011 5th International Conference on*, pages 1–6. IEEE, 2011.

[4] R. Kuhn. Smart phone tool test assertions and test plan. *National Institute for Science and Technology*, 2010.

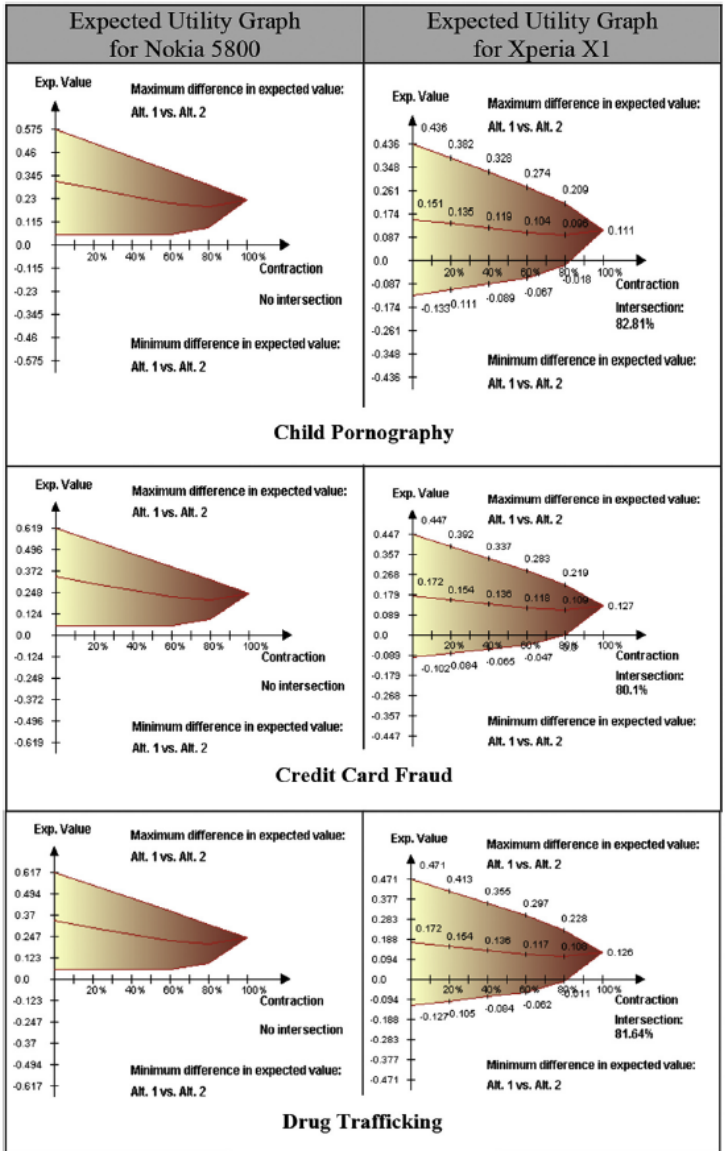
[5] S. P. T. S. NIST. Version 1.1, 2010.

[6] M. Rogers. Dcsa: A practical approach to digital crime scene analysis, vol. 3, 2004.

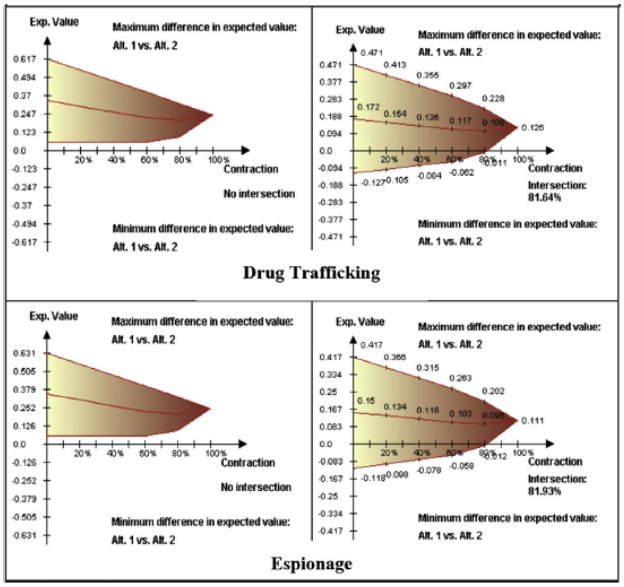
[7] S. Saleem, I. Baggili, and O. Popov. Quantifying relevance of mobile digital evidence as they relate to case types: A survey and a guide for best practice. *Journal of Digital Forensics, Security and Law*, 9(3):3, 2014.

[8] S. Saleem, O. Popov, and O. K. Appiah-Kubi. Evaluating and comparing tools for mobile device forensics using quantitative analysis. In *International Conference on Digital Forensics and Cyber Crime*, pages 264–282. Springer, 2012.

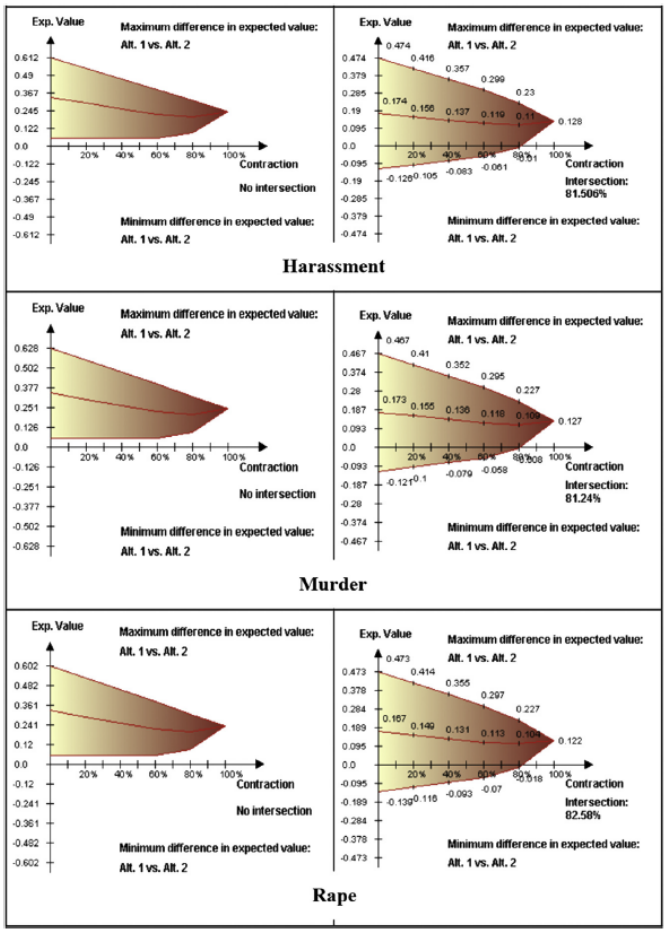
[9] S. Saleem, O. Popov, and I. Baggili. A method and a case study for the selection of the best available tool for mobile device forensics using decision analysis. *Digital Investigation*, 16:S55–S64, 2016.



Slika 3: Pričakovana uporabnost v vseh 14 primerih [9].



Slika 4: Nadaljevanje [9].



Slika 5: Nadaljevanje [9].

Forenzično preiskovanje v programsko definiranih omrežjih

Mitja Rizvič
Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
mr8961@student.uni-lj.si

***Abstract*—Računalniška omrežja se iz leta v leto povečujejo. Ocenjujejo, da se vsako sekundo v internet poveže približno 80 novih naprav, številka pa se z leti še povečuje. Ogromno število novih naprav tako za načrtovalce omrežij predstavlja izziv. Internetno omrežje kot ga poznamo danes temelji na protokolu IP. Glavni sestavni del takšnega omrežja so usmerjevalniki, ki glede na naslov kamor je paket namenjen, le tega usmerjajo po omrežju. Za ta namen uporabljajo različne usmerjevalne protokole. S povečevanjem omrežja se povečuje tudi kompleksnost usmerjanja. Posledično postavitev takšnega omrežja zahteva veliko dela s konfiguracijo omrežnih naprav. Takšno omrežje je tudi zelo težko spreminjati oziroma posodabljati. Kot odgovor na zgornjo problematiko so se pojavila programsko definirana omrežja. V nadaljevanju je najprej predstavljeno nekaj osnov programsko definiranih omrežij. Nato je predstavljeno, kako programsko definirana omrežja vplivajo na omrežne forenzične preiskave ter kako so nam lahko pri tem v pomoč.**

I. Uvod

Tradicionalna IP omrežja temeljijo na usmerjevalnikih, ki IP pakete usmerjajo po omrežju tako, da prispejo do ciljnega naslova. Kljub vsesplošni uporabi so taka omrežja zelo kompleksna in zahtevna za konfiguracijo. Vsako napravo v omrežju je na začetku potrebno pravilno konfigurirati. Da postane stvar še bolj zakomplicirana, je protokol oziroma vmesnik preko katerega lahko spreminjamo nastavitve naprave od naprave do naprave različen. Delovanje usmerjevalnikov lahko logično razdelimo na dva dela: kontrolno plast (angl. control plane) in podatkovno plast (angl. data plane). Kontrolna plast določa, kako ravnati s podatkovnim prometom, podatkovna plast pa obdeluje podatke glede na navodila kontrolne plasti. Obe plasti sta v tradicionalnih omrežjih združeni v eno napravo, kar posledično zmanjša fleksibilnost. Programsko definirana omrežja predstavljajo eno od možnih rešitev zgoraj omenjenih problemov. Osnovna ideja je ločiti kontrolno plast od podatkovne plasti. Omrežni usmerjevalniki tako postanejo preproste naprave, ki delujejo le še na podatkovni plasti. Kontrolna plast je tako implementirana v omrežnem krmilniku, ki daje navodila podatkovni plasti - usmerjevalnikom. Le-ta je lahko implementiran na ločenem sistemu in je lahko centralen, kar pomeni, da lahko nadzoruje več usmerjevalnikov hkrati. Ločitev kontrolne plasti od podatkovne lahko implementiramo tako, da natančno definiramo programski vmesnik preko katerega komunicirata omrežni krmilnik in stikalo. Primer takega protokola je OpenFlow [1]. OpenFlow stikalo vsebuje eno ali več FlowTable tabel. Le-te vsebujejo pravila, katera stikalo povejo kaj naj stori s prejetim paketom. Glavna razlika, ki OpenFlow stikala loči od klasičnih usmerjevalnikov je, da vsebino tabele nastavi OpenFlow krmilnik in ne stikalo samo. Tako stikalo je za razliko od usmerjevalnikov veliko bolj preprosto in ne uporablja kompleksnih usmerjevalnih protokolov. Glede na navodila omrežnega krmilnika lahko deluje kot stikalo, usmerjevalnik, požarni zid ali pa opravlja katero drugo nalogo. Vsi protokoli, ki skrbijo za pravilno usmerjanje paketa, pa se izvajajo centralno na omrežnem krmilniku.

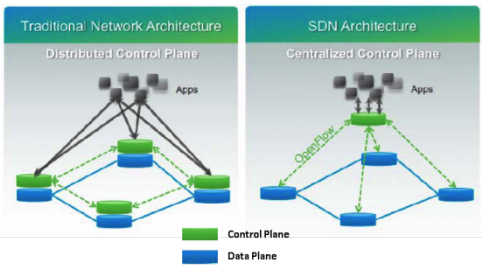


Figure 1. Primerjava tradicionalnega in programsko definiranega omrežja [2]

II. Forenzično preiskovanje omrežij

Kot je bilo omenjeno v uvodu je omrežje, ki tvori internet zelo kompleksno. Posledično je lahko še tako trivialna naloga zelo zahtevna. Za primer vzemimo naslednji scenarij. V podatkovnem centru opazimo sumljiv dogodek - na primer več nepričakovanih poskusov povezave na neznan IP naslov. Naša naloga je preveriti ali dogodek predstavlja varnostno grožnjo – na primer poskus kraje podatkov ali je le posledica nepravilne konfiguracije omrežja. Prva stvar, ki bi jo lahko storili je, da bi se povezali na napravo, ki je poskusila vzpostaviti povezavo in poskusili najti vzrok. Vendar pa lahko hitro naletimo na problem, saj je lahko v primeru vdora napadalec zakril svoje sledi. Tudi če smo našli sledi vdora, ki bi nas potencialno lahko pripeljale do storilca, jim ne moramo zaupati, saj jih je napadalec lahko nastavil namenoma. Podobno je lahko lažne sledi nastavil tudi na drugih napravah v omrežju - stikalih, usmerjevalnikih, požarnem zidu itd. Za omrežno forenziko torej ni dovolj, da znamo podatke najti, ampak je potrebno določiti njihov izvor. Če na primer v konfiguraciji usmerjevalnika najdemo vnos v usmerjevalni tabeli, ki bi nam lahko pomagal pri iskanju storilca, potem moramo določiti, kako se je le ta znašel v tabeli. Le tako lahko z gotovostjo vemo ali je vnos legitimen ali pa je vnos posledica napada oziroma vdora v sistem.

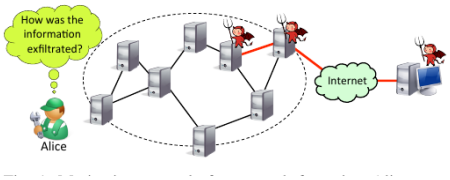


Figure 2. Primer omrežja, kjer sta dve vozlišči ”okuženi”, oziroma dostopna napadalcu. [4]

III. Secure Network Provenance

Če želimo od omrežne naprave pridobiti podatke (na primer zabeležke), potem moramo napravi zaupati, saj v nasprotnem primeru pridobljeni podatki morda niso verodostojni. Tehnika, ki nam zagotavlja verodostojnost podatkov se imenuje Secure Network Provenance (v nadaljevanju SNP). Sistem si lahko predstavljamo kot graf, kjer vsako vozlišče predstavlja stanje omrežja oziroma dogodek, povezave pa odvisnost med njimi. Vsaka naprava v omrežju tako hrani posebno zabeležko, kjer se beležijo vsa sporočila o aktivnostih vozlišča. Poleg tega naprava svoje aktivnosti sporoča drugim napravam v omrežju. Posledično zabeležke v vsaki napravi vsebujejo tako lastna sporočila kot sporočila drugih naprav. Vsa sporočila so digitalno podpisana s privatnim ključem naprave, s čimer preprečimo poneverjanje. Kot omenjeno, nas pri preiskovanju omrežja ne zanimajo le podatki vendar tudi zgodovina, torej od kje je določen podatek prišel. Z uporabo tehnike SNP lahko s pomočjo zabeležk, ki jih vozlišča hranijo za vsak podatek, izsledimo njegov izvor. Poleg tega lahko zabeležke primerjamo z zabeležkami shranjenimi na drugih vozliščih in tako zagotovimo konsistentnost oziroma verodostojnost, tudi v omrežju, ki je napadeno. Ker NSP zagotavlja varnost oziroma konsistentnost podatkov s primerjanjem zabeležk različnih vozlišč, lahko v primeru, da je okuženih več vozlišč, le-ta med seboj koordinirajo laži in se s tem izognejo detekciji. Kot posledica SNP sistem deluje le v omrežjih, kjer je vsaj eno vozlišče neokuženo in popolnoma delujoče. Poleg tega lahko opazujemo podatke le v vozliščih, ki niso okužena. Posledično se z manjšanjem deleža neokuženih vozlišč, manjša tudi del omrežja, ki ga lahko zanesljivo preiskujemo.

IV. Programsko definirana omrežja kot rešitev

V primerjavi s tradicionalnim omrežjem nam programsko definirano omrežje (v nadaljevanju SDN), ponuja dodatne možnosti za forenzično preiskovanje. Namesto, da se zanašamo na sporočila, ki jih beležijo posamezna vozlišča, lahko vsako omrežno povezavo oziroma OpenFlow stikalo spremenimo v nadzorno enoto. Z uporabo OpenFlow tabel lahko nad posameznimi paketi naredimo množico kompleksnih operacij kar v samem stikalu. Takšne operacije vključujejo na primer: filtriranje paketov, spreminjanje glave paketa, kopiranje paketov, posredovanje več napravam itd. Kadar operacija ni mogoča v samem stikalu (na primer: Deep packet inspection) se lahko paket posreduje omrežnemu koordinatorju za nadaljnje procesiranje. Za podoben rezultat bi v tradicionalnih omrežjih morali na povezave med vozlišči, postaviti Proxy enote, kar pa bi bilo zelo zakomplicirano in drago.

A. Primer forenzičnega sistema z uporabo OpenFlow omrežij

Uporaba SDN omrežij nam ponuja veliko dodatnih možnosti, vendar pa so operacije, ki jih lahko naredimo z OpenFlow stikalom omejene. Posamezen paket lahko z uporabo pravil, ki jih zapišemo v Flow tabelo:

- zavržemo,
- posredujemo,
- posredujemo na vse izhode – flooding in
- spremenimo podatke v glavi paketa.

Naprednejše operacije prepustimo omrežnemu krmilniku ali pa namenskim napravam, katerim lahko posredujemo posamezni paket. Na tej ideji temelji tudi predlagan forenzični sistem. OpenFlow stikala ne ponujajo vseh funkcionalnosti, ki so potrebne za izgradnjo forenzičnega sistema. Zato vpeljemo vmesne enote, ki jih imenujemo Provenance Verification Points (v nadaljevanju PVP). Njihova naloga je, da nadzorujejo promet posameznih omrežnih stikal. OpenFlow stikala tako sprogramiramo, da ves promet ustrezno posredujejo PVP napravam, katere pa ga obdelajo glede na potrebe forenzičnega sistema. Glede na to kako promet posredujemo PVP napravam

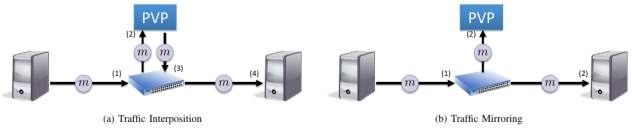


Figure 3. Slika prikazuje dva različna načina posredovanja paketov PVP napravam. [4]

ločimo dva primera. Prvi primer (slika 3 levo) imenujemo ”traffic interposition”. Vsak paket, ki ga stikalo prejme na vhod posreduje PVP napravi. Le-ta ga nato obdela v skladu z zahtevami forenzičnega sistema in ga nato vrne nazaj omrežnemu stikalu, ki ga posreduje dalje na ciljni naslov. Tak pristop omogoča, da PVP naprava poleg samega pregleda paketa, paket spremeni ali pa ga zavrže v primeru, da zazna škodljivo vsebino. Slabost takšnega pristopa je v tem, da se odzivnost omrežja poslabša, saj mora paket dodatno prepotovati do PVP naprave in nazaj do stikala preden se lahko ustrezno posreduje. Alternativni pristop, ki rešuje omenjeni problem je ”traffic mirroring” in deluje tako, da stikalo PVP napravi pošlje le kopijo paketa, originalni paket pa se takoj posreduje ciljni napravi. Tak način občutno zmanjša latenco omrežja vendar pa onemogoča, da PVP naprava v primeru, da odkrije škodljivo vsebino paketa ustrezno prepreči posledice. Kljub vsemu pa pasivno opazovanje omogoča verodostojnost podatkov, ki jih lahko nato uporabimo pri forenzični preiskavi. SDN omrežja omogočajo, da pakete enostavno kopiramo in posredujemo PVP napravam. Težja naloga je kako posredovane pakete v PVP napravi pregledati učinkovito in dovolj hitro, da lahko sledimo hitrosti prenosa podatkov po omrežju. Ker želimo imeti celoten pregled nad omrežjem, si ne moramo privoščiti, da pride do izgub paketov zaradi počasnega procesiranja. PVP naprave zato implementiramo tako, da opravljajo minimalno količino procesiranja, da omogočajo zanesljivo poizvedovanje po stanju omrežja. V prvem poglavju je bil predstavljen sistem

SNP, ki omogoča, da vozlišča v omrežju hranijo zabeležke o omrežnem prometu. Slabost takšnega pristopa je, da so okužena vozlišča lahko med seboj uskladila ”laži” in tako komunicirala med seboj nevidno za forenzični sistem. Omenjeno slabost lahko elegantno rešimo z uporabo SDN omrežij kot je bilo navedeno. Okužena vozlišča tako ne morajo več usklajevati laži, saj celotni promet nadzoruje PVP naprava. Celotna zanesljivost in verodostojnost takšnega forenzičnega sistema tako temelji na zanesljivosti PVP naprav. Le-te lahko naredimo relativno varne oziroma zanesljive tako, da omejimo funkcionalnosti, ki jih imajo. Poleg tega, jih lahko implementiramo kot namenske ASIC naprave, kar še poveča zanesljivost.

V. Zaključek

V primerjavi s klasičnimi omrežji nam SDN omrežja ponujajo veliko možnosti, ki jih lahko uporabimo pri forenzični preiskavi. Z uporabo PVP enot, lahko odkrijemo napade na omrežja, ki jih prej nismo mogli. Uporaba SDN omrežij nam omogoča, da takšne enote relativno enostavno implementiramo z uporabo OpenFlow stikal, s katerimi promet ustrezno usmerjamo po omrežju. Kljub vsemu pa je potrebno tudi pri uporabi SDN omrežij, forenzični sistem skrbno načrtovati in kritičnim točkam kot so PVP enote zagotoviti ustrezno varnost.

References

[1] http://archive.openflow.org/wp/learnmore/

[2] http://blogs.salleurl.edu/data-center-solutions, accesed: 14.5.2017

[3] Diego Kreutz, Fernendo M. V. Ramos, Paulo Verissimo, Christian Esteve Rothenberg, Siamak Azodolmolky, Steve Uhlig: Software-Defined Networking, A Comprehensive Survey

[4] Adam Bates, Kevin Butler, Andreas Haeberlen, Micah Sherr, Wenchao Zhou : Let SDN Be Your Eyes: Secure Forensics in Data Center Networks

Iskanje značilnih vzorcev v programsko definiranih omrežjih

Seminarska naloga pri predmetu Digitalna Forenzika *

Blaž Repas
Fakulteta za računalništvo in informatiko
Univerza v Ljubljani
br9404@student.uni-lj.si

POVZETEK

Programsko definirana omrežja (*SDN - Software Defined Networks*) so pomemben del internetne infrastrukture. Zaradi poenostavitve upravljanja z omrežjem so SDN zanimiva za velike podatkovne centre in ponudnike interneta ter internetnih storitev. Pri le-teh je zelo pomembno nemoteno in stabilno delovanje omrežja, ki ga lahko zmotijo napadi na omrežje. Pomembno je, da potencialni napadalci vedo čim manj o omrežju, saj je s tem oteženo njihovo delovanje.

V SDN se omrežne naprave upravlja centralizirano - ločeno od delovanja (izvajanja funkcije) naprave. Ločeno upravljanje in delovanje ima za posledico drugačno obravnavo paketov (na kontrolni in podatkovni ravni), ki se pretakajo skozi omrežje. V tem delu bom povzel in opisal članek *On the Fingerprinting of Software-Defined Networks* avtorjev *Heng Cui et. al* [7], ki obravnava iskanje značilnih vzorcev (fingerprinting) v delovanju SDN omrežja. Članek predstavi ranljivost SDN omrežja in sicer, da je mogoče razločiti, kdaj omrežje za določeni podatkovni tok uporablja že vzpostavljena pravila in kdaj je potrebna intervencija kontrolne ravni za določitev novih pravil. Pokazano je, kako lahko z veliko verjetnostjo razločimo med tema načinoma delovanja ter kaj lahko s to informacijo naredi napadalec (npr. nad omrežjem izvede napad DoS). Prav tako je predlagan način, kako se možnosti razločevanja zmanjša. V seminarski nalogi bom opisal SDN omrežja ter obravnaval delo in rezultate članka. Obravnaval bom implikacije zbiranja značilnih vzorcev obnašanja omrežja z vidika digitalne forenzike in varnosti.

Osnovni pojmi

SDN - Software Defined Network

Ključne besede

Software Defined Networks, Network Security, Network Fingerprinting

*Študijsko leto 2016/2017

1. UVOD

Množična uporaba interneta od ponudnikov interneta in strežniških storitev zahteva uporabo velikih omrežij, ki iz dneva v dan rastejo. Velika omrežja vsebujejo veliko omrežnih naprav - omrežnih usmerjevalnikov in stikal, ki pa za (pravilno) delovanje potrebujejo konfiguracijo in pravila, kako se obnašati. S premišljeno topologijo in iznajdljivimi pristopi je sicer mogoče omejiti raznovrstnost pravil in konfiguracij, vendar velikokrat omrežja prerastejo sposobnosti klasičnih upravljalskih metod. Zaradi tega se v velikih omrežjih vse pogosteje uporabljajo programsko definirana omrežja (SDN - Software Defined Networks). Glavna lastnost programsko definiranih omrežij je centralizirano upravljanje z omrežjem. Naprave, ki dejansko pretakajo pakete dobijo pravila za delovanje od centralnega strežnika.

V primeru, da omrežna naprava nima nobenega pravila, ki bi ustrezalo toku paketov, mora glede nadaljnjih akcij vprašati centralni strežnik. Tukaj lahko opazimo ločeni kontrolni in podatkovni plasti. Ta razdvojenost je poglavitni razlog, da je možno razločiti med značilnim različnim obnašanjem omrežja. Podatkovna plast je večinoma realizirana z dobro optimiziranimi strojnimi rešitvami, saj mora biti sposobna pretakati več gigabitov podatkov na sekundo. Kontrolna plast pa mora poskrbeti za ustvarjanje novih pravil in njihovo namestitev na naprave. Ker so pri tem potrebni izračunavanje in/ali vpogledi v podatkovne baze, je čas obravnave paketa bistveno daljši, ko se mora narediti novo pravilo.

Na podlagi tega je možno način razpoznavati z različnimi tehnikami. Članek predlaga in opisuje aktivni način, kjer napadalec dejansko ustvarja in pošilja svoj promet preko omrežja, ter pasivni način, kjer napadalec zgolj prisluškuje komunikaciji. Slednji je dosti bolj nevaren, saj ga je izredno težko ali celo nemogoče zaznati s sistemi za zaznavo vdorov (Intrusion Detection Systems), ki delujejo na razpoznavanju anomalij v prometu.

Napadalec lahko torej z uporabo lastno ustvarjenega prometa ali zgolj prisluškovanja ugotovi, kdaj je za paket (tok paketov) potrebna namestitev novega pravila v omrežje. S tem lahko zbere dovolj informacij, da ustvari promet, ki bo vedno zahteval nova pravila in s tem preobremeni kontrolni strežnik ter onemogoči komunikacijo. Lahko pa tudi sklepa, kakšna pravila se uporabljajo v omrežju, to pa lahko zlonamerno uporabi za nove napade na storitve in podobno.

2. PREGLED PODROČJA

Varnost računalniških omrežij je odprta raziskovalna in praktična tema že od samega začetka uporabe računalniških omrežij. Prav tako je varnost in stabilnost pomembna v programsko definiranih omrežjih, ki so vedno bolj popularna [10]. Avtor v članku [15] trdi, da varnostne funkcije SDN omrežij niso dovolj za zagotavljanje varnosti omrežja samega in da je varnost potrebno zasnovati že v sam SND sistem.

S stališča omrežne varnosti in varnosti omrežja je za napadalce pomembno, da lahko določijo značilnosti omrežja, zato je določanje značilnosti omrežja zanimivo tudi na raziskovalnem področju. Znano je [16], da so latence v jedrnih in hrbteničnih omrežjih relativno stabilne in da zasedenost pasovne širine le malo vpliva nanje. Iz tega sledi, da sprememba latence in obhodnega časa paketov ter disperzija para paketov (ki je tesno povezana z latenco) lahko služijo kot mera za določanje neke značilnosti omrežja [20]. Raziskovalci so disperzijo para paketov predlagali tudi kot indikator pasovne širine, ki je na voljo [11, 14, 22], ter za zaznavo ozkih grl v omrežju [6, 8, 13, 12, 19].

Poleg določanja značilnosti omrežij je raziskovalno zanimivo tudi področje zaznavanja in odzivanja na napade (D)DoS [18] s pomočjo programsko definiranih omrežij.

3. SDN - PROGRAMSKO DEFINIRANA OMREŽJA

V tem razdelku bom na kratko opisal programsko definirana omrežja ter napravil primerjavo s klasičnimi omrežji.

3.1 Klasična omrežja

V klasičnih računalniških omrežjih navadno srečamo naprave, kot sta omrežni usmerjevalnik (router) in stikalo (switch). Sestojijo iz logike za usmerjanje/posredovanje ter pravil, po katerih delujejo. Poglavitno je, da so ta pravila določena vnaprej in se načeloma med delovanjem ne spreminjajo. Naprave lahko delujejo na podlagi pravil, lahko pa delujejo tudi na podlagi algoritmov, na primer učenje parov MAC naslovov in IP naslovov ali na katerih vratih (port) se nahaja določen MAC naslov. Pomembno je, da se zavedamo, da se ti algoritmi izvajajo na omrežni napravi. Naprava torej sama sprejema nove odločitve (na podlagi vgrajenega algoritma) in jih izvaja. Ko imamo povezano več naprav, te delujejo bolj ali manj neodvisno druga od druge. Vsaka ima svoje lastno stanje in poganja algoritme nad svojim stanjem. Pravimo, da so te naprave decentralizirane. V primeru, da je potrebno spremeniti delovanje naprav, je konfiguracijo potrebno popraviti na vsaki napravi posebej.

3.2 Programsko definirana omrežja

V programsko definiranih omrežjih sta kontrolna in podatkovna plast ločeni. V takih omrežjih imamo kontrolni strežnik, ki sprejema odločitve in SDN stikala, preko katerih se dejansko pretakajo paketi (podatkovna raven).

Kontrolni del je centraliziran in je ponavadi realiziran s posebno programsko opremo, ki se izvaja na kontrolnem strežniku. Primeri programske rešitev, ki realizirajo SDN kontrolni strežnik so na primer:

- Cisco Open SDN Controller [5]
- Floodlight [1],
- Beacon [9],
- OpenDaylight [3],
- POX [4];

Skoraj vsi SDN upravljalniki podpirajo odprti standard OpenFlow [21], ki definira protokol in programski vmesnik (API) za namestitev pravil na SDN stikala.

Glavna razlika med klasičnimi omrežji in SDN je v nastavljanju pravil. V SDN omrežjih se pravila na stikala vstavljajo dinamično. Ko na stikalo pride paket, ki že ima pripadajoče pravilo, se paket glede na to pravilo posreduje naprej. Če pa pripadajočega pravila ni, potem stikalo pošlje zahtevek (ki vsebuje cel paket ali samo zaglavje) kontrolnemu strežniku. Kontrolni strežnik je zadolžen, da na ta zahtevek odgovori, bodisi z novim pravilom, kam naj se paket (tok paketov) posreduje, ali pa s pravilom, ki promet zavrne. V obeh primerih se zaradi interakcije med stikalom in centralnim strežnikom ustvari dodatna zakasnitev.

4. ISKANJE VZORCEV V SDN OMREŽJIH

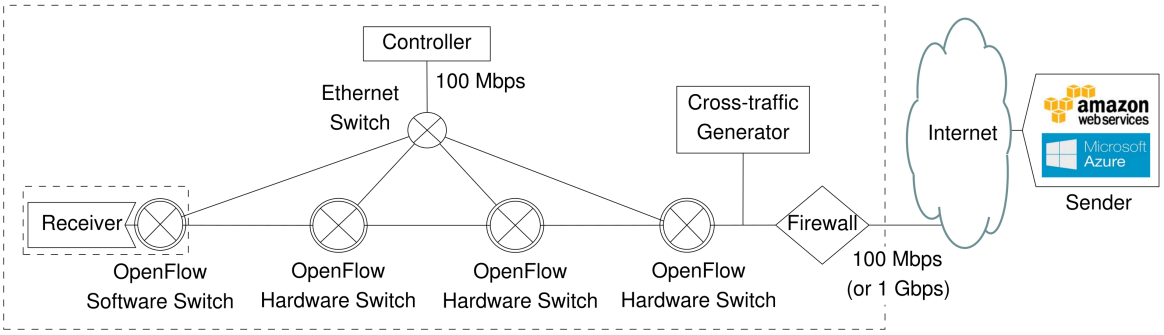
4.1 Problem

Glavni cilj obravnavanega članka je bil preučiti sposobnost napadalca, ki se nahaja zunaj omrežja, da zazna ali je bila za poslani paket (ali tok paketov) potrebna interakcija med stikalom in kontrolnim strežnikom in posledično tudi namestitev novih pravil v omrežje. Kadar omenjene interakcije ni zaznati, lahko napadalec sklepa, da v omrežju že obstaja neko pravilo, ki ustreza poslanemu paketu. V primeru, ko pa stikalo nima primernega pravila za poslani paket (v tabeli pravil ni nobenega ujemajočega zapisa), stikalo sproži zahtevek na centralni strežnik, ki pa nato paket obravnava, ter sprejme odločitev, ki ji v večini primerov sledi namestitev pravil v omrežje. Stikalo mora preden posreduje paket seveda počakati na odgovor centralnega strežnika, kar pa ima za rezultat dodatno latenco, ki je pri že nameščenih pravilih ni.

V raziskavi so avtorji obravnavali dva tipa napadalcev, ki zbirajo podatke o omrežju, aktivni in pasivni tip napadalca. Predpostavljeno je, da ima aktivni napadalec tehnične zmožnosti, da v omrežje pošilja skonstruirane pakete. To lahko doseže s tem, da je prevzel nadzor nad okuženim računalnikom, ki ga uporablja kot izvor paketov. Za razliko od aktivnega napadalca, lahko pasivni le prisluškuje obstoječemu prometu v omrežju. Oba pa potrebujeta sposobnost, da spremljata promet v omrežju, da prisluškujeta. Omeniti velja, da je pasivne napadalce zelo težko zaznati, saj ne tvorijo prometa in jih sistemi za zaznavo vdorov in anomalij (ang. *IDS - Intrusion Detection System* ne morejo zaznati.

4.2 Eksperiment

Avtorji članka so pripravili eksperiment, da bi empirično potrdili hipotezo, da je mogoče določiti, kdaj se v omrežje vstavijo nova pravila. Z potrebe eksperimenta so pripravili merilno okolje (slika 1), ki je podobno dejanskim omrežjem pri ponudnikih internetnih storitev.



Slika 1: Skica omrežja za izvedbo meritev pri eksperimentu določanja značilnosti; Izvor [7]

Okolje se sestoji iz treh OpenFlow strojih stikal (3 NEC PF5240 stikala [2]) in enega programskega OpenFlow stikala OpenVSwitch verzije 2.3.1 [17]. Stikala so na podatkovni plasti povezana preko 100 Mbps povezav (poskus so izvedli tudi z 1 Gbps povezavami). Zamišljeno je, da tri stikala dobro oponašajo dejanske razmere, kjer imamo ponavadi tri nivoje stikal: na vrhu omare (top-of-rack), agregacijske (aggregation) in jedrne (core). Podroben opis okolja (uporabljene računalniške komponente) lahko najdemo v članku [7].

Kontrolna plast se sestoji iz ločenega (out-of-band) ethernet omrežja, ki povezuje stikala in kontrolni strežnik. Omeniti velja, da do tega omrežja napadalec nima dostopa in ne more prisluškovati. Kontrolni strežnik je nastavljen tako, da ima čim manjši odzivni čas na zahtevke za pravila - naredi zgolj vpogled v tabelo in odgovori. S tem se doseže najslabše pogoje za razločevanje. V realnem omrežju bi kontrolni strežnik izvedel še več stvari (kar podaljša čas), kar pa je s stališča razpoznavanja lažji primer.

Da bi simulirali realna omrežja, je v okolje dodan tudi generator prometa (ang. *cross-tarfic generator*). Okolje je bilo povezano v Internet, od koder iz večih različnih lokacij pošiljali testne pakete, da bi zmanjšali vpliv latenc, ki so posledica različnih poti preko internetnega omrežja.

4.2.1 Mere

Za razpoznavanje med predhodno opisanima primeroma so v obravnavanem članku izbrali dve meri:

- disperzijo para paketov (Packet-Pair Dispersion)
- čas obhoda paketa (Round Trip Time).

Pri disperziji med dvema paketoma se meri kot časovni interval med celotno pretočenima paketoma čez omrežje. Ko omrežje ne potrebuje namestitve novih paketov, se oba paketa preneseta približno enako hitro in disperzija med njima ostaja približno enaka, kot je bila poslana. Kadar pa omrežje potrebuje nastavitev novih pravil, pa se se prenos prvega paketa zakasni in s tem se spremeni disperzija.

Pri merjenju časa odhoda se lahko sledi podobni logiki. Kadar omrežje potrebuje nova pravila, se čas obhoda podaljša zaradi dodatne zakasnitve.

4.2.2 Meritve

Za zbiranje časovnih podatkov so avtorji članka uporabili 20 oddaljenih klientov, ki so bili razpršeni po vsem svetu. Eksperiment je trajal od aprila do oktobra 2015. Skupno so poslali in zajeli 869.201 paketov z uporabo vseh oddaljenih klientov in različnih konfiguracij omrežja. Skupno so prenesli skoraj 0,66 GB podatkov merilnih paketov.

Meritve so bile izvedene omrežju, kjer se uporablja eno strojno stikalo, dve strojni stikali, tri strojna stikala, in kombinacija s programskim stikalom.

4.3 Rezultati

Na podlagi eksperimenta so ugotovili, da lahko s pomočjo disperzije dveh paketov zelo dobro napovedo, ali je prišlo do namestitve novih pravil. Kadar je v omrežju delovalo le eno strojno SDN stikalo, so pravilno napovedali z verjetnostjo približno 98.2%. Dodatno strojno stikalo ali dve dodatni strojni stikali sta verjetnost pravilne napovedi povišali na 99%, saj je potreben dodaten čas, da se posodobijo vsa stikala. Ko je bilo v omrežju prisotno programsko stikalo, ki je počasnejše od strojnih, se je napaka povečala na 4,49%. Programsko stikalo potrebuje več časa za posredovanje paketa in je posledično razlika med konfiguracijsko zakasnitvijo in posredovalnim časom manjša, kar je razlog za slabše razpoznavanje.

Pri uporabi obhodnega časa so rezultati podobni. V omrežju s tremi strojnimi stikali je napaka približno 0,43%, z dvema 0,12%, z enim 1,25% ter se poveča na 5,84% ko je v omrežju programsko stikalo.

Obe meri sta zanemarljivo malo odvisni od pasovne širine podatkovne ravni. Rezultati so pokazali tudi, da je mera z disperzijo bolj stabilna skozi čas, saj na obhodni čas bolj vplivajo spremembe v prenosni poti preko internetnega omrežja.

5. NAPADI NA SDN OMREŽJA

Kot je razvidno iz rezultatov članka, je možno s presenetljivo malo napako razpoznati, ali je omrežje potrebovalo namestitev novega pravila ali ne. To je lahko osnova za več možnih napadov na omrežje, ali pa samo za zbiranje informacij za kasnejši napad.

5.1 Zbiranje podatkov o storitvah v omrežju

Napadalec lahko ustvarja sintetične pakete, ki ustrezajo določenim omrežnim storitvam za katere napadalec želi ugotoviti, ali so prisotne v omrežju ali ne. Pare teh paketov pošlje preko omrežja in z prej opisano metodo razpozna, ali je omrežje potrebovalo nova pravila. Če so primerna pravila že bila nastavljena lahko napadalec sklepa, da je tak ali podoben promet že šel čez omrežje in lahko sklepa na obstoj storitve. Na ta način lahko napadalec naredi načrt, kaj je v omrežju in kaj lahko napade.

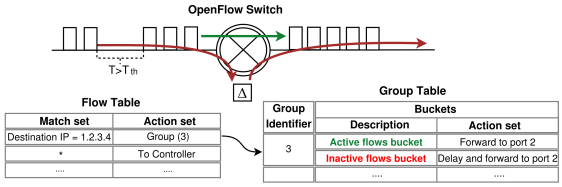
5.2 Napadi zavrnitve storitev - DoS

S podobnim vrednotenjem omrežja lahko napadalec ugotovi, za kakšne vzorce podatkovnih tokov pravila ne obstajajo in generira celotno družino podobnih paketov. Omrežje bo za vsak paket potrebovalo nova pravila, kar zahteva dodatno delo za centralni kontrolni strežnik. V primeru, da se strežnik preobremeni, je lahko rezultat nedelovanje celotnega omrežja.

Tudi v primeru, ko je strežnik dovolj zmogljiv lahko pride do delnega izpada storitve. SDN stikala imajo lahko le omejeno število aktivnih pravil. Če napadalec doseže dovolj veliko število nastavitev novih pravil, lahko pride do tega, da se katero od starih pravil zavrže in se s tem onemogoči pripadajoča komunikacija.

Omeniti velja, da ta dva napada ne izhajata iz specifične ranljivosti, ampak iz normalnega delovanja SDN omrežja, kar je zelo pomembno.

6. PROTIUKREPI



Slika 2: Skica protiukrepa z zakasnitvijo prometa; Izvor [7]

Da bi preprečili zbiranje podatkov o omrežju ali celo napade na SDN omrežje s pomočjo opisane metode, je na nek način potrebno zmanjšati razliko med paketi iz vzpostavljenega toka in novih paketov, ki potrebujejo pravila.

Očitna metoda je zakasnitev celotnega toka paketov. Pakete namesto, da jih posredujemo na izhodna vrata, se najprej posredujejo proti zakasnilnemu elementu in nato spet nazaj na stikalo in proti izhodnim vratom. Namesto da, bi se zakasnil le začetni paket, se zakasnijo tudi vsi ostali paketi v toku. Tako se izenači razlika disperzije in obhodnega časa in ni mogoče več razločevati med aktivnim tokom in začetkom. Seveda pa je to velik poseg v delovanje omrežja in zaradi dodatnega procesiranja paketov je potrebno veliko več računske moči. Posledično se učinkovitost omrežja precej zmanjša.

Avtorji članka v obravnavi predlagajo, da se namesto zakasnitve celotnega toka paketov uporabi drugačna metoda. Slika 2 prikazuje predlog protiukrepa, kjer se zakasni prvih nekaj paketov toka, ki je že ima nameščeno pravilo ampak je bil nekaj časa neaktiven. Na ta način se omrežje zelo malo dodatno obremeni, napadalec pa težko razloči med namestitvijo novega pravila in umetno zakasnitvijo.

7. FORENZIČNA VREDNOST

Na področju digitalne forenzike je izredno pomembno, da poznamo tehnične zmožnosti napadalca. Ker to seminarsko delo obravnava področje računalniških omrežij in njihove varnosti, se s forenzičnega vidika dostikrat obravnavajo zlorabe in napadi preko mreže. Da bi lahko uspešno odkrivali sledi, ki jih morebitni storilec pušča za seboj pri takih napadih, je pomembno, da poznamo, kako se take napade dejansko izvede. Med to spada tudi poznavanje preliminarnega raziskovanja omrežja. To je velikokrat začetni in tudi ključni del napadov, saj napadalec potrebuje čim več informacij o omrežju, da sploh lahko izbere primeren napad.

Določanje pravil v omrežjih lahko pripomore h globokemu razumevanju omrežja, ki ga želi napadalec napasti ali zlorabiti. Če ima napadalec izdelan dober načrt, lahko poskrbi, da ne pušča sledi, ali pa da sledi kažejo drugam.

S stališča odkrivanja forenzičnih sledi ima obravnavani članek bolj preliminarno vlogo, predvsem seznaniti forenzike in preiskovalce o možnih napadih in kaj je lahko storjeno, da se le-ti preprečijo ali omilijo.

8. ZAKLJUČEK

V tej seminarski nalogi sem predstavil programsko definirana omrežja ter njihovo prednost pred klasičnimi omrežji. Po članku [7] sem povzel in predstavil določanje značilnih vzorcev obnašanja SDN omrežja, oziroma, določanje kdaj omrežje deluje brez vstavljanja novih pravil in kdaj so potrebna nova pravila. Predstavil sem dve metodi za ugotavljanje obnašanja, aktivno in pasivno. Povzel sem eksperiment in rezultate, ki kažejo, da je možno z veliko verjetnostjo ugotoviti, kdaj omrežje deluje z obstoječimi pravili in kdaj so bila vstavljena nova, tudi kadar je v omrežju počasnejše programsko stikalo (software-based switch). Predstavil sem možne protiukrepe iz članka ter obravnaval njihov vpliv na učinkovitost omrežja.

Obravnaval sem možne napade in koristi za napadalca omrežja in kako je to relevantno s področja digitalne forenzike ter zaključil, da lahko obravnavane ranljivosti vodijo do zlorab in zlonamernih dejanj, ki jih je forenzično zelo težko obravnavati.

9. LITERATURA

[1] Floodlight sdn controller. [Online; Dostopano: 11-May-2017; http://www.projectfloodlight.org/floodlight/].

[2] Nec programmableflow pf5240 switch. [Online; Dostopano: 11-May-2017; http://www.necam.com/docs/?id=5ce9b8d9-e3f3-41de-a5c2-6bd7c9b37246].

[3] Opendaylight sdn controller. [Online; Dostopano: 11-May-2017; http://www.opendaylight.org/].

[4] Pox sdn controller. [Online; Dostopano: 11-May-2017; http://www.noxrepo.org/pox/about-pox/].

[5] I. Cisco Systems. Cisco open sdn controller, 2017. [Online; Dostopano: 11-May-2017; http://www.cisco.com/c/en/us/products/cloud-systems-management/open-sdn-controller/index.html].

[6] D. Croce, T. En-Najjary, G. Urvoy-Keller, and E. W. Biersack. Capacity estimation of adsl links. In *Proceedings of the 2008 ACM CoNEXT Conference*, page 13. ACM, 2008.

[7] H. Cui, G. O. Karame, F. Klaedtke, and R. Bifulco. On the fingerprinting of software-defined networks. *IEEE Transactions on Information Forensics and Security*, 11(10):2160–2173, 2016.

[8] C. Dovrolis, P. Ramanathan, and D. Moore. Packet-dispersion techniques and a capacity-estimation methodology. *IEEE/ACM Transactions On Networking*, 12(6):963–977, 2004.

[9] D. Erickson. The beacon openflow controller. In *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, pages 13–18. ACM, 2013.

[10] F. Hu, Q. Hao, and K. Bao. A survey on software-defined network and openflow: From concept to implementation. *IEEE Communications Surveys & Tutorials*, 16(4):2181–2206, 2014.

[11] N. Hu and P. Steenkiste. Evaluation and characterization of available bandwidth probing techniques. *IEEE journal on Selected Areas in Communications*, 21(6):879–894, 2003.

[12] G. Karame, D. Gubler, and S. Čapkun. On the security of bottleneck bandwidth estimation techniques. In *International Conference on Security and Privacy in Communication Systems*, pages 121–141. Springer, 2009.

[13] G. O. Karame, B. Danev, C. Bannwart, and S. Capkun. On the security of end-to-end measurements based on packet-pair dispersions. *IEEE Transactions on Information Forensics and Security*, 8(1):149–162, 2013.

[14] S. Keshav. *A control-theoretic approach to flow control*, volume 21. ACM, 1991.

[15] D. Kreutz, F. Ramos, and P. Verissimo. Towards secure and dependable software-defined networks. In *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, pages 55–60. ACM, 2013.

[16] A. Markopoulou, F. Tobagi, and M. Karam. Loss and delay measurements of internet backbones. *Computer Communications*, 29(10):1590–1604, 2006.

[17] B. Pfaff, J. Pettit, K. Amidon, M. Casado, T. Koponen, and S. Shenker. Extending networking into the virtualization layer. In *Hotnets*, 2009.

[18] R. Sahay, G. Blanc, Z. Zhang, and H. Debar. Towards autonomic ddos mitigation using software defined networking. In *SENT 2015: NDSS Workshop on Security of Emerging Networking Technologies*. Internet society, 2015.

[19] S. Saroiu, P. K. Gummadi, and S. D. Gribble. Sprobe: A fast technique for measuring bottleneck bandwidth in uncooperative environments. In *IEEE INFOCOM*, page 1, 2002.

[20] R. Sinha, C. Papadopoulos, and J. Heidemann. Fingerprinting internet paths using packet pair dispersion. *Probe*, 2:P1, 2006.

[21] O. S. Specification. Version 1.3. 2 (wire protocol 0x04), 2013.

[22] J. Strauss, D. Katabi, and F. Kaashoek. A measurement study of available bandwidth estimation tools. In *Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*, pages 39–44. ACM, 2003.

Opportunistic Piggyback Marking for IP Traceback

Analiza članka pri predmetu Računalniška forenzika*

Denis Grabljevec
Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
dg5121@student.uni-lj.si

Nejc Ambrožič
Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
na5478@student.uni-lj.si

Povzetek

Sledenje IP naslova je rešitev za problem iskanja izvora paketa, pri kibernetskih napadih ali spletnih prevarah in je uporabna pri zbiranju in analizi spletnega prometa. Ena izmed rešitev sledenja IP naslova je metoda, ki sloni na principu sledenja z označevanjem (Marking-based traceback - MBT). MBT metoda zelo obeta in je požela veliko pozornosti stroke. Čeprav metoda MBT veliko obeta pa ima tudi pomankljivosti in ena večjih je prenos sporočil namenjenih sledenju paketov. Prenos teh sporočil je ena glavnih funkcionalnosti sledenja paketov. V članku opisujeva rešitev pomankljivosti metode MBT imenovano oportunistično označevanje za sledenje IP paketov (OPM). OPM se od večine metod razlikuje, da ima ločeno vsebino sporočil za sledenje in funkcijo za dostavljanje sporočil. Poleg tega pa učinkovito doseže hitro in robustno dostavo sporočil z izkoriščanjem označevalnih možnosti. Na podlagi predlagane OPM sheme predstavimo prilagodljivo ogrodje na podlagi označevanja, ima nekaj prednosti pred ostalimi metodami za sledenje IP paketov. Z evalvacijo simulacij prikažemo, da predstavljen sistem učinkovito zmanjša število izgubljenih sporočil ter zmanjša obremenjenost usmerjevalnikov.

Kljucne besede

IP sledenje; sledenje na podlagi označevanja; omrežna forenzika; OPM; MBT

1. UVOD

Kibernetski napadi na internetu se vsako leto povečujejo. Kot primer leta 2014 se je vsako uro v poprečju zgodilo 28 DDoS napadov. Ened izmed izzivov, ki nam predstavlja branjenje pred DDoS napadi je skrivanje oziroma prevara izvornega IP naslova, s čimer si napadalci zagotovijo večjo varnost pred razkrikanjem. Sledenje IP naslovom je pogosto uporabljena metoda za odkrivanje izvornega IP naslova, s čimer lahko ugotovimo iz kje je bil napad storjen. Ugotovimo lahko izvorni IP naslov in pot, ki so jo napadalčevi paketi opravili v omrežju. Metoda lahko ublaži učinek in posledice napada in omogoča omrežno forenzično preiskavo napada.

Trenutno je ena najbolj znanih metod za sledenje IP naslovov metoda, ki temelji na sledenju z označevanjem (Marking-based trackback - MBT). Osnovna ideja MBT metode je, da usmerjevalniki prenašajo sporočila oziroma podatke, ki so namenjeni sledenju paketa skozi omrežje, do končnih končnih gostiteljev z označevanjem le teh. Potemtakem si lahko končni gostitelj na podlagi označenih paketov ustvari nek graf oz sliko poti, ki so jo paketi opravili v omrežjo preko najrazličnejših usmerjevalnikov, četudi se je napadalec hotel zaščititi s prevaro izvornega IP naslova. Mehanizmi za sledenje IP naslovov pa se vključijo šele, ko mehanizmi za zaznavanje napadov zaznajo abnormalno povečanje prometa. Namen poznega vklopa mehanizma pa je v zaščiti usmerjevalnikov, saj bi bili lahko te preobremenjeni, če bi bili mehanizmi več čas vključeni.

Raziskave MBT metode pa so pokazale, da se metoda spopada z dvema ključnima težavama. Prva težava je pri individualnih usmerjevalnikih, ki se morajo odločiti ali bodo poslali podatke za sledenje paketov ali ne. V nekaterih primerih pakete za sledenje označi samo usmerjevalnik, ki je najbližji izvoru. V naslednjem primeru pa vsi usmerjevalniki na poti napada. Druga težava pri raziskavi MBT metode pa je kodiranje vsebine sporočila, ki določa informacijo oziroma podatke, katere usmerjevalnik vstavi v glavo IP paketa. Kot primer, usmerjevalnik lahko v IP glavo paketa zapiše IP naslov, zgoščeno vrednost IP naslova ali Huffmanov kod.

Potrebno se je zavedati, da obstaja razlika med velikostjo sporočila za sledenje in prostora prostora v IP glavi, namenjenega sledenju IP paketa. Avtentikacija in ohranjanje zasebnosti sta dva zelo pomembna dejavnika, saj nam preprečujeta, da bi nam ogrožani usmerjevalniki ponaredili in spremenili označene pakete. Zagotavljanje avtentikacije in pa ohranjanje zasebnosti pa občutno povečata velikost podatkov, kar pa lahko predstavlja problem, glede na omejeno velikost, ki jo v ta namen sprejme IP glava. Zaradi česar mora biti sporočilo razdeljeno na več fragmentov, kar pa privede do daljšega časa izvajanja MBT metode, oziroma se lahko v nekaterih primer zgodi, da metoda odpove.

2. PREGLED PODROčJA

Čeprav je bilo na področju IP sledenja predstavljenih že veliko metod, pa se večino metod uvrsti v tri kategorije: metode, ki temeljijo na označevanju, metode beleženja in hibridnem pristopu.

2.1 Označevalne metode

Označevalna metoda deluje po principu, da usmerjevalniki vstavijo svojo identiteto oziroma podatke, ki predstavljajo, da je paket šel mimo usmerjevalnika. Končni gostitelj ima potem vse podatke o poti tega paketa. MBT metode so razdeljene na dva dela. Na Deterministično označevanje paketov (DPM) in Verjetnostno označevanje paketov (PPM). Tipično DPM deluje tako, da se v paket vstavijo identifikacijski podatki prvega usmerjevalnika na poti od izvora, medtem ko PPM deluje na verjetnostnem povečevanju paketov, ki vsebujejo delne informacije o identiteti usmerjevalnikov. Naloga DPM je lociranje izvor napadalca, glavni namen PPM pa je identifikacija poti paketov napadalca.

Čeprav deterministično označevanje povzroči manj napora na strani končnega gostitelja, pa lahko preobremeni mejni usmerjevalnik na strani izvora, saj označuje vsak poslan paket. Pri PPM pa obstaja neločljiv kompromis med število bitov v glavi IP paketa in število paketov, ki nosijo informacije za konstruiranje poti paketov napadalca v omrežju. PPM je zmožen zgraditi pot po kateri je šel paket od napadalca do žrtve le ko končni gostitelj sprejme dovolj paketov, kateri so označeni in nosijo pomembne informacije. Pomembna predpostavka, ki mora biti postavljena v PPM metodi je, da mora skozi napadalčevo pot teči veliko označenih paketov, drugače lahko metoda dolgo sestavlja pot napadalčevih paketov oziroma lahko metoda tudi odpove.

2.2 Metode beleženja

Metoda beleženja temelji na shranjevanju podatkov paketa na strani usmerjevalnika. Usmerjevalnik bi držal vse podatke o paketu zaradi česar nam podatke ne bi bilo potrebno nositi po več paketih. Obstaja pa problem velikosti prostora, saj bi beleženje paketov lahko porabilo veliko prostora.

2.3 Hibridni pristop

Hibridni pristop vsebuje prednosti obeh prej naštetih metod. S tem se zmanjša število označenih paketov, ki so potrebni za konstrukcijo poti paketa in velike porabe prostora na usmerjevalnikih. Ena od možnosti pri hibridnem pristopu je, da ustvarimo dve hibridni shemi (DLLT in PPPM). Shema DDLT deluje tako, da če se usmerjevalnik odloči označiti paket, bo najprej shranil informacijo, ki jo je paketu dodal prejšnji označevalni usmerjevalnik in potem to informacijo prepisal s svojim IP naslovom. Shema ustvari nekakšen povezan seznam usmerjevalnikov.

Metoda za pošiljanje sporočil za sledenje PPPM, je zelo podobna OPM metodi. Obstaja pa nekaj ključnih razlik, in sicer, metoda PPPM predvideva, da ima IP glava zadostno prostora za shranjevanje podatkov usmerjevalnika, medtem ko metoda OPM izrecno podpira fragmentacijo sporočila.

3. UVOD V OPORTUNISTIčNO OZNAčEVANJE

OPM temelji na sledenju na zahtevo. Metoda OPM sproži sledenje na 2 načina. Sproži se, ko se pojavi eden ali več sumljivih tokov, oz ko je potrebno analizirati omrežje. Sproži pa se tudi, ko usmerjevalniki zaznajo pakete oziroma tokove paketov, ki so sumljivi in se jim sledi. Pakete oziroma tok paketov, katerim sledimo rečemo notranji-tok, ostali paketi v omrežju pa so zunanji tokovi. V članku je glavni namen sledenje paketom oziroma ugotoviti celotno pot, ki so jo paketi naredili po omrežju. Rešitev pa nam mora biti sposobna ugotoviti izvor teh paketov in sestaviti delno pot, ki so jo opravili paketi. Zgornji del usmerjevalnika R, nam pove, da govorimo o usmerjevalnikih med izvorom in usmerjevalnikom R. Spodnji del pa nam nakazuje o usmerjevalnikih med destinacijo in usmerjevalnikom R.

3.1 Podatki za sledenje

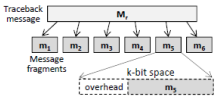


Figure 1: Fragmenti sporočila.

V članku se ne specificira velikosti prostora namenjenega za podatke sledenja v IP glavi. Lahko rečemo, da je v IP glavi K bitov, ki jih lahko uporabimo za prenašanje podatkov, ki jih naloži usmerjevalnik.

Naj M_r označuje podatke, ki jih usmerjevalnik pošlje končnemu gostitelju D. Ker pa je dolžina M_r logično večja od K, pa potrebujejo biti podatki fragmentirani v N IP paketov, ki se ne smejo prekrivati: $m_1, m_2, \ldots, m_n$. Da pa bomo vse fragmente pravilo sestavili nazaj v originalno sporočilo, pa vsak fragment potrebuje tudi nekatere metapodatke, in sicer, podatek za katero sporočilo gre (identifikator sporočila) in indeks fragmenta. Identifikator sporočila, pa nam lahko tudi zmanjša sposobnost napadalcev, ki bi lahko spremenili oziroma podtaknili lažne fragmente. Če pa je sporočilo dovolj majhno za prostor, ki ga ima IP glava, pa nam sporočila ni potrebno fragmentirati.

3.2 Pregled označevalne metode sledenja

Recimo, da pride v usmerjevalnik R paket P. Usmerjevalnik R se odloči, da bo paket P označil s svojimi podatki za sledenje. Sprva usmerjevalnik R generira naključno številko, ki bo služila kot identifikator sporočila Mr. Potem bo sporočilo Mr razdelil v N fragmentov brez prekrivanja. Fragmente bo shranil v svoj lokalni medpomnilnik. Fragmenti bodo nato do destinacije prišli preko označevanja paketov. Končni gostitelj nato čaka, da prejme vse fragmente sporočila. Ko se to zgodi je gostitelj sposoben sporočilo sestaviti in s tem ustvariti celotno pot, po kateri so se prenašali paketi.

3.3 Arhitektura označevalne metode

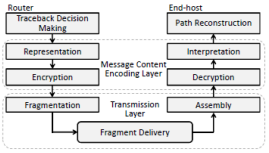


Figure 2: Arhitektura označevalne metode.

V metodi MBT ločimo shranjevanje vsebine sporočila in pa funkcijo za dostavo tega sporočila. V arhitekturi je tudi modul za odločanje, ali se bo paket označil ali ne, potem

modul, ki določa vsebino sporočila. Poleg vsebine pa spada zraven tudi oblikovanje sporočila in kodiranje sporočila. Ena glavnih nalog zgornje plasti je, da čim bolj zmanjša dolžino sporočila, ob tem pa proba zagotoviti avtentikacijo ali ohranjanje zasebnosti. Prenosna plast pa prevzame nalogo prenosa podatkov za sledenje do končnega gostitelja. Glavni problem te plasti je, kako čim bolj zmanjšati število paketov, ki so potrebni za sestavitev sporočila oziroma podatkov. Avtorji so se v članku osredotočili tudi na prenosno plast pri kateri je cilj zagotoviti hiter in robusten prenos fragmentov.

4. OPORTUNISTIčNO OZNAčEVANJE : OSNOVNA OBLIKA

4.1 Motivacija

Omejevanje metode MBT se začnejo že s predpostavko, da fragmente sporočila prenašajo samo paketi, ki se prenašajo po opazovanem toku. Kot že omenjeno je sporočilo razdeljeno na več fragmentov, zaradi česar lahko končni gostitelj čaka kar nekaj časa, predno bo lahko sestavil celotno sporočilo. V tem poglavju, so avtorji govorili, kako lahko fragmente, ki so shranjeni pri usmerjevalikih hitreje prenesemo do končnega cilja. Primer, tok označenih paketov se začne v izvoru S in konča v cilu D. Paketi se bodo do cilja D prenašali preko vmesnih usmerjevalnikov. Poleg prometa med S in D pa bosta usmerjevalnika R_6 in R_5 pravtako svoje pakete pošiljala k destinaciji D. Na poti so prav vsi usmerjevalniki sposobni označevaja paketov. Opazimo lahko, da so paketi v zunanjem toku posredovani do enakega cilja kot paketi v našem notranjem toku, zato lahko tudi paketi zunanjega toka pomagajo pri prenosu informacij o sledenju paketov, s čimer lahko zmanjšamo čas čakanja na sestavljanje poti notranjega toka.

4.2 Osnovni pregled oportunističnega označevanja

4.2.1 Polje za označevanje

Recimo, da nam IP glava ponuja K bitov za sporočila sledenja. Da se lahko zagotovi pravilno sestavo sporočila, so nujna kontrolna polja v vsakem sporočilu oz fragmentu, CL (polje namenjeno povezovanju različnih sporočil usmerjevalnikov), MI (polje namenjeno povezovanju fragmentov sporočila) in FO (odmik fragmentov). Z drugimi besedami OPM ne zagotavlja nobenih posebnih novih polj za shranjevanje podatkov v IP glavo.

4.2.2 Proženje sporočil za sledenje

Glavna naloga metode avtorjev članka je, da je končni gostitelj sposoben skonstruirati pot, po kateri so se prenašali označeni paketi. Vsi usmerjevalniki na poti morajp omogočati sledenje in označevanje paketov. Primer, prvi usmerjevalnik na poti med S in D je usmerjevalnik R_1, ki sprejme paket P. Usmerjevalnik se nato odloči, da bo opravil sledenje paketa v času t_1. R_1 ustvari sporočilo M_r, ki naj se prenese do destinacije paketa P. Sporočilo se razdeli na N fragmentov, kateri se ne prekrivajo. Usmerjevalnik R_1 ustvari naključno število za identifikacijo sporočila tega usmerjevalnika. Za identifikacijo posamezniv fragmentov, ki se bodo sestavili v sporočilo pa lahko uporabimo kar TTL vrednost paketa P. Nato R_1 zapiše prvi fragment v IP glavo paketa P. Paket, ki nosi prvi fragment sporočila prvega usmerjevalnika na poti, opredelimo kot signal sprožitve sledenja (TTS), s čimer pri kasnjejših usmerjevalnikih sprožimo generiranje ustreznih sporočil za sledenje glede na paket P, ter jih shranimo za naknadni prenos do končne destinacije D. Nato usmerjevalnik R_2 sprejme TTS v času t_2. Generira sporočilo za sledenje M_2, katero bo razdeljeno na N fragmentov. R_2 preko TTS paketa dobi podatke kot so CI in MI vrednosti za nastavljanje teh vrednostih pri svojih fragmentih. Fragmenti sporočil pa so shranjeni v lokalnem medpomnilniku. Proces pošiljaja TTS paketa se nadaljuje dokler ne pride do končnega gostitelja.

4.2.3 Dostava fragmentov

Sporočila namenjena sledenju se generirajo na usmerjevalnikih, katere prečka na poti do končnega gostitelja. Ko usmerjevalnik dobi neoznačen paket, preveri če ta paket lahko nosi katerega od fragmentov iz medpomnilnika do namenjene destinacije. Če lahko, potem usmerjevalnik označi paket P s tem, da mu v IP glavo zapiše fragment sporočila namenjenega sledenju.

4.2.4 Rekonstrukcija sporočila

Ko končni gostitelj prejme označeno sporočilo, bo najprej izvlekel fragment sporočila, predno bo paket poslal v višje plasti. Končni gostitelj najprej vse fragmente razdeli glede na njihovo CI vrednost. Potem fragmente iste CI vrednosti sortira po MI vrednosti, tako da so fragmenti v pravem vrstnem redu. S tem končni gostitelj pridobi vsa sporočila, katere je potreboval za sestavo poti, po kateri so se prenašali paketi od vira pa do destinacije.

5. OPORTUNISTIčNO OZNAčEVANJE

Kadar ni dovolj priložnosti za OPM, obstaja verjetnost, da se deli sporočila, ki čakajo v medpomnilniku (angl *buffer* usmerjevalnika že dolgo izgubijo zaradi preliva (angl. *overflow*). Za rešitev tega problema, v tem rzdelku predstavimo napredno zasnovo za izvajanje oportunističnega označevanje (angl. *Advanced Opportunistic Piggyback Marking*) - AOPM.

5.1 Motivacija

Na sliki 3 je prikazano, kako ima lahko dostava fragmentov koristi od *one hop* označevanja. Predpostavimo, da ima usmerjevalnik R_2 v svojem medpomnilniku nekaj fragmentov, ki jih mora dostaviti v D, vendar stopnja prometa proti cilju nizka znotraj nekega časovnega okna. V tem primeru izkoristimo *one-hop piggyback opportunity* in pošljemo sporočilo naprej korak po korak (hop by hop). Kakor je razvidno iz slike 3, bo paket poslan iz R_6 do R_7 prenašal tudi fragment sporočila iz R_2 do R_3. Zaradi pričakovanja, da bodo segmenti poti, ki so bližje cilju bili vsebovani v večih poteh do cilja, lahko ta metoda potencialno poveča verjetnost, da bodo framenti sporočila prišli neposredno do možnosti za neposredno označevanja. Kljub temu, da je mehanizem enokoračnega označevanja intuitiven, preprosta uporaba požrešnege strategije in izkoriščanja vsake možnosti za izvajanje posameznega koraka ni zaželjena, saj bi povečala obremenjenost usmerjevalnikov in tako potencialno povzročila, da bi zamudili naslednjo možnost za pošiljanje sporočila. Za učinkovito izkoriščanje možnosti koračnega označevanja, predlagamo zakasnitveno oportunistino metodo za sledenje IP naslovov, kjer je prenos sporočil voden z zamudo.

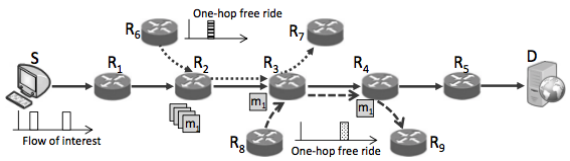


Figure 3: One hop delovanje.

5.2 Zasnova sistema

1. *Two Piggyback Marking Modes*: definiramo dva načina označevanja: neposrednega in one-hop način. Privzeto AOPM deluje v neposrednem. Usmerjavalniki naložijo fragmente sledilnega sporočila v neoznačene pakete, ki potujejo mimo in lahko nosijo te fragmente do cilja. Če fragment sporočila predolgo ostane v medpomniliku usmerjavalnika, se za ta fragment vklopi one-hop način. V tem primeru bo fragment posredovan do naslednjega usmerjevalnika, kjer bo spet prešel v neposreden način, dokler spet ne preteče dovolj časa. Obstaja kompromis med sledilno zakasnitvijo in računsko zahtevnostjo v obeh načinih. V AOPM se ta kompromis da nastaviti s parametrom.

2. Sprožitev sledenja: tudi v AOPM se uporabi TTS za sprožitev sledenja kasneje v omrežju na za to usposobljenih usmerjevalnikih. Za učinkovito označevalno sledenje se uporablja predpomnenje metapodatkov (angl. *metadata cache*) in dve vrsti fragmentov (ena za vsak način delovanja) v vsakem usmerjevalniku.

3. Dostava fragmentov sporočila: Algoritem prikazan na sliki 4 opisuje kako usmerjevalniki sprejmejo paket P po sprožitvenem sporočilu (vrstice 9-20). Če je P označen, ampak ni TTS, algoritem prvo preveri ali je bil fragment sporočila v P označen za one-hop dostavo (vrstica 9). V funkciji $isMarkedForOnehop(P)$, za ugotavljanje načina označevanja. Usmerjevalnik R zajame identifikator zajema iz AOPM označevalnega polja v P in preveri ali se je zgodil zadetek v metadata cache-u. V primeru zadetka, če se označen paket in fragment sporočila razlikujeta v naslovih, to pomeni, da je paket uporabljen kot one-hop prenašalec in funkcija vrne $TRUE$. Nato usmerjevalnik R zajeme informacije o označevanju, generira fragment sporočila in ga ustavi neposredno v ustrezno vrsto. Ta fragment je nastavljen v neposreden način in v vrsti čaka na označevalno priložnost (vrstice 9-11). Po tem, je P označen kot neoznačen (vrstica 12). V primeru, da ni zadetka v metadata cache-u, bo usmerjevalnik posredoval paket nemoteno.

   Ko R prejme neoznačen paket P, prvo preveri ali lahko P nosi kakšen fragment sporočila v one-hop vrsti do njegovih naslednjiih usmerjevalnikov (vrstica 15). To je moč doseči s primerjavo P-jev next-hop in next-hop fragmenta sporočila v metadata cache-u. V primeru, da noben fragment sporočila ne ustreza ali ustreza kakšen fragment iz vrste neposrednega načina, ki je namenjen na isto lokacijo kot P (vrstica 16). Ob odkritju fragmenta sporočila v katerikoli od vrst, bo usmerjevalnik označil P s fragmentov sporočila in ga odstranil iz ustrezne vrste. Nato R posreduje P naprej do naslednjega usmerjevalnika.

```
1  Procedure: Receive(Packet P)
2  if IsMarked(P) == TRUE && P.dest ≠ R.addr then
3      //P is a marked packet and R is not the destination of P;
4      if IsTTS(P) == TRUE then
5          UpdateMetadataCache(P);
6          M=GenerateMessage(P);
7          F=Fragmentation(M);
8          DirectQueueInsert(F);
9      else if IsMarkedForOnehop(P) == TRUE then
10         f=ExtractMark(P);
11         DirectQueueInsert(f);
12         ResetMarkingFields(P);
13
14
15 if IsMarked(P) == FALSE then
16     if IsOnehopQueueCarrier(P) == TRUE then
17         f=GetFragmentFromOnehopQueue(P);
18     else if IsDirectQueueCarrier(P) == TRUE then
19         f=GetFragmentFromDirectQueue(P);
20     if f != NULL then
21         Mark(P, f);
22
23 Forward(P);
```

Figure 4: Algoritem, za označevanje.

4. Rekonstrukcija sporočila poteka na isti način, kot pri tradicionalnem OPM.

6. PRILAGODLJIVO OGRODJE ZA SLEDENJE NA OSNOVI OZNAČEVANJA

V tem razdelku bomo predstavili prilagodljivo ogrodje za sledenje na osnovi označevanja (angl. *Flexible Marking-Based Traceback (FMBT) framework*).

6.1 Pregled

FMBT ogrodje je predstavljeno na sliki 5. Sledilno sporočilo je v ogrodju možno dostaviti do poljubnega gostitelja. Kakor je razvidno iz slike 5, lahko usmerjevalnikih, ki zaznajo sumljive pakete pošljejo sporočilo o tem do sledilnega strežnika (angl. *traceback server*) v oblaku. Ponudniki internetnih storitev (angl. *Internet Service Prodicers* - ISP) imajo lahko postavljene takšne strežnike in s tem zagotovijo storitev, za ugotavljanje izvora sumljivih paketov končnim uporabnikom. Kadar žrtev želi pridobiti podatke o napadu, podatke pridobi s poizvedbo na tak strežnik. Implementacija sistema temelji na zasnovi, da je možno sporočila pošiljati na poljuben strežnik brez dodatnih obremenitev.

Sistem za zaznavo napadov ali mrežni operater lahko sprožita sledilni mehanizem. Na primeru slike 5 vidimo, da je prvi usmerjevalnik z možnostjo sledenje usmerjevalnik R_1 in ta označi paket kot sumljiv in sproži sledenje. Takemu paketu pravimo sprožilni sledilni signal (angl. *traceback trigger signal* - TTS). TTS vsebuje identifikacijo zajemnika in ciljni IP naslov. TTS ne vsebuje nobenega fragmenta sledilnega sporočila, saj bodo vsa sporočila o sledenju poslana na specificiran sledilni strežnik. Ko naslednji usmerjevalnik R_2 prejme TTS, ta generira sledilno sporočilo in ustvari fragmentacijo tega sporočila, za kasnejše posredovanje po omrežju. Prenašanje fragmentov sporoila, ki se nahajajo v

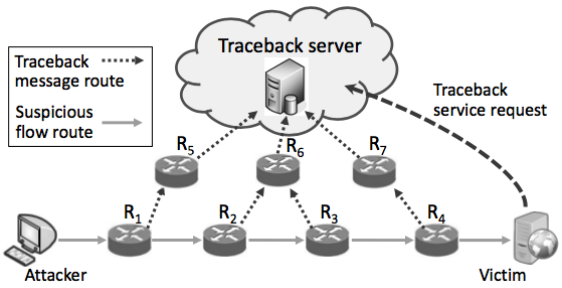


Figure 5: Struktura FMBT ogrodja.

medpomnilnikih usmerjevalnikov se po omrežju prenešajo neposredno ali po principu one-hop piggyback marking.

6.2 Lastnosti FMBT ogrodja

FMBT ogrodje ima nekaj ključnih prednosti pred ostalimi:

1. Ohranja zasebnost: eden glavnih zadržkov uporabe sledenja je razkritje topologije omrežja. Tradicionalni MBT pristopi pošiljajo informacije o označevanju končnim uporabnikom in tako tvegajo razkritje komercialno občutljivih informacij o usmerjevalnikih zunaj ISP-ja. FMBT že v osnovi ohranja zasebnost, saj podatke stranjuje na zaščitenih in točno določenih sledilnih strežnikih. Te strežnike upravlja ISP neposredno in tako so lahko občutljivi podatki ustrezno zaščiteni in niso na voljo zunanjim uporabnikom
2. Omogoča forenzično analizo: z uporabo FMBT so sledilna sporočila lahko dlje shranjena na sledilnih strežnikih v primerjavi z tradicionalnimi MBT pristopi. Tako FMBT omogoča "postmortem"analizo, torej analizo omrežnega prometa dolgo po samem dejanju, ko je zbrano dovolj dokazov, ki omogoča tožilcem podati kazensko ovadbo.
3. Spodbuja uvajanje sledenja: Ker sledilne tehnike zahtevajo aktivno vlogo usmerjevalnikov, lahko ISP igrajo ključno vlogo v sledilnih sistemih. Ena glavnih ovir pri vzpostavitvi takih sistemov so ekonomske narava, saj ISP nimajo nobenih potreb po IP sledenju v njihovih omrežjih, če ne morejo si investije povrniti. FMBT omogoča, da se sledenje izpostavi kot storitev, ki jo je možno tržiti.
4. Skalabilno in robustno sledenje: v obstoječih MBT metodah so vsa sledilna sporočila posredovana do cilja sledenja (žrtve). To lahko deluje dobro kadar je IP sledenje uporabljeno v ne-zlonamernih namenih za analizo omrežja z nizko intentziteto prometa. Vendar v primerih napadov, pošiljanje sledilnih sporočil še dodatno obremenjuje sistem, saj se vsi fragmetni sporočil pošiljajo v isti smeri, kot promet. Sporočila je potrebno analizirati in rekonstruirati omrežno pot. Situacija postane še slabša, če ima žrtev omojena sredstva, ki so zasičena z napadalnim prometom. FMBT pa to dodatno breme prestavi na svoj določen strežnik, ki ima na voljo več sredstev, kar izboljša robustnost in skalabilnost sistema.

7. EVALVACIJA

V tem razdelku prikažemo evalvacijo OPM in AOPM za različne omrežne scenarije z uporabo ns-2 simulatorja.

7.1 Simulacijski setup

V vsakem od poskusov sledimo usmerjevalni poti toka, ki je določen z izvor-cilj parom v linearnem omrežju. Povezave med usmerjevalniki so dvosmerne z $1ms$ zakasnitve in $100Mbps$ kapacitete. Uporabljen je eksponenten vir prometa v omrežju, kar pomeni, da sta čas aktivnega prometa in čas, ko prometa ni modelirana z eksponentno funkcijo. Kot poskus približka realnemu svetu so stopnje prometa v omrežju generirane iz naključnega vzorčenja podatkov iz CAIDA DDoS Attack 2007 Dataset.

7.2 Metrike uspešnosti

Za primerjavo rezultatov so uporabljene naslednje metrike:

- Razmerje dostavljenih fragmentov: razmerje med številom izmerjenih fragmentov, ki jih je prejela žrtev in skupnim številom fragmentov sporočila, ki so jo usmerjevalniki poslali
- Normalizirana sledilna zakasnitev: pretečen čas od časa, ko je TTS generiran na začetnem umserjevalniku in časom, ko je žrtev prejela zadnji fragment sporočila, ki je povezan s tem istim TTS. To razmerje je normalizirano z razmerjem dostavljenih fragmentov.
- Normalizirana čakalna vrsta operacij: skupno število čakajočih operacij, ki nastanejo za pošiljanje fragmentov sporočila, ki jih je sprožil TTS deljena s pripadajočimi razmerjem dostavljenih fragmentov.
- Število prenašalcev: skupno število prenašalcev, ki so bili uporabljeni za dostavo fragmentov sporočila za vsak TTS.

7.3 Rezultati simulacija: scenarij 1

V scenariju 1 je opravljena analiza v razmerah, kjer je dovolj priložnosti za neposredno ali one-hop označevanje, stopnja prometa ciljnega toka je relativno nizka. Čas preklopa med različnima načinoma označevanje je $1s$ in vsako sporočilo je razdeljeno na 10 fragmentov. Pot med izvorom in ciljem je doga 25 povezav. Za promet v ozadju je generiranih 10 naključnih tokov, ki pa se vsi zaključijo v željenem, končnem usmerjevalniku. Te tokovi služijo predvsem kot priložnost za neposredne prenašalce. Poleg tega je vzpostavljenih še 10 povezav med nadključnimi usmerjevalniki.

Slika 6 prikazuje primerjavo za scenarij 1, kjer je ciljni tok omejen na $50kbits/s$. Iz slike 6(a), se vidi da AOPM dostavi 99% vseh framentov sporočila. Slika 6(b) prikazuje CDF krivulje normalizirane sledilne zakasnitve. Slika 6(c) prikazuje primerjavo med normalizirano čakalno vrsto operacij.

7.4 Rezultati simulacija: scenarij 2

V scenariju 2 so odstranjeni vsi naključno generirani tokovi, ki se končajo pri žrtvi in niso del napada. Tako dobimo 20 naključnih povezav med poljubnima usmerjevalnikoma v omrežju. Vsi tokovi v omrežju so omejeni na $50kbit/s$.

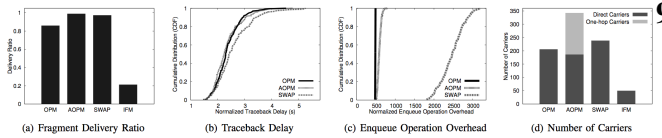


Figure 6: Primerjava zmogljivosti sistemov v scenariju 1.

1. Vpliv dolžine poti: preverimo učinkovitost sheme s spreminjanjem dolžine poti med izvorom in ciljem od 10 do 25. Te številke so izbrane, saj v povprečju paket potrebuje približno 15 korakov, maksimalno pa 25 korakov, da prispe do cilja. Na sliki 7 primerjamo razmerje dostavljenih paketov za različne sheme.

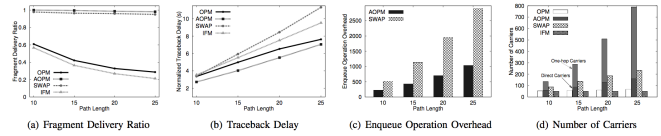


Figure 7: Primerjava zmogljivosti sistemov v scenariju 2: vpliv dolžine poti.

2. Vpliv števila fragmentov sporočila: primerjamo zmogljivost sistema s spreminjanjem števila fragmentov sporočila, ki so določeni z različnimi shemami kodiranja vsebina sporočila. Dolžina poti je nastavljena na 25. Slika 8 prikazuje razmerje dostavljenih fragmentov na sporočilo. Več kot je potrebnih prenašalcev in zato sledenje traja dlje. Posledično so rezultati slabši v vseh shemam, saj se število fragmentov na sporočilo poveča. Kljub temu AOPM ohranjaa visoko raven dostavljenih fragmentov: čez 90% ne glede na povečanje števila fragmentov.

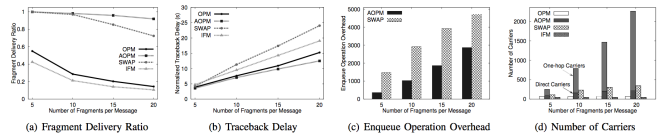


Figure 8: Primerjava zmogljivosti sistemov v scenariju 2: vpliv števila fragmantov na sporočilo.

8. ZAKLJUčEK

V članku je bil predstavljeno oportunistično označevanje paketov, nov pristop IP sledenja. Glavna ideja predlagane rešitve je izkoriščanje možnosti za dostavo paketov za robustno dostavo framentov sporočila do končnega cilja. Predstavljen je bil prilagodljivo ogrodje za označevanje sledilnih paketov (FMBT), ki ponuja neki prednosti pred klasičnimi pristopi označevanja paketov. Izvedena je bila analiza zmogljivosti sistema.

9. REFERENCES

[1] Long Cheng, Dinil Mon Divakaran, Wee Yong Lim, Vrizlynn L. L. Thing, Balancing Agility and Discipline: Opportunistic Piggyback Marking for IP Traceback, *Cyber Security & Intelligence Department, Institute for Infocomm Research (I2R), Singapore.*

Del IV
Storitve in Internet

Forenzična raziskava primerov spletnega zalezovanja, rešenih z uporabo analize vedenjskih karakteristik

[Razširjeni povzetek]

Neža Belej
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana, Slovenija
nb8901@student.uni-lj.si

Blaž Kovačič
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana, Slovenija
bk9242@student.uni-lj.si

POVZETEK

Analiza vedenjskih karakteristik (angl. behavioural evidence analysis) je postopek, ki pripomore k razumevanju digitalnih dokazov in rekonstrukcije zločina. Kljub pomembnosti analize vedenjskih karakteristik še ne obstaja veliko raziskav o apliciranju tega postopka na kriminalna dejanja. V seminarski nalogi bomo opisali pomen analize vedenjskih karakteristik (v nadaljevanju AVK) , pri čemer bomo upoštevali izvlečke iz članka Forensic investigation of cyberstalking cases using Behavioural Evidence Analysis [2]. Izkaže se, da AVK pripomore k sami raziskavi, k razumevanju zločinca in žrtve ter k razbiranju sledi iz digitalnih dokazov. Tako raziskava postane bolj smiselna in natančna.

Ključne besede

Analiza vedenjskih karakteristik, spletno zalezovanje, interpretacija digitalnih dokazov, rekonstrukcija, digitalna preiskava

1. UVOD

Ob množični uporabi interneta je spletno zalezovanje (angl. cyberstalking) postalo pogosto prisoten zločin. Po raziskavah obstaja precejšnje število študentov in mladoletnikov, ki so bili žrtev tega zločina [2]. Spletna anketa 1588 mladoletnikov (10-15 let) je pokazala, da jih je bilo kar 33% žrtev nadlegovanja na spletu; v 15% je šlo za spolno nadlegovanje [11]. Druga anketa 2839 odraslih je odkrila, da je bilo 40% anketirancev žrtev različnega spletnega zalezovanja [4]. Dubajska policija je objavila 39% rast spletnega zalezovanja v letih 2000-2014. Gre torej za pogost tip spletnega zločina, ki pa je precej slabo razumljen.

Uporaba naprednih tehnologij zločincev prinaša nove izzive za digitalne preiskovalce, število raziskav na temo spletnega zalezovanja pa je omejeno. V zvezi s spletnim zalezovanjem se pojavljajo nove vrste dokazov, ki pa jih je potrebno natančno analizirati. V tej fazi si preiskovalec lahko pomaga z **analizo vedenjskih karakteristik** [10]. Ta način naj bi pripomogel k vzpostavitvi natančnega profila zločinca, saj se nanaša tako na prepoznavanje motivacije storilca kot tudi na razumevanje odnosa med žrtvijo in storilcem ter na tveganje zločinca ob fizičnem nadlegovanju žrtve.

V seminarski nalogi bomo na podlagi članka [2] opisali uporabnost AVK v 20 primerih spletnega zalezovanja poročil Dubajske policije.

Najprej bomo opisali ozadje raziskave: definirali bomo pojma analiza vedenjskih karakteristik in spletno zalezovanje. V pregledu področja se bomo osredotočili na dosedanje raziskave, pri čemer bomo posebno sekcijo namenili varnosti modernih socialnih omrežij. V nadaljevanju bomo razložili metodologijo dela - kako in kje je potekala raziskava, rezultat apliciranja teh metodologij na primere bomo predstavili v poglavju Rezultati.

2. OZADJE

2.1 Analiza vedenjskih karakteristik

Analiza vedenjskih karakteristik je raziskovalna strategija, ki uporablja vedenjske vzorce in osebnostne karakteristike z namenom dobiti boljšo sliko o zločincu. Sestoji iz štirih faz:

1. **Nedvoumna forenzična analiza** (angl. equivocal forensic analysis), ki temelji zgolj na dejstvih. Dokaze interpretira z znanstvenim pristopom, celovito in objektivno. Rezultat je teorija, podpkrepljena z dejstvi, ki nam pomaga videti širšo sliko zločina.

2. **Analiza žrtve** (angl. victimology) preiskuje sledi žrtve (izgled, zakonski stan, stil življenja) z namenom odkriti, zakaj je bila ravno ta žrtev cilj napada. Ta faza lahko pripomore h kasnejši analizi storilca in odnosom z njim.

3. **Karakteristike virtualnega mesta zločina** raziskovalcem da vpogled v storilca in njegove cilje. Te karakteristike lahko raziskovalcu dajo nove informacije o morebitnih dodatnih lokacijah dokazov.

4. **Analiza zločinca** na podlagi prejšnjih treh korakov sestavi teorijo o vedenjskih in osebnostnih značilnostih storilca ter tako pridobi njegov profil.

2.2 Spletno zalezovanje

Spletno zalezovanje je vrsta zločina, ki uporablja informacijske tehnologije (epošta, socialna omrežja ipd.) z namenom nadlegovati neko drugo osebo (ali skupino oseb) in ji povzročiti občutke strahu ali grožnje [6]. Storilec torej grozi, po krivem obtožuje to osebo, jo nadzoruje ali celo posnema. Za razliko od fizičnega nadlegovanja (angl. stalking), tu ne gre za situacije, ko bi bila žrtev ves čas pod nadzorom in na očeh zločinca. Tu zločinec na primer žrtvi pošilja e-pošto o njeni trenutni lokaciji in s tem žrtvi daje občutke strahu. Informacijske tehnologije omogočajo zločincem, da ostanejo skriti. Tako lahko pridejo na račun tudi zločinci, ki si sicer ne bi upali nekoga fizično zalezovati [5]. Lahko si naredijo tudi več spletnih profilov in tako še otežijo preiskavo.

2.3 Trajanje in narava spletnega zalezovanja

Angleška psihologa McFarlen in Bocij sta na naboru 24 primerov spletnega zalezovanja raziskovala trajanje in naravo tega zločina. Trajanje spletnega zalezovanja po McFarlenu in Bociju [6] znaša od 1 dne pa do 5 let, s povprečjem 11,5 mesecev. Primer z enodnevnim spletnim zalezovanjem je prerasel v fizično zalezovanje s telefonskimi klici in grožnjami žrtvi. Drugouvrščeni najkrajši primer je tako trajal 17 dni. Najpogostejše sredstvo komuniciranja spletnih zalezovalcev je bila e-pošta, sledi uporaba spleta iz službenega omrežja storilca; pogosti so tudi spletni forumi, kot na primer Usenet ali Bulletin boards.

V kar 13 primerih se je spletno zalezovanje dogajalo v kombinaciji s fizičnim: 6 žrtev je bilo zalezovanih v lastnem domu, 3 v službi, 3 na javnih mestih, 1 oseba pa je bila nadzorovana s kamerami.

3. PREGLED PODROČJA

3.1 Razumevanje spletnega zalezovanja

Na temo spletnega zalezovanja obstaja relativno malo raziskav. Profil teh zločincev je podoben ostalim spletnim zločincem: imajo željo po nadzoru žrtve ali po intimnem razmerju z njo. Pogosto so sami žrtve raznih osebnostnih ali psiholoških motenj. Zločin je lahko tudi posledica razpada zveze, psiholoških težav ali osamljenosti napadalca.

Poznamo več različnih delitev tradicionalnih, fizičnih zalezovalcev. McFarlane in Bocij[6] sta v svoji raziskavi poudarila, da splet prinese fizičnemu zalezovanju novo dimenzijo in tako na podlagi realnih primerov razvila 4 tipe zločincev, ki temeljijo na interpretacijah žrtev spletnega zalezovanja:

- **Maščevalni zalezovalci** (angl. vindictive cyberstalkers), ki trpijo zaradi psiholoških motenj in se brez posebnega razloga neusmiljeno spravijo nad žrtev.
- **Sestavljeni zalezovalci** (angl. composed cyberstalkers), ki neprestano nadlegujejo svoje žrtve in jih spravljajo v stres. Njihov namen je torej povzročiti stres žrtvam in z njimi ne želijo vzpostaviti intimnega odnosa.
- **Intimni zalezovalci** (angl. intimate cyberstalkers), ki želijo pridobiti žrtvino pozornost in z njo imeti intimno razmerje. Ponavadi vedo veliko o svoji žrtvi.
- **Skupina zalezovalcev** (angl. collective cyberstalkers) se nanaša na skupino ljudi, ki zalezujejo svoje žrtve s pomočjo komunkacijske tehnologije.

3.2 Povezava analize vedenjskih karakteristik s primeri digitalne forenzike

Uporaba AVK se je izkazala kot zelo uporabna že v primeru tradicionalnih zločinov (n. pr. umori in posilstva). Izkazalo se je, da je v digitalnih zločinih prav tako uporabna: na računalnikih se najde mnogo podatkov (zgodovina brskalnikov, izbrisane datoteke, pogovori na socialnih omrežjih, časovni žigi), ki razkrijejo pomembne informacije o zločinčevih lastnostih. Ti pomagajo preiskovalcem pri globljem razumevanju dinamike zločina in posledično prinašajo zadovoljivo rekonstrukcijo zločina. Vseeno pa ne obstaja veliko znanstvenih raziskav o apliciranju AVK na digitalne zločine, predvsem na zločin *spletnega zalezovanja*. Silde in Angelopoulov[8] sta razvila metodologijo za uporabo strategije AVK v primeru spletnega zalezovanja (slika 1). AVK sta prepoznala kot sredstvo, ki osredotoči raziskavo na lokacije, ki z večjo verjetnostjo vsebujejo pomembne dokaze.

Rogers[7] je razvil 6 faz integriranja AVK v forenzično preiskavo:

1. klasifikacija primera
2. analiza konteksta
3. zbiranje podatkov
4. statistična analiza
5. časovna analiza / vizualizacija
6. odločitev / mnenje

3.3 Varnostne in zasebnostne težave socialnih omrežjih

Trend socialnih omrežij neprestano narašča. Na sliki 2 vidimo število uporabnikov najpriljubljenejših socialnih omrežij. Naivni uporabniki, predvsem najstniki, na socialnih omrežjih razkrivajo svoje osebne informacije kot so na primer kontaktne informacije, trenutna lokacija, status razmerja in razne fotografije. Če te informacije pustijo odprte javnosti, odprejo možnosti neki drugi osebi za vdor v njihovo zasebnost. 63% uporabnikov omrežja Facebook ima svoj profil odprt za javnost, kar pomeni naslednje: če bo nekdo v spletni brskalnik vnesel niz "Ime Priimek Facebook", bo v trenutku lahko dostopal do informacij, ki jih oseba deli na svojem profilu [3].

4. METODOLOGIJA

Ta del opisuje izvajanje raziskave, katere članek opisujemo. Raziskava je bila izvedena v laboratorijih dddelka za elektronske dokaze Dubajske policije. Uporabljeni podatki so bili kopije vsebine računalnikov, ki so bili zaseženi v primerih spletnega zalezovanja.

4.1 Izbira in število primerov

Raziskava je narejena na podlagi 20 primerov različnih variant spletnega zalezovanja (nadzorovanje, lažne obtožbe,

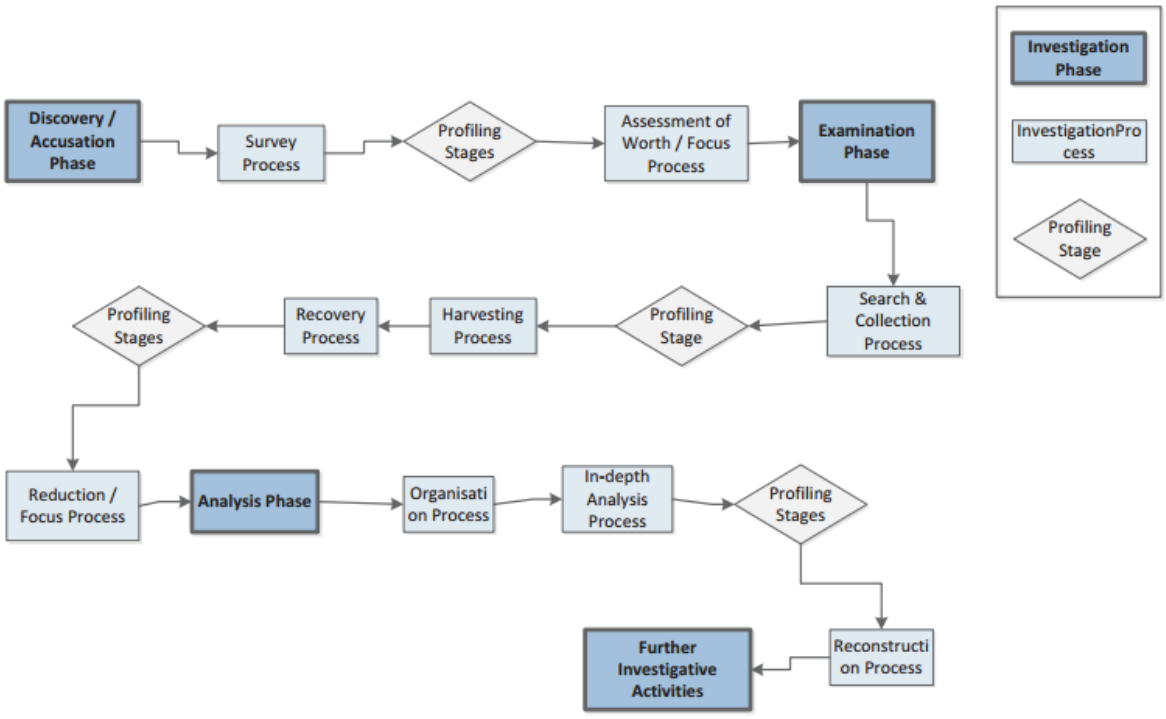


Figure 1: Povezava AVK s tradicionalnim pristopom (Silde, Angelopoulou [8])

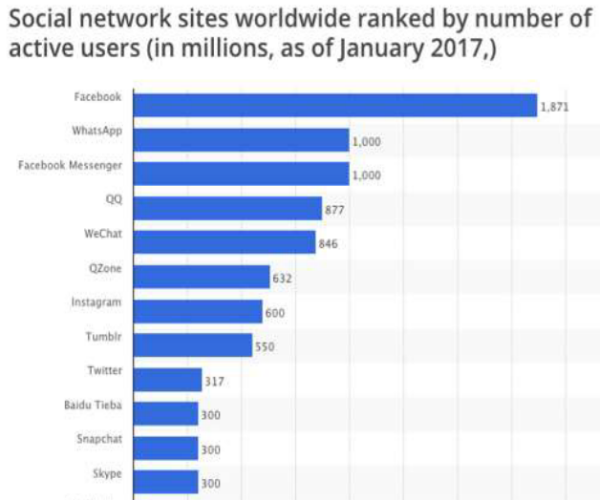


Figure 2: Število uporabnikov najpriljubljenejših socialnih omrežij po članku [3]

ipd.), izvedenega v Dubaju v zadnjih 6 letih. V zločine je bilo vpletenih 31 računalnikov, katerih diski so bili pregledani in arhivirani s strani oddelka za elektronske dokaze Dubajske policije.

4.2 Viri podatkov

Viri podatkov raziskave vsebujejo predvsem gradivo, najdeno na prej omenjenih diskih računalnikov. Sem spadajo tudi razni dokumenti, povezani s primeri. Tukaj so računalniki predvsem last žrtev, medtem ko so, če je to le bilo mogoče, v nekaterih primerih računalniki tudi storilčevi. Zaseženo gradivo je pripomoglo k rekonstrukciji komunikacije med storilcem in žrtvijo, identifikaciji motivacije zločincev, razumevanju konteksta zločina ter odnosa med zločincem in žrtvijo.

4.3 Zbiranje podatkov in analiza

Raziskava vsebuje kombinacijo standardnih tehnik digitalne forenzike in analize vedenjskih karakteristik (AVK).

Za vsak primer je bilo pred samo analizo potrebno razumevanje konteksta zločina. Vsa dokumentacija je bila natančno preučena še pred analizo slikovnih gradiv. Aktivnosti zločinca in žrtve so bile natančno analizirane s pomočjo njunih pogovorov, e-pošte, izbrisanih datotek in drugih sledi (registri, zgodovina brskalnika, slike, videi ipd.). Proces je bil krožen in iterativen, kar pomeni, da se je ponovil večkrat. Uspešno je bila združena strategija tradicionalne digitalne analize z AVK. Vsak digitalni dokaz je bil preučen s pomočjo AVK in je po zaključku le-te predvidel nove lokacije dokaznih gradiv primera in prispeval k razumevanju zločinca

in žrtve. Povzete so bile najdbe vseh primerov.

Med samo analizo so bili zbrani tudi podatki, ki bi se lahko primerjali s preteklimi primeri na tem področju. To se nanaša predvsem na značilke kot so starost žrtve / osumljenca, spol, etnična pripadnost, zaposlitveni status, zakonski stan, računalniška pismenost itd. Vključene so bile tudi spremenljivke kot so prejšnji zločini storilca in njegove psihološke lastnosti: lastnosti zločina kot npr. trajanje nadlegovanja ali tip komunikacije (FB chat, e-pošta ipd.). Upoštevani so bili naslednji faktorji, ki se nanašajo na način in vsebino pogovorov: posnemanje, izpoved ljubezni, komentarji, povezani s spolnostjo, grožnje in nasilne pripombe.

5. REZULTATI

V tem poglavju bomo povzeli rezultate apliciranja prej omenjenih metodologij na dejanske primere. V analizo podatkov ([2]) je bilo vključenih 20 žrtev in prav toliko storilcev.

5.1 Starost

Iz podatkov, pridobljenih v preiskavah primerov, je bilo ugotovljeno, da je starostni razpon žrtev 23 - 48 let, razpon storilcev pa 21 - 63 let. Večina žrtev (skupno 14, kar predstavlja 60%), pripada starostni skupini nad 31 let, od tega spada 30% (3 žrtve) v razpon med 31 in 40 let, ostale tri pa v skupino nad 40 let. Preostalih osem žrtev pripada skupini med 21 in 30 let. Za razliko od žrtev so storilci večinski delež predstavljali v skupini nad 40 let (9 posameznikov, oziroma 40%), nekoliko nižji delež, (7 osebkov, 35%), spada v razpon med 31 in 40 let, ostali štirje (20%) pa v skupino med 21 in 30 let.

5.2 Spol

Moški spol predstavlja večinski delež storilcev. Kar 16 / 20 (80%) storilcev je bilo moških, medtem ko so ženske predstavljale le 20% storilk. Analogno od storilcev je večina žrtev (75%) ženskega spola, kar je potrdila tudi druga raziskava ([9]), ostalih 25% pa predstavlja moške.

5.3 Zaposlitveni status

Večina storilcev in žrtev je bila v času zločinov zaposlenih (80 %), je pa nivo zaposlitvene stopnje precej nihal. Desetina žrtev je imela stopnjo visoke zaposlitve, 60% pa je imelo srednjo stopnjo. Največji delež (70%) storilcev je imelo srednjo stopnjo, 20% pa nižjo stopnjo zaposlitve.

5.4 Etničnost

Večina osebkov v analizi spletnih zalezovanj pripada etničnim skupinam bližnjega vzhoda. Bližnjevzhodne žrtve predstavljajo 45%, storilci pa 40% vseh primerkov. Segmentu z bližnjega vzhoda sledijo kavkazijci. Kavkazijskih žrtev je bilo 30%, storilcev pa 35%. Na zadnjem mestu so posamezniki z daljnjega vzhoda, pri katerih je delež žrtev in storilcev enak, znaša pa 25%.

5.5 Odnos storilec - žrtev

Večina primerov spletnih zalezovanj je potekala med moškim in žensko (60%). S precej manj, le 20% sledi obrnjena vloga, torej ženska - moški. Še manj (15%) primerov je potekalo med dvema ženskama, najmanj (5%) primerov pa med dvema moškima.

5.6 Razmerje med žrtvijo in storilcev

V nasprotju z rezultati drugih raziskav (n. pr. [6]) so imele v tej raziskavi vse žrtve in storilci predhoden odnos. Večina (kar 40%) primerov sta bila žrtev in storilec sodelavca, v 35% primerih pa sta bila nekdanja partnerja. V preostalih 20% sta bili enakomerno zasedani kategoriji oseb, ki so bili bodisi znanci bodisi sta se osebi spoznali na spletu, za zadnjih 5% pa podatek o razmerju ni znan.

5.7 Trajanje primera

Večina primerov spletnega zalezovanja, kar 12 izmed 20 primerov, je trajala šest mesecev ali manj. Le 20% primerov je trajalo v obdobju med sedmimi meseci in enim letom. Najmanj, le 5% primerov, je bilo dvoletnih, za preostalih 15% pa podatka ni.

5.8 Način delovanja spletnega zalezovalca

Največji delež spletnih zalezovanj je bil storjen preko e-pošte; skupno je zavzemal 55%. Drugi najbolj zastopan način izvajanja, ki predstavlja četrtino vseh primerov, je bilo spletno zalezovanje preko socialnih omrežij - večinoma preko omrežij Facebook in Twitter. Storilci zločinov so uporabljali ta omrežja za objavo sovražnih, sramotilnih ali drugih komentarjev o žrtvah. Strani za spletne zmenkarije sledijo socialnim omrežjem s 15%. Na teh omrežjih so storilci oponašali svoje žrtve, pisali lažne objave o spolnih fantazijah ali željah in spodbujali ostale člane teh omrežij k navezovanju stika z žrtvami. Na zadnjem mestu (5%) se nahajajo ostali forumi in spletišča, nekateri storilci pa so svoje žrtve zalezovali tudi v fizičnem svetu, izven računalniškega sveta.

Tematske analize primerov so pokazale, da je večina storilcev uporabljala bodisi nasilni govor bodisi je izkazovala ljubezen in obsesijo do svojih žrtev. V tretjini primerov je zločinec izvedel krajo identitete in se na spletnih omrežjih pretvarjal, da ja on predstavlja oškodovano žrtev, v nobenem od teh primerov pa ni storilec pozval nekoga drugega, da se mu pridruži v njegovem početju.

5.9 Dejanja in citati spletnih zalezovalcev

Preglednica 1 prikazuje najverjetnejše motivacije in pripadajoče akcije ter citate zločincev.

5.10 AVK kot preiskovalno orodje v digitalni forenziki

Združevanje AVK z ostalimi standardnimi postopki digitalne forenzike se je v analiziranih 20 primerih spletnega zalezovanja pokazalo kot zelo dobra praksa. Za prikaz pomembnosti združenih pristopov so v naslednjih razdelkih navedeni primeri interpretativnih in preiskovalnih koristi.

5.10.1 Osredotočenost, hitrost in preiskovalne smernice

V nekem primeru je žrtev poročala o sumljivih aktivnostih na njenem računalniku, saj so se ji na zaslonu neprestavno pojavljala sovražna sporočila. Preiskovalci so najprej pregledali registre žrtvinega računalnika, ki so razkrili nameščeno

Preglednica 1: Najverjetnejše motivacije in pripadajoče akcije ter citate zločincev

Motivacija	Dejanje / citat
Obtoževanje po krivem	Objavljanje nespodobnih fotografij žrtev na socialnih omrežjih ali spletiščih za spletno zmenkarjenje.
	Pošiljanje žaljivih sporočil, nespodobnih fotografij ali informacij preko e-pošte žrtvinim prijateljem, sodelavcem ali družinskim članom, navadno preko žrtvinega e-poštnega naslova.
Oznanjanje ljubezni	Pošiljanje zaporednih ljubezenskih e-poštnih sporočil z namenom izkazovanja storilčeve naklonjenosti, ljubezni ali obsesije z žrtvijo.
	Pošiljanje e-poštnih sporočil, v katerih so bili omenjeni spomini preteklega razmerja med storilcem in žrtvijo.
	Pošiljanje zasebnih, lahko celo pornografskih slik.
Maščevanje / jeza	"Kar si storil(a) boš obžaloval(a) do konca življenja!"
	"Kjekoli že si...Našel/Našla te bom!"
	"[Ime Direktorja Podjetja] je nečasten, neprofesionalen in prevarant."
Sledenje žrtvi	Oddaljen dostop do žrtvinega računalnika.
	Zbiranje in organizacija informacij o žrtvi.

škodljivo programsko opremo, kar je bil dokaz o nepooblaščenem vdoru v preiskovani računalnik. Žrtev je, kot možna osumljenca, omenila dva osebi, s katerima je imela razmerje preko spleta. Preiskovalci so nato s svojo programsko opremo za zaznavanje slabogramja preiskali celoten žrtvin računalnik in odkrili škodljive datoteke in njihove lokacije na disku. Izmed vseh datotek je izstopala ena sama, ki se je nahajala v privzetem imeniku spletnih prenosov, služila pa naj bi kot odjemalec za spletne pogovore. Izkazalo se je, da je dvoklik na datoteko sprožil namestitev zlonamerne opreme, ki je bila najdena v registrih. S pregledom zabeležk o pogovorih se je izkazalo, da ji je škodljivo programsko opremo poslal eden izmed obeh omenjenih možnih osumljencev, s čimer se je potrdil sum o njegovem spletnem zalezovanju.

Ta primer je prikazal pomembnost AVK, saj so koristi, kot n. pr. usmerjanje preiskave v pravo smer in razumevanje vedenjskih vzorcev žrtve ali storilca, primer uspešno pripeljale do zaključka.

5.10.2 *Sklepanje na storilčeve motive in vedenje*

V drugem primeru je zločinec ustvaril nov profil z uporabo žrtvine identitete na spletišču za spletne zmenkarije, kjer je nato naložil in objavil njen e-poštni naslov, nespodobne slike in izmišljena sporočila o njenih spolnih fantazijah. S tem je želel ostale uporabnike spodbuditi k spolnim aktom z žrtvijo, ki je vse to opazila in dejanje prijavila policiji.

S preiskavo predpomnilnikov in zgodovine brskanja spletnih brskalnikov so preiskovalci ugotovili, da je bila žrtev redna obiskovalka omenjene strani za spletno zmenkarjenje. Iz zapisa pogovora med žrtvijo in policijo je razvidno, da je imela s storilcem nekoč odnos preko spleta, ki ga je tudi končala. Z branjem zabeležk so preiskovalci ugotovili, da je zločinec sprva poskušal obnoviti končano razmerje, kasneje pa je, ker se žrtev tega ni želela, pogovor prevesil v agresivnejšo in sovražno smer. S preiskavo njegovega računalnika je bilo ugotovljeno, da je v preteklosti večkrat vdrl v žrtvin profil na omenjenem forumu.

Ta primer je prikazal pomembnost digitalnih dokazov, saj so tekom preiskave odkrili žrtvine stalne obiske foruma in spoznavanje neznanih ljudi, kar jo je nemudoma spravilo v večjo nevarnost spletnih zalezovalcev.

Spletne komunikacije so se v večini primerov izkazale kot dokaz. Pogovori so pogosto vsebovali ponovljive značilnosti pisanja sporočil kot na primer način črkovanja, slovnične napake, nadimki ipd., kar je nakazovalo na iste osebe (storilce), pa njihovo tudi motivacijo za zločin.

Uporabniške datoteke in imeniki pa tudi predpomnilniki in zgodovinske datoteke spletnih brskalnikov so pogosto razkrile žrtvine življenjske navade in interese, med katerimi so bili tudi taki, ki so žrtev izpostavljali nevarnostim spletnih prevarantov.

Določen način spletnega zalezovanja, število datotek, njihove lokacije na disku in prisotnost sumljivih podatkov v datotekah so obstoječim profilom osumljencev dodali nove podatke in tako preiskovalcem omogočili podrobnejši vpogled v misli storilca.

5.10.3 *Identifikacija potencialnih žrtev*

Čas in trud za urejanje in kategorizacija žrtvinih datotek sta prikazovala storilčev odnos do morebitne žrtve. V enem izmed primerov so dokazi pokazali, da trenutna žrtev ni bila edina oseba, napadena s strani istega storilca. Preiskava njegovega računalnika je razkrila imenike s podimeniki, ki so vsebovali slike in podatke treh dodatnih žrtev.

Glede na način spletnega zalezovanja, tip datotek in njihovih lokacij na disku, časovne značke in izbrisane datoteke, najdene na storilčevem računalniku, preiskovalci lahko sklepajo na njegove vedenjske lastnosti. Če so bile na primer izbrisane datoteke žrtvine slike, ki so se nahajale v imeniku z žrtvinim imenom, je to nakazovalo na storilčevo zanimanje v določeno žrtev. Iz najdenih izbrisanih datotek so preiskovalci lahko sklepali, da se je storilec poskušal na tak način zavarovati pred najdbo bremenilnih dokazov.

5.10.4 Izločanje žrtev

Zadnji primer je vseboval žrtev in storilca, ki se je na socialnem omrežju Facebook pretvarjal, da je on izbrana žrtev in objavljal žrtvine slike z žaljivimi komentarji. Žrtvin glavni osumljenec je bil njen bivši soprog, ki se je od žrtve ločil (in na novo poročil z drugo žensko). Na računalniku njenega nekdanjega moža so našli slike žrtve in njunega otroka. S preiskavo zgodovine brskanja so ugotovili, da je bil vprašljiv Facebook profil večkrat dostopan s tega računalnika. V tistem trenutku so najdbe potrjevale žrtvine sume. Kasneje so preiskovalci s časovno analizo ugotovili, da je bil omenjeni Facebook profil večinoma dostopan med 9. uro zjutraj in 2. uro popoldan, ker pa se je osumljeni v teh urah vedno nahajal na delovnem mestu, je bila za krivo spoznana druga, edina preostala možna oseba, nova žena žrtvinega nekdanjega moža.

Da bi preiskovalci zagotovili kar se da zanesljivo teorijo o poteku zločina, mora biti vedno opravljena časovna analiza, dokazno gradivo pa mora biti predmet skupne preiskave. Tak pristop omogoča časovnico aktivnosti med žrtvijo in storilcem, ki pogosto pripomore k boljšemu razumevanju zločina in njegovi rekonstrukciji. Spremenljivi podatki časovnih značk datotek nakazujejo, da so storilci spremenili datoteke, značke same po sebi pa so lahko dokaz o tem, koliko časa je storilec datoteke že posedoval in koliko časa je bodisi načrtoval bodisi izvajal in imel namen izvajati zalezovanje.

5.11 Razprava

Večina zločincev je bila uvrščena v kategorijo sestavljenih in intimnih spletnih zalezovalcev, vendar je raziskava, izvedena v SEIC primerih ([1]) pokazala, da zločinci zaradi spremenljivega izvajanja zalezovanja ne morejo biti uvrščeni le v eno kategorijo. Prepoznane značilke in vedenjski vzorci so bili konsistentni s predhodnimi raziskavami ([6]). To nakazuje na dejstvo, da čeprav ima večina storilcev skupne demografske lastnosti (n. pr. kriminalna zgodovina), so določeni vedenjski vzorci, ki se unikatno pojavijo pri določenemu zločincu in njegovem načinu delovanja ([10]).

Opravljena raziskava je sledila standardnim štirim korakom AVK definiranih s strani Turveya ([10]). Rezultati so pokazali, da je tak postopek odločilno pripomogel k reševanju analiziranih primerov. Preiskavo je usmerjal v pravo smer in nudil informacije o nadaljnjih lokacijah dokazov. Preiskava se je na tak način odvila hitreje in podala boljše razumevanje ter interpretacijo vedenja posameznega storilca. V nekaterih primerih so bile identificirane tudi kasnejše morebitne žrtve določenega zločinca. Postopki so ključno vplivali tudi v primerih, kjer je bil isti račun na določenih spletnih straneh dostopan s strani različnih uporabnikov.

Ker po navadi v primerih spletnih zalezovanj ne gre za fizični kraj zločina (z izjemo primerov, pri katerih je storilec prešel v fizično zalezovanje) je računalnik omenjen kot primaren kraj zločina. Analizirani primeri so pokazali in potrdili Lockardov princip izmenjave podatkov, saj so storilci svoje sledi vedno pustili na vsaj enem računalniku.

Čeprav se je AVK izkazala za pomembno orodje v digitalni preiskavi spletnih zalezovanj ima tudi svoje slabosti. Kakovost podatkov v raziskavi je bila omejena s številom primerov v policijskih arhivih. Prav tako se preiskovalci ne smejo zanašati izključno na podatke pridobljene z AVK, zato izkušnje, znanje in kritično mišljenje preiskovalca igrajo ključno vlogo v priskavi. Po drugi strani so sicer preiskovalci v analiziranih primerih opravili verjetno najboljšo možno analizo, pri čemer pa le ne moremo izključiti njihovega osebnega, pristranskega vpliva in prepričanja o celotnem primeru.

Rezultati raziskave so pokazali da AVK kljub nekaterim pomanjkljivostim in omejitvam pripomore k boljšemu razumevanju dinamike spletnih zalezovanj s povezovanjem digitalnih dokazov z določenimi vedenjskimi vzorci spletnih zalezovalcev. To je zelo pomembno, saj so trenutno znanje in preiskovalne strategije spletnih zalezovalcev še vedno slabo razvite.

5.12 Zaključek

Raziskava je s pomočjo resničnih primerov prikazala aplikacijo AVK na primere spletnih zalezovanj. Z vključitvijo te metode je v članku ([2]) prikazana pomembnost AVK za vpogled preiskovalcev v način delovanja zločincev. Študija je prav tako raziskala psihološke in vedenjske dimenzije take vrste zločina. Preiskave teh zločinov so pogosto omejene s preiskovanjem izključno tehnologij, ki nastopajo v primeru, ne pa tudi preiskovanjem psiholoških in vedenjskih značilnosti žrtev in storilcev.

Iz same raziskave je razvidnih kar nekaj možnosti za izboljšavo in nadaljnje delo. Glavna izboljšava za še boljše razumevanje zločincev in spletnega zalezovanja kot vrste zločina je večje število arhiviranih primerov. Ker je AVK zločinsko orientirana, nudi možnost za prilagoditev bodočih preiskav z vključitvijo te metode v tretnutno obstoječe metodologije preiskovanja primerov spletnih zalezovanj.

6. VIRI IN LITERATURA

[1] N. Al Mutawa, J. Bryce, V. N. Franqueira, and A. Marrington. Behavioural evidence analysis applied to digital forensics: An empirical analysis of child pornography cases using p2p networks. In *Availability, Reliability and Security (ARES), 2015 10th International Conference on*, pages 293–302. IEEE, 2015.

[2] N. Al Mutawa, J. Bryce, V. N. Franqueira, and A. Marrington. Forensic investigation of cyberstalking cases using behavioural evidence analysis. *Digital Investigation*, 16:S96–S103, 2016.

[3] R. Bhagat, R. Modi, P. Patel, and H. Joshi. Privacy and security issues in social online networks. 2017.

[4] M. Duggan. *Online harassment*. Pew Research Center, 2014.

[5] R. M. Kowalski, S. P. Limber, S. Limber, and P. W. Agatston. *Cyberbullying: Bullying in the digital age*. John Wiley & Sons, 2012.

[6] L. McFarlane and P. Bocij. An exploration of predatory behaviour in cyberspace: Towards a typology of cyberstalkers. *First Monday*, 8(9), 2003.

[7] M. Rogers. The role of criminal profiling in the computer forensics process. *Computers & Security*, 22(4):292–298, 2003.

[8] A. Silde and O. Angelopoulou. A digital forensics profiling methodology for the cyberstalker. In *Intelligent Networking and Collaborative Systems*

(INCoS), 2014 International Conference on, pages 445–450. IEEE, 2014.

[9] P. R. Stephenson and R. D. Walter. Toward cyber crime assessment: Cyberstalking. In *Annual Symposium on Information Assurance (Asia), Albany, NY*, pages 7–8, 2011.

[10] B. Turvey. *Criminal profiling: An introduction to behavioral evidence analysis*. Academic Press, 2011.

[11] M. L. Ybarra and K. J. Mitchell. How risky are social networking sites? a comparison of places online where youth sexual solicitation and harassment occurs. *Pediatrics*, 121(2):e350–e357, 2008.

Forenzična analiza podatkov iz oblačnih storitev: študija primera Google Docs

Dejan Benedik
Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
Ljubljana, Slovenija
db9116@student.uni-lj.si

Nejc Kišek
Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
Ljubljana, Slovenija
nk4741@student.uni-lj.si

POVZETEK

V tej seminarski nalogi si bomo pogledali, kakšne so razlike med forenzično analizo oblačnih in lokalno pridobljenih podatkov, pokazali nekaj obstoječih orodij in postopkov za pridobivanje podatkov iz spletnih storitev za shranjevanje in urejanje dokumentov. Na koncu bomo predstavili še lastno implementacijo orodja za iskanje po zgodovini Google Docs dokumentov.

Članek [6] se osredotoča na priljubljeno Googlovo storitev Docs, vendar so predstavljeni koncepti in pristopi uporabni tudi za druge oblačne storitve. Poudarek je na pridobivanju in analizi podatkov, ki se nahajajo v samem oblaku (angl. *cloud-native artifacts*), v nasprotju z analizo sledi, ki jih spletne storitve pustijo na uporabnikovih napravah.

Ključne besede

SaaS, Google Docs, oblačna forenzika, zgodovina dokumenta

1. UVOD

V zadnjih letih vse več ljudi za svoje računske potrebe namesto lokalno nameščenih programov uporablja oblačne storitve, katerih glavna prednost je ta, da jih je mogoče uporabljati ne glede na to, kje se uporabnik nahaja in kakšne vrste napravo ima. Posebej na mobilnih napravah, ki imajo omejen pomnilnik in procesorsko moč, vedno pa imajo dostop do internetne povezave, je tak način delovanja zelo privlačen in zato tudi vse bolj pogost. Storitve, ki jih ljudje uporabljajo za upravljanje dokumentov in datotek, so v forenzični preiskavi lahko zelo pomembne, saj lahko vsebujejo mnogo dokazov.

Ker do podatkov nimamo fizičnega dostopa in ker so urejeni drugače kot datoteke na namiznih in prenosnih računalnikih, klasični pristopi in orodja, ki so v digitalni forenziki v uporabi že desetletja, pri preiskovanju oblačnih storitev pogosto niso dovolj. Z uporabo klasičnih orodij in osredotočanjem na sledi, ki jih storitve pustijo na uporabnikovi napravi, lahko preiskovalec spregleda veliko uporabnih informacij. Potrebno je razviti nova orodja, ki so prilagojena za delo z velikimi količinami oddaljenih podatkov.

Po drugi strani pa lahko forenziku način shranjevanja podatkov v oblačnih storitvah pogosto olajša delo, predvsem z vidika zgodovine in iskanja izbrisanih ali skritih podatkov. V nadaljevanju bomo pokazali, da je možno s pravimi orodji iz oblačnih storitev pridobiti praktično vse podatke, ki so se tam kadarkoli nahajali.

2. RAČUNALNIŠTVO V OBLAKU

Spletne ali oblačne storitve v glavnem delimo glede na delež logičnih nivojev sistema pod nadzorom ponudnika storitve. Ločimo tri velike skupine:

- Infrastruktura kot storitev (angl. *Infrastructure as a service, IaaS*), kjer ponudnik skrbi le za strojno opremo in navidezne stroje. Uporabnik si namesti operacijski sistem z vsemi potrebnimi aplikacijami, ni pa mu treba skrbeti za fizične strežnike. Primeri takih storitev so Amazon EC2, Google Compute Engine in Microsoft Azure.

  Preiskava tovrstne oblačne storitve se bistveno ne razlikuje od preiskave običajnega strežnika – ponudnik mora forenziku priskrbeti sliko uporabnikovih (navideznih) diskov, nad katerimi lahko izvaja preiskavo z znanimi postopki.

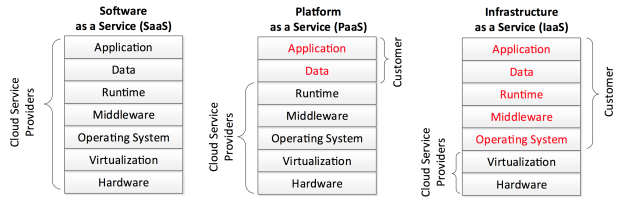
- Platforma kot storitev (angl. *platform as a service, PaaS*), kjer ponudnik priskrbi strojno opremo, operacijski sistem in okolje za izvajanje aplikacij, kamor uporabnik namesti svoj program. Kot primere lahko naštejemo Heroku, Google App Engine in Microsoft Azure.

  Za preiskavo tega tipa storitev potrebujemo sodelovanje ponudnika, ki nam priskrbi dnevniške zapise in kopijo uporabnikovih podatkov, te pa obdelujemo s klasičnimi forenzičnimi orodji.

- Programska oprema kot storitev (angl. *software as a service, SaaS*), kjer je uporabniku na voljo celotna rešitev (npr. spletna aplikacija), ponudnik pa skrbi za strojno opremo, sistem in programsko opremo, ki storitev poganja. Primeri takih storitev so Google Apps, Dropbox in Microsoft Office 365.

  V članku se bomo ukvarjali s to skupino, ker s stališča digitalne forenzike nudi nove izzive. Pri preiskovanju določene osebe moramo zaradi varovanja zasebnosti preprečiti zajem podatkov ostalih uporabnikov

storitve, količine podatkov v oblaku so lahko mnogo večje kot na osebnih računalnikih, podatki pa so lahko tudi razpršeni po različnih gručah strežnikov. Zaradi vseh teh razlogov klasičen zajem slik diskov in njihova analiza nista smiselna, pojavi se potreba po drugačnih, bolj specializiranih orodjih.



Slika 1: Delitev računalništva v oblaku

S stališča forenzike nam vse spletne storitve že same po sebi zagotavljajo nekaj prednosti zaradi delitve na strežnik, ki izvaja storitev in odjemalca, ki skrbi za interakcijo z uporabnikom.

Hranjenje podatkov in računske operacije se dogajajo na strani strežnika, uporabnik pa jih lahko spreminja le na vnaprej predvidene načine, kar pomeni, da so podatki zajeti s tam novejši, bolj zanesljivi in celoviti kot na zaseženi uporabnikovi napravi. Zato se namesto iskanja sledi, ki ostanejo na opazovani napravi, bolj izplača zajem podatkov neposredno iz oblaka.

Spletne storitve beležijo skoraj vse aktivnosti, ki se na njih izvajajo, zaradi česar je skrivanje, spreminjanje in brisanje sledi veliko težje. Če nam uspe dobiti dnevnik aktivnosti na strani strežnika, je zelo verjetno, da preiskovani uporabnik tam ne bo mogel ničesar prikriti, ali pa bo skrivanje podatkov mogoče hitro zaznati.

Zaradi zagotavljanja kvalitetne storitve in varnosti podatkov večina datotek v oblaku ne obstaja zgolj v eni različici, ampak jo spremljajo varnostne kopije in zgodovina verzij. Prav tako izbrisane datoteke nikoli v celoti ne izginejo, da se prepreči težave z neželeno izgubo uporabniških podatkov. To pomeni, da je iskanje zgodovinskih in izbrisanih podatkov veliko lažje kot pri klasičnih lokalnih digitalnih preiskavah.

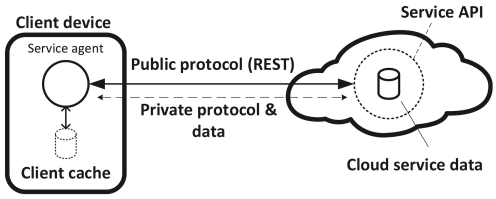
3. LOKACIJA PODATKOV

V tem poglavju si bomo pogledali, kakšna je razlika med podatki, ki se nahajajo na uporabnikovi napravi, in podatki, ki se nahajajo na zalednem delu oblačne storitve, in kako ta delitev vpliva na forenzično preiskavo.

Dokumenti in datoteke se morajo prenesti na uporabnikovo napravo, da jih lahko urejamo ali pregledujemo znotraj odjemalca, ki je običajno spletni brskalnik ali namenska uporabniška aplikacija. Tam se shranijo v predpomnilnik zaradi hitrejšega dostopa, razpoložljivosti ob izgubi povezave do strežnika ali pa služijo le kot lokalna kopija.

Ker se podatki deloma nahajajo na napravi, se v principu njihove analize lahko lotimo s klasičnimi forenzičnimi tehnikami in orodji, vendar ima tak pristop več pomanjkljivosti:

- Delna preslikava (angl. *partial replication*) – lokalni odjemalci pogosto ne hranijo vseh podatkov, ampak predpomnijo le tiste, ki jih potrebujejo zaradi pogostejšega dostopa ali za delo brez internetne povezave. Količina podatkov v oblaku je danes lahko zelo velika: storitev Google Drive na primer nudi zakup več 10 TB prostora, Amazon pa ponuja celo pakete brez omejitve količine shranjenih datotek. Taka količina podatkov je mnogokrat prevelika za uporabniške naprave in jih torej nikoli ne bomo mogli zajeti z odjemalca. Kadar gre za zelo velike dokumente, odjemalci lokalno ne shranijo niti celega dokumenta naenkrat, kar forenziku povzroči dodatne težave pri iskanju sledi.
- Revizije podatkov (angl. *artifact revisions*) – ponudniki storitve pogosto hranijo celotno zgodovino podatkov, vendar pa vsa ni na voljo odjemalskim aplikacijam. Včasih ti lahko dostopajo le do zadnjih nekaj revizij ali do nekaterih točk v zgodovini dokumenta glede na število in čas sprememb, celotna zgodovina pa je na voljo le na strežniku, kjer se storitev izvaja.
- Podatki na strani oblaka (angl. *cloud-native artifacts*) – strežnik odjemalcu pošlje samo tiste podatke, ki jih ta potrebuje za svoje delovanje. Pogosto so izpuščeni metapodatki in druge informacije, potrebne za izvajanje zalednega dela aplikacije. V nekaterih primerih se celo vse računanje dogaja na strežniku, odjemalec pa nato dobi le podatke za prikaz (npr. bitno sliko ali html, ki se pokaže v brskalniku). Zaradi tega odjemalec pogosto sploh nima dostopa do vseh podrobnosti, ki bi jih forenzik potreboval, in jih torej iz njega ni mogoče izluščiti.



Slika 2: Shema strežnik-odjemalec

Pri komunikaciji med strežnikom, kjer teče spletna storitev, in odjemalcem, ki teče na uporabnikovi napravi, imamo običajno dve vrsti vmesnikov: javni, ki je dokumentiran in na voljo razvijalcem aplikacij za uporabo v spletnih aplikacijah, in interni, ki je namenjen le ponudniku storitve in njegovim uradnim (ponavadi) zaprtokodnim aplikacijam.

Nedokumentirani vmesniki nam pogosto dajo informacije potrebne za izvajanje in nadzor nad storitvijo ali reševanje težav. Te niso nujne pri delovanju odjemalca in ne zanimajo uporabnika, zelo pa so lahko koristne pri forenzični preiskavi. Primer takih podatkov so podrobna zgodovina dokumentov, dnevnik dostopov, izbrisane datoteke ali interni metapodatki. Formate sporočil za komunikacijo preko zasebnih vmesnikov je potrebno ugotoviti z obratnim inženiringom in eksperimenti, pogosto pa se tudi spreminjajo, zaradi česar je orodja težko izdelati, njihovo delovanje pa ni vedno zanesljivo.

4. ODKRIVANJE PODATKOV

Kot smo že omenili, je sledi o uporabi spletnih storitev mogoče pridobiti na dva načina: iz lokalnih podatkov na odjemalcu ali neposredno iz oblaka.

Iskanje lokalnih informacij na osebnih računalnikih in mobilnih napravah, ki se povezujejo na spletne storitve, zahteva klasične pristope, kot so pregledovanje dnevniških datotek, podatkovnih baz brskalnikov in odjemalskih aplikacij, pa tudi analiza delovnega pomnilnika. Potrebno je zbrati čim več različnih podatkov z več različnih naprav in iz njih nato sestaviti čim boljšo sliko podatkov, ki se nahajajo v oblaku. Taki postopki so danes že precej pogosti, vendar je očitno, da niso idealen pristop k rešitvi problema.

Iskanje podatkov v oblaku zahteva uporabo novih namenskih orodij, ki podatke iz oblaka pridobijo preko programskih vmesnikov. Na ta način je možno pridobiti celotne podatke, pogosto pa tudi zgodovino, izbrisane datoteke, metapodatke in druge uporabne informacije.

V Google Docs (podobno tudi v ostalih sorodnih storitvah) dokumenti niso predstavljeni kot celovita datoteka, ampak kot zaporedje sprememb od stvaritve dokumenta naprej. Tak način predstavitve datotek je uporaben za lažje razveljavljanje sprememb, olajša pa tudi sočasno urejanje istega dokumenta z več odjemalci. Za forenzika, ki izvaja preiskavo tovrstnih dokumentov, pa ima taka predstavitev še precej drugih prednosti.

Ena od prednosti je ta, da je podatke pri običajni uporabi storitve nemogoče izbrisati, ker je vsako brisanje le nov vnos v seznam sprememb. To pomeni da je vsaka uporabnikova akcija zabeležena in umeščena v čas – vključno s tipkarskimi napakami, ki jih je uporabnik takoj popravil, stavki ki jih je natipkal le na pol in nato odstranil, itd. Forenzik lahko zato iz dokumenta izlušči informacije, za katere uporabnik misli, da jih je že davno izbrisal ali pa je sploh pozabil, da so kdaj obstajale. Še več, za vsako od teh informacij lahko izve do sekunde natanko, kdaj je bila vnesena v sistem.

Poleg samega besedila dokumenta so v zgodovini shranjene tudi vse vstavljene datoteke, na primer slike. Ponudniki spletnih storitev imajo običajno na voljo zelo veliko prostora za shranjevanje podatkov, kar pogosto pomeni, da lahko shranijo vse datoteke, ki potujejo preko njihovih strežnikov. Pri storitvi Google Docs so slike, ki jih shranimo v dokument, na voljo tudi potem, ko jih iz dokumenta izbrišemo, in ostanejo na voljo, dokler v zgodovini revizij dokumenta obstaja referenca nanje – dokončno izginejo šele približno eno uro po izbrisu celotnega dokumenta. Datoteke ostanejo na voljo, da se zagotovi možnost razveljavitve izbrisa, vendar pa to iz uporabniškega vmesnika pogosto ni očitno. Uporabniki spletnih storitev se zato obstoja teh datotek običajno ne zavedajo, računalniški forenziki pa s pravimi orodji tako lahko dostopajo do starih, pozabljenih ali izbrisanih delov dokumenta.

Z vidika izvajanja digitalne preiskave nam dejstvo, da je celotna zgodovina zabeležena, koristi še na drug pomemben način. Vsaka namerna ali nenamerna sprememba, ki se zgodi med pregledovanju takšnega dokaznega gradiva, bo namreč zabeležena in zgolj dodana na konec seznama sprememb. Previdnost pri pregledovanju takega dokumenta zato ni tako kritična kot pri obdelavi lokalnih datotek, kjer forenzikova napaka lahko pomeni izgubo podatkov ali pripelje do negotovosti o tem, ali so bili dokazi spremenjeni. Pri pregledovanju je treba preprečiti le izbris celotnega dokumenta, klasični ukrepi za zagotavljanje nespremenjenosti, kot so kontrolne vsote, pa so potrebni le, ko podatke pretočimo s sistema in jih shranimo v drugačno obliko.

Poudariti moramo še, da so podatki oblačnih storitev shranjeni na oddaljenih strežnikih, zato nimamo zagotovila, da se bodo tam v enaki obliki nahajali še dolgo časa – kot nasprotje si lahko predstavljamo zasežen trdi disk, ki v zavarovanem skladišču lahko ostane nespremenjen več let. Tudi programski vmesniki do spletnih storitev se pogosto spreminjajo, orodja za dostop do njih pa običajno temeljijo na obratnem inženiringu in jih je zato treba sproti posodabljati in prilagajati. Zaradi teh razlogov je pomembno, da podatke iz oblačnih storitev zajamemo in obdelamo čim prej ter da ustvarimo lokalne varnostne kopije.

Zajem podatkov preko zasebnega vmesnika s pomočjo obratnega inženiringa reši tudi problem zasebnosti ostalih uporabnikov storitve. Za izvedbo digitalne preiskave je dovolj, da ponudnik storitve v skladu s sodnim nalogom digitalnemu forenziku podeli ključe za začasen dostop do podatkov preiskovane osebe. Preiskovalec nato s primernimi orodji prenese podatke za nadaljnjo obdelavo. V tem delu postopka se tako povsem izogne datotekam ostalih uporabnikov storitve.

5. STRUKTURA GOOGLE DOCS

V sledečem poglavju bomo podrobneje preučili strukturo notranjih datotek storitve, ki je v slovenščino prevedena kot Google Dokumenti. Vsaj od začetka se je ta program od konkurenčnih razlikoval v načinu prikaza dokumenta (medtem, ko so ostali urejevalniki prikazovali HTML, se je Google Dokument obnašal podobno kot standardni namizni urejevalniki besedil) in v načinu shranjevanja dokumentov. Sistem si ne zapomni le posamezne revizije (angl. *snapshot*), ampak hrani kar celotno zgodovino sprememb dokumenta, revizije pa interno predstavi kot skupine sprememb. Če si zaželimo določeno revizijo, jo dobimo z izvedbo vseh sprememb od stvaritve dokumenta do vključno te revizije.

Vsakič, ko uporabnik odpre dokument, se ustvari nova seja, znotraj katere se beležijo posamezne spremembe. Razmik med beleženji je zelo natančen (do 150 ms), kar strežniku omogoča, da v živo združuje vnose več uporabnikov, ki lahko sproti vidijo akcije drug drugega. Glede na število sprememb in pretečeni čas med posameznimi urejanji se seje grupirajo v tako imenovane glavne revizije (angl. *major revisions*). Preko uporabniškega vmesnika lahko dostopamo le do teh glavnih revizij in dokument tako povrnemo v starejše stanje. Posledica takega sistema je, da podatki v dokumentu nikoli niso zares izbrisani, kar izkoristimo v nadaljevanju.

Posamezen zapis spremembe v dokumentu je sestavljen iz časa zapisanega v standardnem formatu POSIX, identifikatorja številke Googlovega računa uporabnika, ki je spremembo izvedel, identifikatorjev revizije, seje in spremembe ter podatkov o spremembi sami, ki so zapisani v formatu JSON. Seznam sprememb (angl. *changelog*) vsebuje mnogo takih zapisov urejenih po času, iz katerih se da nato zgraditi

končno stanje dokumenta.

[{"ibi": 22, "s": "T", "ty": "is"}, 1493719496680, "00935037001990099604", 9, "6f1e10cf30e01592", 7, null]
[{"ibi": 23, "s": "al", "ty": "is"}, 1493719497034, "00935037001990099604", 10, "6f1e10cf30e01592", 8, null]
[{"ibi": 25, "s": "e s", "ty": "is"}, 1493719497561, "00935037001990099604", 11, "6f1e10cf30e01592", 9, null]
[{"ibi": 28, "s": "t", "ty": "is"}, 1493719497966, "00935037001990099604", 12, "6f1e10cf30e01592", 10, null]
[{"ibi": 29, "s": "av", "ty": "is"}, 1493719498460, "00935037001990099604", 13, "6f1e10cf30e01592", 11, null]
[{"ibi": 31, "s": "ek ", "ty": "is"}, 1493719498952, "00935037001990099604", 14, "6f1e10cf30e01592", 12, null]
[{"ibi": 34, "s": "bom", "ty": "is"}, 1493719499487, "00935037001990099604", 15, "6f1e10cf30e01592", 13, null]
[{"ibi": 37, "s": " iz", "ty": "is"}, 1493719500026, "00935037001990099604", 16, "6f1e10cf30e01592", 14, null]
[{"ibi": 40, "s": "bris", "ty": "is"}, 1493719500628, "00935037001990099604", 17, "6f1e10cf30e01592", 15, null]
[{"ibi": 44, "s": "al", "ty": "is"}, 1493719502140, "00935037001990099604", 18, "6f1e10cf30e01592", 16, null]

Slika 3: Stavek iz 10 zaporednih zapisov

Najpogostejši zapisi so vnosi besedila (*insertion*), ki vsebujejo položaj v dokumentu in vneseni niz, in izbrisi (*deletion*), ki vsebujejo začetek in konec izbrisanega dela dokumenta. Poleg njih obstajajo še druge akcije, kot je vstavljanje slike ali tabele oziroma dodajanje novega odstavka. Več povezanih zaporednih akcij je lahko združenih v skupni zapis (*multiset*), znotraj katerega so gnezdene spremembe, ki se ne morejo izvesti posamič.

Za seznamom sprememb imamo v strukturi še informacije o stilih besedila in naslovov, jeziku ter seznam odstavkov in njihovih pripadajočih stilov. Poleg tega so na tem mestu še seznam komentarjev (angl. *comments*) in predlogov (angl. *suggestions*), ki jih uporabniki lahko dodajajo, ter seznam povezav na dele dokumenta, ki se uporabljajo v kazalu.

Namesto da vsakič, ko odpremo dokument, izvedemo vse spremembe od začetka, nam strežnik lahko vrne del dokumenta v enem kosu, imenovanem *chunked snapshot*, ki vsebuje celotno besedilo dokumenta do nekega trenutka v času, novejše spremembe pa prikaže kot običajen seznam sprememb. Tako lahko odjemalec preskoči izvajanje sprememb pred določeno revizijo in tako pohitri nalaganje aplikacije. To pride do izraza posebej pri dokumentih z dolgo zgodovino.

6. OBSTOJEČA ORODJA

V tem poglavju si bomo pogledali nekaj orodij za pregledovanje in zajem dokumentov in ostalih datotek, shranjenih v priljubljenih oblačnih storitvah. Vsa omenjena orodja potrebujejo veljavno uporabniško ime in geslo preiskovanega uporabnika, s katerimi se nato lahko povežejo na programski vmesnik spletne storitve in preko njega pridobijo želene podatke.

6.1 Kumodd [3]

Kumodd je orodje, razvito za potrebe zajema podatkov iz oblačnih storitev za shranjevanje datotek. Glavna težava, ki jo rešuje, je količina podatkov, ki jo take storitve lahko vsebujejo. Pri zares velikih količinah podatkov je ročno pregledovanje preko uradnega uporabniškega vmesnika preveč zamudno, pretakanje celotne vsebine in filtriranje na lokalnem sistemu pa pogosto tudi ni prostorsko izvedljivo.

Kumodd nam tako omogoča izpis seznama vseh najdenih datotek s podporo filtriranje glede na tip datoteke. S tega seznama lahko nato izberemo, katere datoteke želimo prenesti, pridobimo pa lahko tudi vse njihove starejše različice.

Podprte so štiri priljubljene oblačne storitve: Google Drive, Dropbox, OneDrive in Box, modularna zasnova programa pa omogoča tudi razširitve za podporo drugih ponudnikov.

Ena izmed slabosti orodja Kumodd je, da ne more pridobiti zgodovinskih podatkov o dokumentih ustvarjenih s storitvijo Google Docs – vrne nam le pdf datoteko trenutnega stanja dokumenta.

6.2 Kumodocs [4]

Kumodocs je orodje, razvito za pridobivanje zgodovine Google Docs dokumentov in kot dopolnitev orodja Kumodd. Z uporabo nedokumentiranih vmesnikov za storitve Google Drive in Docs pridobi celotno zgodovino dokumenta in jo izpiše v obliko, ki je primernejša za nadaljnje procesiranje. Poleg tega pridobi tudi vse datoteke, ki so bile kdaj vključene v dokument (npr. slike).

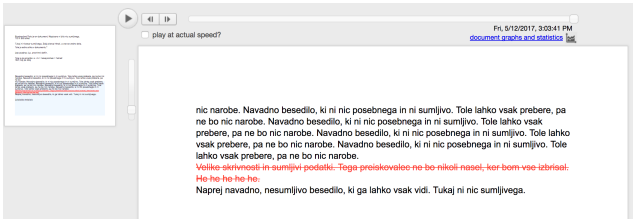
Orodje samo po sebi ni primerno za neposredno uporabo v forenzični preiskavi, saj je njegov izhod namenjen nadaljnji obdelavi z drugim programom – v nadaljevanju si bomo pogledali primer takega programa.

Poleg besedilnih dokumentov (storitev Docs) nam omogoča tudi analizo predstavitev (Google Slides) in slik (Google Drawings). Urejevalnik preglednic (Google Sheeets) uporablja povsem drugačen programski vmesnik kot preostali trije urejevalniki, zato ta v orodju ni podprt.

6.3 Draftback [7]

Draftback je orodje za pregled zgodovine Google Docs dokumentov, prvotno namenjeno za pisatelje, ki si želijo izboljšati svoj stil pisanja s tem, ko opazujejo svoje tipkanje. Orodje deluje kot razširitev brskalnika Chrome, ki v uporabniški vmesnik Google Docs urejevalnika doda možnost pregleda zgodovine. Zgodovino lahko predvajamo kot animirano pisanje besedila ali pa z drsnikom skočimo na katerokoli točko v zgodovini dokumenta. Prav tako lahko vidimo seznam dostopanj do dokumenta in pregled uporabnikov, ki so ga spreminjali.

Orodje je zelo enostavno za namestitev in uporabo in ni primerno le za strokovnjake. Edina njegova slabost je, da ne omogoča obnovitve izbrisanih slik in drugih vstavljenih datotek, označi pa mesto, kamor je bila datoteka vstavljena.



Slika 4: Orodje Draftback

6.4 Kumofs

Kumofs je nadgradnja funkcionalnosti, ki jih ponuja Kumodd. Glavni cilj je pridobivanje metapodatkov, ki so povezani z datotekami shranjenimi v oblaku: vsaka Google Drive datoteka ima na primer lahko več kot sto atributov, kot so GPS koordinate, časi dostopov in sprememb, uporabniški računi oseb z dostopom do datoteke in drugo. Taki metapodatki so s stališča digitalne preiskave zelo uporabni, saj forenziku že v fazah pregleda in identifikacije omogočijo

izločitev ne-relevantnih datotek, kar pohitri in poceni sledeč zajem ter shranjevanje dokaznega gradiva.

Kumofs je implementiran kot bralni (angl. *read-only*) datotečni sistem z uporabo platforme FUSE. Omogoča priklop oddaljene oblačne storitve za shranjevanje podatkov na lokalni sistem. Tako lahko pregledujemo celoten nabor metapodatkov in zgodovino datotek ter prenašamo izbrane datoteke na lokalni računalnik.

Poleg tega, da nam Kumofs ponuja veliko informacij, ima tudi precej funkcionalnosti, ki ga naredijo zelo enostavnega za uporabo

- omogoča prenos datotek iz oblaka kar z obstoječimi sistemskimi orodji kot sta `cp` ali `cat`, tako kot tudi s svojimi vgrajenimi ukazi
- omogoča ustvarjanje navideznih datotek in imenikov, kjer je vsaka zgodovinska revizija kopija datoteke z dodano zaporedno številko
- omogoča pregledovanje Google Docs dokumentov v formatih pdf, txt, odt ali docx
- omogoča pregled izbrisanih datotek na enem mestu ali pa na njihovi prvotni lokaciji
- funkcionalnost imenovana *Time travel* omogoča pregled celotnega sistema v izbranem trenutku v preteklosti.

7. IZDELANO ORODJE

Da bi bilo orodje *kumodocs* uporabno za potrebe forenzične preiskave, ga je potrebno narediti bolj preglednega, saj nam samo po sebi poda le grobo obdelano zgodovino dokumenta, ki je razdrobljena v mnogo posameznih zapisov. Izhod v taki obliki ni primeren za ročno pregledovanje, se ga pa da enostavno procesirati z drugimi programi.

Implementirali smo orodje za iskanje ključnih besed v zgodovini dokumenta. *Zgodogrep* [2] najde vse revizije dokumenta, kjer se je pojavil iskani niz, in jih izpiše. Služi kot primer dodatnega programa, ki naredi orodje Kumodocs bolj uporabno. Tako kot Kumodocs je tudi naše orodje napisano v programskem jeziku python.

Programu moramo podati 2 argumenta: tekstovno datoteko s celotno zgodovino dokumenta (datoteka `revision-log.txt` v izhodu iz Kumodocs) in iskani niz v obliki regularnega izraza. Orodje nato po vrsti bere zapise iz zgodovine in iz njih gradi dokument, v katerem sproti išče želeni niz.

Za vsako najdeno ujemanje na standardni izhod izpiše celotno vsebino in čas, ko je bil dokument v takem stanju. Na ta način lahko dobimo vse zgodovinske verzije preiskovanega dokumenta, ki vsebujejo naš iskani niz. Čeprav je program zelo preprost, lepo demonstrira uporabnost orodja Kumodocs in nam da občutek, kako lahko je priti do podatkov pri dokumentih v oblaku.

Čeprav je orodje Kumodocs precej redno posodobljeno (zadnje spremembe so bile aprila), smo naleteli na nekaj težav s spremenjenimi parametri Googlovega sistema za overitev. To potrjuje našo že prej omenjeno trditev, da je potrebno orodja za analizo oblačnih storitev sproti prilagajati spremembam in so pogosto lahko nestabilna. Druga težava, na katero smo naleteli pri razvoju, je bila podpora posebnih znakov na nekaterih mestih v dokumentu (npr. šumniki v imenu avtorjev), ki v orodju Kumodocs ni bila predvidena.

8. ZAKLJUČKI

Skupaj s priljubljenostjo oblačnih storitev se povečuje tudi delež digitalnih naprav, ki jih s klasičnimi digitalnimi forenzičnimi pristopi ni več možno zadovoljivo pregledati, zato je nujen razvoj novih postopkov in orodij.

V članku, ki je služil kot osnova te seminarske naloge, so avtorji odkrili ključne komponente zasebnih programskih vmesnikov spletnih storitev pod okriljem *Google Documents Suite*. Na podlagi ugotovitev so v tem in v kasnejših člankih predstavili orodja, ki bi lahko bila uporabna za izvajanje forenzične preiskave nad dokumenti v priljubljenih oblačnih storitvah. Eno od teh orodij smo uporabili v lastni implementaciji programa za iskanje ključnih besed po celotni zgodovini dokumenta.

Poleg tega lahko predstavimo še nekaj ugotovitev. Zajem podatkov neposredno iz oblaka je pogosto bolj uspešen kot iskanje sledi na uporabnikovi napravi. Pogosto lahko dobimo celotno zgodovino podatkov, tudi izbrisane datoteke. Če uporabnika skrbi za svojo zasebnost, ne sme niti razmišljati o uporabi oblačnih storitev, ki niso pod njegovim nadzorom. Za preiskovanje podatkov v oblačnih storitvah potrebujemo nova orodja. Če so ta narejena na podlagi obratnega inženiringa, obstaja velika nevarnost, da ponudnik storitve spremeni svoje programske vmesnike in jih tako onemogoči.

9. LITERATURA

[1] H. Chung, J. Park, S. Lee, and C. Kang. Digital forensic investigation of cloud storage services. *Digital investigation*, 9(2):81–95, 2012.

[2] N. Kišek and D. Benedik. Zgodogrep, 2017. `https://github.com/thenejcar/zgodogrep`, accessed 2017-05-29.

[3] S. McCulley. Kumodd, 2017. `https://github.com/andresebr/kumodd`, accessed 2017-05-29.

[4] S. McCulley. Kumodocs, 2017. `https://github.com/kumofx/kumodocs`, accessed 2017-05-29.

[5] V. Roussev, I. Ahmed, A. Barreto, S. McCulley, and V. Shanmughan. Cloud forensics–tool development studies & future outlook. *Digital Investigation*, 18:79–95, 2016.

[6] V. Roussev and S. McCulley. Forensic analysis of cloud-native artifacts. *Digital Investigation*, 16:S104–S113, 2016.

[7] J. Sommers. Draftback, 2013. `http://draftback.com/`, accessed 2017-05-10.

Uporaba analiz sej internetne zgodovine za lažje izvajanje forenzičnih preiskav večuporabniških računalniških okolij

Aljaž Srša
Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
Večna pot 113, 1000 Ljubljana
aljaz.srsa@gmail.com

Blaž Štempelj
Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
Večna pot 113, 1000 Ljubljana
blaz.stempelj@gmail.com

Gregor Sušnik
Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
Večna pot 113, 1000 Ljubljana
susnikg@gmail.com

POVZETEK

Raziskava predstavi nov pristop za identifikacijo upor-abnika, ki je uporabljal računalnik v času zločina. Pristop se izvaja z agregacijo internetne zgodovine obnovljene naprave v seje. S primerjavo pridobljenih sej se lahko določi, ali je posamezna seja enkraten ali ponovljiv dogodek (npr. uporabnikova navada). Pri tem se osredotoči na dva pristopa za agregacijo sej. Seje nespremenljive (fiksne) dolžine in seje spremenljive dolžine. Predstavi tudi pristop za odkrivanje ponovljivih vzorcev, ekstrakcijo teh vzorcev in njihovo predstavitev v obliki binarnih nizov. Za primerjavo sej se uporabi Jaccardov podobnostni koeficient, s katerim lahko določimo mero podobnosti med sejami in z visoko verjetnostjo identificiramo uporabnika neke seje. Eksperimenti so bili izvedeni na dveh testnih množicah, kjer je več uporabnikov imelo dostop do istega računalnika.

Ključne besede

Digitalna forenzika, svetovni splet, analiza sej, analiza konteksta, vzorci življenja, analiza internetne zgodovine

1. UVOD

Raziskovalec mora že med samo preiskavo najti sledi, ki kažejo na uporabo naprave med samim zločinom. Pri tem so najbolj zanimive tarče besedilni dokumenti in medijske datoteke (slike, video in avdio posnetki). Ker računalniki, tablice in mobilni telefoni ponujajo tudi internetno povezljivost, je uporabnikov način uporabe interneta prav tako poln zanimiv sledi. Vse kar uporabnik počne na internetu je zabeleženo v zgodovini brskalnika, urejenem seznamu že obiskanih spletnih strani, ki vsebuje datum in čas obiska ter unikaten naslov obiskane spletne strani. Samo na podlagi teh informacij lahko izluščimo čas dneva, ko je uporabnik najbolj aktiven in tipe spletnih strani, ki jih redno obiskuje. Poleg tega si lahko ogledamo tudi uporabnikove prenesene datoteke in pojme vpisane v iskalnik. Tako izvemo tudi jezike, ki jih govori uporabnik. Zaenkrat je analiza internetne zgodovine še vedno časovno potratna in izvedena ročno. Dodaten izziv prinaša opcija zasebnega brskanja. Ta uporabniku zagotavlja skoraj popolno anonimnost. Vendar internetna zgodovina vsebuje preveč pomembnih informacij, da bi jo enostavno ignorirali. Uporabimo jo lahko kot namige za nadaljnjo preiskavo, ali pa kot lastno dokazno gradivo na sodišču.

Internetna zgodovina predvsem dokazuje, da je napravo uporabljal človek in ne program. Točno s tem problemom so se soočali v primeru Schonfield v Veliki Britaniji leta 2004 [3]. Tožilstvo je bilo primorano umakniti tožbo za posedovanje nezakonitih slik, ker so na računalniku obtoženca našli trojanskega konja in niso omeli dokaza, da je slike prenesel uporabnik in ne trojanski konj. Napaka je bila obravnava slik v izoliranem okolju. Brez konteksta je pravi namen uporabnika zelo težko prikazati. Če bi vnaprej pridobili seznam pojmov vpisanih s strani obtoženca, ki bi namigovali njihovo uporabo za iskanje nezakonitih slik, bi lahko z dodatno analizo mogoče odkrili resnico.

1.1 Enkratni pojavi in ponovljivi vzorci

Predlagamo, da se aktivnosti na napravah klasificira v dva različna tipa pojavov. Enkraten pojav je definiran kot pojav, ki se zgodi največ enkrat (npr. izvedba zlonamernega programa)a, ali kot sekvenca kratkih pojavov, ki se nikoli več ne ponovijo (npr. enkratna uporaba iskalnega niza *"kako narediti bombo"*). Drugi tip so ponovljivi vzorci, ki kažejo na obstoj navad. Da je nek vzorec ponovljiv, se mora zgoditi vsaj še enkrat. Ponovljive vzorce dodatno razdelimo na zaporedne vzorce in časovne gruče. Kadar pojavi A, B in C nastopajo v tem vrstnem redu v nekem časovnem okviru, govorimo o zaporednih vzorcih. Kadar pa pojavi A, B in C nastopajo v kakršnem koli vrstnem redu ali kombinaciji v nekem časovnem okviru, govorimo o časovnih gručah (npr. ACBAB).

Določeni preiskovalni cilji zahtevajo identifikacijo in analizo ponovljivih ali stalnih vzorcev (npr. dostop do gradiva z neprimerno vsebino). Kadar se pojavi dvom kdo je bil upravitelj naprave v nekem časovnem obdobju, lahko tudi na podlagi enkratnega pojava (npr. poslane e-pošte z neprimerno vsebino), ki se znajde v neposredni bližini ponovljivih vzorcev, z veliko verjetnostjo pokaže, da imamo opravka z isto osebo. S tem eliminiramo izjave *"nisem bil jaz"*. Raziskava obravnava sorodna dela za analizo internetne zgodovine in opiše lasten pristop združevanja internetne zgodovine v seje,

katere se primerja in poišče njihova sovpadanja. Eksperimentalni del raziskave opisuje dva različna problema. Iskanje najbolj učinkovite časovne vrednosti za združevanje skupin v seje in učinkovito primerjavo sej. Na koncu je omenjena tudi tehnika kako izboljšati opisani pristop.

2. SORODNA DELA

Eden prvih poskusov izdelave orodja za forenzično analizo je bil Zeitline. Izdelala sta ga Buchholz in Falk [4] leta 2005. Preiskovalcu omogoča, da izdela kompleksne dogodke z uporabo iskanja in filtriranja za poselitev in analizo časovnic. Ker različne aplikacije in operacijski sistemi za seboj puščajo različne sledi sta se Khan in Wakeman [18] odločila za pristop, kjer sta ugotavljala kakšne odtise med normalno uporabo na sistemu puščajo aplikacije. Te značilnosti sta potem uporabila za učenje nevronske mreže, ki bi preiskovalcu med samo preiskavo pomagala pri rekonstrukciji časovnice uporabe aplikacij.

Leta 2009 sta Olsson in Boldt [24] izdelala orodje z imenom Cyber Forensic TimeLab (CFTL). CFTL zna razčleniti disk z uporabo predčasno definiranih artefaktov in sestaviti časovnico v obliki histograma. Ker ne zna avtomatično analizirati posameznih artefaktov, potrebuje pomoč analitika, da označi medsebojne odvisnosti dogodkov uporabe aplikacij.

Leto kasneje je Gudjonsson [9] predstavil orodje log2timeline, ki kreira super časovnico tako, da vse pridobljene informacije prikaže kot dolg seznam. Le-te se nato lahko uporabi za nadaljnjo obdelavo. Carbone in Bean [5] sta orodje log2timeline podprla v njihovem pregledu orodij za izdelavo časovnic in omenila, da po njihovem mnenju vključi preveč nepomembnih datotek. Hargreaves in Patterson [12] sta razvila orodje za rekonstrukcijo visoko nivojskih dogodkov iz nizko nivojskih dejavnosti z uporabo ujemajočih se bližnjih časovnih vzorcev.

James in Gladyshev [16] sta se lotila proučevanja vzroka in učinka rekonstrukcije dogodkov tako, da sta definirala model prehodnih stanj. Če znamo identificirati sledi, ki jih pustijo akcije, lahko sklepamo, da je akcija posledica prehoda določenih stanj na računalniških sistemih. Leta 2014 sta Inglot in Liu [13]] vzela Zeitline in mu dodala nove funkcije. Chabot et al. (2014) [6] so uporabili upravljanje znanja, semantični splet in podatkovno rudarjenje za izgradnjo teoretičnega modela z imenom SADFC (Semantic Analysis of Digital Forensic Cases). Ko bo orodje enkrat implementirano, naj bi bila možna analiza dogodkov in izdelava grafične časovnice. Khatik in Choudhary [19] sta razvila orodje za vizualizacijo časovnic. Uporablja beležke spletnih strežnikov in iskalne nize za rekonstrukcijo časovnice in izdelavo poročila, ki ga lahko uporabimo na sodišču.

Omejitev zgoraj predstavljenih orodij je, da lahko identificirajo samo znane vzorce dogodkov. Interpretacija in identifikacija neznanih dogodkov lahko vidimo v Marringtonovem [22] računalniškem profiliranju in statističnem gručenju datotečnih sistemov, ki so ga predlagali Kälber et al. [17]. Njihov pristop ne potrebuje nikakršnega predhodnega znanja o sistemu. Poskuša namreč identificirati aplikacije in datoteke povezane v času. Gresty et al. [8] so uporabili PCA (Principal Componet Analysis) za poenostavitev časovnic internetne zgodovine.

Zelo zanimive raziskave se izvajajo izven tradicionalne digitalne forenzike na področju rekonstrukcije dogodkov, managementa in prikaza. Kiernan in Terzi [20] se zavzemata, da mora biti prikaz pregleda velikih zaporedij dogodkov zmanjšan in poenostavljen. Hkrati pa moramo imeti tudi širši pogled nad vsemi dejavnostmi, da lahko še vedno zaznamo sumljive. Primera teh problemov sta upravljanje z viri in optimizacija baze podatkov. Avtorja predlagata tehnike za analizo in povzetje velikih revizijskih dnevnikov, ki so nato predstavljeni preiskovalcu. Eagle in Pentland [7] pravita, da ima vsaka oseba lastne navade in vzorce obnašanja, ki jih lahko postavimo v časovni, prostorski ali celo družbeni kontekst za enostavno identifikacijo. Avtorja tem obnašanjem pravita lastna obnašanja (eigen-behaviours). Ye et al. [26] predlagajo notacijo imenovano normalna oblika življenjskega vzorca (life pattern normal form) in ogrodje življenjskega vzorca za določitev in rudarjenje lokacijskih podatkov posameznika in navad uporabe mobilnih naprav. Ti vzorci naj bi kazali na pomembne kraje v življenju posameznika. Pridobljeni morajo biti iz neobdelanih GPS podatkov z uporabo detekcije točk bivanja (stay point detection) in gručenja. Schaefer et al. [25] opisujejo zaporedja dogodkov in naredijo pomembne razlike med metodo sekven-čenja in metodo združevanja. Predstavijo različne načine vizualizacije gruč dogodkov in praznin. Pri tem uporabijo upodobitve, ki niso časovnice, ampak le informacije o dogodkih. Pristop Al Awawdeh et al. [2] (2013) [citat] je realno časovni agent za sprotno shranjevanje podatkov. Hkrati omenijo problem gostobesednosti, kjer so nepomembni podatki prekomerno poročani, zaradi česar se izgubijo pomembni.

Hamid et al. [10] opišejo dogodke kot interakcijo med živimi in neživimi objekti. Izpostavijo, da je območje odkritja aktivnosti ponovljivih vzorcev v zaporedjih podatkov. Le-ti pokažejo razrede aktivnosti, enako kot lahko npr. senzorji v domovih pokažejo, da se nekdo giblje iz kuhinje v dnevno sobo. Minnen et al. [23] opišejo motive kot pod-zaporedja z visoko podobnostjo znotraj daljšega zaporedju podatkov. Pri tem izpostavijo problem odkritja motivov, ker vnaprej ne poznamo njihove dolžine, oblike in velikosti.

Tradicionalno so se raziskave analize časovnic in digitalne forenzike osredotočale na identifikacijo znanih zaporedij dogodkov ali iskanju novih. Če gledamo na zaporedja dogodkov le v izolaciji, lahko spregledamo gruče vedenja, ki nosijo pomembne podatke. To je še kako res za uporabnikovo internetno zgodovino, ki pogosto izraža uporabnikove navade, ne pa zaporedij motivov. V nadaljnjih poglavjih bo predstaviljena metoda za združevanje celotnih zaporednih časovnih artefaktov v seje, ki bood nato primerjani med seboj. Prikazana je tudi uporabnost tega pristopa v situacijah, kjer si več uporabnikov deli isti računalnik. Rezultati so nato primerjani pri različnih konfiguracijah.

3. PRISTOP

3.1 Časovno združevanje sej

Schaefer et al. (2011) [25] je predstavil ***metoda sekvenčenja*** in ***metoda združevanja*** pri postopku analize podatkov. Pri metodi začasnega sekvenčenja so vzorci urejeni v dokaj dolgo zaporedje podatkov. Metoda združevanja pa (angl. Aggregate-against-Aggregate) predstavlja dve različni zbirki združenih primerkov za medsebojno primerjavo.

Za analizo časovnega okvirja zgodovine brskanja ali kakšno koli drugo analizo meta podatkov v okviru digitalno forenzične preiskave je predlagan pristop, kjer je seja časovnega združevanja primerjana s preostalimi sejami, z namenom, da se opredeli v kolikšni meri se preostale seje ujemajo po članih in ostalih komponentah. Ko so vse seje primerjane in ekvivalentne seje združene v skupine, se lahko prične proces analize zaporedij imenovan *intra-session*. Pove nam, ali se pojavi določen vzorec komponente. Samo združevanje sej zagotavlja pomembno kontekstualno analizo na makro nivoju o uporabi naprave, časovno sekvenčnege analize in znatno zmanjšanje količine podatkov potrebnih za procesiranje.

Ključen je sam izbor seje časovnega združevanja. Poznamo dva pristopa izbire takšnih sej:

- Seja s fiksno dolžino, kjer so določena časovna obdobja definirana v naprej (npr. predmet obravnave ima določeno obobje 30 sec, 60 sec ali 60 min).
- Seja z neprekinjeno dejavnostjo spremenljive dolžine. Predmeta obravnave spadata v isto sejo, če sta si dovolj blizu po nekem časovnem razmaku. V nasprotnem primeru spadata vsak v svojo sejo.

Avtorji pričakujejo, da bo pristop seje spremenljive dolžine ustvaril manjše število sej v primerjavi z pristopom fiksne dolžine (trditev je bila potrjena med izvajanjem poizkusov, slika 4 in 5). To velja predvsem za seje z daljšo časovno aktivnostjo. Pristop spremenljive dolžine aktivnosti sledi od začetka do konca seje in je ne razbije na manjše kose. Proces primerjave podobnosti sej je izpostavljen lastni interpretaciji. Seji z enako komponento in člani lahko na prvi pogled izgledata popolnoma enaki, a imata popolnoma različno karakteristiko. Denimo primer pri katerem imamo sejo, ki je dvakrat do trikrat daljša od druge seje in ima zelo različno vedenje na začetku in koncu. Problem pri uporabi sej spremenljive dolžine je zajemanje pravih količin informacij, ki predstavljajo takratno "obnašanje" uporabnika. Denimo, ko si dva uporabnika delita isti uporabniški račun na računalniku, vendar obiskujeta zelo različne spletne strani. Z uporabo pristopa s fiksno dolžino ločevanje med uporabnikoma trivialno.

3.2 Komponente

Komponente so dogodki zajeti tekom seje. Pri analizi internetne zgodovine se obisk spletne strani *Google* lahko šteje kot komponenta. Namesto splošnejše domene je veliko bolj zaželjena specifična stran (npr. *google.co.uk* namesto *google.com*). Vsaka datoteka, sprememba ali dostop do datoteke je lahko komponenta. Če vzorec prenesemo na nekoliko širše področje, je delovanje pametnega doma tudi lahko komponenta. Prav tako dogodki, kot je samodejno prižiganje luči, naprav ali drugih senzorjev. V raziskavi se bomo ukvarjali le s komponentami zgodovine brskanja po internetu. Komponente so ustvarjene v času analize. Čeprav bi lahko beležili število uporabe komponent (npr. ista stran se obišče večkrat v obdobju ene seje), ta podatek ni tako pomemben pri sami analizi primerjave sej. Nasprotno veljala pri analizi sekvenčenja, kjer je pomembno kolikokrat se bo dogodek "A" ponovil v zaporedju dogodkov "ABCAEFAG". Podatek o stanju dogodka - ali se je dogodek zgodil ali pe ne, je zato zadosten pri primerjavi sej. Denimo, da poskušamo pripisati neko sejo točno določenemu uporabniku, ki je znan kot strasten motorist. Podatek o tem kolikokrat je obiskana stran z vsebino motornih koles, je bistveno manj pomemben kot podatek, da je stran z vsebino motorjev bila obiskana.

3.3 Izris števila sej

Pri združevanju sej izpostavimo dva pristopa. Seja fiksne dolžine, ki obsega vse komponente znotraj fiksnega časovnega okna, in seja spremenljive dolžine s komponentami, kjer je časovni razmik še dopusten. Število ustvarjenih sej, ki jih dobimo pri združevanju, lahko predstavimo na grafu. Prva, časovno odvisna komponenta grafa, je definirana v sekundah. Pri pristopu fiksne dolžine je časovna vrednost določena vnaprej, pri pristopu spremenljive dolžine pa je ta določena z neko mejo. Avtorji trdijo, da bodo majhne časovne vrednosti pri obeh pristopih združevanja sej identificirale t.i. sistemsko obnašanje. Posledično bodo seje na premici izgledale kot manjše rezine, pogosto ponavljajoči se dogodki pa bodo poudarjeni. Večje časovne vrednosti bodo omogo-čale identifikacijo uporabnikovih navad.

3.4 Jaccard-ov koeficient podobnosti

Na podlagi podatka o komponeti lahko ustvarimo preprosto predstavitev informacij komponent in sej, kot jih prikazuje slika 1. Gre za nabor podatkov, ki sestojijo iz petih komponent (C1 do C5) in petih sej, pri čemer prva, druga in tretja seja pripadajo uporabniku 1, seji štiri in pet pa uporabniku 2. Tudi pri tako majhnih podatkih so razvidni ponavljajoči se vzorci. Opazimo jih lahko v sejah ena, dva, štiri in pet. Seja sestoji iz preprostega niza ničel in enic. Prav tako ji lahko izračunamo parno primerjavo oddaljenosti glede na preostale seje. Primer: Seji 1 [10101] in seji 2 [00111] je mogoče izračunati razdaljo, ki po Jaccar-dovem koeficientu podobnosti znaša 0.5. (Jaccard 1901 [14]). (Torej imata seji

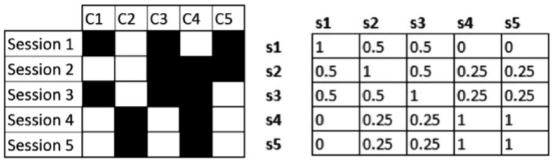


Figure 1: Preprost prikaz tabele s petimi sejami, petimi komponentami in dvema različnima uporabnika (levo) in prikaz tabele z ustrezno parno primerjavo med sejami (desno).

dve komponenti enaki, dve pa različni). Za izračun Jaccardove razdalje med dvema nizoma podatkov uporabimo spodnjo formulo:

$$d_j(A,B) = 1 - J(A,B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|}$$

Prednost uporabe Jaccard-a je, da upošteva le relevantne komponente, ki nastopajo v procesu. V seji ena in dve so to vse komponente, ki si jih seji delita, razen komponente C2, ki ni bila identificirana tako v prvi kot v drugi seji. Ko imamo opravka s 4000 komponentami in so vse izmed komponent enake 0 - tipičen primer je zgodovina domačega raču-

nalnika, je uporaba parnega primerjanja s pomočjo Hammingtove razdalje (Hamming, 1950 [11]) zelo nezaželjena, saj zaradi vseh skupnih komponent, ki imajo vrednost 0, po Hammingtovi razdalji dobimo 0.999 podobnosti. Slika 2 prikazuje primer realnih podatkov predstavljenih s tabelo. Vrstica predstavlja sejo, stolpec pa posamezno komponento. Bela polovica tabele predstavlja uporabnika 1, siva polovica pa uporabnika 2. Kot lahko razberemo iz slike, so nekatere komponente vsebovane pri obeh uporabnikih, vendar sta v resnici prikazana dva dokaj različna nabora internetne aktivnosti. Prav tako je iz tabele razvidno, da se pri obeh uporabnikih nekatere komponente ponavljajo.

3.5 Vzorci primerjave med sejami

Pri uporabi Jaccard-ovega koeficienta podobnosti smo pri sliki 1 spoznali da, prva in četrta seja nimata nobene skupne komponente. Posledično je njuna podobnost enaka 0. Vzorci so ustvarjeni z določanjem skupin po dve ali več sej, ki so nad Jaccard-ovo mero razdalje. Vsaka vrednost večja od 0.0 je potencialno uporabna. Število pridruženih sej se bistveno poveča, če se sprejemljiva Jaccard-ova vrednost zmanjša. Primer: pri vrednosti 1.0 je ustvarjen sledeči vzorec seje: Vzorec 1 = [s4 s5]. Pri Jaccard-ovi vrednosti 0.5 se ustvarita naslednji dve seji: Vzorec 1 = [s1 s2 s3], Vzorec 2 = [s4 s5]. Cilj primerjave je določiti par ali skupek večih sej, ki pripadajo skupnemu ponavljajočemu se vedenju. Na podlagi tega sklepamo, da gre morebiti za istega uporabnika. Če je stopnja povezovanja dovolj nizka, potem bo podobna dejavnost, ki potencialno lahko pripada drugemu uporabniku, opredeljena v našem vzorčenju med sejami. Če pa je raven previsoka, bo veljaveno le natančno ujemanje.

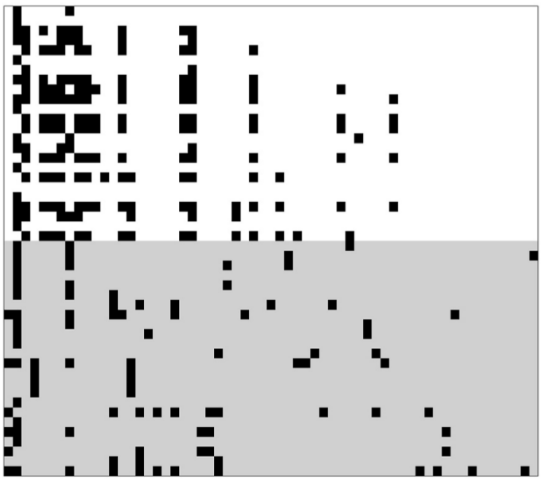


Figure 2: Primerki komponenti pri 50 sejah in dveh uporabnikih.

3.6 Končna ugotovitev pristopa

V raziskavi sta prikazana dva pristopa za izvedbo izbire sej. Prav tako je omenjeno kako podatke internetne zgodovine pretvoriti v komponente. Avtorji so ugotovili, da število pojavitve neke komponente v dani seji ni tako pomembna pri analizi primerjave sej. To jim je omogočilo, da so komponente prepoznali kot prisotne ali neprisotne v seji in jih pretvorili v preprost niz znakov ničel in enic. Podatki, ki jih prikazuje tabela na sliki 2, so še posebej zanimivi, ker prikazujejo različne tipe ponavljajočih se aktivnosti. Pri sistematični primerjavi sej takšne tabele hitro postanejo nepriročne. Zelo pomembna je predvsem uporaba primerne primerjalne metode. Namesto, da metoda upošteva vse možne komponente, je precej boljše, če upošteva le relevantne komponente, kot jih recimo Jaccard-ov koeficient podobnosti.

4. POSKUSI

V temu poglavju avtorji predstavijo dva praktična primera pristopa in poskusne teste s katerim določijo učinkovitost. In sicer metodi za izbiro seje s fiksno ali spremenljivo dolžino in ocenitev komponent med sejami pri različnih nivojih tolerance Jaccard-ovega koeficienta.

Cilj teh poskusov je pokazati, da lahko na podatkih, kjer je znano kateri artefakt spada kateremu uporabniku, izvedemo več različnih poskusov. S tem nastavimo spremenljivke tako, da lahko v prihodnosti analitik nad neznanimi podatki z neko gotovostjo pridobi optimalno sejno analitiko.

4.1 Poskusni podatki

Podatki uporabljeni v poskusih so časovnice spletnih zgodovin. Artefakti so začasno urejena zaporedja URL naslovov. Zatorej je izbira komponent na nivoju domen v delu URL naslova in ne na artefaktih posameznih spletnih straneh.

V obeh naborih podatkov je na voljo ena naprava z enim uporabniškim računom, do katerega dostopa več različnih uporabnikov. V enem naboru obstajajo trije uporabniki, v drugem dva. Tak primer, kjer na napravo z enim uporabniškim računom dostopa več ljudi, je realističen in pogost v preiskavah organa kazenskega pregona.

4.2 Nabor podatkov S

To je posamezen nabor podatkov pridobljen iz treh različnih delovnih postaj v primeru patentov M57 v Digital Corpora Project (Woods et al., 2011 [21]). Posledično te podatki predstavljajo pisarniški računalnik z redko poseljeno spletno zgodovino treh različnih uporabnikov z enakim uporabniškim računom. Časovna razlika med temi uporabniki mora biti dovolj velika, da velike spremenljive dolžine meje ali fiksne dolžine seje ne uvrstijo različnih uporabniških podatkov v isto sejo. Te podatki prikazujejo uporabo enega računalnika deljenega med tremi uporabniki, vendar vsi delajo v različnih izmenskih vzorcih, kar zagotavlja, da se ne pojavijo prekrivanja. Vzorec procesirane spletne zgodovine je viden na sliki 3. Vsak od treh uporabnikov ima približno enako velikost spletne zgodovine glede na časovno obdobje in število artefaktov.

4.3 Nabor podatkov R

Nabor podatkov R prihaja iz hišnega računalnika, z enim uporabniškim računom, ki je bil vključen v resnično preiskavo, kjer je identiteta uporabnika ob danem času neznana. Isti uporabniški račun sta uporabljala dva različna uporabnika. Spletna zgodovina tega računalnika je obsežna in prisotna so daljša neprekinjena obdobja uporabe spleta. Uporabnik 1 je večinski uporabnik računalnika, medtem, ko je uporabnik 2 manjšinski uporabnik z znatno manj dostopa do računalnika.

Time	Component Name	User	# of Artefacts
01/07/2009 22:50:37	wikipedia	1	14
01/07/2009 22:58:19	dell	1	15
01/07/2009 23:02:15	openoffice	1	6
01/07/2009 23:06:07	openoffice	1	6
01/07/2009 23:13:51	openoffice	1	6
01/07/2009 23:18:19	openoffice	1	6
01/07/2009 23:25:53	openoffice	1	6
01/07/2009 23:32:48	ccleaner	1	10
01/07/2009 23:34:53	gamblersanonymous	1	11
01/07/2009 23:35:30	gamblersanonymous	1	11
01/07/2009 23:37:44	gamblersanonymous	1	11
01/07/2009 23:43:50	gamblersanonymous	1	11
01/07/2009 23:44:14	scoresandodds	1	9
01/07/2009 23:44:18	scoresandodds	1	9
01/07/2009 23:45:07	scoresandodds	1	9
01/07/2009 23:45:11	scoresandodds	1	9
01/07/2009 23:52:25	scoresandodds	1	9
01/07/2009 23:52:29	scoresandodds	1	9
01/07/2009 23:54:31	scoresandodds	1	9
01/07/2009 23:56:14	mrklingon	1	10
02/07/2009 00:00:15	syfy	1	11
02/07/2009 00:06:02	syfy	1	11
02/07/2009 00:08:10	syfy	1	11

Figure 3: Primerki obdelane internetne zgodovine iz nabora podatkov S.

4.4 Izris sej

Na sliki 4 in 5 je izrisana analiza sej. Navpična os predstavlja število sej, ki so na voljo, ko nad celotno internetno zgodovino brskanja uporabimo enega od omenjenih pristopov. Vodoravna os pa predstavlja čas in je definirana kot: 30, 60, 120, 300, 600, 900, 1200, 1800, 3600 sekund (oz. polovico minute, 1, 2, 5, 10, 15, 30 in 60 minut). S slik 4 in 5 opazimo, da se število sej znatno zniža, če določimo neko dopustno mejo ali če je fiksna dolžina med 10 in 15 minut. Pri analizi zgodovine brskanja po internetu je to smiselno, ker je lahko uporabnik še vedno aktiven na računalniko. Recimo, ko na spletni strani bere neko vsebino ali pa gleda nek posnetek. Če je časovna os definirana z drugim tipom artefekta, recimo artifekt datotečnega sistema, potem bi opazili drugačen tip obnašanja. Za prilagoditev razlike, lahko nadaljne delo druževanja različnih večnivojskih artefektov zahteva lastno primerjavo med sejami artefaktov. Število sej bo vedno večje v primeru uporabe pristopa z nespremenljivo (fiksno dolžino). Krivulja pri podatkih v primeru pristopa spremenljive dolžine dokaj hitro pade in se potem v neki točki počasi izravna. V tej točki se število sej posledično rahlo spremeni, odvisno od tega, ali je časovni okvir seje definiran pri 900 sekundah ali pa 1500 sekundah. Podobno se dogaja pri pristopu z nespremenljivo dolžino. Krivulja se bo na koncu izravnala v neki točki. To je točka pri 1800 sekundah, v kateri se število sej le malo spremeni.

4.5 Optimizacija podatkov

Podatke so bili zmanjšani tako, da so bile enkratne komponente odstranjene. V splošnem imajo lahko tudi takšne komponente neko raziskovalno vrednost, npr. kot nek pomemben enkraten dogodek. Pri analizi ponavljajočega se vedenja, enkratne pojavitve služijo le zmanjšanju Jaccard-ove razdalje med dvema sejama.

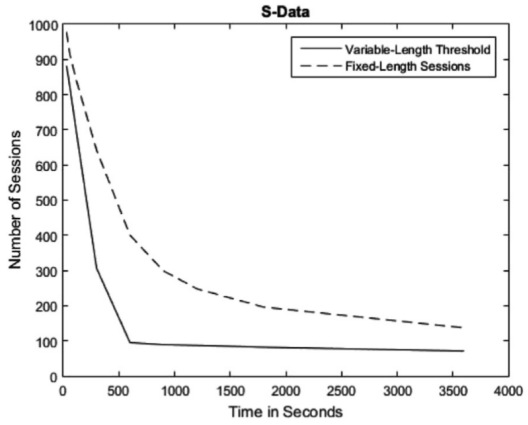


Figure 4: Graf temelji na naboru podatkov S. Prikazuje število sej in čas v sekundah med pristopom uporabe dovoljene vrednosti pri spremenljivi dolžini seje in fiksni vrednosti dolžine seje.

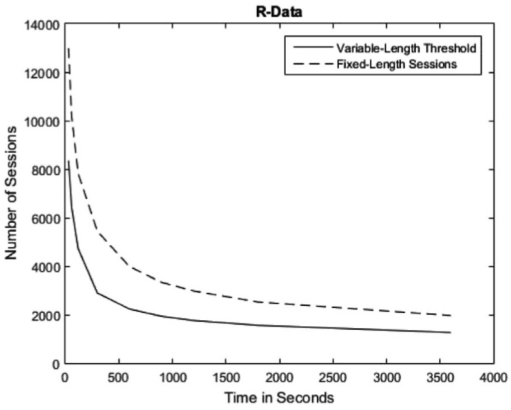


Figure 5: Graf temelji na naboru podatkov R. Prikazuje število sej in čas v sekundah med pristopom uporabe dovoljene vrednosti pri spremenljivi dolžini seje in fiksni vrednosti dolžine seje.

4.6 Pravilna klasifikacija uporabnikove aktivnosti

Hipoteza predstavljena pri preizkusih pravi: če imata dve seji velik Jaccard-ov koeficient podobnosti, potem pripadata istemu uporabniku. Da se trditev preskusi, je bil uporabljen nabor podatkov dveh zgodovin brskanja po spletu. To so bili testni vzorci podatkov, zbranih s strani avtorjev. Zgodovini imata različno karakteristiko in avtorji ne trdijo, da predstavljata pravi model obnašanja. Vsekakor pa opažajo, da je potrebno precej nadaljnega dela za modeliranje navadnega obnašanje uporabnika v številnih okoljih, kot recimo pisarniško okolje, skupno domače okolje, itd. Čeprav se iz rezultatov, ki se nanašajo na poiskus, pravilno določi uporabnika, sam poskus ne opredeli, da seja A pripada uporabniku 1 in seja B pripada uporabniku 2. Če sta v poskusu dva vzorca sej nad dovoljeno mejo Jaccard-ovega koeficienta podobnosti (0.25, 0.50, 0.75, 1.0) in če pripadata različnima uporabnikoma, potem gre za napako. Če pa seje pripadajo

le enemu uporabniku, se to smatra kot pravilna opredelitev uporabnikovega vedenja. Primer pri vzorčnih podatkih na sliki 1 (desno). Pri primerjanju seje 2 in 4 lahko opazimo, da je podobnost med njima enaka 0.25. V primeru, da bi seji predstavljali dva različna uporabnika in da določimo dovoljeno mejo na 1.0, 0.75 ali 0.5, bi ju pravilno opredelili kot ločeni si medseboj. V primeru, da določimo dovoljeno mejo na vrednost 0.25, bi ju opredelili napačno - torej, dobili bi, da pripadata istemu uporabniku. Sliki 6 in 8 prikazujeta izris med dovoljeno mejo pri uporabi pristopa spremenljive dolžine iz R in S nabora podatkov. Sliki 7 in 9 pa prikazujeta časovne vrednosti nespremenljive dolžine za enak nabor podatkov.

5. OCENJEVANJE

5.1 Splošna zmogljivost pristopa

Največ dela pri pristopu povzroča izračun Jaccard-ovega koeficienta za razdalje med parnimi sejami. Število sej je večje takrat, ko se uporablja pristop s fiksno dolžino za izračun agregacije sej. Pri uporabi fiksne dolžine ali nizke meje se čas računanja znatno poveča. Ta čas bi lahko znižali z uporabo gručenja komponent, kot so predlagane v Gresty et al. (2014) [8], vendar se to ni izkazalo za učinkovito, saj je glavna težava v številu sej. Kljub temu je omembe vredno to, da je lahko tak prikaz spletne zgodovine bolj "prijazen poroti", kot pa celoten prikaz prikazan na sliki (Fig 2). Nabor podatkov S (S-dataset) je namerno nastavljen tako, da ni v isto sejo klasificiranih več različnih uporabnikov. Nabor podatkov R (R-dataset) ima en primer, kjer sta dva uporabnika dostopala do istega računalnika v kratkem časovnem okviru, kar je povzročilo napačno klasifikacijo v primerih, kjer je bila meja nastavljena na manj kot 600 s.

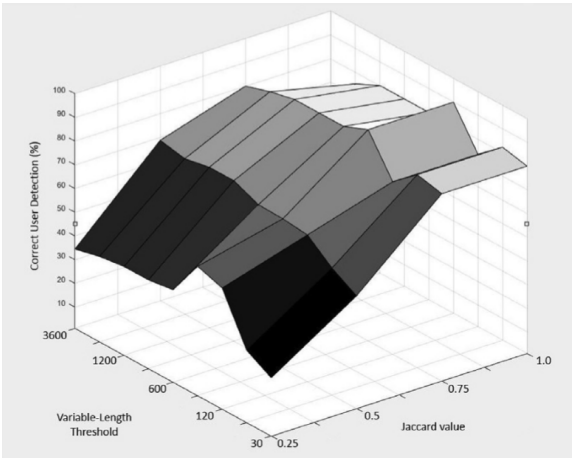


Figure 6: S nabor podatkov pri uporabi pristopa seje s spremenljivo dolžino.

5.2 Primerjava fiksne dolžine s spremenljivo dolžino

Pri spletni zgodovini je bil v splošnem bolj učinkovit pristop s spremenljivo dolžino kot pa s fiksno dolžino. Tak rezultat je pričakovan, saj pristop spremenljive dolžine bolje modelira uporabo naprav namenjenih človeški uporabi. V tem

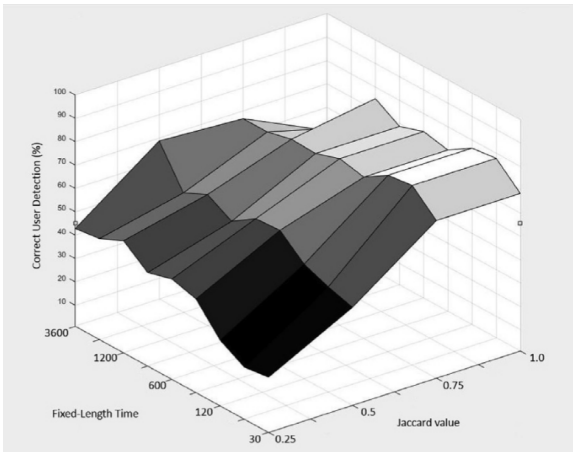


Figure 7: S nabor podatkov pri uporabi pristopa seje z nespremenljivo dolžino.

delu krajši artefakti, ki indicirajo sistemske procese, niso bili preizkuseni.

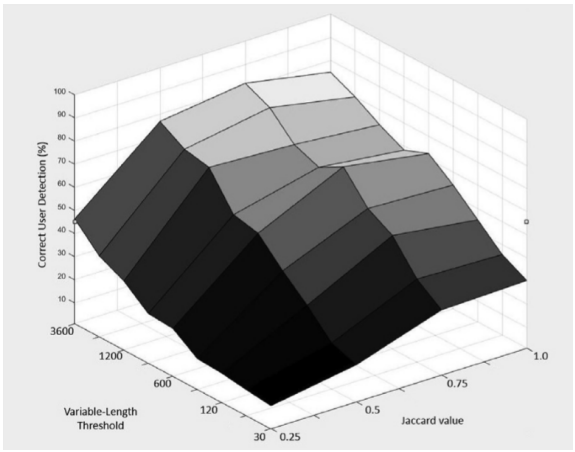


Figure 8: R nabor podatkov pri uporabi pristopa seje s spremenljivo dolžino.

5.3 Nivoji Jaccard-ovega koeficienta

Iz grafov, kjer je uporabljen pristop s spremenljivo dolžino in kjer je Jaccard-ova mera enaka 0,75, opazimo, da je optimalna izvedba med 300 in 900 s. Asociacije med različnimi sejami niso preveč omejene (pri meji 1,0), vendar je klasifikacija pravilnih uporabnikov veliko večja, če za to mejo uporabimo 0,5 ali manjše.

5.4 Stopnja pravilno zaznanih uporabnikov

Z uporabo spremenljive dolžine (0,75 do 1,0) na naboru podatkov S, je natančnost pravilne klasifikacije v rangu 80-90%. Na naboru podatkov R je ta natančnost okoli 56-80%. Te podatki so glede na neoptimiziran pristop zelo spodbudni. Znotraj dolgih primerjav med sejami so se pojavljale seje ("spoiler" seje), ki so vsebovale pogosto pojavljajoče

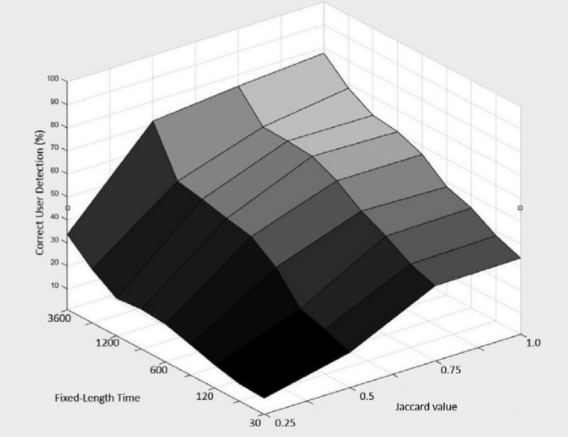


Figure 9: R nabor podatkov pri uporabi pristopa seje z nespremenljivo dolžino.

se komponente. Manjše primerjave med sejami so najredkeje napačno klasificirale uporabnika, saj se je tu uporabljala Jaccard-ova meja bližje 1,0. Z večjo mero se v vzorcu pojavi manj sej, kar posledično pomeni, da se v vzorcu pojavi manj "spoiler" sej. Te seje vsebujejo komponente, ki niso edinstvene posameznemu uporabniku znotraj nabora podatkov, vendar pa vsebujejo spletne strani, za katere bi pričakovali, da jih obiskuje več različnih uporabnikov, kot sta na primer Google in MSN. Prav tako so eksperimentirali z odstranjevanjem pogosto ponavljajočih se komponent. S tem se je drastično povečala stopnja pravilnosti, v nekaterih primerih tudi do 100%. Vendar se je s tem zmanjšalo število sej, kar je v nekaterih naborih podatkov popolnoma izbrisalo manjšinske uporabnike. Primer takega uporabnika (User 2) je v naboru podatku R, kjer je uporabnik imel karakteristično ponovljivo vedenje. Ko pa so izvedli odstranjevanje pogosto se ponavljajočih komponent, je bil ta uporabnik izločen iz postopka odkrivanja. Najboljši pristop za odstranjevanje "spoiler" komponent, je verjetno kombinacija nekakšnega predhodnega učenja podatkov skupaj z odstranjevanjem najbolj pogostih komponent. Nadaljne raziskave se bodo osredotočale na količino redukcije in/ali predhodno učenje, ki je potrebno za uspešno izboljšanje identifikacije uporabnikovih samopodobnih obnašanj.

6. ZAKLJUČEK IN NADALJNJA DELA

Predstavljen je bil pristop za začasno združevanje artefaktov spletne zgodovine v seje, prav tako je bilo argumentirano, da je uporaba primerjave med sejami uporabna v raziskovalne namene, s tem da identificiramo edinstvene in ponavljajoče se dogodke. Pristop in eksperimenti predstavljeni v tem delu so pokazali, da množica podatkov ne potrebuje predhodnih učenj ali kakršnih koli predhodni znanj, in da lahko to preiskovalec uporabi za identificiranje in demonstriranje običajnega obnašanja. Glavni cilj raziskovalnega projekta je izdelati orodje, s katerim je možno preiskusiti preiskovalne hipoteze, kot je na primer "Ali se je ob času in datumu X izvajalo ponavljajoče se obnašanje?". Avtorji verjamenjo, da bi uporaba takega orodja skupaj s sistematičnim pristopom za preiskuševanje pričanj in dejanskih dokazov, kot je prikazano v James et al. (2010) [15], zelo pomagala analitikom ali preiskovalcem pri tem, da bi demonstrirali, da je neka dejavnost dejanski vzorec podprt z dokazi.

Z uporabo reduciranja spletne zgodovine v komponente so pokazali, da je lahko zgodovina predstavljena, kot na sliki 2 2, in da lahko človek prepozna razlike in podobnosti med različnimi sejami. Vendar pa s tisočimi komponentami in potencialno na deset tisoče sej ta vizualizacija celotne spletne zgodovine postane neuporabna. Zato komponente spremenijo v binarni niz, za katere je nato možno izračunati podobnosti med dvema sejama. Rezultati njihovih eksperimentov so pokazali, da lahko različne metode izbire sej izdelajo znatno različne tipe in velikosti sej. Za analizo spletne zgodovine na naborih podatkov uporabljenih znotraj tega dela, menijo, da naj se uporabi meja intervala za ločevanje različnih sej med 10 in 15 minut. Za doseganje najbolj natančnega klasificiranja posameznih uporabnikov naj se uporabi spremenljiva dolžina sej, če je na enem računalniku več uporabnikov.

Eksperimenti v tem delu so uspešno klasificirali 90% aktivnosti v enem izmed naborov podatkov z uporabo metode za združevanje sej. Vendar so mnenja, da bi z uporabo klasificiranja generičnih in pogosto obiskanih spletnih strani, znatno znižali pojavitve napak v vzorcih in boljše prepoznavanje edinstvenih sejnih vzorcev. Pri začetnih eksperimentih, kjer so bile odstranjene nekatere najbolj obiskane komponente, je bila stopnja pravilnih klasifikacij velika, vendar so se pri tem izgubila pomembna ponavljajoča se obnašanja. Zatorej je potrebno nadaljnje delo za določitev pravilnega deleža za odstranitev kvarilnih "spoiler" komponent. To bi bilo lahko mogoče z uporabo predhodnega učenja, kjer bi se določilo katere komponente so generične.

6.1 Nadaljnje delo

Predlagane tehnike v tem delu so samodejno združevanje sej glede na statične vrednosti določene na začetku analize. To lahko pridela rezultate, kjer so podatki ogromni, kar povzroči veliko dela pri procesiranju v parnem zaporedju. Avtorji verjamejo, da se seje lahko reducirajo na sejne profile, kjer se seje s podobnimi karakteristikami, kot so na primer dolžina, trajanje, gostota artefaktov v časovni peroidi, procesirajo skupaj.

V analizi rezultatov eksperimentov so omenili, da je možnost pristranskosti pri uporabi Jaccard-ovega koeficienta v vzorcih, kjer imajo seje manjše število komponent ali v vzorcih z večjim številom komponent, kjer je velika mera odvisnosti. Ta pristranskost se pojavi v sklopu načrta, vendar se z uporabo višjih Jaccard-ovih mer lahko zmanjša število sej v sejnih vzorcih. Lahko bi vključili dinamične izbire, kjer bi seje z manjšim številom komponent uporabile večjo Jaccard-ovo mero preden bi se te dodale v vzorce. Če ima seja različne ali veliko število komponent, potem bi bil uporabljen nižji nivo tolerance. Vrednost vzorca bi bila lahko nadzirana glede na kompleksnost vzorca. Ustrezna mera te kompleksnosti je stvar nadaljnjega dela.

Možna razširitev dela je avtomatično klasificiranje samega spletnega obnašanja, kjer so seje hkrati združene in klasificirane. Prav tako naj se spletne strani preiščejo v spletnem predpomnilniku ali med obnovljenimi spletnimi stranmi na napravi. Ta dodatek bi potreboval možnost razčlenjevanja spletnih strani in določil vsebino glede na razčlenjene besede.

Predpomnjene spletne strani preiskovalcu pogostoma niso na voljo, zato bi se iskanje opravilo na sredstvih, kot so Internet Archive Wayback Machine [1], s katerim je možno poiskati najbližjo časovno periodo, ki ustreza artefaktom naprave v preiskavi. Taka razširitev, bi preiskovalcu omogočala hitro identificiranje enkratnih in ponavljajočih se dogodkov, brez da bi sam moral interpretirati URL naslove spletnih strani.

Pristop z uporabo primerjav med sejami ponuja širok pregled celotne seje, vendar ne upošteva, če obnašanja podsej skupaj tvorijo eno sejo. Za ta namen, poleg analize med sejami, razvijamo še analizo med komponentami.

Pristop predstavljen v tej raziskavi je primerjava med sejami brez upoštevanja vrstnega reda, zaporedja in količine komponent znotraj seje. Naslednji korak profiliranja na videz podobnih sej, je pridobiti zaporedne vzorce znotraj seje za prepoznavanje ponavljajočih se vzorčnih zaporedij komponent. Avtorji menijo, da bi s prepoznavanjem kandidatov sej uporab-ljenih v primerjavi med sejami, znatno zmanjšali čas računanja pregledovanja znotraj ene seje.

7. REFERENCE

[1] Internet archive wayback machine. https://archive.org/web/.

[2] S. A. Awawdeh, I. Baggili, A. Marrington, and F. Iqbal. Cat record (computer activity timeline record): A unified agent based approach for real time computer forensic evidence collection. pages 1–8, Nov 2013.

[3] S. W. Brenner, B. Carrier, and J. Henninger. The trojan horse defense in cybercrime cases. *Santa Clara Computer & High Tech. LJ*, 21:1, 2004.

[4] F. P. Buchholz and C. Falk. Design and implementation of zeitline: a forensic timeline editor. 2005.

[5] R. Carbone and C. Bean. Generating computer forensic super-timelines under linux. *SANS Reading Room*, pages 1–136, 2011.

[6] Y. Chabot, A. Bertaux, C. Nicolle, and T. Kechadi. Automatic timeline construction and analysis for computer forensics purposes. In *Intelligence and Security Informatics Conference (JISIC), 2014 IEEE Joint*, pages 276–279. IEEE, 2014.

[7] N. Eagle and A. S. Pentland. Eigenbehaviors: Identifying structure in routine. *Behavioral Ecology and Sociobiology*, 63(7):1057–1066, 2009.

[8] D. W. Gresty, D. Gan, G. Loukas, et al. Digital forensic analysis of internet history using principal component analysis. 2014.

[9] K. Gu■jónsson. Mastering the super timeline with log2timeline. *SANS Institute*, 2010.

[10] R. Hammid, S. Maddi, A. Johnson, A. Bobick, I. Essa, and C. L. Isbell. Unsupervised activity discovery and characterization from event-streams. *arXiv preprint arXiv:1207.1381*, 2012.

[11] R. W. Hamming. Error detecting and error correcting codes. *Bell Labs Technical Journal*, 29(2):147–160, 1950.

[12] C. Hargreaves and J. Patterson. An automated timeline reconstruction approach for digital forensic investigations. *Digital Investigation*, 9:S69–S79, 2012.

[13] B. Inglot and L. Liu. Enhanced timeline analysis for digital forensic investigations. *Information Security Journal: A Global Perspective*, 23(1-2):32–44, 2014.

[14] P. Jaccard. Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bull Soc Vaudoise Sci Nat*, 37:547–579, 1901.

[15] J. James, P. Gladyshev, M. T. Abdullah, and Y. Zhu. *Analysis of Evidence Using Formal Event Reconstruction*, pages 85–98. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[16] J. I. James and P. Gladyshev. Automated inference of past action instances in digital investigations. *International Journal of Information Security*, 14(3):249–261, 2015.

[17] S. Kälber, A. Dewald, and S. Idler. Forensic zero-knowledge event reconstruction on filesystem metadata. In *Sicherheit*, pages 331–343, 2014.

[18] M. Khan and I. Wakeman. Machine learning for post-event timeline reconstruction. pages 112–121, 2006.

[19] P. Khatik and P. Choudhary. An implementation of time line events visualization tool using forensic digger algorithm. *JCSE Int J Comput Sci Eng*, 2(4), 2014.

[20] J. Kiernan and E. Terzi. Constructing comprehensive summaries of large event sequences. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 3(4):21, 2009.

[21] C. A. Lee, A. Russel, K. Kearton, D. Dittrich, K. Woods, and S. Garfinkel. Creating realistic corpora for forensic and security education. 2011.

[22] A. D. Marrington. *Computer profiling for forensic purposes*. PhD thesis, Queensland University of Technology, 2009.

[23] D. Minnen, T. Starner, I. A. Essa, and C. L. Isbell Jr. Improving activity discovery with automatic neighborhood estimation. In *IJCAI*, volume 7, pages 2814–2819, 2007.

[24] J. Olsson and M. Boldt. Computer forensic timeline visualization tool. digital forensic research workshop, digital investigation 6, s78–s87 (2009).

[25] M. Schaefer, F. Wanner, F. Mansmann, C. Scheible, V. Stennett, A. T. Hasselrot, and D. A. Keim. Visual pattern discovery in timed event data. In *IS&T/SPIE Electronic Imaging*, pages 78680K–78680K. International Society for Optics and Photonics, 2011.

[26] Y. Ye, Y. Zheng, Y. Chen, J. Feng, and X. Xie. Mining individual life pattern based on location history. In *Mobile Data Management: Systems, Services and Middleware, 2009. MDM'09. Tenth International Conference on*, pages 1–10. IEEE, 2009.

Preverjanje avtorstva dokumentov za različne jezike, žanre in tematike

[Povzetek članka Authorship verification for different languages, genres and topics [3]]

Gregor Šobar
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana, Slovenija
gs4015@student.uni-lj.si

Jure Grabnar
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana, Slovenija
jg3236@student.uni-lj.si

Rok Zupančič
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana, Slovenija
rz7306@student.uni-lj.si

POVZETEK

V članku je podan pregled področja preverjanja in določanja avtorstva dokumentov. Jedro članka je osredotočeno na opis ključnih komponent izvornega članka *Authorship verification for different languages, genres and topics*[3], na podlagi katerega smo tudi implementirali predlagano metodo preverjanja avtorstva. Članek navaja rezultate testiranja in ugotovitve avtorjev izvornega članka, podana pa je tudi krajša primerjava uspešnosti algoritma s konkurenčnimi rešitvami. Metoda je bila testirana na 28 različnih korpusih s 16 žanri in različnimi tematikami, ki skupno zajemajo kar 4525 primerov besedil. Rezultati so primerljivi s konkurenčnimi metodami (mediana 75%) in ponekod boljši (v povprečju 5%) od trenutno najboljših metod. Glavni prednosti predstavljene metode so v enostavni razširljivosti z novimi jeziki in nizka računska zahtevnost.

Ključne besede

preverjanje avtorstva, analiza besedil, določanje avtorstva

1. UVOD

Forenzična analiza avtorstva je veja digitalne forenzike z veliko uporabnih scenarijev. Obstajajo primeri, kjer za nek dokument domnevamo avtorja oz. obstaja domneva, da oseba ni napisala dokumenta, vendar avtorstvo v obeh primerih ne moremo natančno določiti. Kot primer lahko podamo zaključne naloge na univerzah [7], ki so lahko bodisi plagiat bodisi plačano delo tretje osebe. Med druge primere uporabe teh metod se navajajo tudi izmišljeni zavarovalniški zahtevki, ponarejene oporoke in odkrivanje spletnih identitet neke osebe [1].

Glavna naloga forenzične analize avtorstva je *preverjanje avtorstva* dokumenta. Uporablja se pri zaznavanju avtorstva dokumenta. Sem spada tudi ugotavljanje ali je nezakonito dejanje (npr. spletno prevaro, distribucijo ilegalnih oziroma ukradenih datotek, trgovino z drogami) povzročila specifična oseba. Druga stran forenzične analize avtorstva je profiliranje avtorja, kjer določimo njegove značilnosti (npr. spol, starost, regijo, socialno ozadje). S forenzično analizo avtorstva ugotavljamo tudi kateri deli dokumenta pripadajo določenemu avtorju, v kolikor je dokument podpisalo več njih.

V nadaljevanju članka bomo na kratko predstavili ključne komponente potrebne za razumevanje obravnavanega področja, predstavljena sorodna dela drugih avtorjev ter nenazadnje predstavitev jedrnega članka ([3]). Ob koncu navajamo tudi njihove rezultate testiranja in ugotovitve.

2. PODROBNEJŠI PREGLED PODROČJA

Pri določanju avtorstva besedila (ang. authorship attribution - AA) se srečujemo s problemom, kjer anonimnemu besedilu pripišemo najbolj verjetnega avtorja iz referenčnega seznama avtorjev [9]. V kolikor je zagotovljeno, da referenčni seznam vsebuje resničnega avtorja potem je to zaprta množica, sicer odprta. Večina obstoječih raziskav se osredotoča na zaprte množice. Množica vseh avtorjev z njihovimi pripadajočimi testnimi besedili imenujemo referenčna množica.

V sklop forenzične analize avtorstva spada tudi problem preverjanja oz. potrjevanja avtorstva (ang. autorship verification - AV) - ali je podpisana oseba resnično avtor analiziranega besedila. Avtorja potrjujemo na podlagi referenčne množice besedil, ki gotovo pripadajo dotičnemu avtorju. Z analizo izločimo značilke besedil v referenčni množici in značilke besedila tretjega avtorja. Z določenim pragom podobnosti glede na referenčno množico lahko potrdimo avtorja tretjega besedila.

Preverjanje avtorstva lahko razumemo tudi kot klasifikacijski problem z enim razredom [11], kjer ugotavljamo ali besedilo pripada množici obstoječih besedil določenega avtorja ali ne. Preverjanje avtorstva je precej zahtevno, predvsem zaradi določanje praga podobnosti. Kljub zahtevnosti se v praksi pogosto uporablja - v poštev pride tudi pri drugih opravilih, kot je npr. odkrivanje plagiatov.

Na podlagi Stamatatos et al. (2014)[10], so lahko AV metode intrinzične ali ekstrinzične. *Intrinzične* metode se navezujejo samo na obravnavani dokument in referenčno množico znanega avtorja za namen določanja avtorstva neznanega dokumenta. *Ekstrinzične* metode po drugi strani uporabijo dodatne dokumente tujih avtorjev za namen transformacije AV naloge iz eno-razredne v binarno klasifikacijsko nalogo.

Izvorni članek [3] na podlagi katerega je bila narejena naša implementacija AV algoritma uvrščamo med intrinzične metode. Sami namreč ne uporabljajo dodatnih besedil za odločanje ali je bil specifičen dokument napisan s strani drugega avtorja.

Glede na Stamatatos (2009) [9] obstajajo pri določanju ali preverjanje avtorja sledeči izzivi:

- Redka, neenakomerna ali težka besedila s pogovornimi frazami, okrajšavami in napakami.
- Vpliva teme, zvrsti, časovne periode in morfoloških značilnosti jezika dokumenta.
- Točnost uporabljenih orodij (npr. orodja za procesiranje naravnih jezikov).
- Iskanje primernih značilk za dani problem.

Enoten problem pa je tudi pomanjkanje standardov, predvsem pri neusklajenosti gradiv in postopkov za dosledno primerjanje ocenjevanj uspešnosti med raziskovalci (in/ali algoritmi) na tem področju.

3. SORODNO DELO

V zadnjih letih so bile raziskave na področju ugotavljanj avtorstva precej uspešne. Omeniti velja *sleparsko metodo* (ang. Impostors Method), ki sta jo predlagala Koppel in Winter (2014)[5]. Metoda išče "drugačno" besedilo iz obravnavanega kot tudi referenčnih dokumentov. Pod "drugačno" besedilo smatramo sestavke, ki po značilkah izstopajo od ostale vsebine. Sleparska metoda enorazredni klasifikacijski problem spremeni v večrazredni, kar pomeni, da besedilo oziroma dokumente glede na podobnost razdeli v več razredov. Podobnosti med besedili se izračunajo preko več iteracij z različnimi kombinacijami značilk. Ob dovolj velikem ujemanju z referenčnim dokumentom metoda določi tudi avtorja besedila. Nekaj težav nastane pri različnih tematikah, jeziku ali žanru. Kljub temu je v praksi metoda precej dobra in uporablja enostavno določljive značilke.

Sleparska metoda je splošno uporabljena. Z nekaj izboljšavami je bila zmagovalka na tekmovanju PAN (tekmovanje v forenzični analizi besedil) leta 2013 in 2014:

- Prva izboljšava avtorja Seidman (2014) [8] temelji na spremembah parametrov: število iteracij, nabor značilk in praga podobnosti.
- Druga izboljšava avtorjev Khonji and Iraqi (2014)[4] temelji na optimizaciji značilk ter ostalih parametrov, povečanju števila značilk in iskanju karakteristik jezika oz. zvrsti besedila.

Druga najbolj uspešna metoda tekmovanja PAN je osnovana na CART algoritmu (ang. Clasification And Regression Trees algorithm), avtorjev Frery et al. (2014) [2]. Več problemov istega jezika tvori eno učno množico. Značilke različnih dokumentov (npr. pogostost besed, črk, ločil) se uporabi za določanje podobnosti med znanimi in neznanimi dokumenti. Algoritem deluje dobro za različne žanre in jezike, poleg tega pa je tudi hiter in učinkovit.

Digitalno forenziko besedil lahko izvajamo tudi s Moreau et al. [6] pristopom. Ideja slednjega je računanje podobnosti med dokumenti s poznanim avtorjem in dokumenti z neznanim avtorjem, ki se nahajajo v učni monžici. V tem koraku določimo prag odločanja. Ocena podobnosti je sestavljena iz petih vrednosti: dve oceni podobnosti Jaccard, dve divergenčni oceni in število poznanih dokumentov. Končno odločitev izračunamo s klasifikacijskim algoritmom na celotni učni množici.

4. METODOLOGIJA

V nadaljevanju bomo predstavili predlagano metode v našem referenčnem članku [3].

4.1 Značilke

Značilke so preprost način za hevristično vrednotenje besedila in posledično določanja avtorstva. Razvite so na podlagi jezikoslovnih, psiholoških in matematičnih značilnosti. Metoda opisana v članku jih razdeli v devet kategorij:

- F_1: n-gram ločil (ang. Punctuation n-grams)
  Primer: "This.is/a:sample-text" $\xrightarrow{\text{n=3}}$ (./:, /:-); $n \in \{1, 2, ..., 10\}$
- F_2: n-gram črk (ang. Character n-grams)
  Primer: "This is a sample text" $\xrightarrow{\text{n=3}}$ (Thi, his, is_, s_i, _is, is_, s_a,...); $n \in \{1, 2, ..., 10\}$
- F_3: n% najpogostejših simbolov (ang. n% frequent tokens)
  Primer: n% najpogosteje uporabljenih simbolov; $n \in \{5, 10, ..., 50\}$
- F_4: predpona simbolov dolžine k (ang. Token k-prefixes)
  Primer: "This is a sample text" $\xrightarrow{\text{n=2}}$ (Th, is, sa, te); $k \in \{1, 2, 3, 4\}$
- F_5: pripona simbolov dolžine k (ang. Token k-suffixes)
  Primer: "This is a sample text" $\xrightarrow{\text{n=2}}$ (is, is, le, xt); $k \in \{1, 2, 3, 4\}$
- F_6: predpona n-gramov simbolov dolžine k (ang. Token k-prefixes n-grams)
  Primer: "This is a sample text" $\xrightarrow{\text{n=2}}$ (This_is, is_a, a_sample, sample_text) $\xrightarrow{\text{k=2}}$ (Th_is, sa_te); $n \in \{2, 3, 4\}$, $k \in \{1, 2, 3, 4\}$
- F_7: pripona n-gramov simbolov dolžine k (ang. Token k-suffixes n-grams)
  Primer: "This is a sample text" $\xrightarrow{\text{n=2}}$ (This_is, is_a, a_sample, sample_text) $\xrightarrow{\text{k=2}}$ (is_is, le_xt); $n \in \{2, 3, 4\}$, $k \in \{1, 2, 3, 4\}$
- F_8: predpone dolžine n in pripone dolžine k (ang. n-prefixes-k-suffixes)
  Primer: "This is a sample text" $\xrightarrow{\text{n,k=2}}$ (Th_is, is, sa_le, te_xt); $n, k \in \{1, 2, 3, 4\}$

- F_9: pripone dolžine n in predpone dolžine k (ang. n-suffixes-k-prefixes)

  Primer: "This is a sample text" $\xrightarrow{n=3,k=2}$ (his_is, ple_te); $n, k \in \{1, 2, 3, 4\}$

Podniz z n zaporednimi besedami oz. simboli se imenuje n-gram.

Članek opisuje metodo, ki s pomočjo regularnih izrazov izlušči značilke, potrebne za računanje podobnosti med besedili. Izluščevanje značilk poteka v najslabšem primeru v linearnem času.

4.2 Podatki

4.2.1 Korpus

Raziskovalci so imeli precej težav pri iskanju besedil testne množice. Poiskati so morali besedila, ki so gotovo avtorska in pridobiti dovoljenje za ponovno uporabo. Pomemben je bil tudi pravi format besedila: *PAN AV korpus*. Vsak problem sestavlja obravnavani dokument in množica znanih dokumentov. Množico sestavlja 1-10 dokumentov, njihov obseg pa je od nekaj sto do nekaj tisoč besed. Porazdelitev besedil med avtorskimi in ne-avtorskimi je približno enakomerna. Znotraj problema je isti jezik, množico vseh problemov pa sestavlja več jezikov: nemščina, angleščina, grščina, španščina in nizozemščina. Korpus so sestavljala besedila iz različnih zvrsti. Razdeljen je bil v dve kategoriji: učna množica, testna množica in pred-testna. Testno množico[1] so sestavljali vsi dokumenti podatkovne baze *PAN AV 2013* in PAN AV 2014 ter še nekaj dodatnih korpusov. Pripravili so tudi svoj nabor podatkov, ki je bil sestavljen iz elektronske pošte in zaključnih del raziskovalcev na njihovem inštitutu, člankov iz znanstvenih revij in kuharskih receptov.

4.2.2 Predprocesiranje podatkov

1. Iz korpusa odstranimo duplikate dokumentov, ki jih določimo na podlagi koeficienta prekrivanja:

$$prekrivanje(X, Y) = \frac{|X \cap Y|}{min(|X|, |Y|)} \quad (1)$$

   Množici X in Y, v enačbi 1, predstavljata množico vseh simbolov dokumentov, ki jih želimo primerjati. Dokumenta sta si podobna, če koeficient prekrivanja preseže neko vrednost. Prag podobnosti določimo eksperimentalno. V članku [3] dokumente s koeficientom prekrivanja nad 0.25 označujejo kot duplikate.

2. Odstranjevanje šuma: odstranimo značke, ne-besede, številke, znake za nove vrstice. Več znakov za presledek pretvorimo v enega. Po tem koraku je besedilo predstavljeno kot en sam dolg niz s presledki.

3. V zadnjem koraku predprocesiranja združimo vse probleme v učni množici posameznega jezika v eno skupno množico za določanja značilk jezika.

[1] Testna množica je dosegljiva na http://bit.ly/1OjFRhJ

4.3 Opis predlagane metode za preverjanje avtorstva

Poglavje se navezuje na metodo iz članka [3]. Prvi korak metode je učenje. Tu pridobimo model z devetimi kategorijami značilk za vsak jezik posebej. Učenje poteka v štirih korakih:

1. Izdelamo vektor značilk in izračunamo podobnosti med pari dokumentov:
   Cilj koraka je računanje podobnosti med neznanim obravnavanim dokumentom in znanimi dokumenti. Znane dokumente združimo v eno samo množico za večji statistični vzorec. Nato izračunamo vektor značilk za obe množici posebej (neznan dokument in vsi znani dokumenti). Vsak element vektorja značilk je relativna frekvenca pojavitve nekega simbola značilke v množici glede na celotno množico, ki ji pripada (Enačba 2).

$$\phi(f_j) = \frac{\text{št. pojavitev simbola} f_j \text{v množici}}{\text{št. vseh simbolov v celotni množici}} \quad (2)$$

   V primeru neznanega dokumenta celotna množica vsebuje le neznani dokument, v primeru znanega dokumenta pa celotna množica vsebuje vse znane dokumente. Simboli enega vektorja značilk so določeni z eno samo značilko, torej ima vsaka značilka svoj vektor značilk. V našem primeru to pomeni, da izračunamo 18 vektorjev značilk (devet značilk, za vsako izračunamo dva vektorja značilk - za neznan dokument in za vse znane dokumente).

   Sedaj lahko za vsako značilko izračunamo manhattansko razdaljo med pripadajočim vektorjem značilk za neznan dokument in vektorjem značilk za neznane dokumente (enačba 3):

$$razdalja(X, Y) = \sum_{j=1}^{n} |x_j - y_j| \quad (3)$$

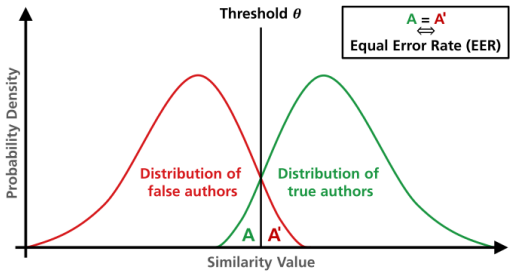
   kjer je X vektor značilk neznanega dokumenta in Y vektor značilk znanih dokumentov. Za končni koeficient podobnosti normaliziramo razdaljo z znano transformacijo (enačba 4):

$$podobnost(X, Y) = \frac{1}{1 + razdalja(X, Y)} \quad (4)$$

   in tako zagotovimo, da se koeficient podobnosti nahaja med 0 in 1.

2. Določitev praga podobnosti:
   Cilj koraka je določiti prag (mejo) podobnosti, kjer lahko rečemo, da sta si vektorja (dokumenta) podobna. Želimo minimizirati razliko med številom nepravilnih pozitivnih odločitev (problemi, ki ne pripadajo avtorju, a jih algoritem označi, da pripadajo) in nepravilnih negativnih odločitev (problemi, ki pripadajo avtorju in jih algoritem označi, da ne pripadajo avtorju). To je vidno na Sliki 1. V ta namen izračunamo mediano koeficientov podobnosti izračunanih na vseh problemih nekega jezika. Tako smo lahko gotovi, da je število

nepravilnih pozitivnih in nepravilnih negativnih odločitev približno enako. Prag računamo za vsako kategorijo značilk posebej. Dokument smatramo, da pripada avtorju, če je koeficient podobnosti višji od tega praga, sicer dokument smatramo, da ne pripada avtorju.



Slika 1: Pri določanju praga podobnosti želimo minimizirati razliko med nepravilnimi pozitivnimi odločitvami in nepravilnimi negativnimi odločitvami.

3. Določitev najboljših parametrov znotraj kategorije značilk:
   Cilj koraka je določitev najboljših parametrov za vsako kategorijo značilk ter za vsak jezik. V ta namen ponavljamo prvi in drugi korak z različnimi parametri za posamezne kategorije značilk. Izberemo tiste parametre, ki imajo največjo točnost za vse dane probleme:

$$\text{točnost} = \frac{\text{število pravilnih odgovorov}}{\text{število vseh problemov}} \qquad (5)$$

4. Gručenje značilk:
   Cilj zadnjega koraka je gručenje posameznih značilk v uporabno skupino. S tem izboljšamo točnost napovedi, saj ima posamezna značilka nižjo točnost kot več značilk skupaj. Tako izberemo le nekaj značilk, ki bodo proizvedle najboljše rezultate. Sestavimo vse možne kombinacije značilk (256 možnih, vedno liho število značilk, da ni izenačenja), izračunamo napoved za vsako značilk v skupini in za končno napoved vzamemo tisto, ki ima največ glasov.

5. TESTIRANJE IN REZULTATI

Avtorji izvornega članka [3] so metodo testirali na dva načina: glede na posamezno kategorijo značilk in glede na grupirane značilke. Prvi poskus je torej zajemal in med seboj primerjal posamezne kategorije značilke. Rezultati so pokazali (Slika 2), da se za največ jezikov najbolje obnese značilka F_2, ki predstavlja n-grame črk. Ta rezultat se ujema s sorodnimi članki [5], [10], kjer se značilka z n-grami črk prav tako obnese najbolje. Avtorji članka si razlagajo to dejstvo s tem, da je značilka z n-grami črk mešanica preostalih kategorij značilk (npr. značilke, ki temeljijo na predponah/priponah). Algoritem najbolje deluje za španski jezik (mediana 73.2%), medtem ko so najslabši rezultati pri grščini (mediana 57.89%). Najslabše se je obnesla značilka F_1 (n-gram ločil), saj ponuja najmanjšo raznolikost simbolov.

V drugem poskusu so avtorji želeli določiti najboljše gruče značilk za vsak jezik na podlagi vseh kombinacij značilk.

F_i	$\mathbb{C}_{nl}$	$\mathbb{C}_{uk}$	$\mathbb{C}_{gr}$	$\mathbb{C}_{es}$	$\mathbb{C}_{de}$	Median
F_1	62.12	65.1	57.37	72.55	68.7	65.1
F_2	**71.21**	**73.33**	60	73.2	**74.78**	**73.2**
F_3	64.65	69.8	56.84	**77.78**	69.13	69.13
F_4	65.66	68.63	53.68	71.24	70.87	68.63
F_5	63.64	67.84	50.53	70.59	66.96	66.96
F_6	65.15	68.63	**67.37**	75.82	71.74	68.63
F_7	63.13	68.63	57.89	68.63	65.65	65.65
F_8	69.19	72.16	64.74	**77.78**	71.3	71.3
F_9	68.18	71.76	61.05	74.51	73.04	71.76
Median	65.15	68.63	57.89	**73.2**	70.87	

Slika 2: Testiranje posameznih značilk za posamezni jezik (prvi poskus). V zadnji vrstici/stolpcu se nahaja mediana za dotični jezik oz. značilko.

Language	Accuracy (%)	Ensemble
Dutch	72.47	$\{F_2,F_8,F_9\}$
English	76.67	$\{F_1,F_2,F_3,F_7,F_8\}$
Greek	67.37	$\{F_6\}$
Spanish	83.33	$\{F_1,F_3,F_4,F_6,F_8\}$
German	78.04	$\{F_1,F_2,F_3,F_5,F_8\}$

Slika 3: Poskus 2 [3]: Najboljše značilke glede na jezik (naučene na testnih podatkih).

Rezultat so vidni na Sliki 3. Vidimo, da se več značilk obnese bolje od posameznih značilk, saj je točnost nekoliko višja. Zanimivo je tudi opažanje, da se F_1 značilka pojavi v najboljši gruči v treh od petih jezikih, čeprav se je obnesla najslabše pri prvem poskusu. Grupirane značilke očitno ponujajo boljše rezultate in se jih uporabi za končno evalvacijo rezultatov.

Language	Our approach	Baselines	
		Moreau	Stamatatos
Dutch	**75**	69.38	72
English	**73.33**	66.67	66.67
Greek	65.42	56.79	**66.07**
Spanish	**72**	71.67	70
German	**79.29**	77	70.50

Slika 4: Rezultati izvajanja podani kot mediana točnosti na posamezen jezik. Najboljši razultati na posamezen jezik so odebeljeni.

Evalvacija rezultatov je potekala s tremi metodami, rezultati katerih so prikazani na Sliki 5 in Sliki 4 . Prva je osrednja nit članka, preostali dve pa sta konkurenčni metodi. Med metodami ni očitnega zmagovalca (metoda, ki bi delovala dobro za vse korpuse in jezike), vendar pa je opisana metoda v povprečju prinesla 5% boljše rezultate od konkurence in se v povprečju ponašala s 75% uspešnostjo (mediana). Nova metoda deluje dobro pri nestrokovnih besedilih (novice), prav tako dobro pa se obnese tudi na strokovni literaturi. Ugotovljeni so bili presenetljivo dobri rezultati za nemški jezik, saj učna množica ni vsebovala vseh žanrov

besedila. Uspešnost gre lahko pripisati morfološki raznolikosti nemškega jezika (npr. nemščina ima mnogo sestavljenih besed, kar pomeni, da lahko povemo več z istem številom besed kot pri kakšnem drugem jeziku). Grški jezik se je obnesel najslabše - avtorji želijo raziskati vzrok temu v prihodnjih člankih. V določenih korpusih (TeNL-PAN14re, TeUK-PAN14es, TeUK-PAN14no) so se vse metode obnesle slabše. Po pregledu teh korpusov so avtorji ugotovili, da za vsak problem v povprečju obstaja samo en znan dokument in število besed v teh dokumentih je povprečno 116.3. To je premalo podatkov za tako statistično metodo. V drugih primerih so avtorji opazili, da se algoritem obnese slabše, če je stil jezika znanih dokumentov drugačen od neznanega dokumenta (npr. znani dokumenti izključno vsebujejo strokovna besedila z nenavadnimi besedami ter neznani dokumenti recepte ali kaj podobnega). Metoda je uspešnejša, če so besedila daljša in različnega žanra. Tako lahko izluščimo mnogo več kvalitetnih značilk.

	Corpus	Our approach	Baselines	
			Moreau	Stamatatos
Dutch	TeNL-PAN14es	75	68.75	**89.58**
	TeNL-PAN14re	53	50	**57**
	TeNL-RadarV	77.5	**82.50**	67.50
	TeNL-Trouw	75	70	**76.50**
English	TeUK-Drexel	77.27	63.64	**81.82**
	TeUK-KoppelBlog	**73**	69.65	69.85
	TeUK-PAN13	**73.33**	66.67	63.33
	TeUK-PAN14es	**58**	48	55
	TeUK-PAN14no	**58.5**	50	54.50
	TeUK-Reddit	**75.33**	70	66.67
	TeUK-Telegraph	77	**84**	78
Greek	TeGR-PAN13	63.33	**73.33**	70
	TeGR-PAN14	**65.83**	48	65
	TrGR-Sintagesparea	**68.57**	55	55
	TeGR-Tovima	65	58.57	**67.14**
Spanish	TeES-ElPais	84.5	**85**	82
	TeES-Ocio	60	**71.67**	70
	TeES-PAN13	**76**	**76**	60
	TeES-PAN14	**72**	52	62
	TeES-Rankia	70	61	**81**
German	TeDE-Amazon	67	57	**70**
	TeDE-CT	76	74	**78**
	TeDE-D120	**82.5**	80	60
	TeDE-Gutenberg	**80**	73	71
	TeDE-Mails	83.33	**87.50**	75
	TeDE-Recht	**81.25**	**81.25**	68.75
	TeDE-Thesen	78.57	**92.86**	57.14
	TeDE-Zeit	76	73.50	**77**
Median Accuracy		**75**	70	69.3

Slika 5: Evalvacija rezultatov metode AV avtorjev izvornega članka [3] v primerjavi z dvema konkurenčnima, glede na 28 testnih korpusov. Najboljši rezultati glede na korpus so označeni odebeljeno.

6. ZAKLJUČEK

Članek predstavlja enostavno in učinkovito metodo potrjevanja avtorstva besedil. Testirana je bila na 28-ih korpusih, ki skupno zajemajo kar 4525 primerov besedil. Njena uspešnost je bila v mediani 75%, v povprečju pa je bila boljša za 5% glede na dosedanje najboljše metode.

Poskusili smo tudi implementirati algoritem v članku[2], vendar ga nam ni uspelo dokončati.

Predstavljena metoda generira kompakten in transparenten model, ki je lahko razširljiv z novimi jeziki ali novimi kategorijami značilk. Model ne potrebuje ročnega nastavljanja praga potrjevanja avtorstva, saj to stori samodejno. Pomembna prednost metode je, da ne potrebuje dodatnega procesiranja naravnega jezika, niti zunanjih storitev. Ima nizko računsko zahtevnost (v povprečju nekaj milisekund na problem). Metoda omogoča razširitve oz. izboljšave. Avtorji vidijo prostor v optimizaciji parametrov tekom učenja na podlagi mrežnega sistema. V načrtu je tudi raziskovanje kako nastaviti parametre, ki bodo optimalni za vse jezike.

7. Literatura

[1] S. Afroz, A. C. Islam, A. Stolerman, R. Greenstadt, and D. McCoy. Doppelgänger finder: Taking stylometry to the underground. In *Security and Privacy (SP), 2014 IEEE Symposium on*, pages 212–226. IEEE, 2014.

[2] J. Fréry, C. Largeron, and M. Juganaru-Mathieu. Ujm at clef in author identification. In *Working Notes for CLEF 2014 Conference*, pages 1042–1048, 2014.

[3] O. Halvani, C. Winter, and A. Pflug. Authorship verification for different languages, genres and topics. *Digital Investigation*, 16, Supplement:S33 – S43, 2016. {DFRWS} 2016 EuropeProceedings of the Third Annual {DFRWS} Europe.

[4] M. Khonji and Y. Iraqi. A slightly-modified gi-based author-verifier with lots of features (asgalf). *CLEF (Working Notes)*, 1180:977–983, 2014.

[5] W. Y. Koppel M. Determining if two documents are by the same author. *JASIST*, pages 178–87, 2014.

[6] E. Moreau, A. Jayapal, and C. Vogel. Author verification: Exploring a large set of parameters using a genetic algorithm-notebook for pan at clef 2014. In *Working Notes for CLEF 2014 Conference*, volume 1180, page 12. CEUR Workshop Proceedings, 2014.

[7] J. Mothe, J. Savoy, J. Kamps, K. Pinel-Sauvagnat, G. Jones, E. San Juan, L. Cappellato, and N. Ferro. Experimental ir meets multilinguality, multimodality, and interaction. In *Sixth International Conference of the CLEF Association, CLEF*, volume 15, pages 8–11. Springer, 2015.

[8] S. Seidman. Authorship verification using the impostors method. In *CLEF 2013 Evaluation Labs and Workshop–Working Notes Papers*, pages 23–26. Citeseer, 2013.

[9] E. Stamatatos. A survey of modern authorship attribution methods. *Journal of the American Society for information Science and Technology*, 60(3):538–556, 2009.

[10] E. Stamatatos, W. Daelemans, B. Verhoeven, P. Juola, A. López-López, M. Potthast, and B. Stein. Overview of the author identification task at pan 2014. In *CLEF (Working Notes)*, pages 877–897, 2014.

[11] B. Stein, N. Lipka, and S. M. zu Eissen. Meta analysis within authorship verification. In *Database and Expert*

[2]Koda naše implementacije je dostopna na https://github.com/jbargu/halvani_dfrws2016

Systems Application, 2008. DEXA'08. 19th International Workshop on, pages 34–39. IEEE, 2008.

Digital Forensics as a Service: an update

Simon Grivc
Fakulteta za računalništvo in informatiko, Univerza v Ljubljani
simongrivc@gmail.com

Uroš Prosenik
Fakulteta za računalništvo in informatiko, Univerza v Ljubljani
uros.prosenik@gmail.com

Klemen Turšič
Fakulteta za računalništvo in informatiko, Univerza v Ljubljani
tursic.klemen@gmail.com

ABSTRACT

V današnjih časih, ko količina podatkov strmo narašča, se v računalniški forenziki pojavljajo vedno novi izzivi. Kako implementirati učinkovito centralizirano storitev, katera bi sprostila forenzike ter omogočila, da svojo preiskavo opravijo bolj učinkovito in podrobno. S tem problemom se ukvarjajo avtorji članka, kateri opisujejo kako na Nizozemskem rešujejo ta problem s sistemom Xiraf ter njegovim naslednikom sistemom Hansken.

Keywords

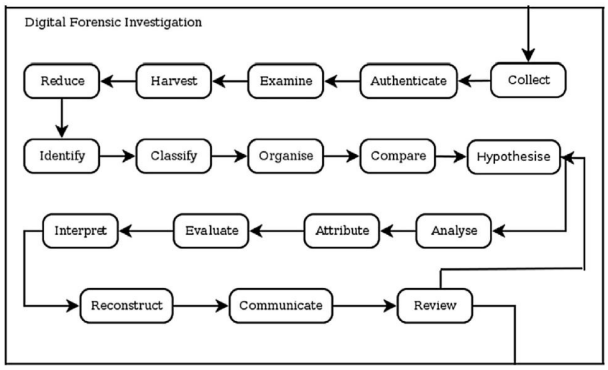
Digitalna forenzika, DFaaS, Digitalna forenzična preiskava, Procesni model, Xiraf

1. UVOD

Dandanes si je nemogoče predstavljati življenje brez digitalnih naprav, saj nas spremljajo na vsakem koraku. Potrebno se je zavedati, da vse te naprave nekje hranijo oziroma puščajo digitalne sledi. To prinaša potrebo po novih načinih forenzičnih preiskav. Digitalne naprave so velikokrat uporabljene oziroma soudeležene v kriminalnih dejanjih, kjer lahko naprave uporabimo kot dokazna gradiva proti storilcu dejanja. Analiza in prepoznava digitalnih podatkov sta tako postali nepogrešljivi.

Od decembra leta 2010 se na Nizozemskem uporablja nov procesni model digitalna forenzika kot storitev (angl. Digital Forensics as a Service - DFaaS). Po treh letih uporabe je ta procesni model postal standard za preiskavo stotine drugih forenzičnih preiskav. Ta model prav tako rešuje veliko problemov ozkega grla. Kljub temu, da model ni standardiziran, pa ga abstraktni dogovor digitalnih forenzičnih modelov vseeno sprejema.

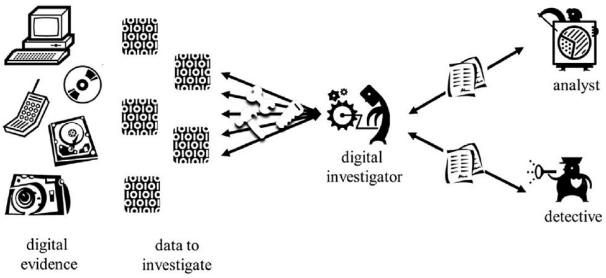
V zadnjem predlogu Kohn (2013) [11] predstavlja model višje ravni, sestavljen iz šestih korakov: dokumentiranje, priprava, incident, odziv na incident, digitalna forenzična preiskava in analiza. Leta 2003 Carrier in Spafford [7] predstavita preiskovalni model, kjer definirata podobnih pet skupin: pripravljenost, uvajanje, fizično preiskavo zločina, preiskavo digitalnega prizorišča zločina in analizo. Casey leta 2011 [9] te korake posploši kot priprava, raziskava oziroma prepoznavanje, ohranjanje in pregled oziroma analiza.



Slika 1: Digitalna forenzična preiskava (Kohn, 2013).

Poleg forenzičnih preiskav obstaja eDiscovery. Gre za elektronski vidik prepoznavanja, zbiranja in proizvodnjo elektronsko shranjenih podatkov (ESI) v odgovoru na zahtevo tožbe ali preiskave. Njegovi koraki so podobni korakom omenjenim v prejšnjem odstavku. Chrisholm leta 2010 [6] razlaga, da je glavna razlika med njima njun obseg dela. Vodilni model pri eDiscovery je Electronic Discovery Reference Model (EDRM).

Članek se osredotoča na pregled digitalnih sledi, definiran iz strani različnih avtorjev, kot proces digitalne forenzične preiskave (Kohn), digitalna preiskava mesta zločina (Carrier) [8] oziroma kot pregled ali analiza (Casey). Opisali bomo tudi, kako je proces digitalne forenzike implementiran na Nizozemskem. Pri tem se uporablja integrirani digitalni forenzični procesni model (IDFPM) in terminologija, katero opisuje Kohn, seveda pa lahko namesto njega implementiramo kateri koli drugi model. Preiskovalni digitalni forenzični model (IDFPM) je predstavljen na sliki 1.



Slika 2: Traditional digital forensics process.

1.1 Tehnične implementacije

Veliko forenzičnih analitičnih sistemov naslednjih generacij je v razvoju ali pa so že implementirane. Glavni cilj teh sistemov je pohitritev in avtomatizacija indeksiranja slik, to pa je prvi korak pri digitalni forenziki kot servis (Digital Forensics as a Service - DFaaS).

Leta 2004 sta v članku Roussev and Richard [15] opisala porazdelitveni procesorski sistem, kateri je veliko hitrejši kot AccessData FTK. Ta je bil izdelan v laboratoriju. Od takrat je FTK 3 s podporo do štirih usposobljenih delavcev uspešno avtomatsko paralelno procesiral podatke. Raziskavo nad avtomatskem procesiranju zaseženih podatkov je izvedel Alink leta 2006 [1]. Ayers (2009) [2] je podal potrebo po takšnem sistemu in zapisal zahteve, katere naj bi tak sistem moral oziroma lahko pokrival. Leta 2012 je Bhoedjang [5] obrazložil, kako je bil razvit in kako se uporablja sistem Xiraf na Nizozemskem. Predlog za izgradnjo DFaaS sistema so podali Lee in Un (2012) [12]. Osredotočili so se na hitrost in končnemu uporabniku ponudili internetni vmesnik za iskanje podatkov.

Veliko je že obstoječih orodij, ki podpirajo eDiscovery tako v zakonodaji izvrševanja kot v podjetjih. Primer sta ZyLAB eDiscovery OnDemand in Symantec eDiscovery Platform, katera poganja Clarewell.

1.2 Pričakovan razvoj

Richard, Russev (2006) [14] in Garfinkel (2010) [10] opisujejo številne možnosti razvoja. Nekateri, kot recimo deljeno procesiranje, so že v razvoju. Garfinkel pričakuje krizo v digitalni forenziki, če se ne najde učinkovitejša metoda analiziranja celotnega forenzičnega materiala. Razlogi za to so povečevanje količine podatkov, šifriranje in širjenje operacijskih sistemov in datotečnih sistemov.

2. TRADICIONALNI PROCES PREISKAVE

Slika 2 prikazuje tradicionalni proces digitalne preiskave na Nizozemskem. Na levi strani vidimo digitalne naprave, na desni strani pa detektive ter analitike, kateri se zanimajo za informacije na napravah.

Odvisno od tipa kriminalistične preiskave, se v preiskavo skoraj vedno vključi tudi digitalni preiskovalec. Primer se lahko začne na več načinov. Na primer: žrtev prijavi kriminalno dejanje, policija je poslana na prizorišče zločina, sistem za zaznavanje vlomov se sproži in sporoči vdor, oziroma na kakršen koli drug način. Temu v IDFPM pravimo proces incidenta ter posledični odzivni proces. Vsa tradicionalna kriminalna dejanja kot so požig, umor, digitalni kriminal, ponavadi vsebujejo digitalno komponento. Detektivi ponavadi nimajo oziroma imajo omejeno znanje v ravnanju z digitalnimi napravami. Zato vključijo v kriminalistično preiskavo digitalnega preiskovalca, kateri odgovori na specifična vprašanja glede digitalnih naprav. Digitalni preiskovalec nima podatkov o posameznem primeru, saj ni bil vključen v preiskavo od začetka, ni prebral nobenih izjav ter ni sodeloval na nobenih zaslišanjih. Njegova koncentracija je na vprašanjih katera mu zastavi detektiv ter na standardnih postopkih. Postopek nalog je na splošno enak in si sledi nekako tako. Prva naloga je izdelava forenzičnih kopij digitalne naprave (zbiranje in preverjanje pristnosti). Po tem z različnimi orodji pregledamo ter poskušamo obnoviti zbrisane datoteke, nedodeljen prostor, zapakirane arhive itd. Naslednja koraka sta indeksiranje podatkov in strukturiranje podatkov v logične formate. Indeksiranje lahko pomeni več stvari, od kreiranja indeksov ključnih besed, shranjevanja prepoznanih časovnih značk v podatkovni bazi do pridobivanja metapodatkov za kasnejšo analizo. Zmanjševanje količine podatkov za analizo je mogoče z odstranjevanjem poznanih datotek iz zgoščene tabele parov.

Običajna praksa v kriminalističnih preiskavah je, da detektivi formulirajo vprašanja glede na ne digitalno informacijo. Ta vprašanja so nato podana digitalnim preiskovalcem. Ponavadi ta vprašanja tvorijo implicitno hipotezo. Če je postavljeno vprašanje, da je potrebno pridobiti vsa email sporočila poslana na določen datum je hipoteza lahko, da je nekdo kontaktiral nekoga in, da je pri tem uporabil email za potrebe komunikacije. Digitalni preiskovalec lahko pomaga pri formuliranju teh hipotez ter pomaga pri testiranju hipoteze. Za odgovor na vprašanje oziroma testiranje hipoteze je potrebno analizirati digitalne dokaze. To lahko storimo na več različnih načinov, na primer z iskanjem po ključnih besedah, ročnim iskanjem po zgodovini spletnega brskalnika ali pa zaženemo ukaz, ki išče v vseh pogovorih na spletu. Običajne metode komunikacije so tiskanje relevantnih informacij, nalaganje informacij na prenosni medij oziroma centralno skladišče ali nalaganje rezultatov na drug sistem, ki lahko upravlja z informacijami. Odvisno od tipa vprašanja, lahko pridemo do veliko rezultatov. Detektiv mora pregledati vse te rezultate preden lahko najde relevantne informacije oziroma vpraša novo pod vprašanje katero zmanjša število informacij. To pomeni, da je IDFPM model bolj zanka kot ravna linija z občasnimi cikli. Na koncu mora biti izdelano uradno poročilo, katerega lahko predstavimo sodišču oziroma vodstvu. Včasih tudi v tem delu pride do novih vprašanj, kar povzroči potrebo po ponovitvi določenega dela preiskave oziroma dodatno iskanje dokazov. To lahko izvedejo detektivi kateri so delali na primeru, drugi detektivi, digitalni preiskovalci ali pa sodni izvedenci.

3. ANALIZA TRADICIONALNEGA PROCESA

Na splošno je proces, opisan v prejšnjem odstavku odvisen od števila dejavnikov. Avtorji navajajo, da so v številnih letih uporabljanja DFaaS sistema nad več stotimi kriminalističnimi preiskavami ugotovili, da število dejavnikov poglavitno vpliva na učinkovitost preiskave. V nadaljevanju bomo opisali upravljanje s sredstvi, tipe vprašanj, zahtevane časovne okvirje, sodelovanje in raziskovanje ter deljenje znanja.

3.1 Upravljanje s sredstvi

Digitalni preiskovalci so ponavadi odgovorni za svoje raziskovalno okolje. To se nanaša na shrambo podatkov, varnostne kopije, omrežje, programe, varnost in veliko drugih nalog, katere pridejo s sistemsko administracijo. Čeprav to ni njihova glavna naloga, se lahko počutijo (in so lahko) odgovorni, če pride do napake. Na splošno niso opremljeni za administracijo preiskovalnega okolja. To lahko vodi do varnostnih vdorov, nedelujočih sistemov za varnostne kopije, nedelovanje samega sistema zaradi nepravilne namestitve, uporaba zastarelega programja in več drugih stvari. To seveda vodi do tega, da se veliko preveč časa porabi za opravljanje administrativnih nalog in manj časa za izvajanje digitalne preiskave.

Od digitalnih preiskovalcev se pogosto zahteva, da so strokovnjaki na večih področjih. Tako se od njih poleg sistemske administracije pričakuje tudi znanje iz kreiranja slik diskov, izvajanje nalog v odzivu na incident, odkrivanje podatkov iz podatkovnih odložišč, zajemanje omrežnega prometa, izvajanje analize, preiskava neznanih formatov datotek itd. Kljub temu, da obstajajo digitalni preiskovalci, katerim uspe združiti vse to znanje, so le ti ponavadi preobremenjeni in ne uspejo uspešno voditi preiskave zaradi pomanjkanja časa. Tukaj se avtorji sprašujejo, ali je res učinkovito imeti nekoga, ki zna hkrati kreirati slike diskov ter izvesti obratni inženiring na datotečnem sistemu.

3.2 Vprašanja

Na splošno poznamo tri tipe vprašanj, katere lahko postavimo v digitalni preiskavi. Prvi tip v bistvu sploh ni vprašanje ampak bolj zahteva po materialu. Na primer: "Dajte mi vse informacije o komunikaciji.", "Poiščite vse zadetke v zvezi z imenom Peter.", "Hočem vse podatke v zvezi z drogami.". Ta vprašanja ponavadi povzročijo veliko dela na strani digitalnega preiskovalca, predvsem zato, ker so zahteve presplošne. Kaj je definirano s komunikacijo? Sporočila, telefonski klici, tekstovna sporočila, vse to lahko spada pod komunikacije. Se informacije v zvezi z drogami navezujejo tudi na informacije o kontaktih, bančnih izpiskih in slike dragih avtomobilov [13]? Naslednji tip vprašanja je nekje v sredini. Še vedno je presplošen, vendar bolj specifičen kot prvi tip vprašanja. Na primer: "Kaj je iskal obtoženi?", "Od kje izvira ta dokument?". Ta vprašanja delujejo bolj detajlna vendar še vedno puščajo preiskovalcu veliko dela. Kako so definirana iskanja? Je to iskanje preko spleta ali lokalno na napravi? Zadnji tip vprašanja je zelo specifičen. Detektiv ima hipotezo, katero hoče preveriti in vpraša vprašanje kot recimo: "S kom si je uporabnik dopisoval dvajsetega marca?", "So te slike bile posnete z digitalno kamero?" itd. Ta vprašanja so zelo specifična in dajejo preiskovalcu dobro idejo kaj iščejo oziroma kaj je njihova naloga. Problem pri tem tipu vprašanja je ta, da potegne preiskovalca v nekakšen tunel. Recimo kaj, če je bilo iskano sporočilo poslano en dan prej ali pa kasneje. Kot smo že prej omenili, digitalni preiskovalec nima znanja iz celotne preiskave zato lahko kljub temu, da odgovori na vprašanje zgreši ključne dokaze.

3.3 Časovni okvirji

Eden od problemov tradicionalnega modela so tudi časovni okvirji. V prvih nekaj dneh od preiskave se oblikujejo hipoteze ter sledi [16] [17]. Ker rezultati digitalne preiskave ponavadi še niso dostopni tako hitro, le tej niso upoštevani pri oblikovanju hipotez in sledi. Rast števila digitalnih preiskovalcev je zelo omejena medtem pa se rast v številu digitalnih naprav, velikosti podatkovnih medijev ter števila primerov z digitalno komponento veliko hitreje povečuje. Ko se digitalni preiskovalec loti primera, so prvi koraki običajno isti. Ustvarjanje forenzične kopije, pridobivanje izbrisanih datotek, pridobivanje zgodovine brskanja po internetu itd. To je eden od razlogov zakaj so digitalni preiskovalci vedno zaposleni. Glede na to, da naslednji primer že čaka na obravnavo, praktično ni časa za inovativnost pri deljenju znanja. Digitalni preiskovalec je prezaposlen z drugimi obveznostmi, kar vodi do tega, da se primeri pri katerih je digitalna tehnologija zelo pomembna ne pregledajo ustrezno oziroma se sploh ne pregledajo.

3.4 Sodelovanje

Trenutni postopek ustvari zelo slabe pogoje za sodelovanje. Če digitalni preiskovalec poroča o celotnih rezultatih se to poročilo porazdeli na več različnih naprav. To lahko storimo na več načinov, na primer: večje število strani, glede na naslovnika oziroma na nalogo. Detektivi lahko nato individualno delajo na svojem delu naloge. Pri tem pogosto pride do prekrivanj saj lahko recimo nekdo, ki išče sledi v elektronski pošti spregleda sled, katera se navezuje na komunikacijo preko facebooka. Prav tako kot pri natisnjeni mapi, delitev dokazov glede na kompleksnost materiala zelo oteži možnost ohranjanja pregleda nad dokazi.

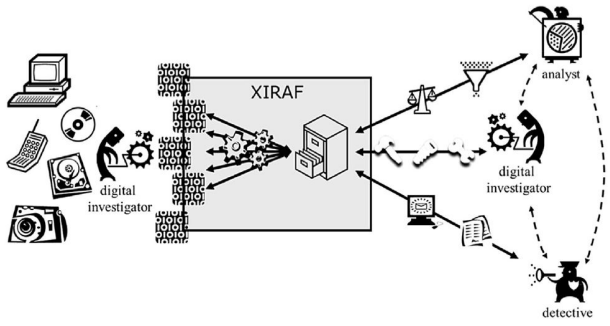
3.5 Raziskovanje, razvoj in širjenje znanja

Digitalni preiskovalci izvajajo preiskavo glede na primer. Na primer iskanje pomena določenih časovnih okvirjev. To informacijo nato posredujejo preiskovalni ekipi, katera uporabi informacijo v primeru. Isti rezultati se seveda lahko uporabijo v različnih primerih, vendar le če se primera povezujeta. Če se pojavi podoben primer preiskave, vendar digitalni preiskovalec za ta primer ne ve, ker se mogoče izvaja v drugem delu države se njegovo znanje izgubi in ni uporabljeno na drugem podobnem primeru. Vsi digitalni preiskovalci morajo slediti novim smernicam v njihovi stroki. Kot že omenjeno v članku so digitalni preiskovalci pogosto preobremenjeni in nimajo vedno časa obnavljati svojega znanja pravočasno.

4. DIGITALNA FORENZIKA KOT STORITEV

Od decembra 2010 se za procesiranje in preiskavo velikih količin zajetega digitalnega materiala uporablja nov storitveni pristop: Digital Forensics as a Service (DFaaS). Servis temelji na zaprtokodnem nekomercialnem produktu Xiraf. Razvit je bil na Inštitutu za Forenziko na Nizozemskem.

Slika 3 prikazuje proces, kako so forenzični primeri obravnavani z uporabo takšnega pristopa. Na desni strani še vedno opazimo detektive in analitike, kateri imajo vprašanja, nanašajoča se na digitalni material kateri se nahaja na levi. Da zagotovimo forenzično celovitost so slike še vedno potrebne. Tako kot pri tradicionalnem procesu je najprej potrebno narediti ustrezno kopijo elektronskih naprav (zajem in preverjanje pristnosti). Glavna razlika je v tem, da se slike kopirajo v glavno odložišče, pregledajo z uporabo standardnih orodij, katera izvlečejo datotečne sisteme, datoteke iz nedodeljenih prostorov, do orodij, ki razčlenijo zapiske



Slika 3: Digital forensics as a service.

oz. beležke, internetno zgodovino in bazo elektronske pošte. Rezultati teh orodij se shranijo v glavnem odložišču. Po shranjenih sledeh lahko iščemo (zmanjšamo obseg, analiziramo) s sledečimi metodami: detektivi se lahko prijavijo v sistem preko spletnega brskalnika, digitalni preiskovalci lahko uporabljajo programski vmesnik v katerem poganjajo avtomatizirana orodja, analitiki pa brskajo po informacijah in analizah z uporabo orodij za vizualizacijo podatkov, vnos sledi ali pa zgradijo mrežo kontaktov. To omogoča opredelitev, identifikacijo, organizacijo in primerjanja podatkov v le nekaj sekundah - odvisno od hipoteze in vprašanja, katere si zastavi preiskovalec. Poizvedbe se lahko izvajajo kadar koli med forenzično preiskavo.

V večini primerov je detektivom brez digitalnega znanja oteženo ovrednotiti, interpretirati in rekonstruirati digitalnih sledi. Za takšne naloge mora razumeti kako in zakaj določena sled obstaja in kateri dogodki lahko vodijo do podane zbirke digitalnih sledi. V primeru odkritja pomembne sledi je ponavadi dobra praksa, da detektiv poišče pomoč pri digitalnem preiskovalcu, kateri lahko detektivu pomaga z odgovori. Za to je potrebna obojestranska komunikacija.

5. ANALIZA DIGITALNE FORENZIKE KOT STORITEV

Nekateri faktorji imajo zelo velik vpliv na učinkovitost procesa digitalne forenzike. Naslednje omenjeni so gledani z zornega kota procesnega modela DFaaS.

5.1 Upravljanje z sredstvi

Administracijo DFaaS izvaja skupina serviserjev. To so sistemski administratorji brez znanja digitalne forenzike. Lahko nalagajo slike na centralno skladišče, indeksirajo slike, podajajo uporabnikom pravice do preiskovalnih primerov in ostala sistemsko administrativna dela. Med indeksiranjem se iz digitalnih naprav kopirajo vsi meta podatki, vključno s časovnimi značkami in se iz njih naredi indeksiranje po ključnih besedah.

Poleg skupine sistemskih administratorjev so tu tudi administratorji aplikacij, podatkovnih baz, infrastruktur in administratorji za ostala dela, katera so ločena od digitalne forenzike, a so potrebna za vzdrževanje, optimizacijo sistemov, prepreko izgube podatkov in nasploh za omogočanje celotne storitve.

S centralizacijo programske opreme poskrbimo, da lahko vsako ne uporabljeno kapaciteto, prostor ali procesorsko moč, uporablja drug uporabnik. Če je preiskava na enem delu države zahtevala veliko prostora ali procesorske moči, so se te kapacitete kupile, kljub temu, da jo je imel drugi oddelek zadosti, saj ni bilo mehanizma za deljenje. S storitvenim modelom pa so na voljo centralizirane kapacitete, katere se lahko deli med vsemi preiskovalci.

Z centralizacijo vseh podatkov je potrebno sistem za varnostne kopije implementirati samo enkrat, prav tako pa tudi varnostni sistem. Še boljše. Na starih modelih v primeru, da je več oddelkov izvajalo preiskavo nad istimi podatki, so se po nepotrebnem delale kopije istih podatkov, nad katerimi so se izvedle analize. S procesnim modelom pa se samo detektivu doda pravice dostopa do primera, kateri se nato razreši v minutah. Prihranijo se dnevi, celo tedni.

5.2 Vprašanja

Ko ima detektiv omogočen dostop s poizvedbami neposredno do digitalnega materiala, lahko uporabi svoje znanje o primeru oziroma iz svojega strokovnega področja. Prav tako lahko prepozna sled, ki potrdi ali ovrže katero od hipotez. Zaradi njegovega strokovnega znanja na določenem področju primera, lahko sled prepozna hitreje kot digitalni preiskovalec. Recimo: detektiv, kateri želi izvedeti, ali je oseba neupravičeno dobila socialno podporo, lahko preveri sliko, ali je njegov osumljenec odšel v tujino na počitnice. Detektiv, ki izvaja preiskave drog pozna mnogo besed, katere se lahko uporabijo za opis le-teh. Detektiv, ki išče alibi osebe, ki trdi, da je ob določenem času sedela za računalnikom, si lahko v zakup vzame nekaj dodatnih ur, da preveri, ali je morda osumljenec res bil na omenjenem prostoru ob določenem času.

Če detektiv sprašuje vprašanja navezujoča se na digitalno gradivo, pa je to lahko zastavljeno preobširno ali preozko. Nazaj v trenutku dobi rezultate, katere pa z ustreznejšimi iskalnimi nizi filtrira. Iskanje z iskalnimi nizi vrne na tisoče rezultatov, katere je dodatno potrebni omejiti, preden so uporabni. Če preiskovalec gleda specifično obliko komunikacije na določen datum s specifično temo, lahko spremeni katerega od filtrov za pridobitev dodatnih informacij.

5.3 Časovni okvir

Kot omenjeno v oddelku tehnične implementacije, je s procesnim modelom smiselna uporaba porazdelitvene implementacije za zmanjšanje in analizo sledi, pridobljenih iz zasežega materiala. Za posamezni oddelek z nekaj ducatom detektivov morda ni izvedljiva implementacija večnamenskega sistema, kateri je večino časa nedejaven. Na veliki ravni pa je smiselno implementirati centralne sisteme, katere lahko uporablja več oddelkov. S takšnim modelom se pričakuje, da je sistem večino časa dejaven (posredovanje podatkov). Če je digitalni material dosegljiv pred preiskavo, se lahko uporabi za oblikovanje hipotez, namesto, da se uporabi samo za njihovo overjanje.

Uporaba digitalne forenzike kot servis prihrani digitalnim preiskovalcem mnogo časa. Na eni strani zmanjša administrativne naloge, katere niso del njihovih zadolžitev, na drugi strani pa morajo preiskovalne naloge izvesti detektivi, kateri imajo veliko več znanja o primeru. To omogoči digitalnemu

preiskovalcu bolj podrobno preiskavo. Sčasoma to privede do več odkritij kar, da preiskavi bolj stabilen temelj. Če se to novo pridobljeno znanje vključi v servis, sistem prevzame vlogo informacijskega centra. Na ta način shranjeni rezultati novih raziskav pomagajo ostalim preiskovalnim ekipam, ki se ukvarjajo z istim problemom. Digitalni material tako postane bistveni del vsake preiskave

Uporaba procesnega modela omogoči delo detektivom iz pisarniške mize njihovega oddelka. To prihrani nepotrebno porabljen čas in zmanjša potrebo po zapolnitvi neobdelanih ur.

5.4 Sodelovanje

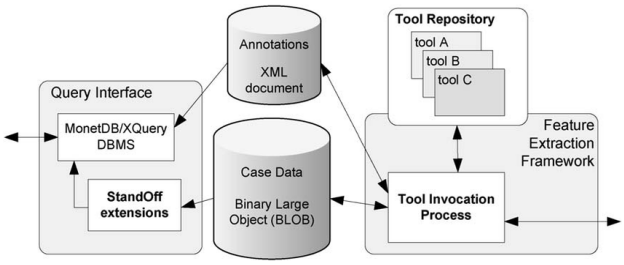
Vsak forenzični analizni sistem mora imeti možnost deljenja napredka na primeru. To pomeni, da ko detektiv najde zanimivo sled, se mora ta zabeležiti v sistemu in je tako vidna drugim detektivom. Na sliki 3 so prikazane povezave med detektivi, digitalnimi preiskovalci in analitiki. Če detektiv naleti na sled, katero ne razume, enostavno povpraša digitalnega preiskovalca za pomoč. Ta lahko detektivu pomaga s pojasnitvijo pomena sledi, vodi dodatno preiskavo ali pa pomaga pri pisanju poročila.

Razdelitev podatkov je enostavnejše s procesnim modelom, kateri omogoča sodelovanje. Preiskovalne teme so ponavadi že razdeljene med detektive in, če je mogoče generično iskati podatke na vseh napravah, ne potrebujejo delitve podatkov glede na velikost in tip naprav. Naj ne bi bilo pomembno, če je bilo elektronsko sporočilo prebrano na mobilni napravi, tablici, namiznem računalniku, kateri uporablja Windows ali pa prenosnik z OS X. Na začetku detektiva zanima, če je sporočilo sploh bilo poslano, komaj kasneje raziskuje, kako je bilo sporočilo poslano. Te tipe poizvedb Xiraf (Bhoedjang 2012) [5] podpira.

5.5 Raziskava, razvoj in deljenje znanja

Pričakujemo specializirane digitalne preiskovalce na enem ali več forenzičnih nalogah. Trenutno so prepoznane tri naloge. Prva naloga je izdelava forenzične slike. Da lahko jamčimo celovitost je ta postopek obvezen. Druga naloga je interpretacija rezultatov, katere so pridobili detektivi, da obrazložimo pomen pridobljenih sledi in kako jih uporabiti v primeru. Tretja naloga je nizkonivojski bitni pregled slik, z namenom iskanja specifičnih informacij, pomembnih za primer. To zadnjo nalogo velikokrat usmerjajo sledi pridobljene iz strani detektivov, za katere je potrebna poglobljena digitalna preiskava.

Z uporabo DFaaS smo zasledili veliko digitalnih forenzičnih oddelkov v zaostanku, kako so si opomogli pri neopravljenih vrstah preiskav. Digitalne preiskave se izvajajo bolj učinkovito, digitalni preiskovalci pa lahko zaradi razbremenitve izvajajo bolj poglobljene preiskave. Kot omenjeno, izvedba preiskave, pa naj jo vodi primer, tehnično vodena, vodena po trendu ali radovednostjo vodena, je lahko vključena v DFaaS. Storitev služi kot informacijski center. Z več kot milijon aplikacij samo v Apple App Store, je ključnega pomena deliti znanje o tem, kako so lahko aplikacije forenzično pregledane (Garginkel, 2010). Enostavno mora biti dodajanje orodij, katera razčlenjujejo forenzične sledi, katere pušča aplikacija in jih poslati v centralizirano okolje. Nova orodja so na voljo vsem hitečim preiskovalcem, posredno pa vplivajo na rezultat raziskave primera oziroma več primerov. Če tudi se rezultati uporabljajo za samo en primer, že samo to, da je aplikacija analizirana, lahko pripelje do rezultata, kateri v obratnem primeru ne bi bil dosežen.



Slika 4: Ogrodje Xiraf sistema.

6. IZKUŠNJE IN DELO V PRIHODNOSTI

Kljub temu, da procesni model ustreza forenzičnim preiskavam in da se je sistem Xiraf v mnogih primerih izkazal za zelo učinkovitega, je še veliko možnosti za izboljšave, ki bi povečale izkušnje ter zmanjšale čas pridobivanja rezultatov. V tem odstavku bomo pogledali možnosti izboljšave.

6.1 Odvečno shranjevanje

Kot smo napisali v odstavku upravljanje s sredstvi v tradicionalni preiskavi, v primeru, da preiskavo vodi več oddelkov hkrati, morajo biti kopije podatkov kreirane in deljene med oddelki. Državna preiskava, kjer morajo imeti vsi oddelki dostop do določenih delov digitalnega materiala posledično povzroči izdelavo velikega števila kopij. Ne le, da za to potrebujemo veliko število shranjevalnih medijev ampak tudi večjo logistiko, ki skrbi za prenašanje teh podatkov, kar lahko privede do občutljivosti na napake. V trenutnem Xiraf postopku se slike, katere so bile zajete na digitalni napravi skopirajo v centralen sistem, iz katerega so nato dostopne vsem, ki želijo in imajo dostop do sistema. Kljub temu, da ta postopek zmanjša število medijev za shranjevanje ter zmanjša število korakov za dostop do podatkov bi lahko ta proces še optimizirali z nalaganjem slik direktno iz digitalne naprave. To se lahko naredi s tako imenovanimi nalagalnimi napravami.

Druga optimizacija iskanja sledi v forenzičnih slikah je lahko ta, da se sledi iščejo že v procesu kreiranja slike diska. Še bolje, postopek analize lahko vpliva na kreiranje slike z določanjem prioritete različnih blokov podatkov. Ob pregledu trdega diska, na katerem je datotečni sistem NTFS se kot prvo analizira MFT (master file table). Če v analizo prioritetno pošljemo MFT mogoče podaljšamo čas kreiranja slike diska, toda s tem zmanjšamo skupen čas analize diska.

6.2 Indeksiranje

Kot je opisano v odstavku povezano delo, je veliko razvoja že šlo v paralelizirano indeksiranje. Edini način, ki bi dodatno paraleliziral ta proces s Xiraf je s pomočjo več strojev za indeksiranje in ločenimi indeksi slik. Po indeksiranju se rezultati združijo v enotno bazo podatkov, tako imenovano objavo. Ta proces je zapleten in zahteva indekse, da je v celoti na voljo preden se lahko poizveduje baza podatkov. Paraleliziranje na Xiraf strežniku zahteva popolno prenovo,

da bi bil prilagodljiv in lahko izvajal poizvedbe medtem ko se indeksiranje še vedno izvaja.

6.3 Dodatne možnosti poizvedb

Xiraf dovoljuje le indeksirane poizvedbe, kar zmanjšuje fleksibilnost. Včasih uporabnik želi postaviti vprašanje, na katero ni mogoče odgovoriti samo s poizvedovanjem podatkov baze. Bolj zapletene poizvedbe, kot so na primer ujemanje polnih besednih zvez, regularni izrazi ali poganjanje dodatnih orodij, je težko izvajati v Xiraf, poleg tega pa je to zelo neučinkovito. Če v sedanjo implementacijo dodamo novo orodje in pridobimo nove rezultate, jih je potrebno najprej shraniti v bazo, potem pa lahko nad njimi izvajamo poizvedbe. Dodajanje novih orodij je zahtevna naloga, saj orodij ni možno avtomatizirati za vsak aktualen primer. Če bi želeli uporabiti ta orodja na vseh aktualnih primerih, bi za delovanje ves čas potrebovali operaterje. To bi bilo potrebno optimizirati, tako da bi lahko avtomatsko poganjali nova orodja na vseh primerih in bi lahko uporabniki sami ustvarjali poizvedbe kot je iskanje z regularnimi izrazi ali pa uporabljali orodja za procesiranje slik.

6.4 Slabosti programa kot storitve

Iste slabosti modela programa kot storitve se aplicirajo na DFaaS model. Nekatere slabosti so zakasnitve, odvisnost od internetne povezave in shranjevanje podatkov v drugih aplikacijah. Katera koli DFaaS implementacija mora poskrbeti, da je za te slabosti ustrezno poskrbljeno.

6.5 Varnost in zasebnost

Xiraf se je začel kot projekt magistrskega dela in zrasel v raziskovalni projekt (Alink e tal., 2006). Nikoli ni bilo mišljeno, da se uporablja v primerih iz resničnega življenja v sedanjem obsegu. Vsi ukrepi, sprejeti za izvajanje varnosti in zasebnosti so postavljeni na vrh trenutne implementacije. Novejše različice Xiraf, vključujejo upravljanje uporabnikov, uporabniške pravice, upravljalno orodje in druge ukrepe, ki se pričakujejo za rešitev. Vsekakor, če je varnost in zasebnost del zasnove, katerekoli kode, procesa ali infrastrukture je potrebno, da upošteva vsa načela. V tehnični zasnovi, dokumentaciji, pregledu kode in zasnovi infrastrukture, mora biti zajamčena varnost in zasebnost vsakega
vpletenega. Dodajanje varnosti in zasebnosti v kasnejši fazi, pusti varnostne luknje in napake. Ukrep varnosti in zasebnosti mora zagotavljati, da nihče nima nikjer neavtoriziranega dostopa, to pa velja tudi za programske razvijalce, sistemske administratorje, morebitne hekerje in čistilce. Kot je opisano v odstavku za Upravljanje z viri, sistemska administracija ni več upravljana s strani digitalnih preiskovalcev, ampak s strani sistemskih administratorjev. To pomeni varnost in zasebnost v zasnovi.

7. UGOTOVITVE

V članku smo analizirali tradicionalen digitalno forenzičen proces in digitalno forenziko kot servis (DFaaS) model. Primerjali smo obe implementaciji, pri tem pa je bil uporabljen digitalno forenzični model (IDFPM, avtor: Kohn (2013)) [3].

Digitalni preiskovalci ne bi smeli biti zadolženi za sistemsko administrativne naloge, saj jih bolje opravljajo namenski sistemski administratorji. V tradicionalnem procesu so digitalni preiskovalci odgovorni prevzemati odgovornost celotne preiskave (shranjevanje, omrežje, programska oprema, varnost, itd). V kombinaciji s svojo osrednjo vlogo pri zagotavljanju in analizi digitalnih dokazov med drugim to vodi v velike administracijske stroške, morebitne kršitve varnosti, napake pri varnostnih kopijah in uporabo zastarele programske opreme. Digitalni preiskovalci so bodisi premalo usposobljeni ali pa preveč izobraženi za vsakodnevne naloge, ki jih opravljajo. V DFaaS na Nizozemskem so se osredotočili na forenzične naloge kot so zaseg materiala in pridobivanje podatkov iz njega. Podatki se pošljejo v centraliziran sistem, ki samodejno izvleče sledi iz podatkov in daje digitalno preiskovalcem in analitikom dostop do sledi. Več administratorjev izvršuje domensko specifične naloge povezane s tem servisom, kot so skrbniki aplikacij, administratorji podatkovnih baz, administratorji za shranjevanje in administratorske infrastrukture. Mogoče bi bilo uporabiti katerokoli od zmogljivosti, ki so trenutno na voljo v isti organizaciji ali celo preko organizacije za obravnavo primerov. Veliko oddelkov ima rezervno shranjevanje in obdelavo zmogljivosti, medtem ko lahko drug oddelek začasno zahteva to sposobnost za obravnavno večjega primera. Navadno ne obstaja način za deljenje teh kapacitet. S centralizacijo pa bi oddelki lahko uporabljali vso procesorsko moč, ki je na voljo za pohitritev procesiranja digitalnega materiala. Mešani oddelki preiskav imajo na voljo eno kopijo predhodno obdelanih podatkov, omogočanje dostopa za druge oddelke pa je lahko urejeno v nekaj minutah. Pomanjkljivost centraliziranega sistema pa je, da morajo vsi podatki biti omogočeni za centralni sistem. Na Nizozemskem trenutno deluje sistem tako, da ustvarijo dodatno kopijo podatkov, nad katero se izvaja lokalna analiza.

Detektivi bi morali biti tisti, ki iščejo digitalno gradivo, saj imajo informacije o primeru, katerih digitalni preiskovalci nimajo. Tako v tradicionalnem procesiranju detektivi pridobivajo podatke od digitalnih preiskovalcev. Pomanjkanje ključnih informacij o primeru pa pomeni težje povezovanje dokazov s primerom. Tako je pogosto digitalni raziskovalec bolj kombajn sledi, kot pa analitik. Čas za preobrat je tako lahko več dni ali celo tednov. Digitalni preiskovalec zagotavlja pridelane sledi (ki so ali pa niso povezane z zahtevano informacijo) detektivom kateri kasneje zmanjšajo sledi na majhen nabor najbolj ustreznih. V storitvenem modelu so vse pridobljene sledi nato dostavljene detektivu. Ti lahko naredijo poizvedbo sledi s filtriranjem in tako izločijo neuporabne informacije v nekaj sekundah. V primeru najdenih sledi imajo direkten dostop do izvornega materiala, kot so slike, dokumenti in email naslovi. Čeprav je res, da naletijo na težave z razumevanjem nekaterih pojmov, kot so izrezljane datoteke ali časovni žig, se jih lahko z izobraževanjem naučijo. Kot opisano s strani Bhoedjan get al., 2012, koristi od nestrokovnjakov odtehtajo tveganja. Digitalni preiskovalci lahko pomagajo detektivom z razlago rezultatov poizvedbe in poglobljeno preiskavo o tehničnih podrobnostih ustreznih sledov [4].

Digitalno gradivo mora biti na voljo v prvih nekaj dneh v preiskavi, tako da ga je mogoče uporabiti za oblikovanje hipotez. V tradicionalnem postopku rezultati digitalne preiskave niso možni v tako hitrem času. Kot rezultat, dokazi najdeni v digitalnem materialu, služijo za oblikovanje hipotez. Zagotavljanje digitalne forenzike kot storitev, pospeši

proces iskanja sledov, zaradi česar detektivi prej dobijo podatke in lahko nadaljujejo s svojim delom preiskave. To skrajša čas in znižuje prag za zapolnitev praznih ur. Če med preiskavo zaidejo v negotovosti, lahko v vsakem trenutku ponovno pregledajo digitalno gradivo.

Sodelovanje med detektivi in digitalnimi preiskovalci je ključnega pomena za razumevanje digitalnega materiala. Pri tradicionalnem postopku, je to težje izvedljivo, saj digitalni preiskovalci poročajo veliko podatkov, ki so poslani več detektivom. Na splošno, namesto materiala se na detektive razdelijo teme preiskave. Model storitev omogoča, da analiziramo vse podatke v zvezi s to temo. Sled, ki je pomembna za eno zadevo, je lahko odločilna pri drugi temi istega primera. Če je detektiv našel zanimivo sled ali sled, ki je ne razume, model storitev omogoča, da zabeleži sledi. Drugi detektivi in digitalni preiskovalci imajo nato dostop do zaznambe in lahko delajo na tem. Sprostitev digitalnega preiskovalca pomeni, da bo lahko le ta opravljal bolj poglobljene raziskave, kar se bo odražalo s pozitivnimi rezultati. Če bi razvili več preiskovalnih metod, bi to pomenilo bistveno zmanjšanje časa porabljenega za digitalne preiskave. Z vključitvijo na novo pridobljenega znanja v storitev, se sistem uporablja kot center znanja. Vsi novi rezultati raziskav shranjenih na ta način, pa pomagajo ostalim preiskovalnim ekipam, ki se borijo s podobnimi težavami. Od decembra 2010, se na Nizozemskem v kombinaciji s Xiraf (Alink e tal. , 2006; Bhoedjang e tal., 2012) uporablja metoda digitalne forenzike kot servis (Digital forensics as a service – DfaaS). Zdaj tri leta kasneje je ta pristop z velikim uspehom postal standard za stotine kazenskih postopkov in več kot tisoč detektivov. Trenutno se razvija naslednik Xiraf-a imenovanem Hansken, kateremu dodajajo nove stvari, katere so se naučili iz serviranja Nizozemskim organom pregona.

Literatura

[1] W. Alink i dr. "XIRAF – XML-based indexing and querying for digital forensics". *Digital Investigation* 3 (rujan 2006), str. 50–58. DOI: 10.1016/j.diin.2006.06.016. URL: https://doi.org/10.1016%2Fj.diin.2006.06.016.

[2] Daniel Ayers. "A second generation computer forensic analysis system". *Digital Investigation* 6 (rujan 2009), S34–S42. DOI: 10.1016/j.diin.2009.06.013. URL: https://doi.org/10.1016%2Fj.diin.2009.06.013.

[3] R.B. van Baar, H.M.A. van Beek i E.J. van Eijk. "Digital Forensics as a Service: A game changer". *Digital Investigation* 11 (svibanj 2014), S54–S62. DOI: 10.1016/j.diin.2014.03.007. URL: https://doi.org/10.1016%2Fj.diin.2014.03.007.

[4] H.M.A. van Beek i dr. "Digital forensics as a service: Game on". *Digital Investigation* 15 (prosinac 2015), str. 20–38. DOI: 10.1016/j.diin.2015.07.004. URL: https://doi.org/10.1016%2Fj.diin.2015.07.004.

[5] R.A.F. Bhoedjang i dr. "Engineering an online computer forensic service". *Digital Investigation* 9.2 (studeni 2012), str. 96–108. DOI: 10.1016/j.diin.2012.10.001. URL: https://doi.org/10.1016%2Fj.diin.2012.10.001.

[6] Chisholm C. "Integrating forensic investigation methodology into eDiscovery". (Jan 2010).

[7] *Carrier B. Autopsy.* URL: http://www.sleuthkit.org/autopsy/;%202003%E2%80%932013..

[8] "Carrier B, Spafford EH, et al. Getting physical with the digital investigation process. Int J Digital Evid". 2 (2003), str. 1–20.

[9] Casey E. "Digital evidence and computer crime: Forensic Science, Computers, and the Internet." 4th ed. Elsevier Science (2011).

[10] Simson L. Garfinkel. "Digital forensics research: The next 10 years". *Digital Investigation* 7 (kolovoz 2010), S64–S73. DOI: 10.1016/j.diin.2010.05.009. URL: https://doi.org/10.1016%2Fj.diin.2010.05.009.

[11] M.D. Kohn, M.M. Eloff i J.H.P. Eloff. "Integrated digital forensic process model". *Computers & Security* 38 (listopad 2013), str. 103–115. DOI: 10.1016/j.cose.2013.05.001. URL: https://doi.org/10.1016%2Fj.cose.2013.05.001.

[12] Jooyoung Lee i Sungyong Un. "Digital forensics as a service: A case study of forensic indexed search". *2012 International Conference on ICT Convergence (ICTC)*. IEEE, listopad 2012. DOI: 10.1109/ictc.2012.6387185. URL: https://doi.org/10.1109%2Fictc.2012.6387185.

[13] Alcindor Y. Leger DL. "Petraeus and Broadwell used common e-mail trick. USA Today". (2012). URL: http://www.usatoday.com/story/tech/2012/11/13/petraeus-broadwell-email/1702057/.

[14] Golden G. Richard i Vassil Roussev. "Next-generation digital forensics". *Communications of the ACM* 49.2 (veljača 2006), str. 76. DOI: 10.1145/1113034.1113074. URL: https://doi.org/10.1145%2F1113034.1113074.

[15] Richard III GG. Roussev V. "Breaking the performance wall: the case for distributed digital forensics. In: Proceedings of the 2004 Digital Forensics Research Workshop". (2004).

[16] Joyce T. "Closing the case: solving violent crimes quickly and efficiently with public records." 79 (2012), str. 50–6.

[17] "U.S. Department of Justice, Office of Justice Programs, Office of Juvenile Justice and Delinquency Prevention. When your child is missing: A family survival guide." (May 1998). URL: http://www.ojjdp.gov/pubs/childismissing/.

Del V
Razno

Avtomatizirano generianje profila za analizo živega Linux pomnilnika *

[Računalniška Forenzika 2016/2017]

Bogdan Golobič
Fakulteta za računalništvo in informatiko
Večna Pot 113
Ljubljana, Slovenija
bg4850@student.uni-lj.si

Staš Hvala
Fakulteta za računalništvo in informatiko
Večna Pot 113
Ljubljana, Slovenija
sh4851@student.uni-lj.si

Timotej Osolin
Fakulteta za računalništvo in informatiko
Večna Pot 113
Ljubljana, Slovenija
to1098@student.uni-lj.si

POVZETEK

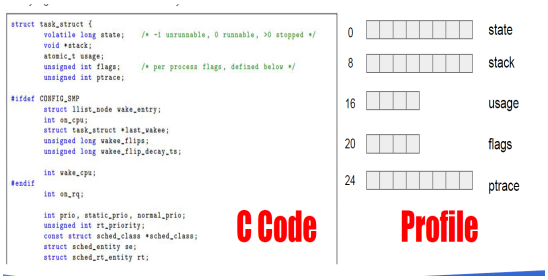
Analiza živega pomnilnika na Linux platformi zaradi narave jedra že od zmeraj predstavljala težavo računalničarjem. Zahteva namreč izredno veliko znanja o sami razporeditvi vsebine pomnilnika, ki ga je običajno veliko lažje pridobiti z razhroščevalnimi simboli generiranimi v času prevajanja programa. Jedro Linuxa je običajno brez razhroščevalnega načina, poleg tega pa je izredno konfigurabilno, kar običajno preprečuje, da bi se informacije o razhroščevanju širile med ostale sisteme, ki si jih ne lastijo. Trenutno je kakršnokoli pridobivanje informacij za odzivne aplikacije na varnostne incidente postalo izjemno zapleten in časovno potraten postopek, kar pomeni, da je tovrsten način v praksi neprimeren. Avtorji članka [6] so razvili orodje z imenom *Layout Expert*, ki omogoča izračun razporeditve vsebine pomnilnika kritičnih struktur jedra med izvajanjem programa brez uporabe dodatnih orodij (npr. prevajalna veriga). Namen njihovega orodja je adaptacija generiranih profilov za Linuxova jedra poljubne verzije, kjer profili označujejo začetne in končne naslove za strukture v pomnilniku. Rezultat je sistemsko specifičen profil z natančno informacijo o postavitvi. Orodje je bilo dodano kot razširitev odprto kodni programski opremi *Rekall* za analizo pomnilnika v forenzičnih preiskavah. V tem članku predstavimo problem izvajanja analize nad pomnilnikom v Linux sistemov, in pregledamo obstoječe rešitve. Opišemo in demonstriramo tudi orodje za analizo na živem pomnilniku (angl. *live memory*).

Ključne Besede

Linux, živi pomnilnik, jedro, Layout Expert, profili

1. UVOD

Analiza pomnilnika postaja vse bolj popularna in efektivna oblika preiskave v kontekstu digitalne forenzike in detekcije zlonamernih programov (angl. *malware*). Cilj analize pomnilnika je zajeti sliko pomnilnika oz. trenutno kopijo fizičnega spomina na delujočem sistemu. S hitrim naraščanjem količine pomnilnika tako v industrijskih kot v domačih računalnikih, postajajo efektivne tehnike za njegovo analizo kritičnega pomena. Na prvi pogled fizični pomnilnik deluje kot ogromna kolekcija podatkov brez logične ureditve, ampak, če vzamemo programsko opremo (angl. *software*) pod drobnogled, lahko opazimo, da je razdeljena v urejene strukture ter sledi neki konvenciji. Razvijalci programske opreme namreč uporabljajo logične konstrukte kot npr. `struct` v jeziku C, da povezane podatke združijo v logične enote. Primer strukture in pripadajočega profila lahko vidimo na sliki 1. Prevajalnik nato poskrbi, da se strukture v pomnilnik shranjujejo konsistentno in generira kodo za dostop do različnih pripadnikov strukture glede na razporeditev.



Slika 1: Prikaz pretvorbe C kode v profil [5].

Profil je model (angl. *template*), ki nam pove kje se vsaka enota v pomnilniku začne in konča. Poleg tega lahko vključuje še [4]:

- **Metapodatke**: podatki kot so na primer ime operacijskega sistema, verzija jedra, verzija projekta (angl. *build number*), itd.
- **Informacije o sistemskih klicih**: indeksi in imena sistemskih klicev,
- **Konstantne vrednosti**: globalne spremenljivke fiksno zapisane v naslove,
- **Osnovne tipi**: nizkonivojski tipi za osnovne jezike

*Članek je povzet po [6]

(običajno C), kar vključuje tudi velikosti celih števil, realnih števil, itd.

- **Sistemske mape**: naslovi kritičnih globalnih spremenljivk in funkcij (samo na Linux in Mac sistemih)

Vsak profil ima unikatno ime, a običajno izvira iz imena operacijskega sistema, verzije, servisnega paketa (angl. *service pack*) ter arhitekture [4]. Torej, vsaka abstraktna struktura katerega koli programskega jezika se prevede v strojno kodo in zasede svoj kos pomnilnika. Ista vrstica kode (angl. *line of code*, LOC) se lahko prevede v različno strojno kodo med različnimi operacijskimi sistemi, verzijami jedra, verzijami prevajalnika, itd.. To pomeni, da so drugače strukturirani tudi profili, kar oteži avtomatizirano analizo le-teh. Da bi uspešno ekstrahirali informacije iz spominske slike (angl. *memory image*), potrebujemo deskriptiven model razporeditve pomnilnika.

V začetkih analize pomnilnika je bilo potrebno ročno definirati razporeditev memorije, ampak to zaradi prej omenjenih različnosti že dolgo ni več mogoče. Prav tako bi bilo nesmiselno sestavljati ureditev pomnilnika iz izvorne kode operacijskega sistema, ker se prevajalniki konstantno posodabljajo in se izhodna strojna koda spreminja. Primer razlike med prevajalniki je zapolnjevanje bajtov (angl. *padding*) različnih elementov v strukturi, da zagotovijo poravnanost (angl. *alignment*) v pomnilniku. To npr. pri jeziku *C* ni definirano v nobenem standardu in se implementacija močno razlikuje med prevajalniki.

Zaradi podpore razhroščevalnim orodjem so prevajalniki začeli izpisovati razporeditve vsake uporabljene strukture v pomnilniku z uporabo razhroščevalnih tokov (angl. *debugging streams*) kot so PDB (*Program DataBase*) datoteke in DWARF (*Debugging Information Format Committee*) tokovi. Operacijski sistem Microsoft Windows shranjuje razhroščevalne simbole (angl. *debugging symbols*) v eksterne PDB datoteke, ki jih lahko prenesemo iz centralnega simbolnega strežnika. *The Volatility Memory analysis Framework (2014)* je bilo prvo od orodje, ki je podpiralo branje in interpretacijo teh tokov za uspešno restavracijo logične ureditve pomnilnika. Sledili so preprostemu postopku, kjer so iz razhroščevalnih informacij sestavili profile specifične za verzijo operacijskega sistema. Orodje *Rekall (2014)* pa omogoča prenos razhroščevalnih simbolov iz Microsoftovega simbolnega strežnika za poljubno verzijo jedra ter pretvori PDB datoteko v profile za analizo tekočega sistema.

Analiza pomnilnika na Linux sistemih je bolj kompleksna. Prvi problem je, da Linux jedra običajno niso prevedena z razhroščevalnimi informacijami, prav tako pa nimamo na voljo simbolnega strežnika kot pri Windowsih. Za izkoriščanje teh podatkov bi torej morali na novo prevesti dele jedra z razhroščevalnimi zastavicami (angl. *debug flags*), kar predstavlja težavo za uporabnike, ki nimajo potrebnega znanja ali orodij (npr. prevajalnih verig) za ugoditev te zahteve. Drugi problem predstavlja visoka prilagodljivost Linux jedra. Uporabniki lahko namreč pred prevajanjem jedra specifirajo velik nabor konfiguracijski opcij, ki močno vplivajo na razlikovanje med jedri iste verzije. Kako to predstavlja težavo za ureditev pomnilnika je razvidno iz leve strani slike 1, kjer je kar nekaj elementov strukture vgnezednih pod predprocesorske makroje (angl. *preprocessing macros*). To pomeni, da je izgled struktur močno odvisen od uporabljenih opcij za prilagoditev jedra, saj prevajalnik ne rezervira prostora za elemente, ki v strukturi niso definirani zaradi makrojev. Kljub vsem omejitvam na Linux sistemih, je želja po avtomatizaciji analize pomnilnika ostala. V sledečih razdelkih opišemo kako so avtorji članka [6] rešili ta problem z orodjem Layout Expert.

2. ANALIZA POMNILNIKA

V idealnem primeru se orodja za živo analizo pomnilnika zaganjajo iz zunanjih medijev, kot so npr. USB ključi v bralnem načinu ali DVD zgoščenke. Ti imajo nameščena vsa potrebna orodja in dodatke, potrebne za triažo poljubnega Linux sistema. Tako orodje (*Rekall*) je že pogosto uporabljeno v Windows sistemih [3], cilj avtorjev pa je bil izdelava podobnega avtomatiziranega sistema tudi za Linux okolja. Pri izdelavi takega orodja je potrebno upoštevati nekaj omejitev oz. pogojev, in sicer:

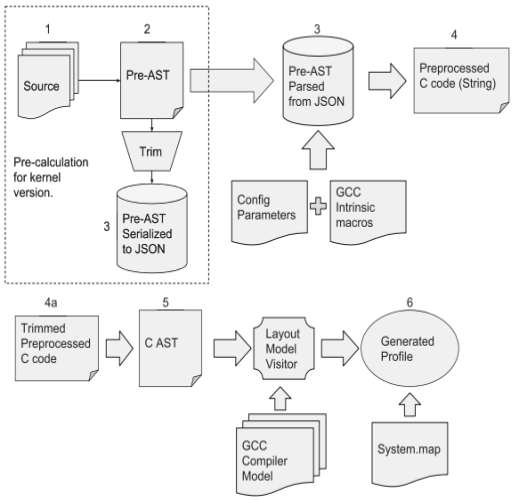
- S stališča uporabnika generiranje prilagojenih profilov za tekoče jedro ni praktično,
- Verzija jedra je znana,
- Konfiguracija ciljnega jedra je znana in posodobljena (pogosto jo najdemo v */boot/* particiji).

2.1 Obstoječe rešitve

Obstoječe rešitve ponujajo t.i. lokalno prevajanje profilov. To pomeni, da potrebujemo celotno orodje, potrebno za prevajanje, nameščeno na ciljnem sistemu. Velikokrat pa tehniki, ki se na varnostne incidente odzovejo, zaradi različnih vzrokov ne morejo nameščati programske opreme na sisteme, še posebej na takšne, ki so bili kompromitirani. To težavo rešujejo tako, da se podatki o ciljnem sistemu prenesejo na oddaljen strežnik, ki prevede programsko kodo, to pa potem reševalec incidenta prenese nazaj na ciljni sistem. Tak princip sicer deluje, toda potrebuje konstantno mrežno povezavo, poleg tega pa se analiza za čas prenosa in prevajanja ustavi.
Podoben način delovanja uporablja orodje *Redline* [1], ki je na voljo le za Windows sisteme. Na voljo ima več načinov uporabe, med njimi prednjači namestitev na kompromitiran sistem. Možnost je tudi namestitve bralca pomnilnika na izmenljivi medij, toda ker orodje podpira le Windows sisteme, pri tem ni potrebno brati nastavitev jedra ipd., kot je to praksa pri raznolikih Linux sistemih. Namensko razširjeno orodje za analizo Linux sistemov je odprtokodno orodje *Volatility* [2].

2.2 Layout Expert

Avtorji članka tako predlagajo nov način - vse potrebne informacije glede jedra ter prevajanja so na voljo znotraj orodja za analizo pomnilnika. S tem se omogoči orodju, da postane neodvisno od omrežja in je tako na voljo za takojšnjo uporabo. Razvili so orodje, imenovano *Layout Expert*, ki temelji na Pythonu in generira konfiguracijske datoteke za *Rekall* orodje. Zaradi omejitev hitrosti Pythona ter kompleksnosti generiranja profilov je sestavljen iz številnih korakov (slika 2), ki izvorno kodo jedra v jeziku C, pridobljeno iz ciljnega sistema, pripravijo za analizo:



Slika 2: Razporeditev *Layout expert-a*.

2.2.1 Razčlenjevalnik predprocesiranja

Predprocesiranje jezika C je prvi člen v prevajanju C datotek v izvršljive datoteke. Običajno to pomeni pretvorbo v t.i. čisto C kodo, kjer se makri, ukazi include in pogojni ukazi ifdef pretvorijo v primerne ekvivalente. Avtorji so ustvarili svoj razčlenjevalnik predprocesiranja, ki za razliko od generičnega C predprocesorja ne potrebuje vseh konfiguracij. Tako *Layout Expert* predprocesor ustvari t.i. Pre-AST datoteke (Preprocessor Abstract Syntax Tree). Ta se potem v naslednjih korakih s pomočjo takrat znanih konfiguracij zmanjša z obrezovanjem nepotrebnih struktur.

2.2.2 Obrezovanje Pre-AST

Pre-AST datoteka generirana v prejšnjem koraku vsebuje celotno izvorno kodo. Pri generiranju profila tipično potrebujemo le manjše kose kode oz. struktur, definiranih in uporabljenih v jedru. Tako *Layout Expert* uporablja t.i. obrezovalnik Pre-AST datotek. Tako odpadejo vse t.i. inline funkcije, vse nastavitvene vrednosti, ki se ne tičejo ciljnega sistema ipd. S tem se ne izreže uporabne kode, hkrati pa se znatno zmanjša velikost izvorne kode.

2.2.3 Predprocesiranje Pre-AST

V tem koraku se pridobi nastavitve jedra, s čimer se v kombinaciji s prej optimizirano kodo tvori C datoteka, ki ne vsebuje več nobenih parametrov predprocesiranja.

2.2.4 Obrezovanje druge stopnje

Nastala C datoteka je zelo velika in vsebuje vsako podatkovno strukturo uporabljeno v jedru. Kot omenjeno, *Layout Expert* temelji na Pythonu, kjer je kompleksno razčlenjevanje časovno zelo zahtevno. Tako avtorji uporabijo osnovno in hitro obrezovanje tudi na tem nivoju. S pomočjo *Rekall* pomnilniške analize, se v tem koraku izluščijo relevantne strukture. Tako se znantno zmanjša velikost C kode, s tem pa omogoči hitrejše delovanje v naslednjem koraku.

2.2.5 C razčlenjevalnik

Layout Expert uporablja svoj C razčlenjevalnik, ki razčlenjuje le osnovne strukture. V kombinaciji s prejšnjim korakom 2.2.4, ki je porezal le uporabljene strukture, je tako razčlenjevanje hitro. Generira se t.i. C-AST (C Abstract Syntax Tree) datoteka, sorodna Pre-AST datoteki.

2.2.6 Model oblikovanja

C prevajalnik načrtuje pomnilniško strukturo za vse novo definirane strukture glede na številna tako enostavna kot kompleksna pravila. Ker so nekatera izmed teh pravil potrebna za pravilno delovanje sistema, jih je v splošnem mogoče uporabiti za predvidevanje natančne in konsistentne pomnilniške strukture. Ker pa prevajalniki omogočajo uporabnikom tudi metode za optimizacijo in poravnavanje, se ti lahko med seboj delno razlikujejo. Avtorji so tako razvili številne teste, ki preverijo vse robne primere, kjer je struktura težavna za predvidevanje.

2.2.7 Sestava

Po samem procesiranju C-AST datoteke, se izvede še obhod skozi to drevo. Za vsako strukturo se predvidi končna postavitev, končni rezultat pa se shrani v *Rekall* profile v JSON formatu.

3. UPORABA LAYOUT EXPERTA

Za uporabo *Layout managerja* je potreben Python verzije 2.7 (Opomba: ne bo deloval na verziji 3 ali novejši). Za najlažjo namestitev je priporočena uporaba orodja *Pip*:

```
$ pip install rekall-layout-expert
```

Zatem je za analizo pomnilnika potrebno zgraditi *Pre-AST*, ki je specifičen za vsako verzijo jedra (za ta korak niso potrebne konfiguracijske datoteke).
Pri tem se lahko uporabi paket zaglavja jedra (angl. kernel headers package) ali pa celo izvorno kodo jedra. Ni pa potrebe da se prevede to izvorno kodo. Primer uporabe (ta korak v povprečju potrebuje minuto časa):

```
$ layout_tool build_pre_ast --source_file_path
/usr/src/tools/linux/module.c --linux_repository_path
/usr/src/linux-headers-4.2.0-22-generic/
pre_ast_4.2.0-22.json
```

Zdaj pa se lahko uporabi *Layout expert* za izračunavanje razporeditev struktur pomnilnika kritičnih jedrnih. Napisati mu je potrebno pot do sistemske konfiguracijske datoteke in *System.map*:

```
$ layout_tool make_profile --config_file_path
/boot/config-4.2.02.0.smp --system
/boot/System.map-4.2.02.0.smp pre_ast_4.2.0-22.json
profile.json
```

Generirani profil lahko uporabimo naprej v orodju *Rekall* za analizo v potrebe primeru.

```
"task_struct": [6784, {
 "acct_rss_mem1": [1896, ["unsigned long long", {}]],
 "acct_timexpd": [1912, ["unsigned long", {}]],
 "acct_vm_mem1": [1904, ["unsigned long long", {}]],
 "active_mm": [928, ["Pointer", {
     "target": "mm_struct"
  }]],
 "alloc_lock": [1736, ["spinlock", {}]],
 "atomic_flags": [1024, ["unsigned long", {}]],
 "audit_context": [1696, ["Pointer", {
     "target": "audit_context"
  }]],
```

Slika 3: Delni izpis *Rekall* profila ustvarjenega z *Layout expert*.

```
"task_struct": [6784, {
 "acct_rss_mem1": [1896, ["long long unsigned int"]],
 "acct_timexpd": [1912, ["long unsigned int"]],
 "acct_vm_mem1": [1904, ["long long unsigned int"]],
 "active_mm": [928, ["Pointer", {
  "target": "mm_struct",
  }]],
 "alloc_lock": [1736, ["spinlock"]],
 "atomic_flags": [1024, ["long unsigned int"]],
 "audit_context": [1696, ["Pointer", {
  "target": "audit_context",
  }]],
```

Slika 4: Delni izpis *Rekall* profila ustvarjenega s tradicionalnim črpanjem iz DWARF toka.

4. REZULTATI

Layout expert kot orodje lahko ustvari profil razporeditve pomnilnika osrednjih jedrnih strktur želenega sistema Linux platforme. Ta profil lahko potem uporabimo v *Rekall-u* za nadaljnjo analizo.
Za namen prikazovanja natančnosti generiranja profilov živega pomnilnika so avtorji [6] primerjali rezultate *Layout expert-a* in *Rekall-ove* standardne skripte za generiranje profila iz DWARF tokov na Linux jedru verzije 4.2.0 in s privzeto konfiguracijo priloženo z distribucijo Ubuntu 15.10. Pri tem je treba dodati, da je za uporabo *Rekall* orodja potrebno nasneti celotno prevajalno verigo tako za virtualko, kot tudi glave jedra. Za tem so še spremenili konfiguracije tako, da so se podatkovni strukti čim bolj spremenile. V naslednjem koraku so z *Layout expert* generirali *Pre-AST* in ga obrezali z jedrnim izvornim drevesom (angl. kernel source tree). Prav tako so z njegovo pomočjo ustvarili *Rekall* profil, ki je lahko nadaljnjo analiziran z *Rekall-om*. Ta profil in profil generiran s pomočjo tradicionalne DWARF ekstrahcijske metode so potem primerjali s splošnimi vtičnikimi za *Rekall (pslist, module in lsof.* Dobljeni rezultati so prikazani na slikah 3 in 4.

Razvidno je, da so odmiki identični. Razlike so le v imenih osnovnih podatkovnih tipov, ki pa predstavljajo isti tip (npr. na sliki *unsigned long long* in *long long unsigned int*). Še ena razlika, ki pa ni vidna na slikah je, da ima profil generiran s pomočjo DWARF tokov, še dodatne strukte, vendar jih ta *Rekall* ne uporablja, tako da praktične razlike potemtakem ni.

4.1 Pomankljivosti

Eden izmed pomanjkljivosti *Layout expert-a* je, da je napisan v Pythonu. Razlog za to izbiro je lahek dostopa do močnih knjižnic za razčlenjevanje. Vendar pa je veliko počasnejši od GCC prevajalne verige. V praktičnih primerih celotno procesiranje vzame približno 1 minuto časa.
Druga potencialna pomanjkljivost je, da je za delovanje *Layout expert-a* potreben dostop do lokalne konfiguracije jedra in tabele simbolov (System.map). V kolikor bi napadalec karkoli izmed tega izbrisal, orodje ne bi delovalo. Treba je tudi dodati, da to tudi ni bil cilj avtorjev.
Naslednja pomanjkljivost je, da je potrebno za vsako verzijo jedrnega izvornega drevesa narediti Pre-AST datoteko. Vendar v primerjavi z dosedanjimi orodji, datoteka za posamezno verzijo pokriva vse možne konfiguracije.

4.2 Diskusija

Eden izmed večjih problemov, ki ga to orodje reši je neomogočena naprava */proc/kcore*, ki uporabnikom dopušča, da imajo dostop do fizičnega pomnilnika. Tako je nastavljeno v sistemih, kjer je poudarek na varnosti. Da se to napravo zopet omogoči je potrebno naložiti jedrni modul v jedro, ki trenutno teče. Za to so potrebne identične konfiguracije, ki so bile uporabljene za izgrajevanje jedra. Razlog je v tem, da mora jedrni modul komunicirati z jedrom, ki teče, posledično morata biti razporeditvi struktov enaki [7].
Layout expert to reši s tem, da generira pravilno razporeditev struktov med izvajanjem sistema, ki jih nato naložimo v jedro. Zaradi tega je orodje uporabno tudi pri zajemu podatkov in ne samo pri analizi.

5. ZAKLJUČEK

V forenziki in tudi na splošno je analiza pomnilnika časovno potraten proces, zaradi potrebe v naprej prevedenih profilov in gonilnikov za vsak določen sistem posebaj. Nemogoče je predvideti razporeditev pomnilnika jedrnih struktur, brez pomoči prevajalnika in izločenih informacij odpravljanja napak (angl. debug), kar pa uteži delo pri hitrih odzivih na incidente.
Rešitev *Layout expert*[6], ki smo jo opisali, pa omogoča izračunavanje porazdelitve pomnilnika pomembnih jedrnih struktur s pomočjo nastavitev jedra na ciljnem sistemu, brez potrebnega dodatnega nasnemavanja programov na ciljnem sistemu, potrebe po zunanji pomoči in zanašanja na prevajalno verigo (ki ni vedno prisotna).

6. REFERENCE

[1] Redline memory analysis tool. Dosegljivo: `https://www.fireeye.com/services/freeware/redline.html`. [Zadnji dostop: 13.05.2017].

[2] The volatilify foundation. Dosegljivo: `http://www.volatilityfoundation.org/`. [Zadnji dostop: 13.05.2017].

[3] M. Cohen. Installing rekall on windows, rekall memory forensics blog. Dosegljivo: `http://rekall-forensic.blogspot.si/2015/09/installing-rekall-on-windows.html`. [Zadnji dostop: 13.05.2017].

[4] M. H. Ligh, A. Case, J. Levy, and A. Walters. *The Art of Memory Forensics: Detecting Malware and Threats in Windows, Linux, and Mac Memory*. Wiley Publishing, 1st edition, 2014.

[5] A. Socała and M. Cohen. Automatic profile generation for live linux memory analysis. Dosegljivo: `http://www.dfrws.org/sites/default/files/session-files/dfrws-eu-2016-p2.pdf`. [Zadnji dostop: 13.05.2017].

[6] A. Socała and M. Cohen. Automatic profile generation for live linux memory analysis. *Digital Investigation*, 16, Supplement:S11 – S24, 2016. {DFRWS} 2016 Europe - Proceedings of the Third Annual {DFRWS} Europe.

[7] J. Stüttgen and M. Cohen. Robust linux memory acquisition with minimal target impact. *Digital Investigation*, 11, Supplement 1:S112 – S119, 2014. Proceedings of the First Annual {DFRWS} Europe.

Obnovitev močno fragmentiranih JPEG datotek

Erik Janežič
Fakulteta za Računalništvo in Informatiko
Večna pot 113
Ljubljana, Slovenija
ej6097@student.uni-lj.si

POVZETEK

Obnavljanje izbrisanih dokumentov predstavlja pomemben del forenzične raziskave. V tem delu so se avtorji osredotočili na obnavljanje JPEG datotek, s poudarkom na datotekah, ki so močno fragmentirane. Glavna težava pri obnavljaju močno fragmentiranih datotek nastopi pri procesu sestavljaja fragmentov v pravilni vrstni red. Za soočanje s to problematiko, avtorji predlagajo novo metriko CED (Coherence of Euclidean distance), za boljše predvidevanje sosednjosti dveh fragmentov. Razvili so tudi svoj algoritem, ki ima poleg drugačne metrike tudi nekatere druge prednosti pred obstoječimi algoritmi. Učinkovitost novega algoritma so primerjali proti dobro poznanemu orodju APF (Adroit Photo Forensic). Teste so izvajali na slikah iz SD kartice neke digitalne kamere.Rezultati primerjave so pokazali, veliko prednost novega orodja pred obstoječim APF. Avtomatsko so uspeli obnoviti kar 97% fragmentiranih JPEG slik, med tem ko je bil APF uspešen le pri 79% slik.

Ključne besede

Fragmentacija slik, obnavljanje, CED, SOD, ED,

1. UVOD

Primeri v digitalni forenziki od forenzika velikokrat zahtevajo ekstrakcijo in analizo multimedijskih datotek. Pogosto so primeri takšni, da je osumljenec že izbrisal podatke iz diskov. V takšnih situacijah postanejo pomembna orodja za obnavljanje izbrisanih datotek (ang. file carving tools). Kadar so boloki datoteke razporejeni sekvenčno na disku, obnavljanje datoteke ne predstavlja prevelikega problema. Vsak tip datoteke ima namreč svoj začetni in končni marker (pri JPEG datotekah: 0xFFD8-> začetek, 0xFFD9 -> konec). Obstaja veliko orodij, ki so sposobna rešiti tovrsten problem (npr. Encase, FTK, The Sleuth Kit). Za orodja, ki se zanašajo na sekvenčnost blokov datoteke, nastopijo težave takrat, ko so datoteke fragmentirane. Fragmentacija je problematična predvsem za velike datoteke, ki so bile shranjene na mediju, ki ima malo prostega prostora. Največkrat na takšne situacije naletimo pri prenosljivih medijih ki uporabljajo EEPROM. V takšnih medijih je velikokrat uproabljena tehnologija proti izrabi pomnilnika, kar vodi do močno fragmentiranih datotek.

Za obnnavljanje fragmentiranih datotek so bila razvita naprednejša orodja, katerih osnovni princip delovanja je sledč; preverimo zaporedne binarne bloke pridobljene iz medija ter probamo določiti, če je med katerima blokoma fragmentacijska točka. Za določevanje fragmentacijskih točk v JPEG datotekah vsa obstoječa orodja izhajajo iz predpostavke, da morajo imeti meje zaporednih blokov JPEG datoteke podobne barve. Za določitev podobnosti meje se uporabljajo mere podobnosti kot na primer vsota razlik (ang. Summ Of Differences, SOD) ali evklidska razdalja (ang Euclidean Distance, ED). Takšne podobnostne metrike dobro delujejo samo na gladkih delih slike, kjer prehod med različnimi barvami ni oster. Kadar imamo opravka s slikami, ki imajo veliko ostrih barvnih mej, te metrike pogosto naredijo napake, kar povzroči napake v vseh nadaljnih korakih obnavljanja.

V tem raziskovalnem projektu so avtorji opazili, da kljub temu, da imajo slike več območij z različnimi tipi barvnih variacij, bo na majhnih lokalnih območjih barvni variacijski vzorec podoben. Za izkoriščanje te lastnosti pri določevanju fragmentacijskih točk so razvili novo metriko imenovano koherenca Evklidske razdalje (ang. Coherence of Euclidean Distance, CED). V nasprotju z nekaterimi obstoječimi algoritmi, ki vse bloke analizirajo sekvenčno, so avtorji v novem algoritmu najprej združili vse sosednje JPEG bloke v JPEG fragmente in te uporabili kot najmanjšo enoto za nadaljno procesiranje.

V nadaljevanju bo najprej opisano še nekaj ozadja za lažje razumevanje novih pristopov, nato bomo predelali nekatere lastnosti že obstojećih algoritmov, sledi razlaga predlaganega algoritma, zaključili pa bomo s primerjavo in analizo predlaganega algoritma z obstoječimi pristopi.

2. TERMINOLOGIJA

2.1 Vizualizacija datotek na disku

Slika 1 upodablja vse termine, ki so pomembni za nadaljno razumevanje teksta. Blok oziroma cluster je najmanjša alokacijska enota datotečnega sistema. Fragment je en ali več zaporednih blokov, ki pripadajo isti datoteki. Fragmentirana datoteka je datoteka z dvema ali več fragmentoma. Segment pa je skupek večih zaporednih fragmentov, ki lahko pripadajo različnim datotekam.

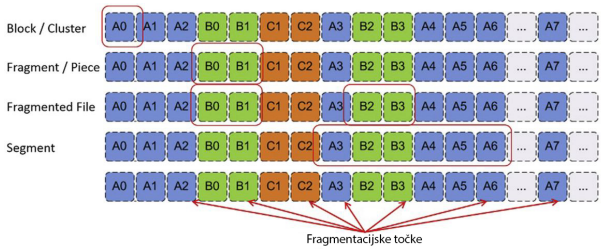


Figure 1: Predstavitev blokov datotek na disku. Bloki z enako črko predstavljajo bloke iste datoteke

3. OBSTOJEčE TEHNOLOGIJE

Kot že omenjeno obstaja veliko osnovnih orodij, ki so uspešna za obnavljanje sekvenčnih datotečnih blokov, ki pa odpovejo pri prisotnosti fragmentacije znotraj datotek.
Za obnavljanje fragmentiranih datotek je bil potreben razvoj naprednejših orodij. Cohen v letu 2007 predlaga metodo, ki izrablja različne metapodatke datotek (npr pdf, zip,...) za obnovitev datotek. Vendar pa metapodatki niso zmeraj prisotni zato metoda ni uporabna za JPEG datoteke.
V istem letu avtor Garfinkel predlaga rešitev za bifragmentirane datoteke (datoteke, ki imajo samo dva fragmenta). Pri tej metodi z "brute force" pristopom generiramo vse kombinacije fragmentov z začetnim markerjem in s končnim markerjem, nato pa generirane datoteke preverimo z JPEG renderiranjem.
Leta 2008 je bil predlagan pristop za rekonstrukcijo JPEG datotek z "restart" markerji. Vloga "restart" markerjev v JPEG datoteki je preprečevanje napak v bitnem toku zaradi napak pri prenosu ali pri pokvarjenih datotekah. Ker se v praksi JPEG datoteke večinoma prenašajo po kanalih brez napak, "restart" markerji večinoma niso potrebni. Posledično večina JPEG datotek nima "restart" markerjev, in tudi če jih ima ni zagotovo da jih bo imel vsak fragment datoteke. Zradi tega ta metoda v večini primerov ni uporabna.
Ena bolj sofisticiranih metod, ki so jo razvili Memon in sodelavci, problem fragmentiranih JPEG datotek preslika v problem iskanja Hamiltonove poti med bloki JPEG datoteke. Vsak blok predstavlja vozlišče v grafu, vrednost povezave med dvema vozliščema pa je odvisna odvisna od podobnosti med obema blokoma. Za izračun podobnosti ta pristop uprablja SOD metriko, ki pa kot že omenjeno ne deluje najbolje kadar je v sliki veliko ostrih robov, oziroma kadar je datoteka močno fragmentirana. Ta metoda je uporabljena v enem bolj znanih forenzičnih orodij APF (Adroit Photo Forensic).
V letu 2011 sta Guo in Xu predlagala pristop z uporabo nevronskih mrež za napovedovanje naslednjega fragmenta. Vendar pa se je izkazalo da metoda ni najbolj uporabna kadar fragmenti JPEG datoteke niso zaporedni.

Problem vseh omenjenih metod je, da v primeru ko imamo v mediju več močno fragmentiranih datotek, kjer so fragmenti posameznih datotek pomešani. Do tega pride predvsem zaradi suboptimalnih metrik za določevanje podobnosti med dvema blokoma JPEG datoteke (ED,SOD). Poleg tega vse naštete metode temeljijo na posameznem bloku kot osnovni enoti za analizo, kar predstavlja težavo za uporabo v primerih kjer je potrebno pregledati velike diske.

3.1 SOD in ED metriki

SOD (Sum of differences) in ED (Euclidean distance) sta pomembni metriki v obstoječi tehnologija obnavljanja fragmentiranih JPEG datotek. V nadaljevanju bom obe na kratko opisali ter ju primerjali z novo predlagano metriko CED (Coherence of Euclidean distance).

Obe mrtriki izrabljata lastnosti gladkih prehodov v sliki za ovrednotenje sosednjih blokov datoteke. Za fragmentirano sliko se SOD vrednost izračuna kot vsota razlik RGB vrednosti slikovnih točk ki so na mejah mmed dvema blokoma.

$$SOD = \frac{1}{n}\sum_{i=1}^{n}|x_i - y_i|$$

kjer je x_i vrednost slikovne pike enega fragmenta, y_i vrednost pripadajoče slikovne pike iz drugega segmenta, n pa število vseh slikovnih pik v meji med fragmentoma. Kljub temu, da je računanje SOD vrednosti enostavno, hitro in učinkovito ni zmeraj najbolj zanesliva. Do problemov na primer pride kadar so v RGB vrednosti v enem od sosednjih blokov enakomerno razporejene (na primer R:50 ,G:50 ,B:50) v drugem bloku pa je celotna vrednost vsote RGB vrednosti v enem parametru (na primer R:0, G:0, B:150).
Za primer si predstavljamo da imamo pet slikovn ih točk: R_0, R_1, R_2, R_3, R_4 RGB vrednosti posamenih točk so sledeče:

1. R_0 : R=0, G=0, B=0
2. R_1 : R=50, G=50, B=50
3. R_2 : R=0, G=0, B=150
4. R_3 : R=0, G=150, B=0
5. R_4 : R=150, G=0, B=0

Vsako iz med točk od R_1 do R_4 primerjamo s točko R_0 in izračunamo SOD vrednosti za posamezen par:

1. $SOD_{(R_0,R_1)} = (|0-50| + |0-50| + |0-50|)/1 = 150$
2. $SOD_{(R_0,R_2)} = (|0-0| + |0-0| + |0-150|)/1 = 150$
3. $SOD_{(R_0,R_3)} = (|0-0| + |0-150| + |0-0|)/1 = 150$
4. $SOD_{(R_0,R_3)} = (|0-150| + |0-0| + |0-0|)/1 = 150$

Vidimo, da so SOD vrednosti enake, če pa vizualno primerjamo sosede je takoj očitno, da je R_1 veliko bolj podobna

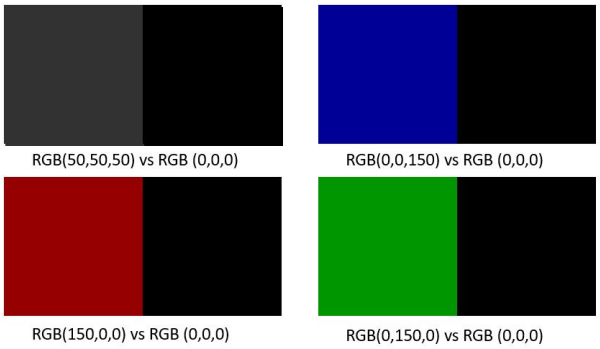


Figure 2: Vizualna primerjava sosednjih slikovnih točk

R_0 kot katera koli ostala točka (slika 2). Iz tega stališča nam da ED metrika boljšo sliko realnost. ED se izračuna na zalo podoben način kot SOD, le da os razlike znotraj vsote kvadrirane, celotno vsoto pa na koncu še korenimo.

$$ED = \frac{1}{n}\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

Če sedaj za isti primer kot SOD poračunamo še ED vrednosti:

1. $ED(R_0, R_1) = \sqrt{((0-50)^2 + (0-50)^2 + (0-50)^2)}/1 = 86.6$
2. $ED_{(R_0,R_2)} = \sqrt{((0-0)^2 + (0-0)^2 + (0-150)^2)}/1 = 150$
3. $ED_{(R_0,R_3)} = \sqrt{((0-0)^2 + (0-150)^2 + (0-0)^2)}/1 = 150$
4. $ED_{(R_0,R_4)} = \sqrt{((0-150)^2 + (0-0)^2 + (0-0)^2)}/1 = 150$

Vidimo, da nam da ED metrika bolj realen rezultat. V obeh metrikah SOD in ED nižja vrednost pomeni večjo podobnost med sosednima slikovnima točkama. Obe metriki za predvidevavnje podobnosti upoštevata le eno vrsto slikovnih točk na meji, kar vodi do napačnih rezultatov v primeru, ki imamo na mediju več fragmentiranih slik s podobno vsebino ali kadar je v slikah veliko ostrih robov.

4. PREDLAGANI ALGORITEM

4.1 Nova metrika - koherenca Evklidske razdalje (CED)

Namen predlagane metrike je izboljšanje zanesljivosti določevanja zaporednih datotečnih fragmentov. Definirana je kot:

$$CED = |ED_{meja} - ED_{bliznji}|$$

Na sliki 3 je bolj nazorno prikazano katere vrste slikovnih točk obravnava CED metrika pri izračunu. S tem da pri izračunu upoštevamo bližnjo okolico meje že obnovljenega dela datoteke, dobimo boljši občutek kakšno mejo lahko pričakujemo pri naslednjem zaporednjem fragmentu.

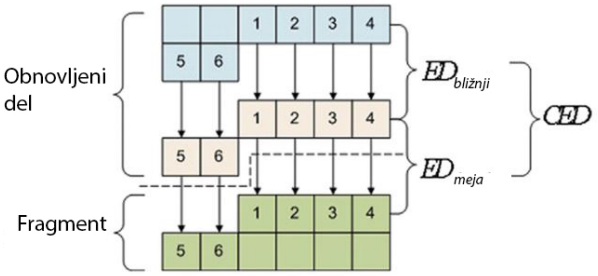


Figure 3: Shema CED metrike

4.2 Primerjava CED,SOD in ED

Za primerjavo lastnosti med CED, SOD in ED metrikami so v članku oporabili sliko 4. Na sliki 5 je oprizorjena primer-

Figure 4: Slika na kateri so bile izvedene primerjave

java vrednosti posameznih metrik. Na X osi so v obeh grafih vrste JPEG datoteke, na Y osi pa so vrednosti metrik za posamezno vrsto.
Z SOD/ED grafa je jasno razvidno, da pri SOD in ED metrikah prihaja do velikih fluksuacij v območju slike kjer je veliko kontrastov in ostrih mej (območje med 20 in 590 vrsto). Na sliki je v tem predelu veliko vej in listov. Porast

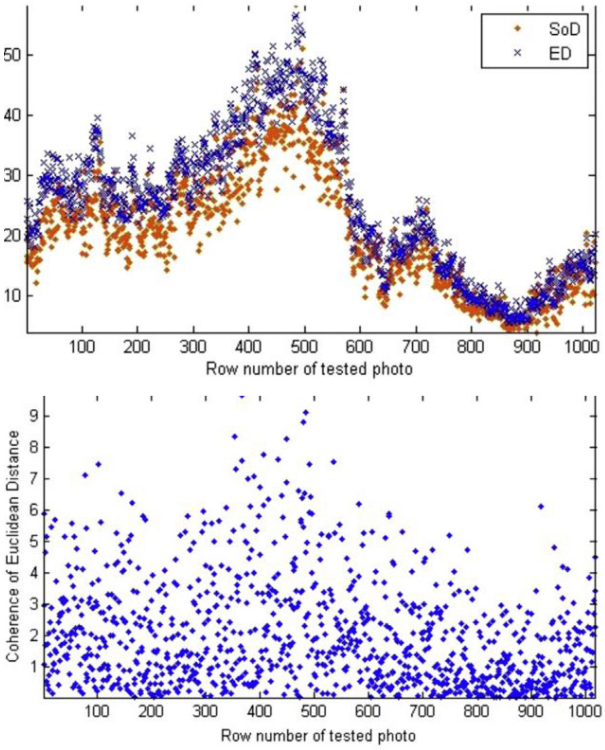


Figure 5: Primerjava vrednosti SOD, ED in CED vrednosti za posamezne vrste v JPEG datoteki

v metričnih vrednosti je opazen tudi na območju med 650 in 720 vrsto, kjer so na sliki prisotni ostri robovi hiše. Takšne fluksuacije v praksi onemogočajo pravilno določitev zaporednih segmentov, če bi bila fragmentacijska točka ravno v eni od takih fluksuacij

Na grafu CED vrednosti so opazne bistveno nižje fluksuacije ter veliko bolj homogeno razporeditev vrednosti po vseh vrstah slike. Tudi maksimalne vrednosti so bistveno nižje kot pri SOD in ED metrikah ($MAX_{SOD,ED} = 50+$, $MAX_{CED} = 10$).

4.3 Primerjava detekcije fragmentacije med posameznimi metrikami

Za primerjanje zmožnosti detektiranja fragmentov posameznih metrik so avtorji članka pripravili fragmentirano različico slike 4 (slika 6). Na sliki 7 so navedene vrednosti posameznih metrik v odvisnosti od vrste na sliki. Vidimo, da vse tri metrike uspešno zaznajo mejo med fragmentom 0 in 1. Pri mejah med fragmentoma 1 in 2 ter 2 in 3 pa opazimo, da os vrhovi pri SOD in DE metrikah relativno nizki v primerjavi z prvim vrhom, ter območjem med 20 in 400 vrsto. Pri CED metriki, je bazna vrednost drastično nižja in bolj homogena in tudi vrhova 2 in 3 sta lahko zaznavna saj močno izstopata od bazne vrednosti.

Za ilustracijo težav SOD metrike so avtorji članka na takšen način fregmentirano sliko (slika 6) poizkušali obnovit z orodjem APF, ki uporablja SOD metriko za detekcijo fragmentov. APF je sposoben pravilno združiti segmenta fragmenta

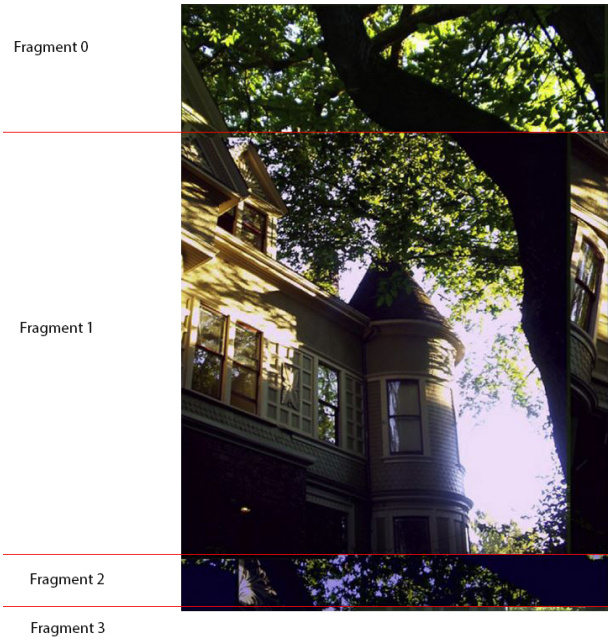


Figure 6: Fragmentirana slika 4

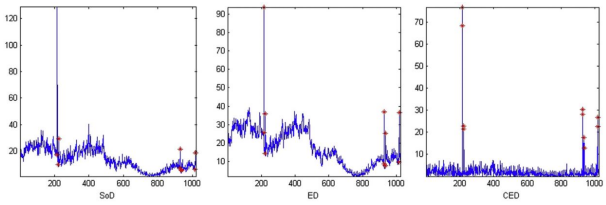


Figure 7: Primerjava detekcije fragmentacij za posamezne metrike

0 in 2. Za fragmenta 1 in 3 pa ne mora uspešno določiti meje, zaradi premalo izrazitih vrhov.

4.4 Algoritem

Poleg drugačne metrike je druga ključna razlika predlaganega algoritma uporaba JPEG segmentov kot osnovnih enot za nadaljne procesiranje (za razliko od ostalih algoritmov, ki uprabljajo JPEG bloke). Shematski prikaz delovanja algoritma je prikazan na sliki 9. V nadaljevanju bo na kratko opisan vsak od šestih korakov algoritma.

4.4.1 Identificiranje JPEG fragmentov

Za ločitev JPEG blokov od blokov ki pripadajo drugim tipom datotek že obstajajo učinkoviti algoritmi kot na primer Oscar.

4.4.2 Združevanje fragmentov v JPEG segmente

Pri združevanju fragmentov v segmente se algoritem združuje zaporedne JPEG fragmente dokler ne naleti na fragment z

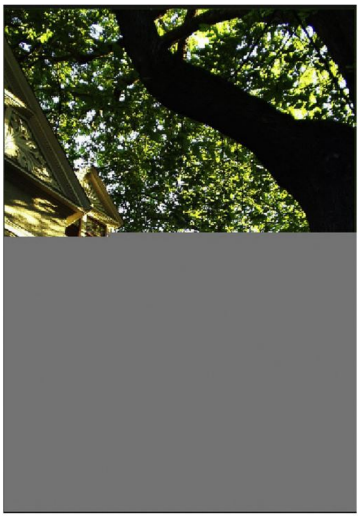


Figure 8: Rezultat orodja APF na fragmentaciji iz slike 6

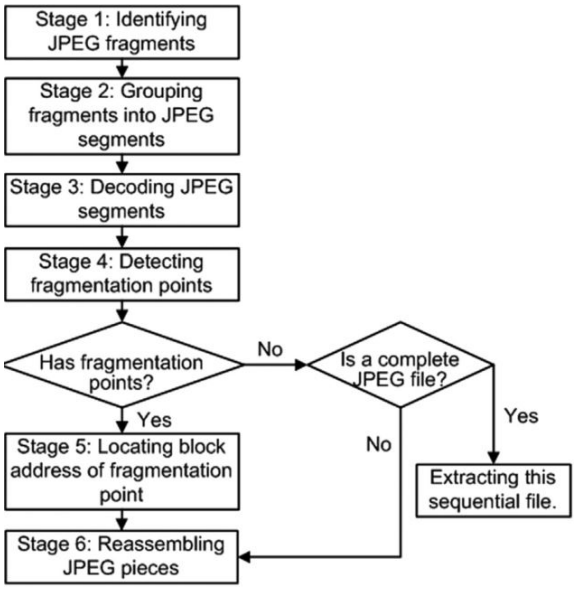


Figure 9: Rezultat orodja APF na fragmentaciji iz slike 6

začetnim oziroma končnim markerjem. V tako pripravljenih segmentih bodo lahko fragmenti različnih JPEG datotek.

4.4.3 Dekodiranje JPEG segmentov

Za dekodiranje JPEG segmentov je v vsakem segmentu potreben pravilen začetni marker. V primeru da takšen marker obstaja ga uporabimo za dekoderanje vseh fragmentov v tem segmentu. Če slučajno takega markerja za določen segment ni, mu lahko poizkusimo prilegati vse headerje iz "data dump -a". Kadar ta pristop ne deluje lahko segmentu priredimo psevdo začetni marker.

4.4.4 Detektiranje fragmentacijskih točk

Pri detektiranju fragmetacijskih točk se uporablja CED metrika. Iz empiričnih opažanj je bilo ugotovljeno, da imajo fragmentacijske točke nekajkrat večjo CED vrednost kot je povprečna CED vrednost. V primeru, da ni najdenih nobenih takšnih vrhov, lahko sklepamo, da segment ni fregmentiran in, da ga lahko obravnavamo kot sekvenčene bloke ki pripadajo eni datoteki. V primeru, da so vrhovi najdeni lahko iz njih razberemo vrsto v kateri pride do fragmentacije. Fragmenti v okolici fragmentacijske meje morajo biti nadaljno analizirani v naslednjem koraku. Po tem pa lahko JPEG segment razdelimo na več delov glede na lokacijo fragmentacijskega bloka.

4.4.5 Lociranje naslova bloka fragmentacijske točke

V praksi je navadno velikost blokov majhna (ponavadi 4096 byt-ov). Ker so dandanes visokoločljivostne kamere nekaj vsakdanjega, imamo ponavadi opravka s slikami, ki imajo visoke resolucije (3648X2736 in više). To predstavlja dodaten problem za obnavljanje fragmentiranih JPEG datotek, kajti ponavadi posamezna vrsta slike ne paše v en sam blok. Zato je potrebno po določitvi fragmentacijske točke iz prejšnjega koraka, izvesti dodatno analizo. Drugače lahko pride do napak pri sestavljanju slik.
Po dekodiranju algoritem izračuna tri vrednosti:

1. Celotno število blokov v segmentu: N_{blocks}
2. Število vrst v dekodiranem segmentu: N_{rows}
3. Zaporedno število vrste kjer nastopi fragmentacijska točka: fp_{row}

V naslednjem koraku ocenimo približen naslov bloka kjer nastopi fragmentacija (fp_{block}):

$$N_{blocks}/N_{rows} = fp_{block}/fp_{row}$$

Nato z binarnim iskanjem določimo zgornjo in spodnjo mejo fp_{block}, s čimer zagotovimo, da bodo bloki bloki v okolici fragmentacijske meje vključeni.
Iskanje točnega naslova zaključimo z preverjanjem CED vrednosti sosednjih blokov v okolici fp_{block}. Na mestu kjer CED vrednost močno naraste je blok s fragmentacijsko točko.

4.4.6 Sestavljanje JPEG delov

Sestavljanje celotne slike je prevedeno na problem iskanja najkrajše poti v grafu. V grafu vozlišča predstavljajo posamezni JPEG deli, povezave med njimi pa so uteženе verjetnostjo, da sta dva dela sosednja. Verjetnost se izračuna s pomočjo CED metrike. Začetna točka pri iskanju je fragment z začetnim markerjem, končna pa segment s končnim markerjem.

4.5 Rezultati eksperimentalnih primerjav

Novonastali algoritem, ki uporablja CED metriko so avtorji za zaključek primerjali z obstoječim orodjem APF. Najprej so testirali zanesljivost CED metode testirali na setu 1000 slik z resolucijo 1021x768 in 100 slik z visoko resolucijo 3648X2048 . Kot mero za učinkovitost metode so uporabili razmerje med številom napačno združenih vrst in številom vseh vrst v sliki. Rezultati so prikazani v sliki 10. Iz rezultatov je razvidno, da CED metrika deluje bistveno bolje kot

The number of false matches using CED, SoD and ED.

Testing	1000 low resolution			100 high resolution		
Photos	(1021 × 768)			(3648 × 2048)		
Similarity Metric	CED	ED	SoD	CED	ED	SoD
False Matches	132,529	381,112	491,775	1829	115,142	128,805
False Matching Rate	17.26%	49.62%	64.03%	0.89%	56.22%	62.89%
Files with False Match	954	1000	1000	59	100	100

Figure 10: Primerjava predlaganega algoritma in obstoječega APF

SOD metrika. Še posebaj je razvidno izboljšanje pri obnavljanju velikih datotek, kjer se je predlagani algoritem odrezal kar 60x bolje.

V drugem testu so uporabili 4Gb SD kartico, ki so jo najprej formatirali v FAT32 format nato pa so set 207 visokoresolucijskih slik (3648X2736) s povprečno velikostjo 5MB zapisali na kartico. Slike so nato nekaj časa naključno brisali in dodajali nove. Na ta način so simulirali običajno uporabo SD kartice. PO končani simulaciji so bili podatki na kartici sestavljeni iz 87 sekvenčnih datotek, 100 fragmentiranih datotek in pa 20 datotek kjer je do fragmentacije prišlo v začetnem markerju in niso bile vključene v analizo. Rezultati eksperimenta so zbrani v sliki 11.

Category	JPEG files	Proposed algorithm	APF
sequential file	87	87	87
2-piece file	79	78	65
3-piece file	18	17	11
4-piece file	2	1	2
6-piece file	1	1	0

Figure 11: Primerjava predlaganega algoritma in obstoječega APF pri obnavljanju datotek iz SD kartice.

Zadnji eksperiment je testiral zmožnost restavriranja močno fragmentiranih datotek. Set testnih datotek za ta eksperiment je bil sestavljen iz 184 močno fragmentiranih datotek, med katerimi je bilo 109 3-delnih JPEG datotek in 75 4-delnih JPEG datotek. Segmentacijske točke so bile narejene naključno s pomočjo računalnika. Rezultati so zbrani v sliki 12. Vidimo, da je novi algoritem vsaj za tretjino boljši pri

Category	JPEG files	Proposed algorithm	APF
3-piece file	109	96	66
4-piece file	75	61	32

Figure 12: Primerjava predlaganega algoritma in obstoječega APF pri obnavljanju močno fragmentiranih datotek.

obnavljanju 3-delnih datotek, ter skoraj še enkrat boljši pri obnavljanju 4-delnih datotek.

5. ZAKLJUčEK

V raziskavi so avtorji predlagali nov algoritem za obnavljanje fragmentiranih JPEG datotek, ki temelji na novi metriki CED za odkrivanje sosednjih fragmentov slike. Algoritem se je izkazal za veliko bolj robustnega in natančnega v primerjavi z uveljavljenim forenzičnim orodjem APF. Algoritem ima kljub temu še nekaj možnosti za izboljšavo, težavo z določitvijo fragmentacijske meje ima namreč v primerih ko slika vsebuje oster rob v povsem horizontalni smeri.

6. VIRI

1. Yanbin Tang, Junbin Fang, K.P. Chow, S.M. Yiu, Jun Xu, Bo Feng, Qiong Li, Qi Han.Recovery of heavily fragmented JPEG files. Digital Investigation 18: 108-117, 2016
2. Struktura trdega diska: http://www.dewassoc.com/kbase/hard_drives/clusters.htm
3. Orodje Oscar: https://link.springer.com/chapter/10.1007%2F0-387-33406-8_5
4. Fragmentacija: https://sites.ualberta.ca/dept/chemeng/AIX-43/share/man/info/C/a_doc_lib/aixprggd/genprogc/file_space_allocation.htm
5. M.I. Cohen. Advanced carving techniques. Digital Investigation 4:119-128, 2007
6. Simson L. Garfinkel.Carving contiguous and fragmented files with fast object validation. Digital Investigation 4:2-12, 2007
7. Karresand M, Shahmehri N. Reassembly of fragmented jpeg images containing restart markers. 2008 European conference on computer network defense; 2008.