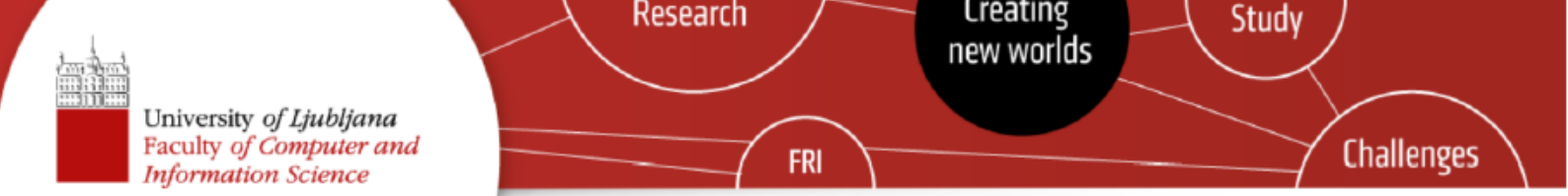


University of Ljubljana
Faculty of Computer and
Information Science

Zbornik

Digitalna forenzika

Seminarske naloge, 2014/2015

Ljubljana, 2015

Zbornik
Digitalna forenzika, Seminarske naloge 2014/2015

Editors: Andrej Brodnik, Luka Krsnik, Anže Mikec, Nejc Novak, študenti

Ljubljana : Univerza v Ljubljani, Fakulteta za računalništvo in informatiko, 2015.

Kazalo

Uvod

Forenzična znanost je znanstvena metoda, ki zbira, proučuje in hrani informacije preteklosti, ki jih lahko uporabimo na sodiščih. Z razvojem na različnih področjih se je znanost močno razširila, zato najverjetneje ne obstaja nihče, ki bi poznal celotno področje forenzike. Forenzično znanost tvori več podvej. To so npr. digitalna forenzika, bioforenzika, forenzična kemija itd.

Zbornik se osredotoča na digitalno forenziko. To je veja forenzične znanosti, ki se nanaša na materialne dokaze iz digitalnih naprav. Digitalno forenziko bi lahko dalje razdelili na računalniško (datotečno) forenziko (hranjenje in analiza računalnikov), omrežno forenziko (hranjenje in analiza prometa ter logov omrežja), mobilno forenziko (hranjenje in analiza pametnih telefonov ter sistemov GPS) in forenziko slabogramja (hranjenje in analiza zlonamerne kode).

Študentje magistrskega študija Fakultete za računalništvo in informatiko Univerze v Ljubljani, ki smo obiskovali predmet Digitalna forenzika, smo med študijskim letom dobili nalogo, katere rezultat je ta zbornik. Med nalogo smo bili razdeljeni v skupine z največ tremi člani. Vsaka skupina je dobila članek, ki se je navezoval na eno izmed vej digitalne forenzike in je bil v času pisanja aktualen (večina jih je izšla v letu 2014). Skupine so dodeljeni članek morale prebrati, raziskati podobne članke in napisati njihovo obnovo, za boljšo oceno pa so lahko tudi izvedle eksperiment, ki je bil opisan v članku. Vse te naloge, ki smo jih lahko napisali v slovenščini ali angleščini, so strnjene v seminarsko nalogo. Vsak študent je po zapisani seminarski nalogi dobil dva druga članka, ki ju je moral recenzirati (kot bi rokovali s pravimi članki). Po recenziji so bile vse naloge še enkrat pregledane in popravljene.

Ta zbornik predstavlja skupek vseh končnih seminarskih nalog, ki so bile narejene v študijskem letu 2014/2015. Namenjen je vsem, ki jih zanima področje digitalne forenzike ali pa le specifična tema, opisana v zborniku. Upam, da ob prebiranju izveste kaj novega, zanimivega in uporabnega.

Luka Krsnik

Poglavje 1

Povzetki (abstracts)

1.1 Distributed filesystem forensics: XtreemFS as a case study

Članek opisuje postopek forenzične preiskave porazdeljenih datotečnih sistemov v oblaku, pri čemer se osredotoči na znan datotečni sistem XtreemFS. Porazdeljeni datotečni sistemi nam nudijo cenovno ugodne in efektivne rešitve za shranjevanje velikih količin podatkov v oblaku. Tudi te tehnologije so lahko izkoriščene v zlonamerne namene, zato potrebujemo postopke in orodja za digitalno forenzično preiskavo. V članku avtorja uporabita predlagan splošen postopek preiskave porazdeljenih datotečnih sistemov na sistemu XtreemFS, pri čemer ga razširita s specičnimi lastnostmi sistema XtreemFS, ki jih je potrebno upoštevati pri zbiranju dokaznega gradiva. Avtorja na koncu članka tudi predlagata natančen proces zbiranja digitalnih dokazov iz porazdeljenih datotečnih sistemov.

1.2 Pregled "Asset Risk Scoring in Enterprise Network with Mutually Reinforced Reputation Propagation"

Članek Asset Risk Scoring in Enterprise Network with Mutually Reinforced Reputation Propagation predstavlja metodo za vrednotenje ogroženosti v omrežju imenovano MUSE (Mutually-reinforced Unsupervised Scoring for Enterprise risk). Pregledali bomo algoritme, na katerih je metoda osnovana in področje uporabe, ki mu je namenjena. Metoda uporablja ideje, ki so že znane iz analiz varnosti na spletu in jih uporabi za analizo omrežja v podjetju. Dobljene rezultate je mogoče uporabiti za bolj učinkovito zaznavanje groženj ter za pomoč pri preiskavi. Na koncu bomo predstavili še podrobnosti same metode in njeno analizo.

1.3 Pregled analiz in gradnja kriminalnih mrež

V začetnem delu pričujočega besedila je predstavljen članek Constructing and Analyzing Criminal Networks, v katerem se avtorji članka sprašujejo, ali lahko s pomočjo javno pridobljenih e-poštnih naslovov (s pomočjo Facebook profilov) kriminalcev zgradijo smiselno mrežo in na tak način odkrijejo kriminalne

skupine, vodje teh skupin in osebe, ki kriminalne skupine povezujejo. V članku avtorji predstavijo metodologijo, orodja in mere, ki so jih uporabili pri raziskovanju, vse troje pa je predstavljeno tudi v tem članku. Kratek pregled obstoječih raziskav, ki je predstavljen v besedilu, zagotovo ne zajema vseh objavljenih del in celotnega obravnavanega področja, obsega pa dela s področja socialnih mrež in analiz ter s področja kriminalnih socialnih mrež. Podobna metodologija, orodja in mere, kot so jih uporabljali omenjeni avtorji, so uporabljene tudi v zadnjem delu članka, ki opisuje uporabo teh orodij na mreži plezalnega centra Ljubljana.

1.4 Uspešnost kampanj z nezaželeno pošto in botneti

Pošiljanje nezaželene pošte povzroča težave tako končnim naslovnikom kot tudi vmesnim členom - ponudnikom raznih spletnih storitev, lastnikom računalnikov, ki so del botnetov in mrežni arhitekturi. Pošiljatelji se poslužujejo veliko različnih metod za doseganje čim višje učinkovitosti in uspešnosti oz. dostavljivosti nezaželenih sporočil - v tem delu smo naredili kratek pregled stanja na tem področju in skušamo izpostaviti najpomembnejše. Nezaželena sporočila se širijo ne le v domeni elektronske pošte, temveč tudi preko veliko drugih elektronskih kanalov. Za lažje in učinkovitejše pošiljanje se avtorji kampanj poslužujejo botnetov - omrežja okuženih računalnikov, ki služijo namenu pošiljanja nezaželene pošte. V delu je predstavljena arhitektura botnetov in opisan eden izmed največjih, Cutwail.

1.5 Forenzična analiza z DNSSEC

Zastrupljanje DNS predpomnilnika je pogosta oblika računalniškega napada, s katero je mogoče nadzorovati žrtvin spletni promet, ji uvajati omejitve pri dostopu, oziroma do nje prenašati škodljivo programsko opremo. Njegovo obvladovanje je eden ključnih dejavnikov za zagotavljanje pravilnosti in dostopnosti internetnega omrežja in storitev. V članku so predstavljene slabosti DNS infrastrukture, ki temelji na izziv-odgovor obrambi in, nasprotno od splošnega prepričanja, ponuja obilico možnosti za različne napade. Nato predstavimo DNSSEC način obrambe proti predpomnilniškemu zastrupljanju, ki je zelo učinkovit in hkrati tudi edini obrambni mehanizem, ki omogoča analizo napada tudi dolgo po samem dogodku.

1.6 Preiskovanje napadov povezanih z izvornimi vrati 0

V članku razložimo osnovne lastnosti preiskovanja omrežja s pomočjo postavljanja vrednosti izvornih vrat paketov TCP na 0 ter kako takšno preiskovanje detektiramo. Opiramo se na referenčni članek z naslovom Multidimensional investigation of source port 0 probing. V prvem delu članka predstavimo postopek detekcije, ki so ga razvili avtorji referenčnega članka tj. s pomočjo gručenja vektorjev značilk paketov TCP odkrijemo glavna žarišča napada. Za posamezno žrišče znamo s pomočjo posebne zbirke poročil o znanih škodljivih programih celo ugotoviti, kateri DLL je bil uporabljen za napad. V drugem delu si ogledamo nekaj alternativnih pristopov k odkrivanju neželenega skeniranja omrežja. Izkaže se, da obstaja veliko različnih načinov in pristopov kako zgodaj zaznati napad. V tretjem delu predstavimo

najodmevnejše žrtve kiber napadov, ki so se v zgodnji fazi začeli ravno s skeniranjem omrežja. Ogledamo si povzetke postopkov vdora ter posledice, ki so jih vdori za seboj pustili. Predstavimo tudi nekaj statistik v zvezi s tovrstnimi napadi.

1.7 Detecting NAT gateways and differenting host devices behind NAT for purposes of digital forensic investigations

One of the of the most common misconceptions emerging from the popular culture is the assumption that IP addresses can uniquely identify the source from which network traffic originates. While it is not impossible, the field of network forensics must adopt more general techniques that allow reliable identification of end-machines, for instance, if they are suspected of involvement in a criminal act. An example of scenario where identification solely with the use of IP addresses could lead to erroneous conclusions, are networks with NAT routers, widely used in residential, home and enterprise networks. We present a structured overview of approaches for identification of NAT devices and diffrentiation of host machines behind NAT and discuss their performance, reliability and generality.

1.8 Steganografija v LTE (Long Term Evolution) sistemi

Seminarska naloga govori o tehnologiji LTE, njenem delovanju in omrežni steganografiji. Predstavili bomo steganografsko metodo LaTEsteg, ki je bila zasnovana za sisteme LTE. Metoda omogoča uporabniku prenos podatkov, ki je neviden nepooblaščenim osebam, katere se ne zavedajo skrite komunikacije. V seminarski nalogi je opisano njeno delovanje, učinkovitost in varnost.

1.9 Overview of PeerShark and other software for detecting peer-to-peer botnets

1.10 Cloud Forensics: iCloud

Cloud forensics has been a hot topic since the increase in popularity of cloud storage. There are already numerous cloud storage providers, one of them being Apple with their iCloud service. In this work we update the research carried out by Oestreicher in 2013 using iCloud 2013 and Mac OS X Maverics. The aim of that research was to develop a forensically robust method for iCloud data acquisition. This was done by checking the MD5 hash values and timestamps on the synchronized data via iCloud. We carry out similar research using a newer version of iCloud (2015) along with the new Mac OS X Yosemite (2015). We show that in two years time not much has changed and we obtain similar results. There are however some differences, one of them being Apple's newest preinstalled Photos application, which produces matching timestamps as well as MD5 hash values as data is synchronized. Timestamps and MD5 hash values still mismatch for the preinstalled applications (except for Photos and Calendar) as we compare the original and downloaded data.

1.11 Towards a Forencsic-Aware Database Solution: Using a secured Database Replication Protocol and Transaction Management for Digital Investigations

Podatkovne baze so danes zelo razširjene in vsebujejo tako veliko količino informacij, da bi bilo življenje brez njih nepredstavljivo. Ker pogosto vsebujejo osebne podatke in na splošno informacije, ki so privatne narave, podatkovne baze vsebujejo sisteme za zaščito. Klub vsemu, pa lahko pride do zlorabe, kraje ali prirejanja podatkov. Podatki pridobljeni v forenzični analizi so pomembni v naslednih primerih. Večino podjetji skrbi za podatke svojih uporabnikov in je inforamcija ali so bili podatki razkriti, oziroma na kakršen koli način kršena privatnost uporabnikov velikega pomena. V ta namen se potrebuje orodja s katerimi se lahko preveri, oziroma dokaže pristnost podatkov in ali je prišlo do zlorabe sistema. Drug primer je analiza napadov in ugotavljanje šibkosti sistema, ki jih je napadalec izkoristil, ter bodoča preprečitev. V ta namen bomo predstavili rešitev forenzično zavednega sistema za upravljanje podatkovnih baz, ki uporablja interne strukture za podvajanje in transakcije kot osnovo za preiskavo in za rekonstrukcijo podatkov v forenzični analizi. Velika prednost predstavljene metode je, da za analizo ne potrebujemo pregledovati dodatnih dnevniških zapisov. Prav tako metoda zagotavlja pristnost dokazov in utrdi dokazno verigo skrbnitva. Za lažjo oceno je predstavljen tudi prototip izvedbe v MySQL podatkovni bazi in celovita ocena varnosti, glede na najbolj relevantne napade.

1.12 Prednosti in slabosti žive in mrtve forenzične analize

Članek obravnava različne pristope forenzičnega preiskovanja, ki se izvaja na osumljenčevem računalniku. Podrobneje sta predstavljeni metodi žive in mrtve analize, njune prednosti ter slabosti. Mrtva analiza se uporablja predvsem zaradi dogovora, da se s takim načinom forenzične preiskave ohranja celovitost podatkov na podatkovnem nosilcu, živa analiza pa nam služi v primerih, ko je za preiskavo kritično, da zajamemo trenutno stanje sistema. Da bi ugotovili, če do sprememb pri mrtvi analizi prihaja na praktičnem primeru, smo poskusili izvesti podoben eksperiment kot v referenčnem članku - z zgoščevalno funkcijo SHA1 smo izračunali zgoščene vrednosti osumljenčevega diska pred in po izvajanju mrtve analize ter opazovali morebitne spremembe, ki jih tovrstna analiza naredi na disku.

1.13 VMI-PL: Jezik za nadzor navideznih platform z vpogledom v navidezne stroje

V računalniški forenziki imamo vse večkrat opravka s sistemi, ki na tak ali drugačen način tečejo v navideznih strojih, npr. v oblaku ali drugačnih okoliščinah z deljeno strojno opremo. Navidezne platforme omogočajo nevidno oz. skoraj nevidno opazovanje delovanja sistema, kar lahko močno olajša pridobivanje dokazov ter hkrati ne spreminja dokaznega gradiva (t.j. obnašanja programske opreme v opazovanem stroju).

1.14 Impacts of increasing volume of digital forensics data: A survey and future research challenges

Rapid development of digital media devices and its availability has contributed to the development of cyber-crime. In order for police to cope with the increase of cyber-crime new methods for digital forensic need to be developed. In this paper we will present and discuss methods of digital forensic analysis and current problems in this field.

1.15 Video Evidence as Used in Court of Law and its Source Identification

In the court of law, image and video is used more and more throughout new court cases as a means to help establish a valid case. With gaining more weight, it is also becoming more important to assure proper source of said material and to establish its authenticity. In this paper we begin with a broad overview of how video and image evidence is nowadays used in the court of law. Later, we describe several techniques that are currently being used for image and video source identification. We conclude the paper with a description of the source identification method by Chen et. al and the improvements that should be introduced into surveillance systems that employ wireless IP cameras.

1.16 Pridobivanje skritih sporočil iz steganografskih slik

Steganografija omogoča skrivanje podatkov v različne krovne medije (ang. cover media/images) tako, da se originalni in steganografsko obdelani medij na pogled ne razlikujeta. Pogost pristop k steganografiji je skrivanje sporočil v najmanj pomembne bite medija, t.i. LSB steganografija. Pri tem lahko tudi definiramo kateri deli medija bodo uporabljeni. Če se sporočilo skrije kar po vrsti od začetka, gre za preprosto LSB steganografijo. Obstaja pa tudi taka, ki uporablja kodirne ključe, ki skrito sporočilo razpršijo po celotnem krovnem mediju. Za odkrivanje skritih podatkov iz steganografskih medijev obstaja mdr. tehnika lociranja tovora (ang. payload location). Le-ta računa slike ostankov na podlagi originalnega in steganografsko obdelanega medija. Uporabna je, da izvemo kateri biti so bili spremenjeni, ponavadi pa ne zna sama po sebi ugotoviti zaporedja bitov, posebej če je bil uporabljen kodirni ključ. V članku pokažemo, da zaporedje vendarle lahko rekonstruiramo iz pričakovanih srednjih vrednosti slike ostankov (ang. expected mean residuals). V članku tudi pogledamo nekaj primerov steganografske programske opreme, ki je na voljo za prosto uporabo in primerjamo njihove rezultate.

1.17 Analiza stisnjenega RAM-a na operacijskih sistemih Mac OS X in Linux

V forenziki se čedalje bolj uporablja analiza računalniškega pomnilnika z namenom izboljšanja že uveljaljenih forenzičnih metod za pregledovanje podatkovnih nosilcev, saj je v neobstojnem pomnilniku veliko

število informacij, ki jih na statičnih pomnilnikih ni. Z zajetjem neobstojnega pomnilnika dobimo vpogled v delujoče stanje sistema, informacije o programih, ki se izvajajo, podatke o tekočih internetnih povezavah in še več. Še vedno pa nastopajo težave pri analizi izmenjevalnih datotek (ang. swap file), ki običajno kot particije na disku služijo kot dodatni prostor za shranjevanje informacij, do katerih lahko potem po potrebi dostopa operacijski sistem. Čeprav so lahko izmenjevalne datoteke bogat vir informacij, je njihova analiza vsebovala le iskanje znakovnih nizov in majhnih binarnih struktur. Bolj napredna analiza izmenjevalnih datotek je otežena zaradi vse bolj pogoste uporabe enkripcije na izmenjevalnih datotekah in zajemanja medsebojno skladnih izpisov pomnilnika in izmenjevalnih datotek. Vendar novejši operacijski sistemi skušajo zmanjšati izmenjavanje na disk z uporabo stiskanja. Hramba stisnjenih strani v RAM-u poveča učinkovitost delovanja, poleg tega pa predstavlja novo možnost zajemanja digitalnih dokazov, ki so bili v preteklosti izmenjani na disk. Ta članek razpravlja o težavnosti analize izmenjevalnih datotek v večjih detajlih, možnostih stisnjenega RAM-a na operacijskih sistemih Mac OS X in Linux ter o novih orodjih za analizo le tega.

1.18 Skrivanje podatkov v šifriranih video vsebinah H.264/AVC s pomočjo zamenjave kodnih besed

Digitalne video vsebine lahko shranimo v šifrirani obliki. Na ta način zagotovimo večjo varnost in ohranimo zasebnost. V namen označevanja vsebin in zaznave nedovoljenih posegov oziroma podatkovih sprememb, šifriranim video enotam v nadaljevanju dodamo podatke, ki so skriti. S tem ohranimo zaupnost vsebine brez potrebe po dešifraciji. Omenjeni postopek je zato hitrejši in boljši od podobnih tehnik skrivanja podatkov. V besedilu raziščemo tehniko skrivanja podatkov neposredno v šifrirano obliko H.264/AVC video vsebin. Opisana tehnika vključuje podatkovno šifriranje, vstavljanje podatkov in njihovo izvlečenje. Pri šifriranju in skrivanju se osredotočimo na tri lastnosti kodeka H.264/AVC, ki so intra-napovedni načini, razlika vektorjev in koeficiente ostankov, kjer iskoriščamo značilnosti zapisa nihovih kodnih besed. Neodvisno od pomena video vsebine lahko potem vstavimo oziroma skrijemo podatke z uporabo tehnike zamenjave kodnih besed. Izvlečenje skritih podatkov je sicer možno, če so podatki šifrirani ali ne. Količina podatkov, ki jih lahko skrijemo je odvisna od same dolžine in značilnosti (količine gibanja, količine tekstur) video zapisa. Glavna prednost predlaganega postopka je, da ta ne spremenijo velikosti končnih datotek.

1.19 Samodejno prepoznavanje kopiranih kriminalnih spletnih strani z metodo gručenja

V interesu spletnih kriminalcev je, da pride v stik z njihovo prevaro kar se da veliko število ljudi. S tem v mislih si želijo čim ceneje izdelati veliko skupino spletnih strani, ki so vsaj na videz čim bolj različne. Ker pa je izdelava spletne strani vsakič na novo enostavno predraga, za izdelavo uporabljajo že narejene predloge ali pa delčke preteklih spletnih strani, kar vključuje HTML, CSS in JavaScript kodo, še bolj pogosto pa njihovo zvito sestavljeno in kar se da prepričljivo besedilo. Uporabili bomo metodo nenadzorovanega strojnega učenja za združevanje podobnih spletnih strani v gruče. Na ta način lahko

ugotovimo, katere izvirajo iz iste kriminalne združbe. Kakovost rezultatov bomo tudi ovrednotili, tako da bomo rezultate primerjali z mnenjem človeškega poznavalca te tematike.

Del I

Digitalna forenzika na diskih

Distributed filesystem forensics: XtreemFS as a case study

Predstavitev postopka forenzične preiskave datotečnega sistema XtreemFS

Klemen Možina
Fakulteta za računalništvo in informatiko
Ljubljana, Slovenija
km4054@student.uni-lj.si

Dejan Kostadinovski
Fakulteta za računalništvo in informatiko
Ljubljana, Slovenija
dk9158@student.uni-lj.si

Domen Petrič
Fakulteta za računalništvo in informatiko
Ljubljana, Slovenija
dp5304@student.uni-lj.si

POVZETEK

Članek [6] opisuje postopek forenzične preiskave porazdeljenih datotečnih sistemov v oblaku, pri čemer se osredotoči na znan datotečni sistem XtreemFS. Porazdeljeni datotečni sistemi nam nudijo cenovno ugodne in efektivne rešitve za shranjevanje velikih količin podatkov v oblaku. Tudi te tehnologije so lahko izkoriščene v zlonamerne namene, zato potrebujemo postopke in orodja za digitalno forenzično preiskavo. V članku avtorja uporabita predlagan splošen postopek preiskave porazdeljenih datotečnih sistemov na sistemu XtreemFS, pri čemer ga razširita s specifičnimi lastnostmi sistema XtreemFS, ki jih je potrebno upoštevati pri zbiranju dokaznega gradiva. Avtorja na koncu članka tudi predlagata natančen proces zbiranja digitalnih dokazov iz porazdeljenih datotečnih sistemov.

Ključne besede

XtreemFS, porazdeljeni datotečni sistemi, forenzična preiskava, računalništvo v oblaku

1. UVOD

V zadnjih letih se močno povečuje količina zajetih in shranjenih podatkov v elektronski obliki, zato postaja t.i. big data eden izmed glavnih tehnoloških trendov. Po raziskavi podjetja Gartner [1] bo big data tudi v naslednjih letih velik vir dobička podjetij. V zadnjih letih se kot platforma za hranjenje in analizo velikih količin podatkov uporabljajo razni ponudniki oblačnih storitev (IaaS), kot je na primer Amazon. Računalništvo v oblaku je kot tudi druga omrežna infrastruktura, prav tako izpostavljena raznim kriminalnim dejanjem, zato potrebujemo proces, imenovan digitalna forenzika, da lahko pridobimo dokazno gradivo o zločinu. Glavna težava preiskave računalniških okolij v oblaku in porazdeljenih datotečnih sistemov pa je v porazdeljenosti podatkov po raznih podatkovnih centrih po celem svetu, do katerih je ponavadi težko dobiti dostop v preiskavi (in pri tem identificirati vse sisteme, ki vsebujejo dokazno gradivo).

Za področje digitalne preiskave porazdeljenih sistemov obstaja kar nekaj priročnikov, ki pa so večinoma usmerjeni na SaaS oblačne storitve (npr. Dropbox in Google Drive) in večinoma ne obravnavajo podrobnih postopkov zbiranja dokaznega gradiva iz. oblačnih strežnikov oz. porazdeljenih datotečnih sistemov. Ta članek uporablja XtreemFS kot za prikaz tehničnih in procesnih težav pri zajemu dokaznega gradiva iz porazdeljenih datotečnih sistemov, ki se v oblačnih storitvah pogosto uporabljajo.

V naslednjem poglavju je predstavljeno področje, ki ga zajema članek, sledita mu poglavji s predstavitvijo splošnega procesa preiskave digitalnih sistemov in datotečnega sistema XtreemFS. Nato sledijo ugotovitve izvedenega postopka preiskave sistema XtreemFS in predlagan splošen postopek zbiranja dokazov iz porazdeljenih datotečnih sistemov.

2. PREGLED PODROČJA

2.1 Računalništvo v oblaku

NIST(National Institute of Standards and Technology) je računalništvo v oblaku opredelil kot model, ki omogoča priročen dostop do skupnih razpoložljivih računalniških virov, ki jih je mogoče z minimalnim naporom in brez posebne interakcije uporabnika hitro omogočiti (zagnati) in sprostiti. Računalništvo v oblaku se lahko šteje za novo paradigmo, saj prinaša večjo prilagodljivost in razpoložljivost pri nizkih stroških. Posledično je v zadnjem času prejelo velik del pozornosti. [5].

Do sedaj so bili razviti trije storitveni modeli:

- *Software-as-a-Service (SaaS)*
- *Platform-as-a-Service (PaaS)*
- *Infrastructure-as-a-Service (IaaS)*

in obstajajo štirje načini uvajanja (deploying) računalništva v oblaku [7]:

- *Zasebni oblak*
- *Javni oblak*
- *Skupni oblak*
- *Hibridni oblak*

Čeprav je znižanje stroškov primarna motivacija za premik k ponudniku oblačne storitve, to ne bi smelo vplivati na zmanjšanje varnosti in zasebnosti. S stališča varnosti je potrebno zagotoviti zaupljivost, arhitekturno upravljanje, identiteto, izolacijo opreme, varovanje podatkov in razpoložljivost.

2.2 Značilnosti oblačnih storitev

Po članku [4] oblačne storitve zaznamujejo predvsem sledeče značilnosti, zaradi katerih je računalništvo v oblaku postalo pomemben člen v gospodarstvu.

- *Večnajemniško okolje oblaka* - Zaradi tega načrtovalskega pristopa, se lahko vire učinkovito dodeljuje uporabnikom storitev, saj omogoča skalabilnost, dostopnost in upravljanje.
- *Elastičnost* - Možnost dodeljevanja in odvzemanja virov po potrebi in s tem boljšega izkoriščanja le teh.
- *Sprotno plačevanje* - Vire plačujemo glede na to koliko smo jih količinsko in časovno potrebovali.
- *Zanesljivost* - Uporabnika za zanesljivost ne skrbi. Omogoča se zamenjavo nedelujočih virov, brez da bi uporabnik za to vedel.

2.3 Ponujene storitve in namestitve

Podjetja se glede na zaupnost podatkov in cenovne ugodnosti storitev računalništva v oblaku odločajo za različne modele namestitve.Obstajajo javni oblaki, katere ponujajo velika podjetja kot so Google, Amazon in drugi. Za te oblake je značilno, da si ponudniki storitev lastijo infrastrukturo, katero nato ponudijo uporabnikom. Ponudniki niso vedno osredotočeni le na prodajo podjetjem ampak tudi na prodajo storitev v oblaku posameznikom. Kot opisuje članek [2] so običajno take storitve cenejše kot privatne namestitve. Za privatne namestitve je značilno, da so v lasti iste organizacije, ki oblak uporablja. Takšno namestitev ponavadi zaznamujejo iste značilnosti kot javne. Zavedati pa se je treba, da morajo ustanove, ki si namestijo privatni oblak le tega vzdrževati. Obstaja pa še srednja pot katero imenujemo hibridni oblak. Ta način namestitve omogoča, združevanje privatnega in javnega oblaka.

Pri ponudbi računalništva v oblaku se je glede na izkušenost in potrebo uporabnika razvilo nekaj modelov storitve. Uporabnik lahko v najem dobi oddaljen virtualiziran računalnik, na keterega sam namesti operacijski sistem. Uporabnik mora sam poskrbeti za varnost ter trajnost podatkov. Tak model imenujemo Infrastruktura kot storitev (IaaS). Primer takega modela je Windows Azure. Naslednji model se imenuje platforma kot storitev (PaaS). Uporabnik v tem primeru razvije aplikacije z virtualnim razvojnim okoljem in jih nato namesti v oblak. Te aplikacije izkoristijo oblak za izvajanje ter so ponujene kot storitve. Primer take storitve je IBM Bluemix. Tretji model storitev v oblaku je Aplikacija kot storitev (SaaS). Uporabnik uporablja storitve, ki jih ponudnik ponuja preko omrežja in se izvajajo v oblaku. Pogosto so storitve dosegljive preko brskalnika. Primer take storitve je Microsoft Office as a Service.

V članku bomo podrobneje preučevali XtreemFS, ki spada v model infrastrukture kot storitve.

2.4 Forenzične preiskave v oblaku

Odkrivanje in pridobivanje dokazov od oddaljene, elastične in s strani ponudnika kontrolirane oblačne platforme se razlikuje od tradicionalne digitalne forenzične preiskave [3].

Digitalnim preiskovalcem največkrat manjkajo primerna orodja za izvedbo digitalnih oblačnih forenzičnih preiskav. Če nimajo zagotovljene alternativne tehnike in orodij, se preiskovalci zanašajo na svoje obstoječe strokovno znanje v orodjih kot so Guidance, EnCase ali AccessData Forensic Toolkit (FTK).

Digitalna forenzika za računalništvo v oblaku prinaša nove tehnične in pravne izzive. Računalništvo v oblaku naredi forenzično preiskavo drugačno, zlasti glede na oddaljeno naravo dokazov, pomanjkanje fizičnega dostopa ter zaupanje, ki je potrebno pri celovitosti in avtentičnosti podatkov. Medtem ko so cilji forenzičnega preiskovalca enaki kot prej, nekonvencionalni problemi vključujejo težko pridobivanje podatkov na daljavo, velike količine podatkov, porazdeljene in elastične podatke, sledljivost in lastništvo podatkov.

Zaseg in pridobitev digitalnih artefaktov so začetni koraki v forenzičnem procesu [3]. Obstajata dva možna scenarija: preiskovalci na daljavo lahko sami zbirajo forenzična dokazila od določenega vira, ali pa ta dokazila dostavijo ponudniki. Vsak scenarij zahteva drugačno stopnjo zaupanja pri dostavljanju podatkov. Nadalje, vsak scenarij uporablja drugačne tehnične izvedbe pri obnovitvi podatkov.

Narava spletne oddaljene forenzike uvaja nova varnostna razmišljanja. Na primer [3], forenzična delovna postaja mora imeti dostop do interneta, da lahko preiskovalec pridobi dokaze. Ukrepi, kot so požarni zidovi in strežniki proxy, pomagajo pri zaščiti delovne postaje pred napadom. Vendar možnost okužbe podatkov postane s tem bolj verjetna, kot če delovna postaja ne bi bila povezana z omrežjem, ali pa bi bila v izoliranem omrežju. To tveganje je treba sprejeti ali sanirati z ustrezno tehnologijo (spremljanje, popravljanje, itd.).

3. OGRODJE ZA FORENZIčNO PREISKAVO SISTEMOV V OBLAKU

V članku je kot splošen postopek preiskave porazdeljenih oblačnih sistemov uporabljeno ogrodje za forenzično preiskavo sistemov v oblaku (*Cloud forensics framework*), ki sta ga avtorja originalnega članka [6] razvila že v okviru prejšnjih raziskav. Postopek sledi splošnim korakom preiskave digitalnih naprav, poudarjena pa je iterativna narava, ki je pomembna za uspešno preiskavo kompleksnih okolij, kot je XtreemFS. V splošnem se najprej opravi preiskava odjemalca, ki se uporabi za identifikacijo oblačnih storitev in zajem podatkov na samem odjemalcu. Nato se izvede analiza strežniških okolij. Postopek forenzične preiskave poteka v štirih korakih:

1. Identifikacija virov dokaznega gradiva in ohranjanje: v prvi iteraciji se ponavadi uporabi odjemalca datotečnega sistema za identifikacijo, ki nas ponavadi vodi do drugih komponent sistema. V drugi iteraciji se identificira še druge komponente sistema, ki bi bili morda relevantni za primer in se opravi proces ohranjanja (za-

varovanja) dokaznega gradiva.

2. Zbiranje (zajem) dokazov: ta faza se ukvarja z dejanskim zajemom dokaznega gradiva.

3. Preiskovanje in analiza: faza se ukvarja s preiskovanjem in analizo dokaznega gradiva, ki sta ključni fazi forenzične preiskave porazdeljenih datotečnih sistemov. Preiskava je pomembna za razumevanje komponent sistema, analiza pa za njegovo rekonstrukcijo.

4. Predstavitev: faza se ukvarja s predstavitvijo zbranih dokazov.

4. XTREEMFS

XtreemFS je porazdeljen datotečni sistem, ki ponuja storitve za shrambo podatkov ponudnikom oblačnih storitev (npr. shramba virtualnih naprav), s tem da omogoča razne storitve kot sta replikacija in porazdeljevanje datotek. Da bi ponudil replikacijo podatkov, odporno na napake, so podatki navadno porazdeljeni in replicirani po raznih strežnikih oz. podatkovnih centrih.

Glavni značilnosti XtreemFS sta porazdeljen in repliciran datotečni sistem, za dosego tega pa se uporabljajo tri glavne komponente sistema: direktorijska storitev (DIR - *Directory Service*), katalog metapodatkov in replik (MRC - *Metadata and Replica Catalog*) in naprave za hranjenje objektov (OSD-ji - *Object Storage Devices*).

DIR je pristojen za vzdrževanje registra vseh storitev in logičnih diskov oz. volumnov (*volumes*), ki jih ponuja storitev XtreemFS. Zato ostali deli arhitekture XtreemFS pogosto komunicirajo s storitvijo DIR in posledično je DIR dober vir podatkov za identifikacijo lokacije ostalih komponent, ki so povezane v okolje (npr. OSD-ji).

MRC je pristojen za shranjevanje in vzdrževanje metapodatkov glede shranjenih datotek (podatkov). Posledično je tudi MRC dober vir za forenzično preiskavo.

OSD je pristojen za hrambo dejanskih podatkov, ki jih odjemalci pošljejo instanci XtreemFS. OSD je torej glavna komponenta forenzične preiskave, saj vsebuje vsebino podatkov, ki jih je odjemalec naložil v virtualni datotečni sistem. V praksi imamo znotraj ene XtreemFS instance več OSD-jev, saj nam to omogoča replikacijo in porazdelitev podatkov po več fizičnih napravah. XtreemFS uporablja koncept logičnih diskov oz. volumnov za virtualno segregacijo podatkov. Vsak volumen ima lahko določene različne pravice in politike dostopa in je primarna enota za odjemalce, ki z njimi rokujejo (npr. odjemalci priključijo volumen, jim nastavljajo pravice,...).

XtreemFS odjemalec se uporablja za povezavo z datotečnim sistemom in je odgovoren za vse administrativne operacije nad sistemom.

5. PREISKAVA SISTEMA XTREEMFS

V tem poglavju so opisane glavne komponente sistema v okviru digitalne preiskave le-teh.

5.1 Direktorijska storitev (DIR)

DIR vsebuje informacije za identifikacijo in lokacijo ostalih komponent in je zato logična začetna točka preiskave. Na DIR obstajajo naslednji tipi podatkov:

- Spremenljivi podatki okolja: lokacija (IP naslovi) in unikatni identifikatorjih vozlišč datotečnega sistema (UUID), tipi vozlišč, informacija o lastniku in konfiguracija.
- Nespremenljivi podatki okolja: varnostne kopije zapisov,
- Konfiguracijske datoteke: omrežna konfiguracija (porti, naslovi), avtentikacijske informacije, formati podatkovnih baz,...

V prvi fazi preiskave (Identifikacija vira dokazov) se DIR uporablja za najdbo in dekodiranje podatkov o ostalih komponentah sistema. Najprej je seveda potrebno lociranje DIR komponente, kar opravimo z analizo priklopljenega datotečnega sistema na odjemalcu. Če to ni možno in ima preiskovalec dostop do mreže, v kateri se naj bi nahajal DIR, lahko izkoristi DIR avtodetekcijo. MRC in OSD komponente lahko namreč detektirajo DIR z uporabo UDP broadcasta. OSD pošlje UDP paket na broadcast naslov LAN segmenta na vrata DIR storitve (32638). Lahko pa tudi pošlje omenjen paket nazaj na broadcast naslov na vrata OSD-ja (32640). To povzroči, da vse OSD komponente odgovorijo z napako in tako lahko pridobimo naslove vseh OSD-jev. Ko ima preiskovalec dostop do DIR sistema, se naj osredotoči na konfiguracijsko datoteko (*/etc/xos/xtreemfs/dirconfig.properties*) s prej omenjenimi informacijami. DIR ponuja tudi HTTP storitev za pregled statusa in nastavitev sistema sistemskemu administratorju (dostopna na vratih 30638). Privzeto je dostopna brez avtentikacije in ponuja pregled naslednjih informacij:

- Preslikava IP naslovov k idenfitikatorjem storitev.
- Register storitev: informacije o vsaki komponenti, vključno s tipom storitve (OSD, VOLUME,...), ime (ponavadi UUID), status (online, locked) in datum zadnjega dostopa. Preiskovalec lahko tu ugotovi, katera storitev se skriva za določenim UUID.
- Konfiguracije storitev: porti, omogočanje SSL-a, avtentikacija,...

V naslednjih fazah preiskovalec najprej opravi zajem prej najdenih podatkov. Kot najlažji način se izkaže zajem preko prej omenjenega HTTP vmesnika. Če pa preiskovalec ne preiskuje delujočega DIR strežnika (recimo le kopijo strežnika), pa mora preiskati BaduDB podatkovno bazo za temi metapodatki (*/var/lib/xtreemfs/dir/*). Kot najboljši način preiskave se izkaže rekonstrukcija podatkovne baze iz njenih zgodovinskih slik (snapshots) v podmapi *db-log*. Storitev DIR vsebuje loge (zapise operacij) na lokaciji */var/log/xtreemfs/dir.log*, ki pa privzeto ne vsebujejo veliko podatkov.

5.2 Katalog metapodatkov in replik (MRC)

MRC vsebuje veliko podatkov, ki se nanašajo na volumen in datoteke oz. direktorije, ki so shranjeni v njih. To vključuje

podatke o številu OSD-jev in njihovih lokacijah ter UUID-jih, po katerih je datoteka porazdeljena oz. replicirana in nizkonivojske podatke o samih datotekah (ime, velikost, časi, dovoljenja,...). Uporaba teh podatkov v kombinaciji s prej zajetimi omogoča preiskovalcu boljše razumevanje podatkov v tej XtreemFS instanci. Glavni podatki v MRC so:

- Metapodatki konstruktov: definicija internih konstruktov XtreemFS (volumni)
- Metapodatki datotek: prej omenjeni podatki o OSD-jih in samih datotekah.
- Konfiguracijske datoteke: omrežna konfiguracija (porti, naslovi), avtentikacijske informacije, formati podatkovnih baz,...

V prvi fazi preiskave (Identifikacija vira dokazov) preiskovalec najprej pregleda konfiguracijo storitve (da jo preveri s tisto v DIR), ki se nahaja na lokaciji */etc/xos/xtreemfs/mrcconfig.properties*. Poleg teh konfiguracij se na MRC nahajajo še konfiguracije o posodabljanju časov datotek (dostopa,...). Tudi MRC ponuja HTTP storitev za dostop do statusa in konfiguracije te storitve (vrata 60636). Tu pridobimo podatke o lokaciji storitve DIR, statistike o prometu in zahtevah ter informacije o logičnih diskih. Slednja je za preiskovalca še posebej zanimiva, saj mu lahko pove tip volumna v smislu politik porazdeljevanja, repliciranja in dostopa, število datotek in količino zasedenega prostora na disku. MRC je tudi pristojen za avtentikacijo dostopa do storitev. Ponuja dva tipa avtentikacije: *"NullAuthProvider"*, ki se zanaša na podatke, ki jih posreduje odjemalčev operacijski sistem in uporabo X.509 certifikatov.

Faza zbiranja dokazov je tudi zelo pomembna, saj so omenjeni metapodatki potrebni za rekonstrukcijo datotek, ki so porazdeljenih po več OSD-jih. Možni sta dve metodi zbiranja podatkov MRC. Kot že omenjeno, je ena možnost rekonstrukcija BaduDB podatkovne baze in dostop do podatkov preko API-jev (v primeru, da strežnik ne deluje). Obstaja pa tudi bolj preprosta možnost zbiranja podatkov z uporabo orodja xtfsmrcdbtool, ki je del paketa xtreemfs-tools. Orodje nam omogoča izvoz baze v XML format in obnovitev baze iz njega. Primer uporabe ukaza:

```
xtfs_mrcdbtool -mrc pbrpc://localhost:32636 dump
 /tmp/mrcdump.xml
```

V fazi preiskovanja in analize dokazov lahko pridobljeno XML datoteko delno analiziramo z uporabo urejevalnika tekstovnih datotek. Lahko pa seveda uporabimo oz. sprogramiramo svoj razčlenjevalnik XML datotek. Struktura XML dokumenta je naslednja (navedeni so elementi):

- FILESYSTEM (korenski element)
  - DBVERSION (identifikator verzije pod. baze)
  - VOLUME: predstavlja posamezen logičen disk. Ima atribute UUID, NAME (ime) in ACPOLICY (identifikator politike dostopa)
  - DIR: predstavlja posamezen direktorij v volumnu (prvi vnos je korenski direktorij). Ima atribute ID, NAME, UID in GID (lastnik - uporabnik in skupina), ATIME, CTIME, MTIME (čas dostopa, kreiranja in spreminjanja, predstavljeni kot POSIX zapisi), RIGHTS (numerična predstavitev POSIX pravic dostopa. Določeno pravico predstavlja posamezna številka (potence števila 2, ki naraščajo), vrednost RIGHTS se izračuna kot vsota teh številk, ki predstavljajo posamezno pravico (RWX za posamezno osebo oz. skupino).
  - ATTRS: ki vsebujejo elemente ATTR, ki so tipa ključ-vrednost, kodirani v Base64 formatu. Vsebujejo podatke o omogočanju izdelovanja posnetkov volumna, OSD politiko, in politiko porazdeljevanja volumna (striping policy)
  - FILE: predstavlja metapodatke datoteke, ki se nahaja v posameznem direktoriju. Vsebuje atribute ID, NAME, SIZE (velikost), EPOCH, ISSUEDEPOCH, UID in GID (kot v DIR), ATIME, CTIME, MTIME in RIGHTS (vrednost se izračuna podobno kot pri vnosu DIR, le da se pri datotekah izračunana vsota pravic še odšteje od števila 32768) in READONLY.
  - XLOCLIST, ki vsebuje elemente XLOC s podatki o lokacijah delov porazdeljenih datotek. Vsebuje atribut PATTERN, ki pove, na kakšen način se datoteka razdeli.
  - OSD, ki z atributom LOCATION pove lokacijo dela datoteke (UUID OSD-ja)

5.3 Naprava za shranjevanje objektov (OSD)

OSD sistemi so jedro forenzične preiskave sistema XtreemFS, saj vsebujejo dejanske datoteke oz. njihove dele. OSD-ji nam tudi omogočajo obnovitev izbrisanih datotek z uporabo znanih forenzičnih pristopov. OSD-ji hranijo podatke dveh tipov:

- Podatke iz datotek: shranjene datoteke in zapisi.
- Konfiguracijske datoteke

V fazi identifikacije virov dokazov lahko z do sedaj pridobljenim znanjem o XtreemFS sistemu preko DIR in MRC storitev lociramo ustrezne fizične OSD naprave. Pomembna je tudi v tem primeru konfiguracijska datoteka na lokaciji */etc/xos/xtreemfs/osdconfig.properties*, ki poleg osnovnih konfiguracij vsebuje še nekatere za preiskavo zanimive podatke. Najpomembnejša direktiva je *object_dir*, ki vsebuje lokacijo dejanskih podatkov (hrambe podatkov) na napravi.

V fazi zbiranja podatkov je najpreprostejši način za pridobitev datotek uporaba XtreemFS odjemalca, s čimer priklopimo datotečni sistem in zbiramo relevantne datoteke. Če te metode ni možno uporabiti, pa moramo s podatkov iz MRC in DIR sistema ročno locirati in pridobiti datoteke. Preiskovalec lahko zbere dele datotek in jih z uporabo metapodatkov logično združi v datoteke in jih kopira na zunanjo shrambo ali pa naredi fizično sliko naprave za hrambo. Prav pridejo tudi zapisi (dnevniki), ki hranijo podatke o opravljenih datotečnih operacijah v sistemu.

Object_dir direktorij vsebuje množico podmap in datotek z metapodatki, ki se nanašajo na posamezne dele datotek. Strukturirani so kot hierarhija direktorijev z dvema heksadecimalnima številoma v imenu
(npr. [object_dir]/AA/BB/CC/DD/...), v zadnjem direktoriju je še en direktorij, ki ima ime sestavljeno iz UUID logičnega diska (volumen) in ID-ja datoteke. V tem direktoriju se nahajajo dejanski deli datotek. Izkaže se, da je lokacijo datotek (hierarhijo direktorijev s hekasdecimalnimi vrednosti) možno izračunati, in sicer se to izvede tako, da se izračuna kontrolna vsota imena zadnjega imenika (ki je sestavljen iz UUID volumna in ID datoteke). Če je hash vrednost npr. D0BEB653, se bo zadnji imenik z deli datoteke nahajal v [obj_dir]/D0/BE/B6/53/. Imena delov datotek so sestavljena iz treh blokov po šestnajst heksadecimalnih števil (skupaj 48 heksadecimalnih števk). Prvi blok predstavlja "ObjNo" (zaporedno številko dela datoteke), drugi "ObjVersion" in tretji "checksum" dela datoteke.

5.4 XtreemFS odjemalec

Odjemalske aplikacije XtreemFS nudijo pomembno orodje za pridobivanje dokazov iz porazdeljenega datotečnega sistema. Najpreprostejši način preiskave je priklop (mount) XtreemFS sistema preko odjemalca in ekstrakcija podatkov iz njega. Z uporabo orodja "xtfsutil" pa lahko pridobimo tudi vse pomembne metapodatke logičnih diskov, direktorijev in datotek.

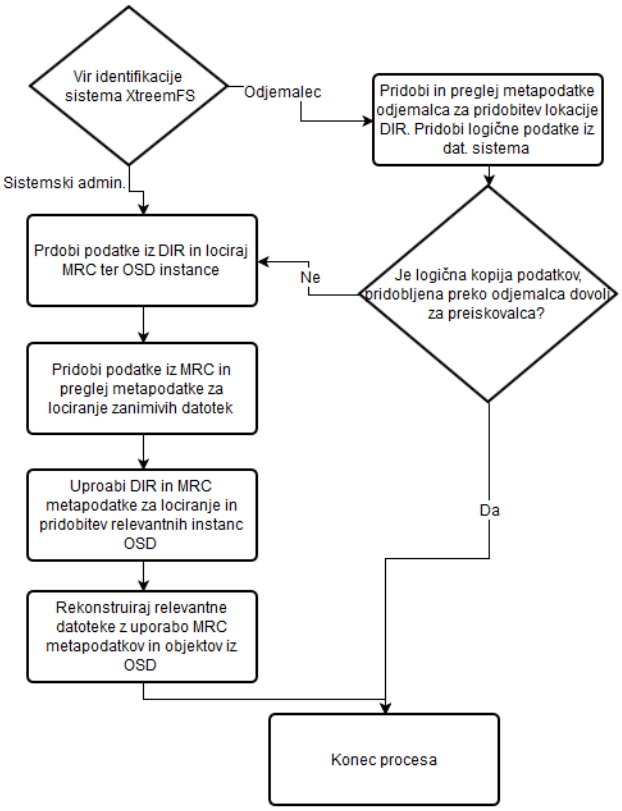
5.5 Povzetek korakov preiskave sistema XtreemFS

V tem podpoglavju je predstavljen celoten proces preiskave sistema XtreemFS, ki ga lahko izvede preiskovalec. Če je le mogoče, naj preiskovalec pri preiskavi uporabi odjemalca, saj to bistveno pohitri preiskavo. Preiskavo sestavljajo naslednji koraki:

- Identifikacija uporabe XtreemFS: v prvih korakih preiskovalec preišče XtreemFS instanco. Največkrat se začne z lociranjem XtreemFS odjemalca oz. z obvestilom sistemskega administratorja, da se uporablja XtreemFS kot datotečni sistem. V primeru XtreemFS odjemalca naj preiskovalec z uporabo le-tega locira DIR in ustvari logično kopijo podatkov, ki so priklopljeni na odjemalca. Pregleda naj tudi zapise odjemalca. Če je preiskovalec zadovoljen s pridobljenimi logičnimi podatki, lahko na tem mestu zaključi proces pridobivanja podatkov (ni potrebno identificiranje še ostalih sistemov v XtreemFS). V nasprotnem primeru pa nadaljuje z naslednjim korakom procesa.

  Če preiskovalec začne preiskavo s posvetovanjem s sistemskim administratorjem glede okolja (ki omeni uporabo XtreemFS), pa naj preiskovalec od njega pridobi podatke o lokaciji strežnikov in avtentikacijske podatke (certifikate, gesla,...) in če je možno, dostop do XtreemFS odjemalca, nastavljenega na ciljni datotečni sistem. Kot pa je bilo že omenjeno, lahko uporabo XtreemFS ugotovi s pošiljanjem paketkov za samodejno zaznavo po omrežju.

- Zbiranje in preiskava podatkov iz DIR: ko je bil DIR najden in lociran, naj preiskovalec z uporabo HTTP storitve pridobi podatke o statusu DIR in lokacijah MRC ter OSD sistemov v tem XtreemFS okolju.



Slika 1: Proces preiskave sistema XtreemFS

- Zbiranje in preiskava podatkov iz MRC: ko pridobi dostop do MRC-ja, preiskovalec pridobi omenjene datoteke z metapodatki iz podatkovne baze z uporabo orodja *xtfs_mrcdbtool*, če je mogoče. Ko pridobi metapodakte, jih preiskovalec analizira z namenom ugotovitve, kateri volumni, direktoriji in datoteke so zanimivi in še niso bili pridobljeni preko odjemalca.

- Zbiranje, preiskava in analiza podatkov iz OSD-jev: ko identificira relevantne OSD-je iz podatkov DIR (lokacija) in MRC (relevantni deli datotek), lahko preiskovalec pridobi relevantne podatke. Primaren vir podatkov so deli datotek, ki tvorijo posamezne datoteke na datotečnem sistemu. Te podatke lahko pridobimo z uporabo datotečnega sistema gostitelja (npr. ext4). Lahko pa ustvari forenzično sliko diska za analizo.

6. POSTOPEK PRIDOBIVANJA DOKAZOV IZ PORAZDELJENIH DATOTEčNIH SISTEMOV

V tem poglavju je opisan splošen postopek pridobivanja dokazov iz porazdeljenih datotečnih sistemov. Proces je sestavljen iz treh delov:

- Direktorijske storitve: lociraj direktorijske storitve, pridobi podatke in jih preglej, da ugotoviš komponente sistema v uporabi in njihove lokacije.

- Hramba metapodatkov: z uporabo direktorijskih storitev lociraj metapodatkovni strežnik, pridobi metapodatke in jih preglej, da ugotoviš, katere datoteke, mape in logični diski so zanimivi za preiskavo

- Hramba podatkov: z uporabo do sedaj zbranih podatkov o okolju sistema, pridobi relevantne naprave za hranjenje podatkov oz. logične podatke in jih rekonstruiraj z uporabo metapodatkov.

V sledečih podpoglavjih so vsi koraki podrobneje predstavljeni.

6.1 Direktorijske stortive

Direktorijske storitve hranijo metapodatke o vozliščih porazdeljenega podatkovnega sistema. V primeru XtreemFS je storitev implementirana v obliki centraliziranega DIR strežnika. Lahko pa so implementirane tudi kot bolj porazdeljen model. Neodvisno od implementacije mora preiskovalec najprej najti direktorijsko storitev (metapodatke sistema). Z uporabo teh podatkov bo bolje razumel delovanje sistema in pridobil lokacije vozlišč (IP naslovi). Ko so ti podatki najdeni, jih mora preiskovalec zavarovati (ohraniti) in pridobiti. Analiza teh podatkov sestoji iz dekodiranja (če je potrebno) in dokumentiranja pomembnih detajlov vozlišč (IP naslov, tip, identifikatorji,...).

6.2 Hramba metapodatkov

Z uporabo pridobljenih podatkov iz direktorijske storitve lahko preiskovalec identificira vire metapodatkov v okolju. Metapodatki so lahko shranjeni skupaj s podatki, pogosteje pa so shranjeni v posebni podatkovni bazi. Preiskava metapodatkov (po zajemu) vključuje zmanjševanje podatkov, ki jih bomo morali pregledati, torej se fokusiramo le na metapodatke, ki nakazujejo na lastnika, ki je osumljenec oz. v povezavi z njim. Pomembni metapodatki so tudi začasne datoteke, hashi datotek,...

6.3 Hramba podatkov

Po opravljenih prejšnjih fazah ima preiskovalec dovolj podatkov za lociranje in rekonstruiranje datotek, ki so porazdeljene po datotečnem sistemu. Z uporabo do sedaj zbranih informacij se preiskovalec lahko odloči, ali bo izdelal bitno sliko (kopijo) naprav za hrambo in jo kasneje analiziral ali pa bo podatke pridobil logično (z uporabo odjemalskih aplikacij). V obeh primerih je potrebno paziti na ohranjanje dokazov (blokiranje pisalnega dostopa, ...). Če se je preiskovalec odločil za logično pridobivanje, lahko z uporabo metapodatkov rekonstruira datoteke.

7. ZAKLJUčEK

Z naraščanjem digitalizacije podatkov in uporabe oblačnih storitev za obdelovanje velikih količin podatkov (Big data) se tudi povečujejo možnosti za izkoriščanje teh podatkov v zlonamerne namene. Posledično je pomemben razvoj digitalne forenzike na tem področju.

V članku je bila podrobno predstavljena preiskava porazdeljenega datotečnega sistema XtreemFS, ki se pogosto uporablja v oblačnih storitvah. Glavne ugotovitve v članku so naslednje:

- Direktorijske storitve omogočajo preiskovalcu pregled nad samim sistemom, ki ga preiskujejo.

- Hrambe metapodatkov vsebujejo pomembne podatke za identifikacijo relevantnih datotek.

- Hhrambe podatkov pa vsebujejo same podatke, ki jih lahko preiskovalec rekonstruira z uporabo podatkov o sistemu in metapodatkov.

8. LITERATURA

[1] Top 10 technology trends impacting information infrastructure. `https://www.gartner.com/doc/2340315/top--technology-trends-impacting`. Obiskano: 7.5.2015.

[2] A view of cloud computing. `http://delivery.acm.org/10.1145/1730000/1721672/p50-armbrust.pdf?ip=88.200.105.31&id=1721672&acc=OPEN&key=4D4702B0C3E38B35.4D4702B0C3E38B35.4D4702B0C3E38B35.6D218144511F3437&CFID=673407145&CFTOKEN=61110282&__acm__=1431334524_cecb07af7ac21e40ce0322413f5fa3b8`. Obiskano: 9.5.2015.

[3] J. D. in Alan T. Sherman. Acquiring forensic evidence from infrastructure-as-a-service cloud computing: Exploring and evaluating tools, trust, and techniques. *Digital Investigation*, pages 91–92, 2012.

[4] A. H. in Anton Zvonko Gazvoda in Benjamin Kastelic. Tehnični izzivi pri forenzičnih raziskavah v oblaku. *Digitalna forenzika, Seminarske naloge 2012/2013*, pages 40–47, 2013.

[5] W. A. Jansen. Cloud hooks: Security and privacy issues in cloud computing. pages 1–5, 2011.

[6] B. Martini and K. R. Choo. Distributed filesystem forensics: Xtreemfs as a case study. *Digital Investigation*, 11(4):295–313, 2014.

[7] F. OGIGAU-NEAMTIU. Cloud computing security issues. pages 141–143.

Del II

Digitalna forenzika na omrežjih

Pregled "Asset Risk Scoring in Enterprise Network with Mutually Reinforced Reputation Propagation"

Matija Rezar
mr3193@student.uni-lj.si

POVZETEK

Članek *Asset Risk Scoring in Enterprise Network with Mutually Reinforced Reputation Propagation* predstavlja metodo za vrednotenje ogroženosti v omrežju imenovano MUSE (*Mutually-reinforced Unsupervised Scoring for Enterprise risk*). Pregledali bomo algoritme, na katerih je metoda osnovana in področje uporabe, ki mu je namenjena. Metoda uporablja ideje, ki so že znane iz analiz varnosti na spletu in jih uporabi za analizo omrežja v podjetju. Dobljene rezultate je mogoče uporabiti za bolj učinkovito zaznavanje groženj ter za pomoč pri preiskavi. Na koncu bomo predstavili še podrobnosti same metode in njeno analizo.

1. UVOD

Ogledali si bomo način, kako iz mase poročil, ki prihajajo iz različnih delov omrežja izluščiti uporabne podatke. V večjih omrežjih nastaja veliko sporočil o dogajanju in večina od teh nima velike vrednosti. Če pa želimo ohranjati varnost, pa je potrebno ugotoviti katera sporočila dejansko govorijo o varnostnih grožnjah in katera ne. To storimo tako, da entitetam v omrežju dodelimo ocene zaupanja in se nato odzivamo na njihova sporočila tem ocenam primerno.

S filtriranjem uporabnih informacij iz velike količine podatkov se že dolgo ukvarjajo spletni iskalniki. Ti morajo iz velikega števila spletnih strani ugotoviti, katere nosijo uporabne informacije. To pa jih otežujejo zlonamerna spletišča, katerih namen je s prevaro priti na vrh lestvice rezultatov pri iskanju.

Sporočila z omrežja uporabljamo zato, da v omrežju zaznavamo in preprečujemo vdore. Sistemi namenjeni temu nato iz različnih poročil o dogajanju v omrežju odkrivajo možne vdore in se nanje odzivajo.

2. PREGLED PODROČJA

Članek predstavlja srečanje področij varnosti in analize grafov. Iz grafa omrežja lahko ugotovimo, katerim entitetam lahko zaupamo, nato pa te informacije uporabljamo za preprečevanje vdorov.

2.1 IDPS

Sistemi za zaznavanje in preprečevanje vdora (*Intrusion Detection and Prevention Systems*) so sistemi, ki aktivno nadzirajo dogajanje v omrežju in se nanj odzivajo [7]. Upravljavce omrežja obveščajo o napadih, oziroma slednje zatirajo z zapiranjem povezav in dodajanjem pravil v požarno pregrado.

2.1.1 Načini zaznavanja

Sistemi lahko uporabljajo različne načine zaznavanja. Posamezna implementacija navadno uporablja več takih načinov hkrati [7].

Zaznavanje na podlagi podpisov. Sistem vdore zazna na podlagi v naprej predpisanih vzorcev. Ti vzorci so statični in osnovani na znanih napadih. Posluša za dogodke, kot so poskus prijave z uporabniškim imenom ”root”ali izvedljive datoteke v e-pošti [7].

Zaznavanje na podlagi anomalij. Takšen način zaznavanja potrebuje obdobje učenja. V tem času se sistem nauči, kaj je za omrežje običajno, nato pa zaznava dogodke, ki statično gledano niso običajni. Ker se omrežje s časom spreminja je potrebno tak sistem občasno ponovno naučiti. Lahko pa se tudi zgodi, da se napad zgodi v času učenja in se sistem nauči, da je tako dogajanje običajno. Primeri anomalij so velika količina e-pošte ali močno povečan promet storitve [7].

Analiza protokola s stanjem. Vdore je mogoče zaznati tudi na podlagi zlorabe protokola. V tem primeru sistem išče poskuse uporabe protokolov, na način, ki ga slednji ne dopuščajo. Na primer FTP v neavtenticiranem načinu ne dopušča določenih ukazov. Če poskuša nekdo te ukaze uporabiti gre verjetno za napadalca. Veliko protokolov zahteva hrambo stanja, zato je tudi za njihovo analizo potrebno hraniti stanje. Iščemo pa lahko tudi nelegalne oziroma nenavadne argumente, kot so na primer uporabniška imena s 1000 znaki [7].

Zaznavanje vdora lahko vršimo neposredno na strežnikih, na povezavah med napravami, na celotnem omrežju ali na brezžičnih povezavah [7].

Ker pa taki sistemi delujejo hevristično proizvedejo veliko rezultatov. Večina od teh je lažno pozitivnih, zato jih nima smisla podrobno analizirati. Tako sporočila končajo v dnevnikih in jih človek pregleda šele ob forenzični analizi [4].

2.2 Omrežje

V poslovnem omrežju je udeleženih veliko različnih entitet, ki komunicirajo ena z drugo. Močno pa se razlikujejo po tem, koliko jim zaupamo. Tako lahko sporočilom dodelimo različno težo, glede na to od kod prihajajo [4].

Zunanji strežniki. Zunanji strežniki so naprave, nad katerimi nimamo nadzora. Lahko gre za običajne spletne in e-poštne strežnika, ki so bili okuženi. Mogoče pa so to strežniki postavljeni z namenom škodovanja in vdorom.

Naprave. Tu gre za vse naprave nad katerimi imamo nadzor. Od naših strežnikov, do osebnih računalnikov in mobilnih naprav naših zaposlenih.

Uporabniki. Uporabniki so ljudje, ki uporabljajo naprave. Oceniti moramo kako bo njihovo obnašanje vplivalo na varnost omrežja.

Pooblastila. Različne entitete se morajo pogosto predstaviti ena drugi, zato je pomembno, da je zagotovljena varnost dokumentov s katerimi se predstavljajo. Tu gre predvsem za uporabniške račune in certifikate.

Sredstva z visoko vrednostjo. Posebno je treba izpostaviti sredstva, ki imajo visoko vrednost. To so prav tiste entitete, ki jih želimo zaščititi pred vdori. Višina tveganja na teh napravah je to kar nas najbolj zanima.

2.3 Zaznavanje lažnih vsebin

Splet in internet imata že od samega začetka težave z razločevanjem legitimnih in lažnivih vsebin. S tem so se spopadali prvi iskalniki, ki so želeli prikazati vsebino, ki ne laže o svoji legitimnosti. Razvite so bile različne metode odkrivanja in razvrščanja vsebina glede na stopnjo avtentičnosti [6].

PageRank. Uspeh iskalnega orjaka Google gre pripisati implementaciji razvrščevalnega algoritma, ki se ne zanaša na vsebino strani temveč na povezave na stran. Ne ozira se torej na to, kar stran pravi sama o sebi, temveč na to kaj drug strani pravijo o njej [1].

Ena od interpretacij delovanja PageRank algoritma je tako imenovani "naključni obiskovalec", ki se naključno sprehaja po povezavah. Pomembnost strani je torej določena kot verjetnost, da se ob določenem času obiskovalec nahaja na določeni strani. Če ima stran več vhodnih povezav bo verjetnost, da se bo obiskovalec nahajal tam večja. Občasno obiskovalec preneha s sprehodom in ga ponovno prične na naključni drugi strani [1].

TrustRank. Osnovni PageRank je dovzeten za goljufanje, saj lahko nepridipravi s pravilno povezanimi stranmi umetno zvišujejo svojo pomembnost. Zato je bil razvit algoritem, ki za osnovo vzame strani, ki jim zaupa [3].

Pri algoritmu TrustRank naključni obiskovalec preiskovanje pričenja na straneh, za katere ve, da so vredne zaupanja. Vsakič, ko konča sprehod se vrne na eno od zaupanja vrednih strani. Te mora v naprej določiti človek, ali pa zanje jamči nekdo, ki mu zaupamo, npr. .edu, .mil in .gov domene [6].

HITS. Tehnika HITS (*Hyperlink-Induced Topic Search*) je namenjena odkrivanju zaupanja vrednih virov v omrežju. Osnovana je na ideji, da v omrežju obstajajo vozlišča in zaupanja vredni viri. Boljša vozlišča so tista, ki kažejo na več zaupanja vrednih virov, oziroma na čim boljše vire, boljši viri pa so tisti, na katere kaže več vozlišč [5].

Vsaka entiteta v omrežju dobi oceno "vozliščnosti"in oceno zaupanja. Oceni sta enaki normalizirani vsoti zaupanja oziroma oceni entitet, s katerimi je dana entiteta povezana.

$$\boldsymbol{v} = \lambda \boldsymbol{S}\boldsymbol{z}$$
$$\boldsymbol{z} = \mu \boldsymbol{S}^T \boldsymbol{v}$$

$\boldsymbol{S}$ je matrika sosednosti, $\boldsymbol{z}$ je vektor zaupanja entitet, $\boldsymbol{v}$ vektor ocen vozliščnosti, λ in μ pa normalizirata vektorje.

Iz tega lahko dobimo naslednjo enačbo:

$$\boldsymbol{z} = \lambda\mu \boldsymbol{S}^T \boldsymbol{S}\boldsymbol{z}$$

Zaradi lastnosti matrike $\boldsymbol{S}$ vektor $\boldsymbol{z}$ konvergira v glavni lastni vektor matrike $\boldsymbol{S}^T\boldsymbol{S}$. Tako dobimo ocene zaupanja v posamezne entitete v omrežju [6].

3. ČLANEK

V članku avtorji opisujejo uporabo zgoraj navedenih metod za iskanje najbolj ranljivih delov omrežja. Tako pridobljene podatke je mogoče uporabiti tako pri vzpostavitvi varovanja, kot pri preiskavi.

3.1 Omrežje

Zanima nas kakšna je verjetnost, da neka entiteta v omrežju predstavlja varnostno grožnjo. Verjetnosti bomo izračunali iz začetnih vrednosti entitet za katere poznamo stopnjo zaupanja, ki jih bomo razširjali med ostale entitete v omrežju.

Omrežje lahko predstavimo kot k-partitni graf, ki ga sestavljajo: zunanji strežniki, naprave, uporabniki, pooblastila in sredstva z visoko vrednostjo, povezave v grafu pa predstavljajo odnose med njimi.

Zunanji strežniki - p_s. V tem primeru gre predvsem za strežnike, ki gostijo zlonamerno programsko kodo. Komu-

nikacija med takim strežnikom in drugo entiteto v omrežju ogrozi to entiteto in ji zvišuje stopnjo tveganja.

Naprave - p_d. Tveganje naprave predstavlja verjetnost, da je bila naprave okužena.

Uporabniki - p_u. Uporabniki so najpogostejši vir varnostnih groženj, zato nas zanima do kakšnih strežnikov dostopajo in kakšne naprave uporabljajo.

Pooblastila - p_c. Tu ocena predstavlja verjetnost, da so bila pooblastila ukradena in tako ne moremo več zaupati entitetam, ki jih uporabljajo.

Sredstva z visoko vrednostjo - p_a. Najpomembnejši del omrežja, kjer nas zanima, če so bili narejeni nepooblaščeni dostopi, ukradeni podatki.

3.2 Algoritem

Varnost entitete v omrežju je torej odvisna od entitet s katerimi je v stiku. Vsako napravo opišemo s stopnjo zaupanja, ki je komplementarna stopnji nevarnosti.

$$P_{\text{naprava}}(x_z) = 1 - P_{\text{naprava}}(x_n)$$

Nato opišemo enačbe zaupanja za posamezno vrsto entitete.

$$p_d \propto \omega_{ds} \sum_s m_{ds} p_s + \omega_{du} \sum_u m_{du} p_u + \omega_{dc} \sum_c m_{dc} p_c$$
$$p_u \propto \omega_{ud} \sum_d m_{ud} p_d + \omega_{uc} \sum_c m_{uc} p_c$$
$$p_c \propto \omega_{cu} \sum_u m_{cu} p_u + \omega_{cd} \sum_d m_{cd} p_d$$
$$p_a \propto \omega_{ad} \sum_d m_{ad} p_d + \omega_{au} \sum_u m_{au} p_u + \omega_{ac} \sum_c m_{ac} p_c$$

Parameter ω je utež glede na vrsto povezave, m je utež povezave in p je zaupanje entitete.

3.3 Začetni podatki

Zunanji strežniki. Začetna vrednost zaupanja zunanjih strežnikov je odvisna od tega za kakšno vrsto škodljive programske opreme gre. Strežniki, ki so povezani z botnet-i dobijo stopnjo zaupanja 0.05, manj škodljive vsebine pa dobijo 0.4 ali 0.25.

Notranje entitete. Za notranje entitete so začetne vrednosti pridobljene iz različnih virov kot so sistemi za upravljanje z viri in sistemi za preprečevanje vdorov. Uporabni podatki so na primer različice in operacijskih sistemov in, predvsem za uporabnike, zapisi poskusov nepooblaščenih dostopov v dnevnikih.

4. ANALIZA REZULTATOV

Rezultati, ki jih predstavljajo avtorji članka so zelo skopi. Iz 200GB/dan podatkov iz dnevnikov usmejevalnikov, DNS strežnikov, HTTP zaglavij ipd. so sestavili graf z 11790 vozlišči in 44624 povezavami, ne navajajo pa metodologije sestavljanja grafa.

Izmed končnih rezultatov so predstavljeni le podatki o uporabnikih, kjer je opisano kako je eden od uporabnikov od svojih sosedov "podedovalšlabe ocene varnosti.

Pri preizkušanju zaradi varovanja zasebnosti niso bili prisotni podatki o pooblastilih.

5. ZAKLJUČKI

Metoda opisana v članku vpeljuje tehnike, znane iz zaznavanja lažnih vsebin na spletu, v svet omrežne varnosti.

Zanimivo bi bilo videti več rezultatov, vendar so avtorji z njimi zelo skopi. Navajajo le, da se rezultati ujemajo s pričakovanimi, vendar ne podajajo empiričnih podatkov.

Avtorji prav tako ne pišejo o implementaciji. Navajajo le, da je algoritem konvergiral v petih korakih, ter da so opazili kvadratično časovno zahtevnost. Manjka pa dejanski opis algoritma, oziroma, če gre le za implementacijo obstoječega algoritma, omemba za kateri algoritem gre.

Prav tako manjka kakšna resna utemeljitev zakaj je celotna stvar sploh potrebna. Članek govori o uporabi v IDPS in pri preiskavah, vendar ne navaja konkretnih predlogov.

Tema kot taka je predstavlja dobro odskočno desko za nadaljnje raziskovanje vendar bi bilo potrebno opredeliti še veliko podrobnosti. Zagotovo je koristno v podjetju poznati najbolj ranljive dele, vendar to samo po sebi ni uporabno, če ne znamo ukrepati. Prav tako ni dodelana metodologija določanja začetnih vrednosti. Na tem področju bi bilo mogoče preizkusiti veliko različnih načinov ocenjevanja tveganja vključno z obstoječimi štatičnimi"metodami.

Navsezadnje pa je praktično nemogoče objektivno primerjati metode, saj je težko priti do podatkov. Pred vdori ne vemo katere entitete v omrežju bi morale imeti kakšno oceno, po incidentih pa podjetja ne izdajajo podatkov, saj jih ti sramotijo [2]. Prav tako so, kot omenjajo avtorji članka, problematični osebni podatki. Glede na to, da so ljudje eden od najbolj pogostih vektorjev za napade [**?**], je izpuščanje človeškega faktorja iz analize zelo velika pomanjkljivost pri preizkušanju.

6. REFERENCE

[1] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1):107–117, 1998.

[2] E. Byres and J. Lowe. The myths and facts behind cyber security risks for industrial control systems. In *Proceedings of the VDE Kongress*, volume 116, pages 213–218, 2004.

[3] Z. Gyöngyi, H. Garcia-Molina, and J. Pedersen. Combating web spam with trustrank. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pages 576–587. VLDB Endowment, 2004.

[4] X. Hu, T. Wang, M. P. Stoecklin, D. L. Schales, J. Jang, and R. Sailer. Asset risk scoring in enterprise network with mutually reinforced reputation propagation. In *Security and Privacy Workshops (SPW), 2014 IEEE*, pages 61–64. IEEE, 2014.

[5] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999.

[6] J. Leskovec, A. Rajaraman, and J. D. Ullman. *Mining of massive datasets*. Cambridge University Press, 2014.

[7] K. Skarfone and P. Mell. Guide to intrusion detection and prevention systems.

Pregled analiz in gradnja kriminalnih mrež

[Predstavitev izbranega članka, predstavitev področja in predstavitev praktične implementacije]

Anže Mikec
University of Ljubljana
Faculty of Computer and Information Science
Večna pot 113
SI-1001 Ljubljana
am2810@student.uni-lj.si

Saša Saftić
University of Ljubljana
Faculty of Computer and Information Science
Večna pot 113
SI-1001 Ljubljana
ss5178@student.uni-lj.si

POVZETEK

V začetnem delu pričujočega besedila je predstavljen članek Constructing and Analyzing Criminal Networks [12], v katerem se avtorji članka sprašujejo, ali lahko s pomočjo javno pridobljenih e-poštnih naslovov (s pomočjo Facebook profilov) kriminalcev zgradijo smiselno mrežo in na tak način odkrijejo kriminalne skupine, vodje teh skupin in osebe, ki kriminalne skupine povezujejo. V članku avtorji predstavijo metodologijo, orodja in mere, ki so jih uporabili pri raziskovanju, vse troje pa je predstavljeno tudi v tem članku.

Kratek pregled obstoječih raziskav, ki je predstavljen v besedilu, zagotovo ne zajema vseh objavljenih del in celotnega obravnavanega področja, obsega pa dela s področja socialnih mrež in analiz ter s področja kriminalnih socialnih mrež.

Podobna metodologija, orodja in mere, kot so jih uporabljali omenjeni avtorji, so uporabljene tudi v zadnjem delu članka, ki opisuje uporabo teh orodij na mreži plezalnega centra Ljubljana.

Ključne besede

Kriminalne mreže, socialne mreže, socialne vloge

1. UVOD

Živimo v obdobju, v katerem podatki neprestano pridobivajo na veljavi. So stranski produkt vsake transakcije vsakdana. Rudarjenje ogromnih količin podatkov, ki nastajajo vsako sekundo, prinaša informacije na vseh področjih in s tem ključne prednosti tistim, ki jih znajo uporabiti pravočasno in na pravi način.

Skupine nastanejo tako, da posamezniki med seboj tvorijo povezave. Skupine med seboj komunicirajo, tvorijo med seboj nove povezave in se združujejo v skupnosti. Vse odnose in strukture v družbi lahko opišemo formalno, iz teh metapodatkov pa se lahko učimo o družbi sami. Popularna družabna omrežja kot sta Facebook in Twitter nam omogočata, da preučujemo družbo v pristnem okolju. Čeprav je okolje digitalno, so odnosi in komunikacije pristne preslikave iz fizičnega sveta. Razumljivo je, da se z digitalnim medijem družbeni procesi spreminjajo, vendar to ne pomeni, da so kaj manj pristni.

Veliko raziskovalcev pristopa k analizi omrežij z namenom izboljšanja družbenih procesov in struktur. Eden izmed aspektov tega pristopa je tudi odkrivanje negativnih vplivov družbe. To so skupine posameznikov, ki kršijo zakone in izkoriščajo ostale s ciljem večjega dobička ali družbenega statusa. Z analizo družabnih omrežij lahko kriminalne skupine odkrijemo, preučujemo njihovo delovanje in se na njih učimo. Cilj ni le poiskati in kaznovati zločince ampak preučiti njihovo delovanje, načine združevanja, hierarhije znotraj skupin in se naučiti take skupnosti onemogočiti in hitreje izločiti iz družbe. Področje je zato pomembno tako za znanstveno sfero kot tudi za napredek družbe same.

2. PREGLED PODROČJA

Ta odstavek vsebuje pregled dosedanjega dela in uporabljenih metod na tem področju. Kratkemu pregledu sledi celovitejša analiza študij, osredotočenih na kriminalna družabna omrežja.

2.1 Družabna omrežja

Mislove [1] je s sodelavci opravil obsežno analizo strukture več spletnih družabnih omrežij z namenom boljšega razumevanja in načrtovanja družabnih omrežij. Analizirali so omrežja Flickr, YouTube, LiveJournal in Orkut. Raziskava naj bi bila prva takega obsega, izluščili so več kot 11,3 mio uporabnikov in 238 mio povezav. Rezultati so potrdili nekatere temeljne lastnosti družabnih omrežij. Avtorji opažajo tudi, da omrežja vsebujejo gosto povezano jedro vozlišč z visokimi stopnjami in da to jedro povezuje manjše močne gruče vozlišč nizkih stopenj z roba omrežja.

Kumar [11] je s sodelavci analiziral orodje za izmenjavo fotografij omrežja Flicker in družbeno omrežje Yahoo 360 ter opazoval, kako se omrežji spreminjata s časom. Njihove meritve kažejo na delitev omrežij v tri regije: osamelci, ki ne sodelujejo v omrežju; izolirane skupnosti, ki imajo pretežno

zvezdasto strukturo; in ogromna komponenta zasidrana z dobro povezano jedrno regijo, ki vztraja tudi brez zvezdastih struktur. Model karakterizira uporabnike kot pasivne člane omrežja; uporabnike, ki vzpodbujajo prijatelje, da migrirajo na splet; in uporabnike, ki ostale povezujejo in v celoti sodelujejo v družbeni evoluciji omrežja.

Kwak [3] je s sodelavci zbral podatke z omrežja Twitter, primerjal različne mere rangiranja in analiziral vpliv posredovanja oziroma ponovnih objav. V procesu zbiranja podatkov so pridobili 41,1mio uporabniških profilov, 1,47mrd družbenih povezav, 4262 popularnih tem in 106mio objav. Iz opažanj so ugotovili, da razvrščanje po številu privržencev in razvrščanje z metodo PageRank vračata podobne rezultate.

V zadnjem času je Ugander [5] s sodelavci opravil popolno analizo družabnega grafa omrežja Facebook. Izračunali so številne značilke, med drugimi tudi število uporabnikov in prijateljstev, stopnjo porazdelitve, dolžine poti, grozdenje in mešanje vzorcev. Rezultati so pripeljali do treh glavnih opažanj:

- Družabno omrežje je skoraj popolnoma povezano (99,91% posameznikov pripada eni povezani komponenti), prav tako rezultati potrdijo obstoj fenomena "šestih stopenj povezanosti".
- Koeficient lokalnega grozdenja in degeneriranost sosesk grafa kaže na goste strukture v grafu sosek uporabnikov, čeprav je graf omrežja Facebook v bistvu redko poseljen.
- Opazi se jasen vpliv starosti in nacionalnosti na strukturo skupnosti. Ni opaziti, da bi na strukturo vplival tudi spol.

2.2 Kriminalna družabna omrežja

Na temo uporabe metod analize družabnih grafov za pridobitev boljšega razumevanja strukture kriminalnih družabnih omrežij in odkrivanja ključnih članov kriminalnih skupin je bilo opravljenih že veliko raziskav. Xu in Chen [16] sta vpeljala ogrodje CrimeNet, ki služi samodejni analizi omrežij in vizualizaciji. Trdita, da je s tem ogrodjem identifikacija ključnih članov in vzorcev interakcij med skupinami hitrejše. Xu je s sodelavci predlagal tehniko analize povezav, ki uporablja algoritme najkrajših povezav za odkrivanje najmočnejših asociacijskih povezav med entitetami kriminalnih omrežij. Qin [4] je s sodelavci [15] z namenom analize terorističnih omrežij uporabil tehnike rudarjenja spletnih struktur. Harper in Harris sta uporabila analizo povezav v poskusu, ki je zajemal 29 analistov s področja pravnega pregona, da sta prikazala razmerja znotraj kriminalne organizacije. Sparrow [13] je uporabil tehnike omrežne analize in analiziral kriminalna omrežja s poudarkom na identifikaciji ranljivostih kriminalnih organizacij. Kerbs [8] je uporabil javno dostopne podatke in analiziral tragične dogodke, ki so se zgodili septembra 2001. Xu in Chen [14] sta analizirala več zločinskih omrežij na osnovi podatkov zločinskih incidentov, ki jih je prisrkbel policijski oddelek okrožja Tucson. Lu [17] je s sodelavci z metodami analize družabnih omrežij na osnovi tekstualnih podatkov, pridobljenih s strani časopisov, sodišč in transkripcij sodnih postopkov omrežij analiziral skupnost hekerjev. V opisani množici raziskav so avtorji uporabili informacije iz novic, podatke medijev ali podatke pridobljene z empiričnimi študijami. Posledica te vrste pridobivanja podatkov so manjša omrežja z neskladji in netočnostmi.
Druga množica raziskav je opravila večje analize zločinskih omrežij. Yang je s sodelavci [2] analiziral omrežje zločincev na družabnem omrežju Twitter. Uporabili so profile uporabnikov, ki so objavljali povezave, ki jih je Google označil kot zlonamerne. Za rangiranje uporabnikov so vpeljali algoritem podoben algoritmu PageRank.

3. PREDSTAVITEV IZBRANEGA ČLANKA

V nadaljevanju tega poglavja je predstavljen članek Constructing and Analyzing Criminal Networks [12] avtorjev Hamed Savari, Ehab Abozinadah, Alex Mbaziira in Damon McCoy. Glavna motivacija za nastanek članka je ugotovitev, da kibernetski kriminalci pogosto ne delujejo sami. Za komunikacijo preko spleta uporabljajo običajno forume, ki jim omogočajo deljenje informacij in trgovanje [9]. Avtorji članka se zavedajo, da kljub temu, da imamo poglobljeno znanje o poslovnem modelu kibernetskih kriminalcev, nimamo poglobljenega znanja o njihovi socialni strukturi. Analiza forumov nam lahko razkrije povezave med posameznimi akterji ne pa tudi njihove socialne mreže.

V članku so se osredotočili na tri korake

- **Gradnja velike socialne mreže iz javno dostopnih e-poštnih naslovov.** S tehniko povezovanja drugih profilov iz Facebooka s profili kibernetskih kriminalcev se je znatno povečalo število podatkov za analizo (število profilov).
- **Evalvacija tehnik za identifikacijo kriminalnih profilov znotraj mreže.** Analiza evalvacijskega algoritma PageRank za identifikacijo ostalih potencialnih kibernetskih kriminalcev
- **Ročna analiza posameznih kriminalnih profilov na Facebooku.** Ročna analiza posameznih kriminalnih profilov omogoča dodaten vpogled v skupino in metode, ki jih uporablja neka kriminalna skupina.

3.1 Podatki

Za analize in ugotovitve v članku so uporabljeni podatki, ki so bili javno razkriti. Podatki vsebujejo nekaj več kot tisoč e-poštnih naslovov, ki so sodelovali v napredni Nigerijski prevari [7].

Podatki so pridobljeni iz Facebook profilov, za katere so kibernetski kriminalci pri prevari uporabili enak spletni naslov kot za Facebook profil. Med več kot tisočimi e-poštnimi naslovi je bilo takšnih 262. Med 262 profili je bilo 183 javnih. Skupaj z njihovimi prijatelji tvori mrežo preko 40 tisoč profilov. Med temi profili je bilo tudi 79 kibernetskih kriminalcev (prijatelji prijateljev), ki nimajo javnega profila. Opis podatkov je povzet v tabeli 1.

V množici podatkov so vključeni samo prijatelji kriminalcev, ne pa tudi njihovi prijatelji. Stopnja teh povezav je 1. V primeru, da je stopnja kakšne povezave višja kot 1, to pomeni,

Table 1: PODATKI

E-poštni naslovi	Najdeni profili	Javni profili	Privatni profili	Končno število pregledanih profilov
1036	262	183	79	43125

da je ta profil povezan z več kriminalci (največja stopnja, ki je bila odkrita, je 20).

3.2 Vizualizacija

Vizualizacija, ki je prikazana v članku, je ustvarjena z orodjem Force Atlas 2 [6]. Ker vsi podatki vsebujejo preko 43 tisoč vozlišč, so se v članku omejili na vizualizacijo samo tistih vozlišč, ki imajo povezave s stopnjo višjo kot 1 (torej so povezani vsaj z dvema kriminalcema). Rezultat take vizualizacije je prikazan na sliki 1.

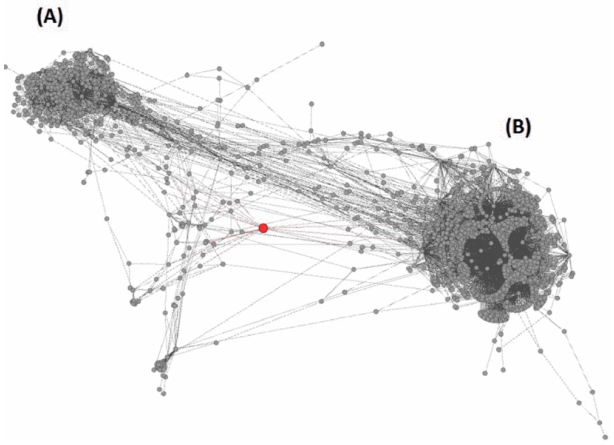


Figure 1: Vizualizacija vozlišč, ki imajo stopnjo povezave višjo kot 1. Dve ločeni skupini sta označeni z A in B

Pomemben del vizualizacije je interakcija med vozlišči kriminalcev in ne njihovih prijateljev. Vizualizacija samo teh vozlišč je prikazana na sliki 2. Vizualizacija vsebuje 53 vozlišč, kar pomni, da je med 262 kriminalci bilo 53 takšnih, ki so imeli direktne povezave med sabo. Tako na sliki 1 kot na sliki 2 lahko opazimo dve ločeni skupini, ki sta označeni z A in B. To, da najdemo ločeni skupini na obeh grafih, kaže na to, da kljub temu da kriminalci iz različnih skupin niso direktno povezani, imajo prijatelje, ki jih povezujejo.

3.3 Analiza

Pri analizi so se avtorji članka osredotočili na naslednja vprašanja:

- Kateri kriminalec ima centralno pozicijo v grafu?
- Katere podskupine in družbe lahko najdemo v grafu?
- Kateri krimalci delujejo kot posredniki med kriminalci?
- Kakšen je status kriminalcev glede na njihovo pomembnost in vpliv v grafu?

Pri iskanju odgovorov na zgornja vprašanja so avtorji članka uporabili mere za merjenje centralnosti in tehnike za detekcijo podskupin.

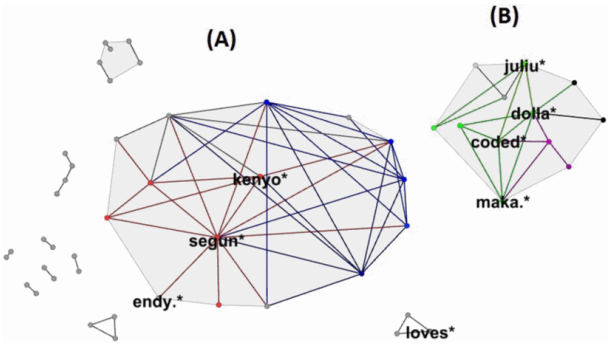


Figure 2: Vizualizacija, ki vsebuje samo vozlišča kriminalcev, ki imajo direktno povezavo med sabo. Vozlišča, ki so v nadaljevanju članka označena kot najpomembnejša, so označena z imeni. Skupini A in B sta enaki skupini, kot na sliki 1

3.3.1 Mere centralnosti

Najbolj pogosto uporabljene mere centralnosti so stopnja, vmesnost in bližina vozlišča.

- **Stopnja vozlišča** Pri tej meri privzamemo, da število povezav, ki jih ima vozlišče, korelira z aktivnostjo in vplivom, ki ga ima to vozlišče na svoje sosede in je zato ključni akter. Stopnjo vozlišča izračunamo z enačbo:

$$D_i = \frac{k_i}{N-1} = \frac{\sum_{j \epsilon G} a_{ij}}{N-1} \quad (1)$$

  kjer je k_i stopnja vozlišča, a_{ij} je ij-ti element matrike sosednosti in $N-1$ je faktor normalizacije (N je število vozlišč v grafu).

- **Vmesnost vozlišča** Vmesnost vozlišča je število najkrajših poti, ki vključujejo to vozlišče. Vozlišče z visoko vmesnostjo je pomembno za povezovanje različnih skupin. Enačba vmesnosti vozlišča:

$$B_i = \frac{\sum_{j<k \epsilon G} n_{jk}(i)/n_{jk}}{(N-1)(N-2)} \quad (2)$$

  n_{jk} je število najkrajših povezav med vozliščema j in k in $n_{jk}(i)$ je število vozlišč, ki gredo skozi to vozlišče. $(n-1)(n-2)$ je normalizacijski faktor.

- **Bližina vozlišča** Bližina (oz. oddaljenost) vozlišča je definirana kot vsota najkrajših poti med tem vozliščem in vsemi ostalimi vozlišči v grafu. Vozlišče z nizko bližino lahko do ostalih vozlišč dostopa hitreje in na enostavnejši način. Enačba:

$$C_i = (L_i)^{-1} = \frac{N-1}{\sum_{j \epsilon G} d_{ij}} \quad (3)$$

  d_{ij} je razdalja med vozliščema i in j. L_i je normalizirana razdalja tega vozlišča do vseh ostalih vozlišč v grafu.

- **Centralnost lastnih vektorjev** Centralnost lastnih vektorjev definira, do katere mere je vozlišče povezano z ostalimi dobro povezanimi vozlišči. Enačba:

$$x_i = \frac{1}{\lambda} \sum_{j \epsilon M(i)} x_j = \frac{1}{\lambda} \sum_{j \epsilon G} a_{ij} x_j \quad (4)$$

$M(i)$ je skupina sosedov $i$tega vozlišča. λ je konstanta. aij je ij-ti element matrike sosednosti.

3.3.2 PageRank

PageRank [10] lahko uporabljamo za rangiranje med vozlišči grafa in tako določamo relativno pomembnost vozlišč. Zaradi podobnosti med povezavami med spletnimi stranmi in mreže socialnih profilov je uporaba PageRanka smiselna. Namen PageRanka je izmeriti globalno pomembnost nekega vozlišča. PageRank je definiran z naslednjo enačbo:

$$PR(A) = (1-d) + d * \sum_{B \epsilon M(A)} \frac{PA(B)}{L(B)} \quad (5)$$

Kjer so $M(A)$ vozlišča, ki so povezana (sosedi) z vozliščem A, $L(B)$ je število odhodnih povezav. B in d sta faktorja dušenja.

3.4 Rezultati analize in interpretacija

Z enačbami (1) do (5) so v članku izmerili mere centralnosti in PageRank. Najpomembnejših 10 vozlišč je prikazano v tabeli 2.

Če pogledamo tabelo 2, lahko vidimo, da je oseba Coded uvrščena najvišje pri vseh merah. Dejstvo, da rezultati po metodah korelirajo, nam kaže, da je ta oseba zelo povezana, je v centru skupin, obstaja veliko najkrajših poti (vmesnost vozlišča), ki vodijo skozi to vozlišče in da je to vozlišče povezano z dobro povezanimi vozlišči (lastni vektorji). Tudi PageRank označi podobna vozlišča za najbolj pomembna.

Povezave desetih najbolje poveazanih vozlišč so prikazane v vizualizaciji na sliki 3.

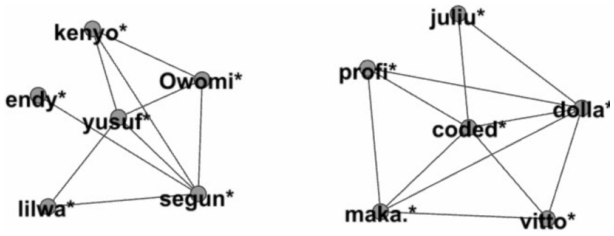


Figure 3: Vizualizacija najpomembnejših vozlišč

Zanimiva ugotovitev, ki jo avtorji članka izpostavijo, je ta, da na sliki 3 nimamo vseh vozlišč iz tabele 2. Vozlišča, ki je v tabeli 2 poimenovano z loves, ne najdemo med vozlišči, ki so direktno povezana s kriminalci. Če pogledamo pozicijo tega vozlišča na sliki 3, vidimo, da ni v množici A ali B. Najdemo ga na sliki 1, kjer je označeno z rdečo barvo. To vozlišče ima pomembno vlogo pri povezavi skupine A s skupino B in ima zaradi tega visoke vrednosti pri merah centralnosti vozlišča.

Zanimivo je, da vizualizacija najpomembnejših vozlišč tvori enaki skupini A in B kot vizualizacija vseh vozlišč.

Table 3: Kriteriji za krimianlne profile

Kriterij	Opis
Skupine	Ali je član skupin, ki se fokusirajo na kriminalne aktivnosti
Všečki	Ali všečka objave, ki so povezane s kriminalnimi aktivnostmi, kot so vdiranje in prevare.
Slike	Ali ima portrete, na katerih so vidne velike količine denarja, ponarejene izkaznice ali veliko telefonov.
Objave	Ali objavlja objave, ki so povezane s kriminalnimi aktivnostmi oz. ali se z njimi hvali.

3.5 Detekcija skupin

Pomembna naloga pri analizi omrežja je odkrivanje vzorcev in skupin znotraj omrežja. V članku so se odločili, da bodo za določitev skupin znotraj omrežja uporabili tehniko modularnosti [7]. S to tehniko dobi vsaka skupina modularno vrednost. Modularna vrednost je vrednost med -1 in 1 in meri razmerje števila povezav med vozlišči v skupini in številom povezav med skupinami.

V začetni mreži s 43 tisoč vozlišči so s tehniko modularnosti odkrili 108 skupin. Z namenom ločevanja pomembnejše skupine od manj pomembnih so se avtorji odločili, da pogledajo, koliko napomembnejših vozlišč vsebuje katera skupina. V članku izpostavijo dve skupini. Prva ima 8962 vseh vozlišč, od tega 9 kibernetskih kriminalcev in 4 najpomembnejše kriminalce. Druga skupina pa ima 3144 vseh vozlišč, 8 kibernetskih kriminalcev, od teh sta 2 med najpomembnejšimi kriminalci. V članku za ti dve skupini sklepajo, da sta potencialni skupini, kjer se osebe združujejo z namenom, da bi počeli kriminalne aktivnosti.

3.6 Ročna analiza

V članku skušajo z ročno analizo odkriti dodatne kriminalne profile, ki jih običajne mere niso odkrile. Za potrebe ročne analize so določili kriterije, po katerih bodo merili, ali je nek profil kriminalen ali ne. Kriteriji upoštevajo skupine, v katere je profil vključen, všečke in vsebine fotografij. Kriteriji so prikazani v tabeli 3.

V članku so z ročno analizo prišli do treh kategorij:

- **Nedoločeno**. Na profilu ni znakov kriminalnih dejanj.
- **Član kriminalne skupine**. Sodeluje v skupinah, ki so označene za kriminalne skupine. Na profilu ima fotografije orožja, drog in denarja. Na profilu prikazuje ukradeno blago.
- **Najverjetneje kriminalec**. Sodeluje v skupinah, ki so označene za kriminalne skupine. Njegovi komentarji vsebujejo informacije o kriminalnih dogodkih. Je lastnik več telefonov in večje količine denarja (prikazana na fotografijah). Med slikami na profilu ima fotografije bančnih računov in kreditnih kartic.

Z ročno analizo in zgoraj omenjenimi kriteriji so med stotimi najvišje uvrščenimi prijatelji kriminalcev odkrili 8% oseb,

Table 2: PODATKI

Vmesnost vozlišča	Stopnja vozlišča	Lastne vrednosti	PageRank
coded	coded	coded	coded
kenyo	dolla	dolla	loves
loves	loved	juliu	dolla
dolla	juliu	maka	kenyo
nkemd	maka	loves	juliu
adefo	kenyo	kenyo	maka
juliu	devoe	segun	nkemd
maka	segun	profi	devoe
devoe	nkemd	devoe	endy
sagun	endy	nkemd	segun

ki so najverjetneje kriminalci in 19% takšnih, ki so člani kriminalnih skupin.

4. IMPLEMENTACIJA

Z namenom boljšega razumevanja rezultatov in uporabljenih metod je bila opravljena analiza podobna analizi kriminalnega omrežja v članku. Jedro analize je stran Plezalni center Ljubljana, z družabnega omrežja Facebook. Stran je odprta, torej so informacije o njej in njenih članih javne narave. V času pisanja na strani sodeluje 3145 članov, avtorji strani objavljajo novosti dnevno. Ciljna skupina so ljudje, ki se ukvarjajo s športnim plezanjem. Stran temelji na fizičnem plezalnem centru z istim imenom, ki ga lahko najdemo v Ljubljani.

4.1 Programska oprema

Podatki o objavah in članih strani so bili pridobljeni s programskim orodjem Netvizz (verzija 1.2). Orodje deluje na družabnem omrežju Facebook in omogoča izvoz podatkov skupin in strani v formatu GDF. Vsi pridobljeni podatki o uporabnikih so maskirani, za to poskrbi že orodje samo. Podatki izvoženi v formatu GDF so bili nato uvoženi v programsko orodje Gephi, ki omogoča izris in analizo grafov.

4.2 Analiza

Izvoženi podatki vsebujejo samo podatke zadnjih 250 objav na strani Plezalni center Ljubljana. Iz prvotnega grafa, ki ga ob uvozu zgenerira orodje Gephi, ni mogoče razbrati nobenih uporabnih informacij. Vozlišča so razporejena naključno znotraj regije pravokotne oblike, kot je vidno na sliki 4. Povezave in vsa vozlišča so sive barve. Vsako vozlišče predstavlja enega člana omrežja Facebook, ki je sodeloval na strani Plezalni center Ljubljana.

Orodje Gephi omogoča izbiro večih različnih načinov razporeditve vozlišč grafa. Izbrana je bila razporeditev z algoritmom Force Atlas, ki graf preoblikuje v bolj berljivo in kompaktno obliko na podlagi povezav med vozlišči. Za rangiranje vozlišč izberemo stopnjo vozlišča in vozlišča obarvamo z gradientom, rezultat je viden na sliki 5.

Poleg izrisa grafa omogoča orodje tudi izračun raznih statistik, tudi tistih uporabljenih v članku. Ena izmed izbranih mer je vmesnost vozlišč, ki je na prikazanem grafu izbrana za izračun velikosti vozlišča (premer kroga, ki označuje vozlišče). Izris oznak vozlišč zaradi dolžin oznak ni mogoč, čeprav bi bil za domeno problema smiseln. Spomnimo, da

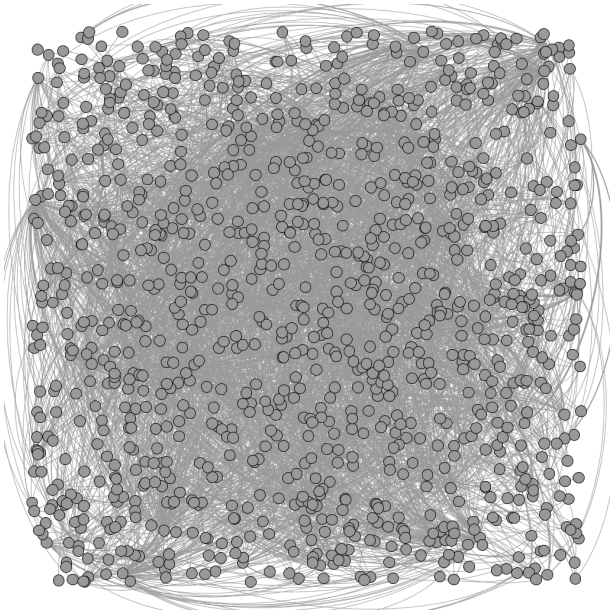


Figure 4: Usmerjen graf zgeneriran iz podatkov iz datoteke .gdf.

so imena uporabnikov maskirana in ne prinašajo dodatnih informacij.

Eden izmed pomembnejših korakov je odkrivanje skupnosti. Z orodjem izračunamo modularnost grafa, nato pa na podlagi rezultatov obarvamo pridobljene skupine, kot je vidno na sliki 7.

4.3 Vrednotenje

Iz končnega grafa je razvidno, da večino grafa sestavlja eno veliko, dobro povezano jedro. Opazimo lahko nekaj jasno razpoznavnih skupin, ki vztrajajo na robu grafa. Pravtako vidimo nekaj osamelcev, ki jih je glede na velikost omrežja zelo malo. Graf povezuje okoli 5 uporabnikov, ki delujejo kot stična točka za ostale uporabnike in skupine. Tudi ti 'popularnejši' uporabniki so med sabo povezani.
Podrobnejša analiza grafa in družbene strukture omrežja bi pokazala pomen in vlogo posameznih skupin. Ena izmed možnosti je, da vsaka samostojna gruča predstavlja skupino tečajnikov športnega plezanja ali pa podskupino plezalcev, ki redno trenirajo ali plezajo skupaj. Glavna vozlišča (hubi) so najverjetneje športni plezalci z daljšo tradicijo

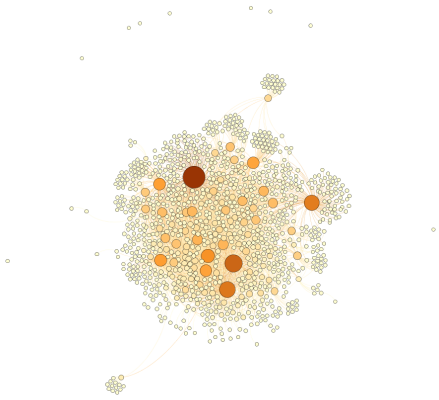


Figure 5: Graf z označenimi stopnjami vozlišč.

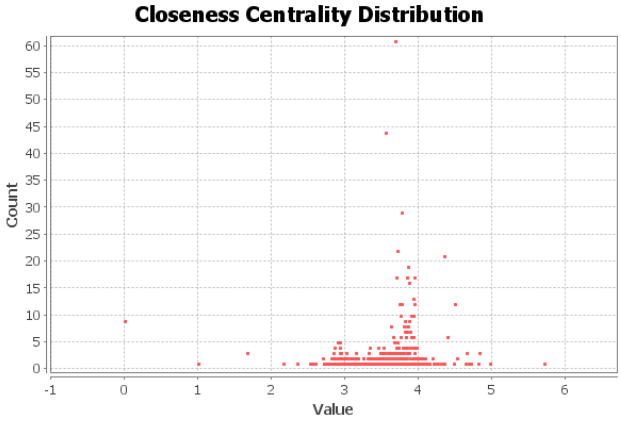


Figure 6: Statistika povprečne razdalje od danega vozlišča do vseh ostalih v grafu.

plezanja, večjimi dosežki in/ali inštruktorji v plezalnem centru. Močna povezanost vozlišč, obstoj enega velikega jedra in majhno število osamelcev govorijo o tesni povezanosti članov te podmnožice omrežja.
Kljub temu, da je bila izbrana majhna podmnožica objav na strani Plezalni center Ljubljana, lahko iz izdelanega grafa razberemo veliko zanimivih struktur. Čeprav je struktura omrežja zanimiva, bi bilo še bolj zanimivo v analizo vključiti prava imena uporabnikov in rezultate analize preslikati v realni svet. Zavedamo se, da to orodja namenoma preprečujejo in s tem ohranjajo integriteto in privatnost uporabnikov družabnih omrežij.

5. ZAKLJUčEK

V analiziranem članku in implementaciji lahko vidimo moč, ki ga ima analiza podatkov. Z razmeroma majhno množico podatkov, ki ne vsebuje pomembnih informacij, lahko izvemo veliko. Z nekaj e-poštnimi naslovi, ki jih lahko povežemo s Facebook profili, dobimo dovolj veliko socialno mrežo, ki jo lahko uporabimo za analizo.

Z analizo mreže, ki je bila sestavljena iz e-poštnih računov kibernetskih kriminalcev lahko ugotovimo, da so kibernetski

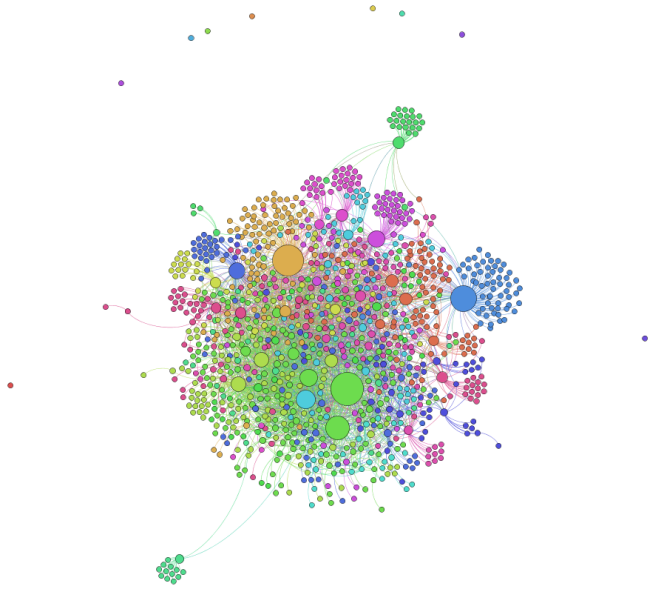


Figure 7: Graf skupine Plezalni center Ljubljana po obdelavi

kriminalci organizirani v skupine in podskupine znotraj teh skupin. Z uporabo različnih mer lahko ugotovimo, kateri so pomembni člani kriminalnih skupin. Vidimo, da lahko uporabimo različne mere pri iskanju pomembnih članov in da so različne mere približno enako uspešne in vračajo podobne rezultate.

6. REFERENCES

[1] K. P. G. P. D. A. Mislove, M. Marcon and B. Bhattacharjee. Measurement and analysis of online social networks. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, page 29–42. ACM, 2007.

[2] J. Z. S. S. C. Yang, R. Harkreader and G. Gu. Analyzing spammers' social networks for fun and profit: a case study of cyber criminal ecosystem on twitter. In *Proceedings of the 21st international conference on World Wide Web*, page 71–80. ACM, 2012.

[3] H. P. H. Kwak, C. Lee and S. Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th international conference on World wide web*, page 591–600. ACM, 2010.

[4] D. H. M. S. J. Qin, J. J. Xu and H. Chen. Analyzing terrorist networks: A case study of the global salafi jihad network. In *Intelligence and security informatics*, page 287–304. Springer, 2005.

[5] L. B. J. Ugander, B. Karrer and C. Marlow. The anatomy of the facebook social graph. In *arXiv preprint*, page 1111.4503. arXiv, 2011.

[6] M. Jacomy, S. Heymann, T. Venturini, and M. Bastian. Forceatlas2, a continuous graph layout algorithm for handy network visualization. *Medialab center of research*, 560, 2011.

[7] B. Krebs. Yahoo boys have 419 facebook friends, 2013.

[8] V. E. Krebs. Mapping networks of terrorist cells. In *Connections*, page 24(3):43–52, 2002.

[9] M. Motoyama, D. McCoy, K. Levchenko, S. Savage, and G. M. Voelker. An analysis of underground forums. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, pages 71–80. ACM, 2011.

[10] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. 1999.

[11] J. N. R. Kumar and A. Tomkins. Structure and evolution of online social networks. In *Link Mining: Models, Algorithms, and Applications*, page 337–357. Springer, 2010.

[12] H. Sarvari, E. Abozinadah, A. Mbaziira, and D. McCoy. Constructing and analyzing criminal networks. In *Security and Privacy Workshops (SPW), 2014 IEEE*, pages 84–91. IEEE, 2014.

[13] M. K. Sparrow. The application of network analysis to criminal intelligence: An assessment of the prospects. In *Social networks*, page 13(3):251–274, 1991.

[14] J. Xu and H. Chen. Criminal network analysis and visualization. In *Communications of the ACM*, page 48(6):100–107. ACM, 2005.

[15] J. J. Xu and H. Chen. Fighting organized crimes: using shortest-path algorithms to identify associations in criminal networks. In *Decision Support Systems*, page 38(3):473–487, 2004.

[16] J. J. Xu and H. Chen. Crimenet explorer: a framework for criminal network knowledge discovery. In *ACM Transactions on Information Systems (TOIS)*, page 23(2):201–226. ACM, 2005.

[17] X. L. Y. Lu, M. Polgar and Y. Cao. Social network analysis of a criminal hacker community. In *Journal of Computer Information Systems*, page 51(2):31–41, 2010.

Uspešnost kampanj z nezaželeno pošto in botneti

Jože Kulovic
Fakulteta za računalništvo in informatiko
Univerza v Ljubljani
Slovenija
jk7441@student.uni-lj.si

Boris Savič
Fakulteta za računalništvo in informatiko
Univerza v Ljubljani
Slovenija
bs3348@student.uni-lj.si

Žiga Emeršič
Fakulteta za računalništvo in informatiko
Univerza v Ljubljani
Slovenija
ze9741@student.uni-lj.si

ABSTRACT

Pošiljanje nezaželene pošte povzroča težave tako končnim naslovnikom kot tudi vmesnim členom – ponudnikom raznih spletnih storitev, lastnikom računalnikov, ki so del botnetov in mrežni arhitekturi. Pošiljatelji se poslužujejo veliko različnih metod za doseganje čim višje učinkovitosti in uspešnosti oz. dostavljivosti nezaželenih sporočil – v tem delu smo naredili kratek pregled stanja na tem področju in skušamo izpostaviti najpomembnejše.

Nezaželena sporočila se širijo ne le v domeni elektronske pošte, temveč tudi preko veliko drugih elektronskih kanalov. Za lažje in učinkovitejše pošiljanje se avtorji kampanj poslužujejo botnetov – omrežja okuženih računalnikov, ki služijo namenu pošiljanja nezaželene pošte. V delu je predstavljena arhitektura botnetov in opisan eden izmed največjih, Cutwail.

Keywords

nezaželena pošta, botnet, spam, varnost

Uvod

Nezaželena pošta[1] je ena izmed glavnih gonilnih sil sive ekonomije svetovnega spleta [9]. Maja 1978 je svetovni splet doživel rojstvo prvega nezaželenega poštnega sporočila. *Gary Thuerk*, prodajalec pri podjetju *Digital Equipment Corporation* je na ta dan poslal nezaželeno poštno sporočilo[2] 400 osebam, od skupnih 2.600, ki so bile v tistem času prisotne na omrežju ARPAnet – tako imenovanem prvem spletu. Sporočilo je bilo prodajno naravnano, kar je razburilo precej prejemnikov. Sporočilo je kljub razburjenosti doživelo tudi nekaj uspeha, saj je nekaj prejemnikov izkazalo interes za prodajno vsebino. To je bil začetek povsem nove ekonomije, ekonomije nezaželene pošte.

Prvo nezaželeno sporočilo je bilo poslano ročno, danes je več kot 85% poslanih s pomočjo tako imenovanih *botnetov*. Uspešne kampanje nezaželene pošte njihovim avtorjem prinesejo tudi do 1.000.000 $ na leto. Prvo nezaželeno sporočilo je prodajalo računalnike, danes pa v okviru nezaželenih sporočil prevladuje prodaja nelegalnih zdravil in ponarejenih izdelkov. V verigo prodaje preko nezaželene pošte so vključene različne institucije, od proizvajalcev tovrstnih izdelkov, do plačilnih procesorjev in bank.

Pošiljanje nezaželene pošte je postalo žrtev miselnosti kvantitete pred kvaliteto, saj je za uspešno kampanjo, ki prinese večje dobičke avtorjem kampanje, potrebno poslati na tisoče, celo milijone sporočil [10]. K temu so pripomogle številne študije nezaželenih sporočil, kot tudi razvoj orodij za detekcijo tovrstnih sporočil, ki pri končnem prejemniku sporočilo označijo kot nezaželeno – kar povzroči, da večina tovrstne pošte s strani prejemnika nikoli ni ogledana oz. prebrana.

V tem delu se bomo osredotočili na pošiljanje nezaželene pošte v okviru botnetov. V poglavju 2 je obravnavana definicija botneta, kaj to sploh je in kako nastane. V poglavju 1 so opisane vrste nezaželenih sporočil, zgodovina, kako takšna sporočila filtrirati in vrste kanalov skozi katere se širijo. V poglavju 3 je podrobneje opisan eden največjih botnetov Cutwail. Sledeče poglavje 4 predstavi faktorje za uspešno kampanjo pošiljanja nezaželene pošte. Sledi zaključek v poglavju 5, kjer je povzeta analiza nezaželene pošte, glavne ugotovitve iz področja in predlogi za nadaljno delo.

1. NEZAŽELENA SPOROČILA – SPAM

Čeprav v tem članku primarno obravnavamo nezaželena sporočila v okviru nezaželene elektornske pošte, je pombembno obravnavati definicijo nezaželenega elektronskega sporočila v splošnem. Elektronska nezaželena sporočila so nenaročena sporočila poslana veliki skupini ljudi z uporabo elektronskih sporočilnih sistemov (Skype, elektronska pošta, SMS, objave na forumih ...). Prvotno se je izraz *spam* za nezaželeno sporočilo razvil pri širjenju nezaželenih sporočil preko elektronske pošte, nato pa se je začel uporabljati še za sporočila poslana preko ostalih kanalov. Nezaželenih sporočil se poslužujejo še posebno v namene oglaševanja saj je takšno oglaševanje poceni v primerjavi z ostalimi načini oglaševanja [12].

Nezaželena sporočila je relativno enostavno distribuirati. Običajno najtežji del, ki ga morajo pošiljatelji nezaželnih

[1] Čeprav pod pojem nezaželena pošta v splošnem spada tudi fizična pošta v, običajno, papirnati obliki, se v tem delu omejujemo zgolj na elektronsko obliko nezaželene pošte. Izraz *nezaželena pošta* tako označuje nezaželeno pošto v izključno elektronski obliki.

[2] *Sporočilo* oz. nezaželeno sporočilo je v nadaljevanju obravanano kot posamezna sporočilna instanca elekronske komunikacije. V tem delu je nezaželeno sporočilo tako širši del elektronske komunikacije in ne le del elektronske pošte.

sporočil narediti je zbirati kontaktne podatke čim večje populacije naslovnikov (e-poštni naslov, telefonska številka ipd.).

1.1 Nezaželena e-poštna sporočila

Nezaželena pošta (angl. *Unsolicited bulk email - UBE* ali nezaželena komercialna e-pošta (angl. *Unsolicited commercial email - UCE* je frekvenčno pošiljanje nezaželenih sporočil s komercialno vsebino velikemu številu prejemnikov. Trenutno nezaželena sporočila prevladujejo po količini od ostalih e-poštnih sporočil in obsegajo kar 85% vseh svetovnih e-poštnih sporočil. V nekaterih državah so zaradi obsežnosti nezaželenih e-poštnih sporočil sprejeli pobude o nezakonitosti le teh. Z zakonsko prepovedjo se je obseg nezaželenih sporočil zmanjšal na 66% [3].

1.2 Zgodovina nezaželenih sporočil

Najzgodnejše dokumentirano pošiljanje nezaželene pošte izhhaja iz leta 1978 pri uporabnikih ARPANET-a glede oglasa za predstavitev Digital Equipment Corporation (omenjeno v uvodem poglavju). Takšna sporočila se je pričelo označevati z izrazom ''spam" šele leta 1993, ko je USENET zaradi hrošča v programu poslal sporočilo 200 ljudem. Skupina posameznikov je ta incident poimenovala ''spamming", kar se danes označuje kot pričetek rabe izraza ''spam" za nezaželeno sporočilo [5].

Pošiljanje nezaželenih sporočil je postala poslovna praksa aprila 1994 , ko sta odvetnika najela programerja, da je objavil sporočilo ''Green Card Lottery- Final One?" [14]. Za razliko od predhodnih pošiljateljev nezaželenih sporočil sta bila avtorja ponosna, da sta na takšen način oglaševala in kasneje napisala knjigo o takem oglaševanju. Kasneje sta načrtovala odprtje svetovalnega podjetja za pomoč oglaševanja drugim ljudem vendar neuspešno.

Večina nezaželenih sporočil se trenutno širi preko e-poštnih sporočil. Do leta 2009 je bila večina sporočil v angleškem jeziku. Kasneje so pošiljatelji nezaželenih sporočil začeli uporabljati avtomatske prevajalske storitve tako, kar je povečalo količino sporočil tudi v drugih jezikih.

1.3 Delovanje pošiljateljev nezaželenih sporočil

V primerjavi s fizičnim - papirnatim oglaševanjem, e-poštna sporočila oglaševalca ne stanejo skoraj nič. Večina stroškov je pokritih s strani prejemnika, ker oglaševalcu ni potrebno plačati prejemnikove internetne povezave. Ker so stroški oglaševalcev tako nizki se veljavnost e-poštnih naslovov na katere se pošilja e-poštna sporočila ne preverjajo, saj iz ekonomskega stališča to ne predstavlja velike razlike. E-poštne naslove avtorji nezaželenih poštnih kampanj največkrat pridobijo s/z [16]:

- programi, ki samodejno pridobivajo e-poštne naslove s spletnih strani
- nakupom uporabniških podatkov od podjetji, ki takšne podatke posedujejo
- generiranjem e-poštnih naslovov na podlagi uporabniških in lastnih imen spletnih strani

Veliko pošiljateljev nezaželenih sporočil uporablja raznovrstne trike s katerimi nas lahko pritegnejo, da preberemo sporočilo. Nekateri triki, ki jih pošiljatelji nezaželenih sporočil uporabljajo so [6]:

- skušajo pritegniti z ''zadevo sporočila"
- skritje ciljnega e-poštnega naslova iz polja prejemnika (angl. polje ''To")
- skrivanje ostalih prejemnikov sporočila
- skrivanje e-poštnega naslova pošiljatelja ali sprememba zaglavja sporočila za namene spreminjanja pošiljateljevega e-poštnega naslova

Najpogostejši trik, ki ga pošiljatelji nezaželenih sporočil uporabljajo je pošiljanje sporočil preko e-poštnega strežnika tretje osebe. Takšen način pošiljanja sporočil povzroči dvojno škodo – in sicer poleg škode na strani prejemnika, tudi škodo na vmesnem strežniku, ki se ga preplavi z nezaželenimi sporočili. Za vsako nezaželeno sporočilo, ki gre skozi (sicer nedolžni) poštni strežnik lahko dobi strežnik pritožbe od uporabnikov. Številni pošiljatelji nezaželenih sporočil, sporočila pošiljajo preko brezplačnih računov, ki jih nato opuščajo in zaporedno ustvarjajo nove za vsak napad posebej.

1.4 Filtri nezaželenih sporočil (spam filtri)

Filtri nezaželenih sporočil imajo dolg seznam kriterijev na podlagi katerih odločijo ali je določeno sporočilo nezaželeno ali ni. Vsak kriterij vrne določen faktor, ki na koncu preko vseh kriterijev vpliva na odločitev ali je določeno sporočilo nezaželeno ali ne. Primer kriterija filtra nezaželenih sporočil:

- Omenjen denar $(0,193$ točke$)$
- Govori o nekakšnem preboju $(0,232$ točke$)$
- Vsebuje nujno zadevo $(0,288$ točke$)$
- Garantirana vrnitev denarja $(2,051$ točk$)$

Meja skupne vsote faktorjev pri kateri se obravnava neko sporočilo kot nezaželeno je prilagodljiva in se jo lahko nastavi [8]. Nekateri filtri nezaželenih sporočil imajo tudi možnost samodejnega učenja na podlagi uporabnikovih dejanj, ko neko sporočilo označijo kot zaželeno oz. kot nenezaželeno. Pri samodejnem učenju filtrov nezaželenih sporočil se ti sinhronizirajo med seboj, tako da učenje poteka hitreje in efektivneje.

Natančni kriteriji obstoječih, široko uporabljanih filtrov za nezaželena sporočila niso javno objavljeni, saj bi avtorji kampanj s tem pridobili znanje s katerim bi nato lažje obšli take filtre.

1.5 Izogibanje nezaželenim sporočilom

Izogibanje nezaželenim sporočilom je sestavljeno iz preventive in kurative. Za preventivno obnašanje proti nezaželenim sporočilom je pomembno, da se zavedamo kaj počnemo na spletu, komu posredujemo informacije in komu so dostopni naši osebni podatki, ki jih puščamo na spletu. Pri kurativnem procesu izogibanja neželene pošte je potrebno uporabljati ustrezne filtre nezaželene pošte. Priporočila za izogibanje nezaželenih sporočil lahko v grobem strnimo v slednja [17]:

- uporaba filtrov nezaželenih sporočil
- pazljivo ravnanje z lastnim e-poštnim naslovom in ostalimi zasebnimi podatki
- uporaba sekundarnega ali večih e-poštnih naslovov

- izogibanje rabi kratkih uporabniških imen
- uporaba različnih vzdevkov za različne domene
- preverjanje pošiljatelja pri prejetih sporočilih
- pozornost ob prejemu nezaželenega sporočila [7] (neobiskovanje povezav v sporočilu, izogibanje odpiranju priponk in prijava prejetega nezaželenega sporočila ustreznim organizacijam)
- poučevanje ostalih o nezaželenih sporočilih

1.6 Drugi kanali širjenja nezaželenih sporočil

Zaradi priročnosti in nizke cene je e-pošta glavni kanal širjenja nezaželenih sporočil. Kljub temu se nezaželena sporočila pošiljajo tudi preko drugih kanalov vendar ne v takšnem obsegu.

1.6.1 Sistemi za takojšnje sporočanje

Sistemi za pošiljanje takojšnjih sporočil (angl. Instant messaging systems) običajno niso blokirani s strani požarnih zidov, kar je še posebno ugodno za pošiljatelje nezaželenih sporočil. Primer instantnega sporočilnega sistema sta Skype in Google Hangout.

1.6.2 Forumi

Pošiljanje nezaželenih sporočil na forumih zajema predvsem oglasna sporočila na spletnih forumih. Običajno jih postavijo z avtomatiziranimi agenti na spletu. Večina forumov vsebuje povezave do zunanjih spletnih strani z dvojnim ciljem. Povečati prepoznavnost v iskalniku (ang. *search engine*) v zelo iskanih/konkurenčnih področjih (hujšanje, zdravila, igre na srečo, pornografija, nepremičnine ...) ter ustvarjanje večjega prometa na komercialnih spletnih straneh. Spletne povezave lahko vsebujejo tudi kodo za sledenje indentitete. S tem spletne strani pridobijo informacijo kdo in od kod je prišel na njihovo spletno stran – to omogoča natačnejšo analizo in vodenje statistike.

1.6.3 Mobilni telefon

Nezaželena sporočila na mobilnih telefonih so usmerjena predvsem v storitev besedilnih sporočil mobilnega telefona (SMS). To je lahko za prejemnike sporočil še posebno neprijetno ne le zato, ker je že samo nezaželeno sporočilo nadležno ampak tudi zato, ker se prejeti SMSi lahko tudi obračunavajo, kar vodi v direkten finančni strošek prejemnika. Zaradi vse večje razširjenosti SMS nezaželenih sporočil so sprejeli zakon , da morajo vsa nezaželena sporočila na mobilnih telefonih poslana preko SMS sporočil imeti opcijo HELP in STOP, s katerim se lahko odjavimo od takšnih sporočil.

Kljub velikemu številu uporabnikov mobilnih naprav je obseg SMS nezaželenih sporočil majhen v primerjavi z ostalimi kanali, ker morajo pošiljatelji nezaželenih sporočil plačati ceno za pošiljanje SMS sporočil. Ravno tako je namestitev virusov, črvov v telefone drugega za pošiljanje nezaželenih sporočil težko, saj uporabniki aplikacije običajno prenašajo preko centralnih repozitorjev, ki so pod nadzorom.

1.6.4 Socialna omrežja

V kategorijo nezaželenih sporočil pri socialnih omrežji sodijo socialne aplikacije kot so Facebook, Twitter in podobne. Twiter je za preprečevanje objavljenih nezaželenih sporočil preučeval strukture interesov vseh svojih uporabnikov za prejemanje zanimivih twitov in na podlagi analize predhodno filtrira twite [11]. Največja nadloga pri socialnih omrežjih so tudi razni vdori v uporabniške račune in pošiljanje povezav, nezaželenih sporočil pod krinko uporabnikovih zaupanja vrednih stikov (družina, prijatelji ...).

1.6.5 Socialna nezaželena sporočila

Pri socialnih nezaželenih sporočilih je mišljeno predvsem širjenje nezaželenih sporočil izven centralno vodenih socialnih omrežij uporabniške vsebine na poslovnih, vladnih in neprofitnih spletnih straneh po vsem svetu. Sem sodijo tudi lažni uporabniški računi, pod katerimi uporabniki komentirajo in lahko objavljajo žaljiva, sovražna in nasilna sporočila.

1.6.6 Spletne igre

Pri mnogih spletnih igrah lahko igralci vzpostavijo stik drug z drugim preko igralec-igralec (angl. *player-to-player*) sporočilnim sistemom ali preko klepetalnic. Definicija nezaželenega sporočila se razlikuje od igre do igre, vendar se običajno ta izraz uporablja za vse oblike poplav sporočil, kršenja pogojev spletne strani/igre. To je še posebej pogosto pri masovno več igralskih spletnih igrah (angl. *Massively multiplayer online role-playing game - MMORPG*, kjer pošiljatelji poskušajo prodati predmete povezane z igro v realnem svetu za pravi denar.

1.6.7 Indeksiranje nezaželenih spletnih strani v iskalnikih (spamdexing)

Indeksiranje nezaželenih spletnih strani v iskalnikih se nanaša na prakso na svetovnem spletu pri čemer spreminjajo vsebino HTML spletne strani tako, da povečajo svoje možnosti za visoko uvrstitev na seznamu zadetkov spletnega iskalnika (angl. *search engine*). Takšne spletne strani uporabljajo ''black-hat" tehnike za optimizacijo iskalnikov tako, da namerno manipulirajo svojo uvrstitev v iskalnikih. Mnogi sodobni spletni iskalniki spreminjajo svoje iskalne algoritme, da bi čim bolj izključili spletne strani iz svojih rezultatov, ki vsebujejo lažno indeksiranje: iskalni agenti tako npr. zaznajo večkratno ponovitev določenih besed z uporabo gramatične analize kot nezaželeno spletno stran. Če se ugotovi, da lastnik spletne strani nezaželeno spletno stran in uporablja lažno indeksiranje, da bi dvignil pozicijo na iskalniku ga le ta kaznuje tako da se njegova pozicija zmanjša namesto, da bi se dvignila.

1.6.8 Blogi

Nezaželena sporočila v blogih, wikijih, knjigah gostov ali drugo javno dostopnih forumih je tudi oblika indeksiranja nezaželenih spletnih strani. To se naredi z objavo (običajno samodejno) naključnega komentarja, kopiranje materiala od drugod ali spodbujanja komercialne storitve.

Spletne strani, ki omogočajo uporabnikom dodajanje ali urejanje hiper povezav lahko hitro postanejo žrtve nezaželenih povezav. Povezave, ki jih lastniki nezaželenih spletnih strani dodajajo običajno kažejo na spletni naslov za katerega želijo povečati uvrstitev na rezultatih iskalnikov. Višja uvrstitev spletne strani na iskalniku predstavlja večjo priljubljenost spletne strani in tako povečuje število potencialnih obiskovalcev.

1.6.9 Spletne strani z video vsebino

Spletne strani, ki se ukvarjajo z delenjem video vsebin kot je npr. YouTube so pogosto tarče pošiljateljev nezaželenih

sporočil.

Najpogostejše tehnike pošiljateljev nezaželenih sporočil vključujejo objavljanje povezav do spletnih strani s pornografsko vsebino ali do strani s spletnimi zmenkarijami v komentarjih naključnih video posnetkov ali na naključnih uporabniških profilih. Z uporabo sistema glasovanja komentarjev lahko lastniki objav nezaželenih komentarjev svojemu komentarju povečajo priljubljenost tako, da ga uvrstijo med najbolj priljubljene in s tem dosežejo takojšno vidnost komentarja vsem ki stran obiščejo.

Druga pogosta tehnika, ki jo pošiljatelji nezaželenih sporočil uporabljajo, je uporaba agentov za objavljanje sporočil na naključne uporabniške profile. Takšna sporočila običajno vsebujejo vabljiva besedila in slike. Glavni namen takšnih objav je pritegnitev ljudi na povezavo na njihovih domačih profilih. YouTube je v obrambo proti takšnimi povezavami blokiral objavo takšnih povezav, dodatno pa je implementiral CAPTCHA sistem ter s otežil frekvenčno in hitro objavljanje vsebin.

Naslednji popularen način objavljanja nezaželenih sporočil na takšnih spletnih straneh je dejanski nezaželen video pod imenom nekega resnično popularnega video posnetka ali videa celotnega filma s katerim želijo pridobiti pozornost. V video običajno vključijo tudi resnično sliko iz istoimenskega videoposnetka, ki se nato pojavi pri predogledu in s tem zavedejo gledalce, da le-ti hitreje kliknejo nanj. Uporabnik šele po kliku nanj opazi, da je vsebina takšnega posnetka popolnoma nepovezana s filmom za katerega se izdaja ali pa vsebuje opis, da je bil film odstranjen s spletne strani. Temu sledi povezava na spletno stran kjer naj bi se nahajal celotni film. V nekaterih primerih lahko povezava privede do spletne ankete, zlonamerne programske opreme ali druge neželjene vsebine.

1.6.10 Nezaželena sporočila preko internetne telefonije

Nezaželena sporočila preko internetne telefonije (ang. *SPam over Internet Telephony SPIT*) je širjenje oglasnih sporočil preko internetnega telefona. Oglaševalci imajo običajno posnete posnetke, ki se predvajajo uporabnikom ki jih kličejo in s tem si olajšajo delo. Prednost internetne telefonije v primerjavi z navadno telefonijo na strani oglaševalcev je predvsem cena in enostavnost, saj takšen način oglaševanja oglaševalca običajno ne stane nič. Hkrati si oglaševalec zagotovi anonimnost preko interneta in ima način s katerim lahko hkrati opravi veliko število klicev iz iste lokacije. IP naslovi, ki se uporabljajo za širjenje nezaželenih sporočil preko internetne telefonije se kljub temu običajno lahko opazijo na podlagi statistične analize. Tak pošiljatelj ima veliko število kratkih odhodnih klicev, ki so v v večini primerov nezaključeni.

1.6.11 Akademski iskalniki

Akademski iskalniki omogočajo raziskovalcem iskanje akademske literature. Raziskovalci iz University of California, Berkeley in OvGU je pokazala, da večina spletnih akademskih iskalnikov, predvsem Google Scholar niso sposobni prepoznati lažnih, nezaželenih člankov. Raziskovalci so manipulirali navedbe števila člankov in s tem dokazali, da je možno indeksirati v Google Scholar iskalniku popolnoma lažne članke. V nekatere članke so uspeli vključiti celo primere oglaševanja.

2. BOTNET

Botnet je omrežje, sestavljeno iz računalnikov, katerih delovanje je kompromitirano s strani zlonamerne programske kode. Okuženi računalniki izvajajo ukaze in naloge brez vednosti njihovih lastnikov, saj so pod nadzorom oddaljenega napadalca. Okuženi računalniki tako sodelujejo kot pošiljatelji nezaželene pošte, kot tudi v okviru drugih zlonamernih namenov, npr. napadov DDOS (angl. distributed denial-of-service).

Botneti po mnenju ekspertov predstavljajo resno grožnjo svetovnemu spletu, saj je po nekaterih ocenah kar 25% vseh računalnikov povezanih v svetovni splet okuženih z zlonamerno programsko kodo in povezanih v ti. botnete [1].

Tovrstna omrežja okuženih računalnikov so pripravljena na izvajanje nelegalnih aktivnosti na tako visokem nivoju, da lahko ogrozijo delovanje tako zasebnih kot javnih storitev v več državah po svetu. Napadalec, pogosto znan tudi pod imenom *botmaster* [1], prevzame nadzor nad okuženimi računalniki z namenom izvajanja različnih kriminalnih aktivnosti, ter tako zakrije sledi oz. oteži ali onemogoči forenzikom odkrivanje izvora napadalca.

Nekatere študije kažejo na zaskrbljujoč podatek, da se je industrija nezaželene pošte razvila do te stopnje, da je kar 80% vse poslane pošte pravzaprav nezaželene. Večina teh sporočil je poslanih iz omrežij *botnetov* [1]. Nekateri bolj znani so dobili tudi svoja imena, kot npr.: *Grum*, *Bobax*, *Cutwail*, *Rustoc* in so podrobneje opisani v podpoglavju 2.4.

Kljub temu, da zaradi razvoja filtrov nezaželene pošte večina sporočil, ni nikoli prebranih oz. ogledanih, so pravzaprav vsa sporočila prepotovala spletne povezave od izvora do cilja, ter tako po nepotrebnem trošila omrežne vire. Aktivnost botnetov predstavlja visoke finančne izgube tako ponudnikom spletnih storitev, zasebnim podjetjem, vladam in domačim uporabnikom. V letu 2007 je ocenjena škoda povzročena s poslanimi nezaželenimi sporočili presegla 100 milijard $ [13].

2.1 Zgodovina botnetov

Zgodovinsko gledano izvor botnetov leži v *Internet Relay Chat* - IRC omrežjih. IRC je tekstovno bazirano omrežje za izmenjavo sporočil, ki organizira komunikacijo v kanale. Vendar pa v omrežjih IRC koncept *bot*-a, ni nujno pomenil nezaželjenega obnašanja. Primarna ideja tovrstnih botnetov je bila kontrola interakcij znotraj kanalov. S pomočjo tega so omogočali preproste interpretacije ukazov, ponujali so celo preproste igre in ostale storitve svojim uporabnikom.

Kmalu za pojavom IRC botnetov v letu 1993 so se pojavili zlonamerni botneti, ki temeljijo na isti ideji, vendar izvajajo zlonamerno programsko kodo, ter so bili ustvarjeni z namenom izvajanja DDoS napadov. Novi boti so se razvili tako, da danes uporabljajo kompleksne mehanizme za komunikacijo z *bootmaster*-jem, ki izkoriščajo tudi druge dostopne protokole, kar je posledično vodilo do tega, da so botneti danes precej bolj robustni in sofisticirani.

Računalnik se lahko okuži z zlonamerno kodo botneta preko izmenjave datotek, P2P omrežij, email priponk, okuženih spletnih strani itd. [1]

2.2 Komponente botnet-a

Zaradi boljšega razumevanja kako botnet-i delujejo, v tem delu podrobneje poglejmo njihove komponente. Kljub temu, da so botnet-i lahko zasnovani na različne načine, so osnovne komponente enake in so prikazane na sliki 1.

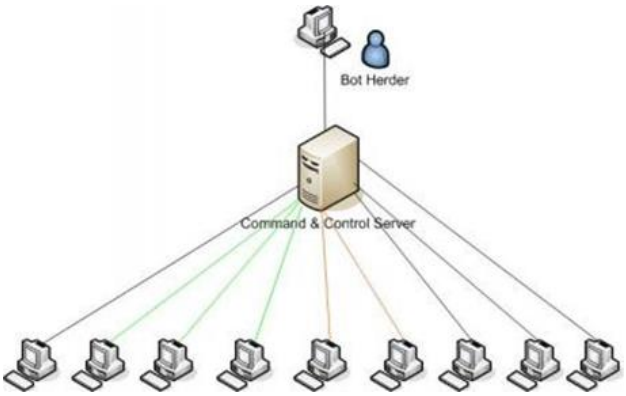


Figure 1: Komponente botnet-a

2.2.1 Bot

Bot je zlonamerna programska koda nameščena na okuženem računalniku. S pomočjo zlonamerne kode je okužen računalnik sposoben izvajati serijo različnih napadov. Tovrstna zlonamerna koda se ponavadi zažene ob vsakem zagonu operacijskega sistema računalnika.

2.2.2 C&C Strežnik

Navodila za izvajanje ukazov na botih se pošiljajo preko tako imenovanih C&C strežnikov. C&C strežnik je centraliziran računalnik, ki pošilja ukaze na okužene računalnike in sprejema poročila o uspešno izvedenih ukazih ali njihovih napakah.

2.3 Topologije Botnetov

Omrežja okuženih računalnikov so ponavadi organizirana v eno izmed sledečih struktur:

- **Zvezda** - okuženi računalniki so organizirani v obliki zvezde okoli centralnega strežnika
- **Redundantni** - računalniki so povezani z več kot enim C&C strežnikom, tako lahko napadelec nemoteno upravlja z njimi tudi v primeru izpada enega strežnika
- **Hierarhično** - nadzorni strežniki so organizirani v hierarhične nivoje
- **Naključno** - nadzorni strežniki so porazdeljeni povsem naključno, med seboj komunicirajo preko P2P povezave.

2.4 Nezaželena pošta in Botneti

Vsak dan je pošlanih preko 150 milijard nezaželenih sporočil. V nadaljevanju so predstavljena omrežja okuženih računalnikov, ki so v največji meri odgovorna za nastanek teh sporočil.

2.4.1 Grum

Grum je po mnenju mnogih prihodnost pošiljanja nezaželene pošte [15]. Kljub temu, da je omrežje relativno majhno, saj vsebuje slabih 600.000 okuženih računalnikov, zakrivi kar 25% vse poslane nezaželene pošte. Večina sporočil poslanih preko Gruma vsebuje oglase nelegalnih farmacevtskih izdelkov.

2.4.2 Bobax

Bobax je po mnenju raziskovalcev najet s strani različnih pošiljateljev, saj sporočila, ki jih pošilja precej varirajo po vsebini [15]. Kar naredi Bobax tako zanimiv je dejstvo, da pošlje vsak dan preko 27 milijard nezaželenih sporočil, pod njegovim nadzorom pa je le 100.000 okuženih računalnikov.

2.4.3 Pushdo/Cutwail

Omrežje Pushdo/Cutwail je nastalo leta 2007 in danes pošilja preko 19 milijard nezaželenih sporočil dnevno [15]. Pod nadzorom ima $1,5$ milijona okuženih računalnikov. Sporočila poslana iz omrežja Pushdo/Cutwail vsebinsko pokrivajo tako faramcevtske izdelke, spletne kazinoje kot tudi ostale vsebine. Omrežje Cutwail je tudi na voljo za najem in je podrobneje opisano v poglavju 3.

2.4.4 Rustock

Pod svojo kontrolo drži 2 milijona računalnikov [15], ter trenutno drži rekord največjega botnet-a. Značilnost tega omrežja je bila, da je v kratkem času poslal ogromne količine nezaželenih sporočil, nato pa za nekaj časa poniknil. Danes pošilja sporočila le med 3. in 7. uro zjutraj, v tem času pa pošlje preko 17 milijard sporočil.

3. CUTWAIL BOTNET

Cutwail botnet je eden izmed največjih botnetov za neželjena sporočila v zadnjih letih [9]. Botnet Cutwail je možno najeti, naročniku je navoljo spletni vmesnik, ki omogoča nadzor botov in vsebino poslanih sporočil [9]. Cutwail vsebuje samodejno spreminjanje delov poštnih sporočil zato, da se oteži detekcijo nezaželene pošte na strani prejemnikov. Naročnik s spletnim vmesnikom tako pravzaprav nadzira le predloge sporočil, ki jih Cutwail nato ustrezno prilagaja.

Ker je večina Cutwailovih razvijalcev in strank iz Rusije so tudi navodila v ruščini [9]. V navodilih je opisano delo z uporabniškim administratorskim vmesnikom kot tudi navodila in predlogi za uspešne kampanje pošiljanja nezaželenih sporočil. Kot ugotavljajo avtorji [9] so splošna navodila podana za uspešno kampanje matematično napačna in verjetno služijo le za komercialne propagandne namene.

Cutwail botnet za širjenje botov uporablja Pushdo zlonamerno programsko opremo, ki se namesti na operacijske sisteme Microsoft Windows. Botnet je znan tudi po imenom *0bulk Psyche Evolution* [9].

4. USPEŠNOST IN UČINKOVITOST POŠILJANJA NEZAŽELENIH SPOROČIL

4.1 Uspešnost

Uspešnost kampanje pošiljanja nezaželenih sporočil je odvisna od tega v kolikšni meri je pošiljatelj dosegel svoj cilj. Če je vsebina sporočil oglasno sporočilo se uspešnost meri v količnih prodanih izdelkov oz. v končnem dobičku, ki ga je takšno pošiljanje prineslo. Vmesne stopnje uspešne kampanje bi lahko opisali kot:

1. pridobitev kakovostnih e-poštnih naslovov
2. uspešno pošiljanje na e-poštne naslove
3. uspešna dostavitev v ciljni poštni nabiralnik (zaobitev filtrov nezaželene pošte)

4. naslovnikov ogled sporočila

5. željena naslovnikova akcija (nakup oglaševanega izdelka)

Postopki pridobivanja e-poštnih naslovov so opisani v poglavju 1.3. Za kakovosten e-poštni naslov se smatra tisti za katerega velja, da bo njegov naslovnik z veliko verjetnostjo pogledal sporočilo. To velja za račune, ki jih naslovnik pogosto preveri in uporablja slabe filtre nezaželene pošte. Uspešno pošiljanje na e-poštne naslove, opisano v točki 2 pomeni, da bo e-poštno sporočilo dejansko prišlo do ciljnega strežnika. Meriti uspešnost skozi točko 3 in 4 je že težje, saj je v sporočilo potrebno vključiti HTML elemente, ki na določen strežnik sporočijo o odprtju sporočila (npr. vključitev nevidne slike s pripadajočimi GET parametri, ki so dovolj, da na strežnik sporočijo informacijo o odprtju) ali povezav, ki pa jih naslovnik mora odpreti, da bi lahko izmeri uspešnost kampanje. Točka 5 presega okvire tega dela in je nismo obravnavali, potrebno pa se je zavedati, da je to končni cilj vsake oglaševalske kampanje.

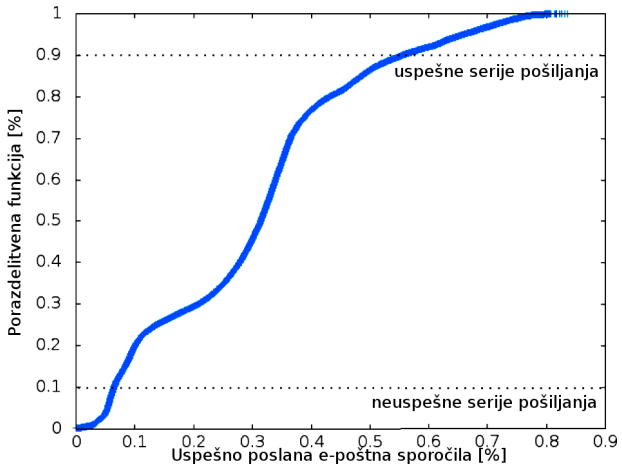


Figure 2: porazdelitvena funkcija uspešno poslanih serij e-poštnih sporočil

Računanje uspešnosti točk 3, 4 in 5 je kompleksno in jo je brez ustreznih meritev praktično nemogoče izračunati. Zaradi tega se bomo kot merilo za uspeh osredotočili na količino uspešno poslanih e-poštnih sporočil (točka 2) – kot je tudi primer v delu [9]. Kriterij v točki 1 (pridobitev kakovostnih e-poštnih naslovov) se odraža skozi druge tri. Avtorji [9] so uspešno pošiljanje sporočila definirali kot odsotnost odgovora o napaki ciljnega strežnika. To sicer ne zagotavlja povsem gotove zanesljivosti, vendar omogoča preprosto evalvacijo.

Kljub doseženim/nedoseženim ciljem uspešnosti je pomemben faktor tudi učinkovitost. Več sporočil ko lahko pošljemo s čim manjšim vložkom virov bolj je kampanja učinkovita. Na sliki 2 je prikazana porazdelitvena funkcija uspešno poslanih e-poštnih sporočil za serije pošiljanj izvedenih v delu [9] – zgornjih 10% serij pošiljanj je označenih kot uspešnih, spodnjih 10% neuspešnih.

Avtorji Cutwaila kot pomemben faktor omenjajo hitrost pošiljanja kampanje (na ta način naj bi filtri imeli manj časa, da se naučijo kaj je nezaželena pošta) in lokacijo strežnika (sporočila poslana iz določenih držav naj bi bila bolj pogosto zavrnjena) to naj ne bi bilo pomembno [9]. Razlogi za takšne trditve so verjetno ekonomske narave ali premajhnega poznavanja področja [9]. Avtorji [9] sicer niso upoštevali filtrov nezaželene pošte, zato takšni sklepi ne veljajo v splošnem.

Pomemben faktor splošne uspešnosti je v sposobnost priprave vsebine, da sporočilo ne izgleda nezaželeno. Veliko tekstovnih filtrov se da pretentati z rabo slik in besedilom shranjenim v slike, ki so nato prikazane kot del HTML sporočil. Na tem področju je bilo opravljenih nekaj raziskav [2], [4].

4.2 Učinkovitost

Hitrost procesiranja sporočil (generiranje, pošiljanje, obravnava odgovora) je tako pomembna le pri merjenju učinkovitosti. Hitrost procesiranja e-poštnih sporočil v_{proc} lahko definiramo kot prikazuje enačba 1 [9]

$$\frac{1}{v_{proc}} = \max\left(\frac{S}{v_{gen}}; \frac{S}{v_{send}} + \frac{1}{v_{response}}\right) \tag{1}$$

kjer S predstavlja velikost sporočila v byteih, v_{gen} hitrost generiranja sporočil iz predlog, v_{send} hitrost pošiljanja sporočila in $v_{response}$ hitrost procesiranja odgovorov iz ciljnega strežnika.

V okviru učinkovitosti se tako pojavi tudi vprašanje števila botov, ki so potrebni za pošiljanje sporočil. Če enačbo izrazimo kot čas potreben za pošiljanje T in z B označimo število potrebnih botov, dobimo enačbo 2, ki jo predlagajo avtorji [9].

$$T = \frac{E}{vB} \implies T = \frac{E}{v \min(B, \frac{N}{n})} \tag{2}$$

kjer T predstavlja čas posamezne serije, E število vseh e-poštnih naslovov, N število e-poštnih naslovov v trenutni seriji, n število e-poštnih naslovov posameznega bota, v povprečno hitrost procesiranja e-poštnega naslova posameznega bota in B število aktivnih botov.

Cilj je čim bolj minimizirati število porabljenih virov in hkrati v najkrajšem možnem času dostaviti čim večje število sporočil.

5. SKLEP

V delu smo predstavili pregled področja nezaželene pošte, delovanje botnetov in kaj so kriteriji uspešne oz. neuspešne kampanje pošiljanja nezaželene pošte. Vsekakor bi bila dobrodošla celostna raziskava in pregled področja v smeri učinkovitosti filtrov in kakšna mora biti vsebina sporočil, da se filtre čim bolje zaobide. Pomemben faktor uspešne kampanje pošiljanja nezaželenih sporočil je tudi kaj naslovniki dejansko preberejo, kaj dojamejo kot očiten oglas in kaj kot, na videz uporabno vsebino. Tu je še veliko prostora za raziskave, saj meritve, kjer je edini kriterij dostavljivost sporočila ni dovolj in lahko vodi v napačne zaključke.

6. ZAHVALA

Zahvalili bi se prof. dr. Andreju Brodniku in dr. Gašperju Fele-Žoržu iz Fakultete za računalništvo in informatiko, Univerze v Ljubljani, ki sta omogočila raziskavo s področja analize nezaželjene pošte in s svojim znanjem znatno prispevala k nastanku tega dela.

7. REFERENCES

[1] D. L. C. J. W. S. B. AsSadhan, J. Moura. *Detecting botnets using command and control traffic*. Eighth IEEE International Symposium on Network Computing and Applications, 2009.

[2] S. Dhanaraj and V. Karthikeyani. A study on e-mail image spam filtering techniques. In *Pattern Recognition, Informatics and Mobile Engineering (PRIME), 2013 International Conference on*, pages 49–55, Feb 2013.

[3] Email spam. `http://en.wikipedia.org/wiki/Email_spam`, 2015. [Online; accessed 5-May-2015].

[4] Y. Gao, A. Choudhary, and G. Hua. A comprehensive approach to image spam detection: From server to client solution. *Information Forensics and Security, IEEE Transactions on*, 5(4):826–836, Dec 2010.

[5] History of email spam. `http://en.wikipedia.org/wiki/History_of_email_spam`, 2010. [Online; accessed 30-April-2015].

[6] How Exactly Does Spam Work? `http://www.webspam.org/`, 2014. [Online; accessed 30-April-2015].

[7] How To Avoid Spam. `http://www.techsupportalert.com/content/how-avoid-getting-spammed.htm`, 2013. [Online; accessed 30-April-2015].

[8] How to Avoid Spam Filters. `http://mailchimp.com/resources/guides/how-to-avoid-spam-filters/html/`, 2015. [Online; accessed 30-April-2015].

[9] J. Iedemska, G. Stringhini, R. Kemmerer, C. Kruegel, and G. Vigna. The tricks of the trade: What makes spam campaigns successful? In *Security and Privacy Workshops (SPW), 2014 IEEE*, pages 77–83, May 2014.

[10] C. Kanich, C. Kreibich, K. Levchenko, B. Enright, G. M. Voelker, V. Paxson, and S. Savage. Spamalytics: An empirical analysis of spam marketing conversion, 2008.

[11] Social spam. `http://en.wikipedia.org/wiki/Social_spam`, 2015. [Online; accessed 11-May-2015].

[12] Spamming. `http://en.wikipedia.org/wiki/Spamming/`, 2015. [Online; accessed 30-April-2015].

[13] R. C. P. R. M. S. Sérgio S.C. Silva, Rodrigo M.P. Silva. *Botnets: A survey*. Computer Networks 57, 2013.

[14] The History of Spam . `http://www.spamhelp.org/articles/HowDoIStopSpam.pdf`, 2010. [Online; accessed 30-April-2015].

[15] New and Improved. `http://www.techrepublic.com/blog/10-things/the-top-10-spam-botnets-new-and-improved/`, 2015. [Online; accessed 3-May-2015].

[16] What is spam? `https://kb.iu.edu/d/afne`, 2015. [Online; accessed 30-April-2015].

[17] What is spam, and how to avoid it. `https://runbox.com/email-school/what-is-spam-and-how-to-avoid-it/`, 2015. [Online; accessed 30-April-2015].

Forenzična analiza z DNSSEC

Sabina Oražem
so.orazem@gmail.com

Jakob Uršič
ju0994@student.uni-lj.si

ABSTRACT

Zastrupljanje DNS predpomnilnika je pogosta oblika računalniškega napada, s katero je mogoče nadzorovati žrtvin spletni promet, ji uvajati omejitve pri dostopu, oziroma do nje prenašati škodljivo programsko opremo. Njegovo obvladovanje je eden ključnih dejavnikov za zagotavljanje pravilnosti in dostopnosti internetnega omrežja in storitev.

V članku so predstavljene slabosti DNS infrastrukture, ki temelji na izziv-odgovor obrambi in, nasprotno od splošnega prepričanja, ponuja obilico možnosti za različne napade.

Nato predstavimo DNSSEC način obrambe proti predpomnilniškemu zastrupljanju, ki je zelo učinkovit in hkrati tudi edini obrambni mehanizem, ki omogoča analizo napada tudi dolgo po samem dogodku.

Keywords

varnost, spletni napadi, cache-poisoning, DNS, digitalni podpisi, kriptografski dokazi

1. UVOD

V zadnjem času je mogoče opaziti porast napadov, ki so hkrati tudi vse bolj dovršeni. Ti napadi spodbijajo pravilnost in stabilnost spletnih storitev, žrtve napadov pa so tako posamezniki, kot tudi podjetja in organizacije. Posledice so lahko zelo škodljive in privedejo do ekonomskih izgub, izgub strank, uporabnikov pri podjetjih, saj vplivajo na upadanje spletnega poslovanja itd.

Tukaj se bomo osredotočili predvsem na DNS, katerega pravilno delovanje je izjemnega pomena. Natančneje, pod drobnogled bomo vzeli zastrupljevanje DNS predpomnilnika. Tako poimenujemo dogodek, kjer napadalec pripravi DNS razreševalnike (angl. DNS resolvers), da sprejmejo in v predpomnilnik shranijo ponarejene DNS odgovore. Ko žrtev poizvaja pri napadenih razreševalnikih, jih ti naslovijo na napačne lokacije, kjer se lahko napad nadaljuje, npr. s krajo občutljivih osebnih podatkov, kot so uporabniška imena in gesla, podatki kreditne kartice, prisluškovanje osebnim pogovorom itd.

Najpogostejša obramba proti omenjenim napadom je izziv-odgovor mehanizem. Omenjen mehanizem, pravzaprav gre za avtentikacijo, deluje na način, da eden od udeležencev pošlje izziv, na katerega mora drugi pravilno odgovoriti. Pogost primer uporabe takšne zaščite je zahteva gesla pri vstopu na spletno stran. Če uporabnik gesla ne pozna, bo odgovor na zastavljeni izziv napačen.

Tak način zaščite pa ni vedno najbolj učinkovit. Predvsem imamo tukaj v mislih t.i. man-in-the-middle (MitM) napade, katerih lastnost je, da napadalec pridobi dostop do prometa med ostalima komunikajočima uporabnikoma brez njune vednosti. V takem primeru lahko zajame izziv v prometu in nanj tudi pravilno odgovori v svojem (ponarejenem) odgovoru. Primer, če komunikacija v obe smeri uporablja enak protokol, lahko MitM izziv zajame, ter pošlje enak izziv naprej naslovniku preko druge vzpostavljene povezave. Ta mu v svojem odgovoru izda rešitev izziva, ki jo napadalec posreduje nazaj prvemu naslovniku in se s tem uspešno avtenticira (angl. reflection attack). Do nedavnega je veljalo, da ja takšen način zaščite ustrezen vsaj pred zunanjimi napadalci, vendar pa zadnje ugotovitve kažejo, da lahko tudi ti najdejo način za zaobitev takšnega varnostnega sistema. Praktičen primer navaja npr. vir [3], ki opisuje napad na google.rw domeno. Na sliki 1 je prikazano, do kakšne mere so napadalci povečali TTL pri tem napadu. TTL pri DNS zapisih določa število sekund hrambe zapisa v predpomnilniku. Zaradi ažurnosti je ponavadi TTL nastavljen na 300 (5 min), napadalci pa so ga v tem primeru nastavili kar na 172800.

Figure 1: Sprememba DNS TTL pri napadu.

First seen	Last seen	IPs
10/25/13	10/25/13	173.194.40.119 (TTL: 300) 173.194.40.120 (TTL: 300) 173.194.40.127 (TTL: 300) 31.170.160.228 (TTL: 172800) 74.125.226.247 (TTL: 300) 74.125.226.248 (TTL: 300) 74.125.236.183 (TTL: 300) 74.125.236.184 (TTL: 300) 74.125.236.191 (TTL: 300)

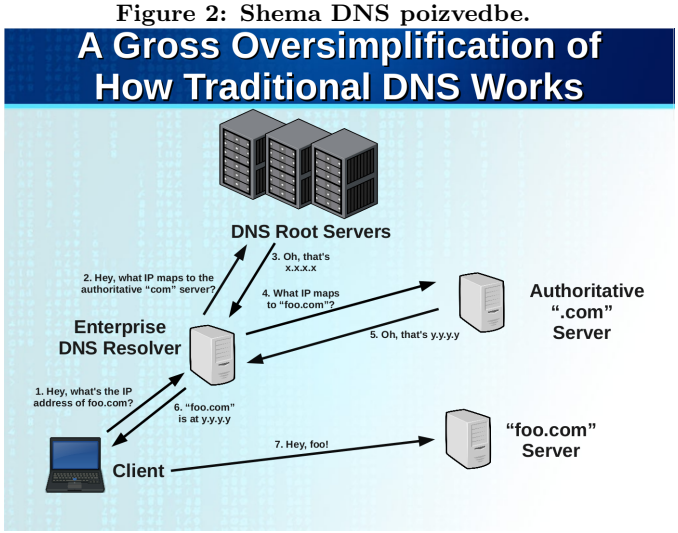
Vse omenjeno nas privede do ugotovitve, da proti sodobnim napadalcem takšen način obrambe ni več ustrezen za zagotovitev varnosti uporabnikov. Najboljša alternativa, ki

je podrobneje predstavljena tudi v tem članku, je uporaba DNSSEC(domain name system security extensions). Gre za kriptografsko zaščito DNS, ki zapise overi z digitalnim podpisom. Njena standardizacija sega že v leto 1997, vendar kljub temu še ni prišla v širšo uporabo. Ta omogoča učinkovito zaščito in kot edina zagotavlja tudi kasnejše preiskovanje napadov, ter posledično omogoča zbiranje dokazov za morebitne forenzične analize, ne glede na spretnosti napadalca.

2. DNS

DNS predstavlja porazdeljeno podatkovno bazo naslovnih preslikav. Ta pretvori imena domene v neko drugo vrednosti (npr. IPv4 naslov). Uporabnik preko razreševalnika (angl. resolver) pridobi ustrezen DNS zapis, razreševalnik pa do lokacije zapisa pride z izpraševanjem imenskih strežnikov. Če strežnik odgovori z ustreznim DNS zapisom (možen odgovor bi bil tudi, da domena ne obstaja), si ta zapis razreševalnik shrani v predpomnilnik. Dokler se ta zapis nahaja v predpomnilniku, so vsa nadaljnja izpraševanja vrnjena direktno od tam. Primer opisanih dogodkov DNS poizvedbe je prikazan na sliki 2.

Figure 2: Shema DNS poizvedbe.



3. ZASTRUPITEV PREDPOMNILNIKA

Razreševalnik deluje na način, da sprejema samo odgovore, za katere je zaprosil. To pomeni, da mora biti prvi del vsakega napada, katerega cilj je zastrupitev predpomnilnika, sprožitev DNS povpraševanja. Šele ko je povpraševanje sproženo lahko sledi drugi del, vrnitev lažnega odgovora.

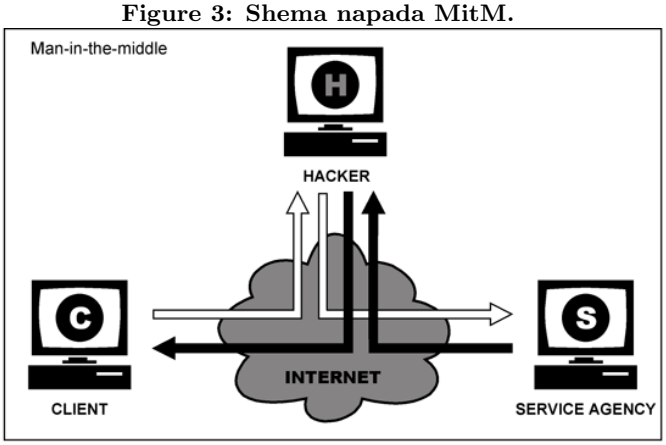
Če ima napadalec direkten dostop do DNS razreševalnika, to se zgodi npr. v situaciji, ko se nahaja na omrežju, ki uporablja skupen razreševalnik, lahko brez večjih težav sproži poljubno število DNS povpraševanj.

Druga možnost je uporaba odprto dostopnega razreševelnika (angl. open DNS resolver). Ta omogoča storitve vsakemu uporabniku, hkrati pa s tako lastnostjo postane prijaznejši za napadalce, ki zastrupljajo predpomnilnik. Ponavadi je zato storitev vendarle omejena samo na uporabnike nekega omrežja, kljub temu pa še vedno obstajajo pristopi, ki omogočajo dostop tudi od zunaj. Primer takšnega pristopa je uporaba zlonamerne skripte (t.i. puppet). Za puppet napade velja, da se lahko zgodijo povsem brez vednosti žrtve, na straneh, ki delujejo popolnoma neškodljive, npr. preko oglasnih sporočil na spletni strani. Ko se enkrat takšna skripta prenese na žrtvino napravo, lahko sproža DNS poizvedbe, ki jih napadalec želi zastrupiti, brez žrtvine vednosti[4]. To seveda ni edini način za doseganje takšnega cilja, obstajajo tudi druge, manj znane tehnike, npr. s pošiljanjem e-pošte ipd.

4. DNS ZASTRUPITEV PREDPOMNILNIKA IN MITM

Splošni internetni protokoli, kot sta DNS in usmerjanje prometa (angl. routing), imajo zaščito osnovano predvsem pred napadalci od zunaj, bolj malo pozornosti pa je namenjeno zaščiti proti MitM. Razlog se nahaja predvsem v prepričanju, da slednjih napadalcev ni oz. jih je zelo malo, kar pa ne drži povsem. Obstaja več načinov, kako si zagotoviti takšno pozicijo. Uporabniki do omrežja ne dostopajo več samo od doma, temveč je vse več uporabnikov tudi na (brezžičnih) javnih omrežjih, kjer se brez večjih težav lahko izvede napad. Zelo učinkovito poseganje v pravilnost delovanja DNS iz pozicije MitM je mogoče izvajati tudi na vozliščih hrbtenice omrežja, npr. cenzura določenih spletnih vsebin. Slednje seveda velja za mogočnejše napadalce, ki sploh imajo dostop do hrbtenice, kot so vladne organizacije. Obstaja tudi možnost preusmeritve prometa preko napadalca (angl. route hijacking), ki se s takim dejanjem spet spravi v pozicijo MitM in si omogoči podlago za nadaljnje napade. Na sliki 3 je predstavljena shema MitM.

Figure 3: Shema napada MitM.



Pogost način zastrupljevanja DNS predpomnilnika se zgodi tudi z izkoriščanjem ranljivosti imenskih strežnikov in registerjev. Za razliko od napadov, kjer je tarča določena žrtev, je tak napad globalen in zadeva vse uporabnike, ki dostopajo do ogroženega strežnika.

4.1 MitM in javno dostopna omrežja

Vse več možnosti obstaja za dostop do interneta preko javnih wifi točk, npr. hoteli, letališča, lokali ... Večina uporabnikov se pogosto poslužuje takega načina dostopanja, redko kdo izmed njih pa se verjetno zaveda, da se z dostopom preko

takšnih omrežij povečajo možnosti napada. Ti so lahko dvojega izvora: zlonamerni operator ali uporabnik.

V obeh primerih postane pod vprašajem pravilnost in dostopnost internetnih storitev. Primer učinkovitega napada je z lažnim DNS odgovorom in preusmeritvijo uporabnikov na neželene gostitelje, kjer npr. prenesjo škodljivo programsko opremo. v primeru, da je zlikovec operater, ima ta seveda vsa pooblastila na mreži, saj poseduje usmerjevalnik, preko katerega gredo vsi podatki. Če je zlonamernež uporabnik omrežja, lahko pridobi MitM ugodnosti z lažnimi DHCP odgovori, ki jih posreduje novo prijavljajočim-se uporabnikom. S takimi odgovori poda napačen IP/MAC naslov lokalnega rekurzivnega razreševalnika (npr. na svojo lastno mrežno kartico(NIC)). Na ta način napadalec preseže vse nadaljnje dns poizvedbe in lahko z njimi poljubno manipulira.

Mogoča pa je tudi izvedba napada brez zavzetja MitM pozicije. Če se nahajamo na javnem wi-fi omrežju, to pomeni, da vsaka naprava lahko prejme vse pakete, ne glede na ciljni naslov. To lastnost je mogoče zlorabiti tako, da se na poslano dns poizvedbo pošlje napačen odgovor, še preden prispe pravi.

Poleg vseh omenjenih nevarnosti velja pozornost nameniti še nadzorovanju uporabnikovih online aktivnosti. S spremljanjem MAC naslova (ki enolično določa napravo), lahko nadzorujemo preko katerih omrežji se uporabnik prijavlja. Hkrati obstajajo tudi datoteke dnevniških zapisov prijav, ki lahko hranijo podatke o uporabnikih še veliko časa in omogočajo tudi kasnejše poizvedovanje.

4.2 MitM v hrbtenici povezav

V takšnih primerih ne gre več za navadne napadalce, temveč za pomembne vladne organizacije, kot so npr. Nacionalna Varnostna Agencija (NSA). Omenjeni si lahko privoščijo sodelovanje z internetnimi ponudniki in tako dobijo dostop do vozlišč na hrbtenici povezav. Tam razporedijo posebne strežnike, ki so namenjeni proizvajanju lažnih DNS odgovorov. Pomembno je poudariti, da zaradi lokacije teh strežnikov, njihov odgovor vedno prvi doseže uporabnika. Kasneje prispe tudi resničen odgovor, ki pa se zavrže, saj se po protokolu obravnava le prvi pravilen odgovor.

4.3 MitM in preusmerjanje prometa

Internet sestavlja več avtonomnih sistemov, ki so med seboj povezani z določenimi usmerjevalnimi protokoli. Obstajajo t.i. BGP (Boder Gateway Protocol) sporočila, ki naznanjajo spremembe v usmerjanju. Usmerjevalniki generirajo takšna sporočila, ko se zgodi sprememba (topologije, rekonfiguracije, ...) Pomanjkljivost teh sporočil je v tem, da ne vsebujejo nikakršne avtentikacije, kar je mogoče izkoristiti za napad. Tako lahko zlonamernež s spremembo BGP sporočila preusmeri promet preko svojega omrežnega operaterja, kjer pridobi položaj MitM, in na ta način pošilja ponarejene DNS odgovore v promet. O takšni obliki napada je bilo poročano v letu 2013 [5], kjer so napadalci promet preusmerili preko Belorusije oz. preko Islandije. Na sliki 4 je prikazano, kako je v času napada potekal promet iz Guadalajare v Washington, z velikim ovinkom preko Belorusije.

Figure 4: Sled preusmerjenega prometa preko Belorusije.

5. NAPAD Z OGROŽANJEM GOSTUJOČE INFRASTRUKTURE

Napad na DNS je napogosteje izveden preko ogrožanja gostujoče infrastrukture DNS, to je na primer napad na register domene ali izkoristek ranljivosti pri nazivih strežnikov.

To je eden najdonosnejših napadov, ko napadalec ni oblike MitM. Raziskave na tem področju so aktualne, saj se napadi na DNS, ki omogočajo prevzem domene, pojavljajo vse pogosteje.

5.1 Ogrožanje registra

Pogosti so napadi, ko napadalec izkoristi ranljivosti ali pomankljivosti uporabniškega vmesnika. Predvsem so izpostaljene pomankljivosti pri preverbah vnosov uporabnika. Vrednosti oziroma podatki, ki jih uporabnik oziroma napadalec vnese preko uporabniškega vmesnika, se ne preverijo ali pa je preverba pomankljiva.

Znani napadi teh vrst so na primer 'injection attacks' to je 'vbrizgavanje' zlonamernih podatkov. Vbrizgavanje napadalcu omogoča spreminjanje DNS zapisov žrtvine domene.

Figure 5: Python koda oziroma program DNSInject.py, ki omogoča spreminjanje DNS zapisa.

```
# Craft the packet with scapy
add_packet = sr1(IP(dst=name_server)/UDP()/DNS(
    opcode=5,
    qd=[DNSQR(qname=dns_zone, qtype="SOA")],
    ns=[DNSRR(rrname=new_dns_record,
        type="A", ttl=120, rdata=ip_value)]))
```

Omenjeno je nekakšna odskočna deska za še bolj raznolike napade, kot so raznašanje nezaželjene pošte (spam), spletno ribarjenje (phishing), širjenje zlonamerne kode (malware) in drugi.

Ugotovljeni so bili možni napadi tudi na druge načine na primer, ob registraciji domene. Domene bi lahko registrirali pod legitimnim imenom, ki pripada drugim domenam (le-te jih nimajo pod nadzorom, kot imajo svoje).

Žal je prepoznavanje in preprečevanje tovrstnih napadov

Figure 6: Opis DNSInject.py programa.

```
root@VG-kali:/mnt/hgfs/gitrepos/PenTestScripts# ./DNSInject.py -h
usage: DNSInject.py [--add] [--delete] [-ns ns1.test.com]
                    [-d mynewarecord.test.com] [-ip 192.168.1.1]

DNSInject is a tool for modifying DNS records on vulnerable servers.

optional arguments:
  --add                 Add "A" record to the vulnerable name server.
  --delete              Delete "A" record from the vulnerable name server.
  -ns ns1.test.com      Nameserver to execute the specified action.
  -d mynewarecord.test.com
                        Domain name to create an A record for.
  -ip 192.168.1.1       IP Address the new record will point to.
```

težko. V te namene bi morali preverjati lastništvo pri sami registraciji.

5.2 Ogrožanje naziva strežnika

Tovrstni napadi imajo malce daljšo zgodovino.

V tem primeru se napadalec spravi nad ranljivost operacijskega sistema ali samega DNS programa (software). To je odskočna deska za pridobivanje nepooblaščenega dostopa do sistema in izvajanje zlonamerne oziroma kode, ki sistemu povzroči škodo.

Opisane ranljivosti so bili ugotovljene na primer pri MS server, Bind versions in PowerDNS. Med najbolj znanimi ranljivostmi so na primer "buffer overflow", površno sprejemanje vnešenih vrednosti in ostale. "Buffer overflow"napadalcu omogoči nedovoljen dostop do sistema in izvajanje naključne kode, ki je lahko tudi zlonamrena.

6. DNSSEC

DNSSEC je kratica, ki označuje **Domain Name System Security Extensions standard**. Le-ta je bil razvit v namen prepoznavanja 'MitM' in 'cache poisoning' ranljivosti pri DNS-ju. Celovite informacije nam omogočajo kriptografski digitalni podpisi v zapisih DNS-ja. Podpis omogoča preverbo ali so podatki zapisani v DNS-ju enaki tistim, ki so bili objavljeni v ciljni coni.

Ko uporabnik želi dostopati do spletne strani, stub resolver, operacijskega sistema uporabnika, zahteva zapise domene. Sledi poizvedba po ključu pripadajoče cone, ključ se pridobi iz DNSSEC. Sledi preverjanje ali se podatka ujemata. Če validacija uspe uporabnik lahko dostopi do željene strani, nasprotno DNSSEC uporabniku prepreči dostop do zolnamerne strani.

DNSSEC definira nove zapise (RRs) za shranjevanje podpisov in ključev pri uporabi preverbe pristojnosti DNS odzivov.

Na primer, **RRSIG** zapis vsebuje podpise, ki zagotavljajo pristnost RR niza. S podpisovanjem samo RR niza in ne specifičnih odzivov, DNSSEC omogoča izračun podpisov izven medmrežja (off-line). To je pomembno tako za hitrejše (računanje podpisa je računsko zahtevno), kot tudi varnejše podpisovanje.

Vsaka cona generira par ključev **(sk, vk)**. Par ključev vsebuje ključ, ki služi podpisovanju ter ključ, ki služi preverjanju le-tega. Ključ za podpisovanje podatkov je vedno tajen

Figure 7: Shema delovanja DNSSEC.

in nikoli na voljo v medmrežju. Strežnik tako poleg zapisov RR posreduje tudi podpis v RRSIG (v RR-ju). Da bi se lahko izognili odzivnim napadom ima vsak podpis tako imenovan datum zapadlosti. Ključ, ki služi preverjanju podpisa, potrditvi izvora in integriteti podatkov **(vk)** je javen in hranjen v **DNSKEY** (v RR-ju).

6.1 Pasti soodvisnoti domen

Tovrstne pasti so pogoste. Pojavijo se kadar neka domena vsebuje zapise (RR) v drugih domenah, torej je odvisna od druge domene. Ko je domena, ki je podpisana z DNSSEC, odvisna od domene, ki tovrstne zaščite ne vsebuje, lahko zaščita upade.

6.2 Operativni izzivi

Pojavlja se več izzivov na področju uvajanja DESSEC-a. Tokrat se osredotočamo na operativne izzive in možne prekinitve delovanja.

Po narejenih študijah so prekinitve večinoma posledica napak v ključih in podpisovalnih postopkih. Večina neuspehov DESSEC je posledica človeških napak in operativne izzive bi lahko zmanjšali, če bi podpisovanje in generiranje ključa avtomatizirali.

7. FORENZIKA, DOKAZI IN ZAZNAVANJE Z DNSSEC

DNSSEC je uporaben pri ugotavljanju napadov na DNS in forenzični analizi le teh in prav lastnost podpisovanja je tista, ki te funkcionalnosti omogoča. Vse to je omogočeno vsakomur, ki ima na voljo javni ključ.

V smislu forenzične analize nam sistem nudi točen datum napada ter gostitelje na katere so bili podatki preusmerjeni.

Podpis nosi lastnosti: datum zapadlosti podpisa, datum na katerega je bil podpis generiran ter oznako kriptografske "opreme", kot na primer algoritem, ki jo uporablja pri zasnovanju podpisa.

Kriptografski podpis naj bi bilo nemogoče ponarediti, oziroma ga ponaredijo lahko le zelo močni napadalci, kot na primer državni in vojaški organi. Odpre se nova veja raziskovanja, kako torej zaznati tudi napade tako močnega or-

gana. Sistem v tovrstne namene, bi bil moral zbrati DNS odzive (skupaj s pripadajočimi podpisi in ključi) preko konfiguracije pravil v požarnem zidu (firewall).

7.1 Forenzična analiza

Prav značka na podpisu nam nudi uporabne informacije, ki omogočajo podrobno analizo o tem kdaj je bila neka povezava tretirana za veljavno, torej kdaj je bil napad zasnovan. Časovne značke v podpisu omogočaju operatorju omrežja analizo, ko so bili 'zastrupljeni' zapisi dostavljeni.

7.2 Dokazi

V večini primerov je potrebno dokazovanje napada oziroma zločina tudi tretji osebi, kot na primer sodniku. Oškodovani lahko predstavi DNS zapise zlonamernega dejanja, skupaj s kriptografskimi podpisi. Tretja oseba lahko podpis preveri, saj ji je na voljo ključ za prevero podpisa, ki je javen, torej ne tajni. Podpis lahko dokaže, da ni šlo za napako, temveč za dejanski napad.

Tovrstni dokazi niso mogoči pri drugih DNS kriptografskih obrambah, kot je na primer Eastlake cookies.

7.3 Zaznava napada

DNS je porazdeljena infrastruktura, torej eno samo domeno zastopa več strežnikov, nadalje so tudi strežniki porazdeljeni. Tako je težko, če ne celo nemogoče ogroziti vse strežnike neke domene hkrati. To bi od napadalca zahtevalo sledenje več internetnim povezavam (linkom) naenkrat (ti pripadajo različnim AS = Automous Systems) ali ogrozitev vseh strežnikov.

Pri razvoju ideje za datekcijo napada, izhajamo iz predpostavke, da zgoraj omenjeno skoraj ne uspe tudi najmočnejšim napadalcem.

Dejstvo, da napadalec ne more hkrati ogroziti vseh strežnikov in povezav (linkov), pomeni, da se bodo različni nazivi strežnikov, odzvali različno. Medmrežje lahko zagotovi zaupanja vredne DNS odzive. Preko proksija lahko pošljemo poizvedbe na serverje in te odzive primerjamo s tistimi, ki nam jih je zagotovil DESSEC, če pride do neskladanj, lahko sumimo na vsiljivca.

Žal še ni bil oblikovan standardiziran sistem za obveščanje uporabnikov o napadu. Možna rešitev v tej smeri bi bil razvoj 'DNSSEC-aware' (DNSSEC zavedajočih se) brskalnikov, ki bi uporabnika obvestili, da je bil preusmerjen na drugo spletno mesto.

8. CONCLUSIONS

Varen DNS je nujen za stabilnost in polne funkcionalnosti intreneta. V obdelanih poglavjih je bila opisana študija DNS varnosti in prikazano je bilo, da je DNS infrastruktura ranljiva ter, da so napadi nanjo pogosti (napdalci so različnih profilov in znanj). Največji problem predstavlja zaznavanje napadov. Napad je težko, morda celo nemogoče zaznati, razen če je tekom napada prekinjena povezava.

Veliko je delov internetne varnosti, DNSSEC lahko ublaži pomisleke glede varnosti s strani MitM napadov in cache poisoning, ni pa to rešitev za popolno internetno varnost. Da bi se zaščitili tudi pred ostalimi prepogostimi grožnjami, kot so na primer spoofing ali phishing, potrebujemo tudi druge plasti zaščite, kakor je SSL certifikat in two-factor avtentikacija.

Četudi se oba tako DNSSEC in SSL zanašata na javni kriptografski ključ, vsak izvaja drugačno oziroma nudi druga?no zaščito. Zatorej sta za popolno internetno zaščito potrebna oba in nikakor eden ne nadomesti drugega. Zelo preprosto rečeno DNSSEC obravnava vprašanje 'Kje', medtem ko SSL odgovarja na vprašanja 'Kdo' in 'Kako'.

Optimalno bi bilo ne samo preprečevanje napadov, zaželjena je tudi sama zaznava napada in zbiranje podatkov o zaznanem napadu. Na podlagi zbranih podatkov o dogajanju lahko dejanje forenzično analiziramo. DNSSEC je priporočljiv saj tudi generira kriptografske dokaze, ki žrtvi omogočajo dokazovanje napada tretjim osebam. Prav tukaj se DNSSEC razlikuje od drugih podobno funkcionalnih aplikacij.

9. REFERENCES

[1] Haya Shulman and Michael Waidner Fachbereich Informatik Technische Universita?t Darmstadt, Towards Forensic Analysis of Attacks with DNSSEC, 2014 IEEE Security and Privacy Workshops, DOI 10.1109/SPW.2014.20, 2014.

[2] Olafur Guomundsson and Stephen D. Crocker, Observing DNSSEC validation in the wild, 2014.

[3] F. Denis, The GOOGLE.RW Hijack, spletna stran 1, 2013.

[4] S. Antonatos, P. Akritidis, V. T. Lam, and K. G. Anagnostakis, Puppetnets:Misusing Web Browsers as a Distributed Attack Infrastructure, ACM Transactions on Information and System Security, vol. 12, no. 2, pp. 12:112:15, Dec. 2008.

[5] K. Bode, Somebody is Hijacking Massive Amounts of Data Via Iceland, http://www.dslreports.com/shownews, 2013.

[6] Nikolaos Alexiou, Stylianos Basagiannis Panagiotis Katsaros, Tushar Deshpande Scott A. Smolka, Formal Analysis of the Kaminsky DNS Cache-Poisoning Attack Using Probabilistic Model Checking, 2010 IEEE 12th International Symposium on High Assurance Systems Engineering, 2010.

[7] Verisign, Domain Name System Security Extension, Authenticating the Internet from end-to-end, spletna stran2, 2015.

[8] Christopher Truncer, CHRISTOPHER TRUNCER'S WEBSITE,A Hacker's Perspective, spletna stran 3, April 30, 2014.

Preiskovanje napadov povezanih z izvornimi vrati 0

Rok Kogovšek
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana, Slovenija
63100185
rok.kogovsek@student.uni-lj.si

Nikolaj Janko
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana, Slovenija
63100248
nj1466@student.uni-lj.si

Miha Pleško
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana, Slovenija
63100330
mp8538@student.uni-lj.si

POVZETEK

V članku razložimo osnovne lastnosti preiskovanja omrežja s pomočjo postavljanja vrednosti izvornih vrat paketov TCP na 0 ter kako takšno preiskovanje detektiramo. Opiramo se na referenčni članek z naslovom *Multidimensional investigation of source port 0 probing* [4].

V prvem delu članka predstavimo postopek detekcije, ki so ga razvili avtorji referenčnega članka tj. s pomočjo gručenja vektorjev značilk paketov TCP odkrijemo glavna žarišča napada. Za posamezno žarišče znamo s pomočjo posebne zbirke poročil o znanih škodljivih programih celo ugotoviti, kateri DLL je bil uporabljen za napad.

V drugem delu si ogledamo nekaj alternativnih pristopov k odkrivanju neželenega skeniranja omrežja. Izkaže se, da obstaja veliko različnih načinov in pristopov kako zgodaj zaznati napad.

V tretjem delu predstavimo najodmevnejše žrtve kiber napadov, ki so se v zgodnji fazi začeli ravno s skeniranjem omrežja. Ogledamo si povzetke postopkov vdora ter posledice, ki so jih vdori za seboj pustili. Predstavimo tudi nekaj statistik v zvezi s tovrstnimi napadi.

1. UVOD

Referenčni članek[4] se osredotoči na dogodek, ko so 2. novembra 2013 Ciscovi senzorji v globalnem omrežju detektirali več kot petkratno povečanje paketov TCP z vrednostjo izvornih vrat 0 (ang. source port 0). Tovrstni paketi TCP sami po sebi sicer niso škodljivi, vendar obstaja velika verjetnost, da predstavljajo zgodnjo fazo kiber napada. Vrednost 0 je za vhodna vrata TCP rezervirana (RFC 6142[13]), torej prepovedana za uporabo, in ravno zato zanimiva za omrežne napadalce. Odziv sistema na to prepovedano vrednost je namreč rahlo različen pri različnih verzijah sistemov, kar omogoča tako imenovano profiliranje sistema (ang. system fingerprinting). Podatki pridobljeni s povpraševanjem (ang. probing) služijo za odkrivanje ranljivosti, ki se potencialno lahko uporabijo pri dejanskem napadu sistema.

Avtorji referenčnega članka so imeli na voljo dovolj izčrpen posnetek omrežnega prometa, ki so ga Ciscovi senzorji zajeli novembra 2013, da so lahko s pomočjo posebne analize poiskali izvor sumljivo povečane količine paketov TCP z nastavljenimi izvornimi vrati na vrednost 0.

2. PRISTOP AVTORJEV REFERENČNEGA ČLANKA

Avtorji referenčnega članka so imeli na razpolago tri vire podatkov, na podlagi katerih so izvedli analizo prometa:

1. posnetek sivega prometa (12 GB/dan)
2. posnetek prometa DNS (1.3 mio sporočil/dan)
3. zbirko poročil o znanih škodljivih programih (30 mio poročil)

Najprej so analizirali posnetek darknet prometa, nato so rezultate primerjali z rezultati analize DNS vnosov ter zbirko poročil. V nadaljevanju si bomo pogledali te tri korake.

2.1 Analiza sivega prometa

Beseda sivi promet (ang. **darknet oz. "temačna stran interneta"**) označuje omrežni promet, ki je naslovljen na obstoječ in veljaven vendar nezaseden naslov IP. Razloge, zakaj bi nekdo pošiljal pakete na nezaseden naslov delimo v tri skupine:

- *skeniranje omrežja* (ang. probing)

  Skeniranje izvajajo roboti (ang. bot), črvi (ang. worm) ali namenski programi (ang. tools) da odkrijejo obstoječe entitete v omrežju. Skeniranje po IPju imenujemo horizontalno skeniranje, skeniranje po vratih pa vertikalno skeniranje.

- *izzvano pošiljanje* (ang. backscattered)

  Do izzvanega pošiljanja pride, če nam nekdo pošlje zahtevek iz lažnega naslova. Tedaj mu odgovorimo na lažni naslov. Primer so DoS (ang. Denial of Service) napadi, kjer žrtev odgovori na lažni naslov napadalca.

- *napaka v omrežni konfiguraciji*

  Sem spadajo napake zaradi nepravilno konfigurarane mrežne kartice ali usmerjevalnika, zaradi katerih paketi TCP včasih uidejo v sivo omrežje.

Promet, ki se pošlje v sivo omrežje, zajamejo posebni Ciscovi senzorji (ang. dark sensors). Analiza tega prometa je pomembna za zgodnje odkrivanje kibernetskih napadov. Že takoj na začetku avtorji izmed množice podatkov v sivem omrežju izluščijo le pakete TCP z nastavljenimi izvornimi vrati na vrednost 0. Nadaljnjo analizo izvajajo le na teh paketih.

Ker je vseh podatkov znotraj posameznega paketa preveč, uporabijo le podmnožico tistih podatkov, ki na dovolj diskriminativen način opisujejo paket. S tem bistveno zmanjšajo dimenzijo preiskovalnega prostora iz nekaj sto dimenzij na 29 dimenzij (Slika 1). Izbor ustreznih atributov temelji na članku [3].

Features		
Data link layer	1	Delta time with previous packet
	2	Packet length
	3	Frame length
	4	Capture length
	5	The flag 'frame' is marked
Network layer	6	IP header length
	7	IP flags
	8	IP flags: reversed bit
	9	IP flags: do not fragment bit
	10	IP flags: more fragments bit
	11	IP fragment offset
	12	IP time to live
	13	IP protocol
Transport layer	14	TCP segment length
	15	TCP sequence number
	16	TCP next sequence number
	17	TCP acknowledgment number
	18	TCP header length
	19	TCP flags
	20	TCP flags: congestion window
	21	TCP flags: ECN-echo
	22	TCP flags: urgent
	23	TCP flags: acknowledgment
	24	TCP flags: push
	25	TCP flags: reset
	26	TCP flags: syn
	27	TCP flags: fin
	28	TCP window size
	29	UDP length

Figure 1: Vektor značilk, ki predstavlja posamezni paket TCP. Izbor 29-ih značilk je po navedbah avtorjev dovolj diskriminativen.

Nad vektorji značilk nato izvedejo dva algoritma gručenja, k-means in EM (ang. Expectation-Maximization), s čimer povežejo sorodne omrežne pakete v gruče (ang. clusters). Rezultat je prikazan na Sliki 2.

Na sliki opazimo, da so dobili štiri gruče, kar pomeni, da so v sivem omrežju obstajali štirje različni tipi omrežnih paketov. Izrisali so graf silhuet gruč EM (Slika 3), ki ponazarja kakovost gruč. Graf silhuet izriše razdaljo posamezne točke

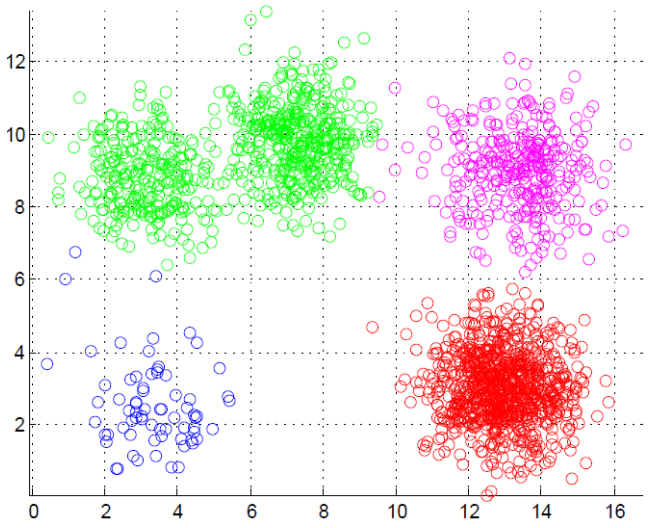


Figure 2: Rezultat algoritma gručenja EM so štiri gruče; vsaka predstavlja drugačen tip omrežnih paketov.

iz grupe do sosedne grupe; vrednost 1 pomeni veliko oddaljenost, vrednost 0 pa da sta dve gruči blizu skupaj. Opazimo, da ima večina točk vrednost silhuete nad 0.6, torej so gruče jasno ločene med seboj. V nadaljnji raziskavi so iskali skupne lastnosti posamezne gruče.

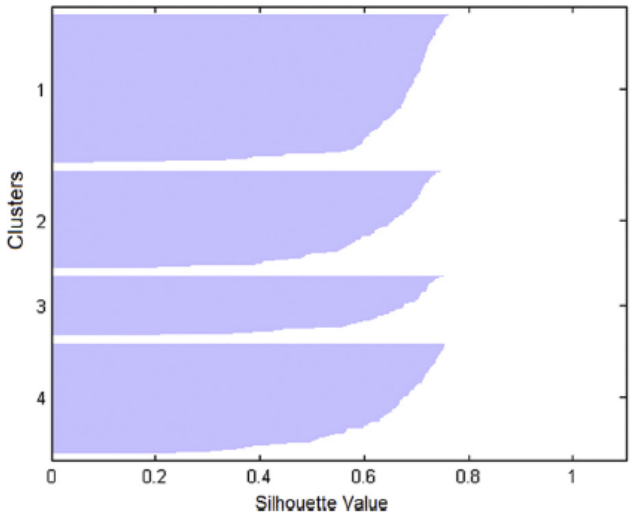


Figure 3: Graf silhuet gruč ponazarja kakovost gručenja. Za vsako točko prikažemo njeno oddaljenost do najbližje sosedne gruče. Večina točk ima vrednost nad 0.6, kar dokazuje visoko kakovost gruč.

Podrobnejša analiza paketov prve gruče je pokazala, da se v njej nahajajo omrežni paketi, ki vsi izvirajo iz istega naslova IP, lociranega nekje v Nemčiji. Vsak izmed 800.000 paketov je naslovljen na drug ciljni naslov, kar je jasen znak horizontalnega preiskovanja omrežja. Takšno preiskovanje je običajno znak prve faze kibernetskega napada, v kateri napadalci s pošiljanjem paketov TCP z izvornimi vrati z vrednostjo 0 pridobivajo podatke o žrtvi.

Druga gruča prav tako vsebuje pakete, ki vsi izvirajo iz istega naslova IP, tokrat lociranega nekje na Nizozemskem. Vendar gre v tem primeru za vertikalno preiskovanje, saj preišče kar 60.000 različnih vrat.

Tretja gruča je drugi primer horizontalnega preiskovanja omrežja, ki se osredotoči na 9.000 različnih IPjev, za katere preišče vrata 445, 22 in 3389. Za vrati 445 se običajno skriva program Microsoft Directory, za 22 varna lupina (ang. secure shell, ssh) in za vrati 3389 Remote Desktop. Ti programi so znani po varnostnih luknjah.

Četrta gruča predstavlja pakete, ki so rezultat napačne konfiguracije omrežja. Vsebuje neškodljive in nenamerne pakete, ki za preiskavo niso relevantni.

2.2 Analiza prometa DNS

Pojem **promet DNS** označuje omrežni promet, ki se odvija rekurzivno med strežniki DNS. Smiselno je preveriti, kakšna je bila aktivnost strežnikov DNS za naslove IP, ki smo jih izpostavili pri analizi sivega omrežja. Za gruče zlonamernih strežnikov je značilno, da imajo naslove IP zaporedne in da nanje kaže veliko vnosov DNS.

2.3 Uporaba zbirke poročil o znanih škodljivih programih

Zbirka poročil o znanih škodljivih programih je posebna zbirka shranjena v datoteki XML. Vsebuje lastnosti (kam se poskuša povezati, katere porte uporablja, ...) in simptome (katere datoteke DLL uporablja, katere registre spreminja, koliko pomnilnika zasede, ...) mnogih zlonamernih programov. Zbirka je redno dopolnjujena z novimi vnosi s strani raziskovalcev, ki vsak novi zlonamerni program temeljito preučijo v kontroliranem okolju. S pomočjo takšnih zbirk je možno ugotoviti kateri program je povzročil napad. Algoritem za iskanje po zbirki je sledeč:

```
1  Input: Dynamic malware analysis reports: XMLs;
2  List of the scanning source IPs: ProbingIPs;
3  Output: List of malware samples that infected the
   probing machines: MalwareList
4  for xml in XMLs do
5      if xml.getMalware().getdestIP() in ProbingIPs then
6          if xml.getMalware().getTime()==Nov, 2013 then
7              MalwareList.add(xml.getMalware().getName());
8          end
9      end
10 end
```

Figure 4: Algoritem ugotavljanja konkretnega zlonamernega programa, ki je bil uporabljen za novembrski napad. Algoritem se sprehodi po zbirki poročil o znanih škodljivih programih in poišče takšne s podobnimi simptomi.

Algoritem vrne, da je bil novembrski napad izveden s pomočjo programa "Virus.Win32.Sality".

3. ALTERNATIVNI PRISTOPI

V tem sklopu si bomo ogledali sorodne primere in njihove pristope pri analizi omrežnega prometa za odkrivanje različnih zlonamernih napadov in zlonamernega skeniranja omrežja.

Za odkrivanje, zaznavanje in posledično tudi preprečevanje takih napadov obstaja že več različnih pristopov, ki jih lahko delimo na dva dela. Prvi (makroskopski) pristop temelji na skeniranju in analizi prometa v omrežju. Drugi (mikroskopski) pa na analizi podatkov pridobljenih iz zlonamerne programske opreme ali zlonamerne kode (analiza izvorne kode zlonamernih programov).

3.1 Analiza preiskovalnih botnet napadov

Botnet-i danes prevladujejo v svetu preiskovalnih napadov. Napadalci preko različnih spletnih virov omogočajo prenos zlonamerne kode na računalnike in naprave različnih uporabnikov. Prenešena zlonamerna koda izvaja preiskovalno analizo omrežja na računalnikih brez lastnikove vednosti. Rezultate analize poroča nazaj napadalcu. Takim okuženim računalnikom rečemo botnet-i. Ko napadalec s pomočjo botnetovih preiskovalnih analiz zazna ranljivost na katerem od sistemu v omrežju, lahko izvede napad.

Cilj avtorjev članka "Towards Situational Awareness of Large-Scale Botnet Probing Events"[12] je bil, da razvijejo metodologijo, s pomočjo katere, bi lahko spletna mesta analizirana iz strani napadalcev z metodo lokalnega preiskovanja spletnega prometa to tudi zaznala.

3.2 Zaznavanje sumljivih aktivnosti s pomočjo IP Gray Space analize

V različnih omrežjih se običajno pojavljajo neopredeljeni IP naslovi, ki v spektru vseh naslovov tvorijo tako imenovani IP Gray space prostor. Paketi, z omenjenimi IP naslovi, pogosto nosijo zlonamerno programsko opremo ali kodo.

Avtorji članka "Identifying and Tracking Suspicious Activities Through IP Gray Space Analysis"[8] so s primerjavo vzorcev prometa proti IP Gray Space naslovi in vzorci prometa v živem omrežju pokazali, da obstaja korelacija med prometom v IP Gray Space prostoru in zlonamernim prometom v omrežju. Demonstrirali so še detekcijo SPAM aktivnosti, zlonamernega skeniranja omrežja in zlonamerne kode, z uporabo IP gray space analize.

3.3 Relacija med zlonamerno programsko opremo in preiskovalnimi napadi

Zaradi hitrega porasta prometa zlonamernih programov v različnih omrežjih je vse več potrebe po tehnikah in načinih odkrivanja omenjenih napadov. Še posebej pomembna je zaščita pri zero-day napadih. To so napadi, ki izkoristijo ranljivosti sistema, katere vzdrževalci še niso odkrili ali pa je še niso imeli uspeli popraviti.

V članku "Practical Correlation Analysis between Scan and Malware Profiles against Zero-Day Attacks Based on Darknet Monitoring"[9] so avtorji s pomočjo analize darknet aktivnosti zmodelirali Scan profile. Iz analize obnašanja zlonamernega programja v kontroliranem okolju so zmodelilari profile generalnih karakteristik zlonamernega programja. Kot rezultat so postavili in diskusirali korelacijo med omenjenimi profili in predstavili, da je mogoče odkriti zero-day napade z scan-malware verigo analize.

Pri drugem sorodnem članku "Correlation analysis between spamming botnets and malware infected hosts." so postavili

korelacijo med spam botnet-i in napravami okuženimi z zlonamerno kodo. Zaključili so, da je večina botnet-ov bilo okuženih vsaj s štirimi različnimi zlonamernimi programi. Razvili so tudi metode, ki omogočajo določitev tipa zlonamernega programa, ki je okužil napravo.

V članku "A proposal of malware distinction method based on scan patterns using spectrum analysis."[6] avtor predlaga metodo za odpravljanje zlonamerne kode, ki temelji na opazovanju določenih vzorcev v analizi spektruma.

3.4 Detekcija preiskovalnih aktivnosti s pomočjo DNS-ja

Avtorji članka "DNS-based Detection of Scanning Worms in an Enterprise Network"[15] so razvili novo tehniko in način kako hitreje zaznati vse pogostejše napade črvov v določenem omrežju. Tehnika deluje na osnovi korelacije med DNS zahtevki in odhodnega prometa. Prednosti omenjene tehnike od prej že obstoječih tehnik detekcije črvov so:

- Detekcija črva že po prvem poskusu infekcije
- Možnost detekcije zero-day napadov črvov
- Majhen delež lažnih preplahov

V članku "Dynamics of Online Scam Hosting Infrastructure"[11] je avtor opisal, kako zaznati in razumeti dinamiko spam kampanje s pomočjo korelacije med znanimi spam URL-ji in DNS prometom na omrežju.

Avtorji članka "Exposure finding malicious domains using passive DNS analysis" se sklicuje na to da velika večina zlonamernega prometa temelji na DNS-ju, saj ga uporabljajo za kontrolo in okužbo različnih naprav v omrežju. Zato danes večina orodij za odkrivanja zlonamernih domen temelji prav na analizi DNS prometa in same vsebine DNS zahtevkov. Avtorji omenjenega članka so prav tako za odkrivanje zlonamernih domen tudi sami razvili sistem, ki se imenuje EXPOSURE. EXPOSURE omogoča odkrivanje zlonamernih domen v realnem času. Razvili so ga iz različnih njihovih tehnik, ki so pri testiranju požele zelo dobre uspehe.

4. PRIMERI NAPADOV IZ PRAKSE

Do sedaj smo zgolj omenili, da so napadi povezani z vrati 0 možni in da predstavljajo resno grožnjo za sisteme tudi v praksi. V nadaljevanju si bomo zato ogledali nekatere primere iz prakse, ki potrjujejo nevarnost in njihove posledice, ki so nastale zaradi nepoznavanja oz. zanemarjanja nevarnosti vrat 0. Kako kritično je poznati posledice tovrstnih napadov, vidimo že iz grafikona na Sliki 5. Tu namreč vidimo, da več kakor polovico tarč napadov predstavljajo industrija, državni organi in organizacije. Žrtve take narave bodisi hranijo občutljive podatke bodisi so občutljive na morebitne izpade lastnih sistemov. Tako lahko kraja podatkov oz. sesutje sistema prinese katastrofalne posledice.

4.1 Cisco poročilo 2013

Spletno poročilo podjetja Cisco je predstavljalo vstopno točko za izvorni članek[4]. Poročilo je zanimivo, ker zaznava velik porast v uporabi vrat 0 pri TCP zahtevkih. V samo

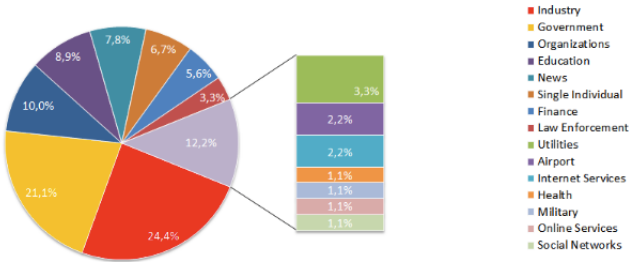


Figure 5: Razporeditev tarč napadov leta 2013 po sektorjih. Grafikon je rezultat raziskave v izvornem članku[5].

treh urah so zaznali največjo količino takih zahtev v istem letu, pri čemer moramo upoštevati, da je bil dogodek zaznan konec leta. Upravičeno so postavili sum, da gre za začetno fazo kibernapada, kjer napadalec preiskuje opazovani sistem za morebitne ranljivosti. Dodatno težo je sum pridobil, ko so ugotovili, da so IP naslovi virov zahtev omejeni na osem naslovov z nizozemske in da je velik delež zahtev usmerjen proti vratom 445. Podana vrata predstavljajo dostop do storitve SMB/DCERPC pri Windows sistemih, ki velja za eno izmed pogosteje izkoriščenih ranljivosti. Poročilo tudi opozarja na pogosto situacijo, ko se izvidnica za napad lahko zgodi lahko tudi nekaj mesecev pred samim napadom in je zato dobra praksa ob zaznanih TCP zahtevkih iz vrat 0, da poleg blokiranja takih zahtev tudi preučimo sistem za morebitne ranljivosti, ki bi jih napadalec s povpraševanjem zaznal[17]. O aktualnosti problema porasta TCP zahtev z izvornimi vrati 0 pričata tudi grafa Slike 6, kjer vidimo, da se je trend uporabe vrat 0 nadaljeval še dolgo po podanem poročilu.

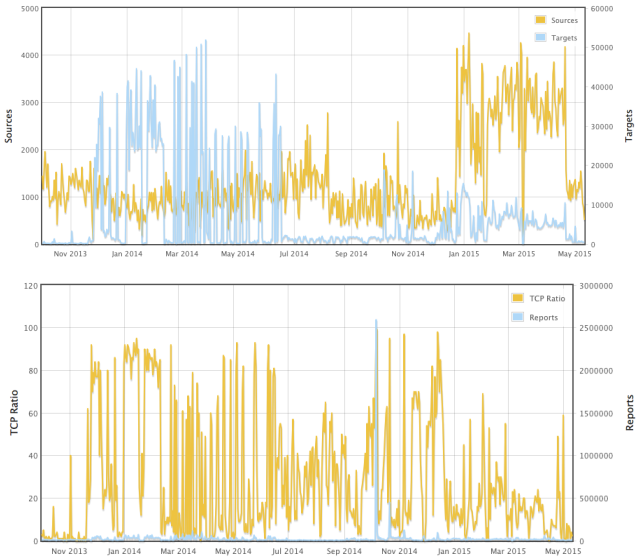


Figure 6: Zgornji graf prikazuje aktivnost TCP in UDP paketov z vrati 0 bodisi kot izvor bodisi kot ponor od trenutka zaznane povečave[17] do danes. Spodnji graf prikazuje delež TCP paketov z vrati 0 in število poročil, ki omenjajo povezane dogodke v istem obdobju. Grafa sta generiana s podatki od DShield/Internet Storm Center[2].

4.2 Primer New York Times 2002

Napad na strežnike New York Times leta 2002 se je zgodil slučajno, ko je zdolgočaseni heker z namenom kratkočasenja zagnal dvominutna poizvedovanja po straneh in pri New York Times naletel na sedem nepravilno nastavljenih strežnikov, ki so mu predstavili vstopno točko v notranje omrežje časopisne hiše. Preko strežnikov je uspel dostopiti do notranjega strežnika, ki je med drugim tudi hranil kopijo podatkovne baze. V tej so se nahajali tudi uporabniški računi zaposlenih, med drugim tudi računi z administratorskimi pravicami, ki so imeli še vedno veljavno geslo. Z uporabo takega računa si je heker generiral uporabnika z vsemi pravicami in si pridobil dostop do celotne podatkovne baze. Čeprav ni pridobil dostopa do urejanja spletne strani, kar je bilo zaradi preteklih izkušenj ločeno, je pridobil s podanimi koraki druge kočljive podatke. Med drugim si je pridobil osebne podatke okoli 3000 strokovnjakov, ki so sodelovali pri pisanju člankov. Tako je pridobil informacije o telefonskih številkah, identifikacijskih številkah osebnih dokumentov, njihovo zgodovino, strokovno področje, zaslužek in, najbolj presenetljivo, zaznamek o podkupljivosti s strani urednikov. Zaradi kvalitete časopisne hiše so bili to podatki elite in politično močnih oseb, kar izgubo podatkov še toliko bolj obremeni. Čeprav je heker po napadu opozoril časopisno hišo, je to lep primer, kako navidez nedolžno poizvedovanje lahko privede do izgube velike količine zasebnih podatkov[14].

4.3 Primer Wordpress 2013

Aprila 2013 je US-CERT [United States Computer Emergency Readiness Team] množično pozvala uporabnike na posodobitev Wordpress ogrodja za spletne strani in zamenjavo uporabniških imen in gesel. Zgodil se je namreč množičen napad po spletu na strani, ki tečejo na Wordpress ogrodju. V večtedenskem napadu so napadalci poizvedovali o prisotnosti Wordpress ogrodja, nakar so na najdenih straneh poskusili z mavričnimi tabelami nekaj tisoč pogostih kombinacij pridobiti dostop do strani. Napad ni predstavljal zgolj nevarnosti ukradenih podatkov in nelegalnih dostopov do strani. Pojavila se je nevarnost zbiranja strežnikov v nove botnete za prihodnje napade, saj so napadalčevi botneti predstavljali predvsem šibke domače osebne računalnike. Podobne taktike so se že uporabile za pripravo večjih napadov, izmed katerih je eden uspel vdreti v Finančne organe ZDA jeseni 2012. Sama moč zbiranja strežnikov z ogrodji Wordpress, se je pokazala s tem, da je npr. ponudnik oblačnih storitev CloudFare prejemal vsako uro po 60 milijonov zahtev svojih Wordpress uporabnikov[18]. Primer kaže moč izrabe preiskovanja vrat pri sistemih, ki uporabljajo popularno ogrodje, ki pa ne upošteva vseh ranljivosti in tako odpira vrata napadom na uporabnike ogrodja in zlorabo njihove strojne opreme za nadaljne napade poleg sprotne škode.

4.4 Primer Playstation Network 2011

Med 17. in 19. aprilom 2011 se je zgodila tedaj največja kraja osebnih podatkov v zgodovini, ko so hekerji skupine Anonymous preko ranljivostih pridobljenih s poizvedovanjem strežnikov pridobili dostop do baze osebnih podatkov 77 milijonov uporabnikov omrežja Playstation Network podjetja Sony. Napad in izguba podatkov, med drugim tudi kreditnih kartic, je povzročila 23 dni nedolovanja omrežja. Zaradi izgube kočljivih podatkov, slabe odzivnosti do javnosti in kršenja nekaterih zakonov o varstvu osebnih podatkov je napad povzročil podjetju več milijard dolarjev odškodnin. Poleg odškodnin so bili prisotne tudi druge finančne škode, med drugim naj bi 23-dnevno nedelovanje omrežja povzročilo izgubo 171 milijonov dolarjev[16]. Primer lepo izriše posledice napadov na sisteme podjetji, ki preko napadenih sistemov služijo denar. Izriše namreč oboje, situacijo kraje podatkov in nedosegljivost omrežja.

4.5 Napad Irana 2013

Kibernapadi so nevarni tudi na državnem nivoju, o čemer priča iranski napad na ameriško energetsko podjetje leta 2013. Napad se je razvil do te stopnje, da so napadalci poleg pregleda podatkov pridobili tudi dostop do nekaterih funkcionalnosti sistema in so lahko celo uravnavali pretok plinovodov in naftovodov, kar je omogočalo sabotažo energijske preskrbe na državnem nivoju, kar je naredilo napad mnogo nevarnejši kot vse dosedanje. Pravzaprav gre v danem primeru za kibernetsko vojno med državama, saj so Američani pred tem že sodelovali pri zloveščih napadih z Stuxnet črvom, ki je sabotiral iranske jedrske elektrarne[7].

4.6 Nevarnost ponornih vrat 0

Omenili smo že, da ranljivost vrat 0 izvira iz rezervirane specifikacije, ki pa v praksi ne preprečuje uporabe teh vrat. Mnogi sistemi imajo tako pomanlkjivo urejene požarne zidove, kar omogoča napade. Do sedaj smo pozornost usmerili na izvorna vrata 0 kot vir podatkov. Vendar se nevarnost tu ne konča. Potrebno je tudi opozoriti na napade, ki izkoriščajo poonorna vrata 0, kot npr. DDoS napadi. V pripsevku na Endace on Network Visibility Blog[10], avtor še pred Cisco dogodkom[17] opozori na porast DDoS napadov in uporabe ponornih vrat 0. V trenutku pisanja so zaznali porast DDoS napadov za 43%, pri čemer se je uporaba ponornih vrat 0 povečala na 25% delež vseh DDoS napadov. Dodatni problem ponornih vrat 0 je tudi, da se zapis uporablja znotraj legalnega prometa pri paketih brez TCP in UDP glave in pri fragmentiranih paketih. Najbolj poznan tak promet je gotovo ICMP protokol. Ta lastnost drastično poveča moč DDoS napada, ki želi izčrpati pasovno širino povezave. Sam potencial takih napadov izraža že cenuzus interneta iz leta 2012[1], ki poroča o več kot 400 milijonih ranljivih sistemov na preučevanih 4.3 milijonih preiskovanih omrežjih /24. Njihovo razporeditev prikazuje zemljevid na Sliki 7. Tudi tu se lahko o aktualnosti prepričamo iz prvega grafa na Sliki 6, kjer vidimo tudi visoke količine ponornih vrat 0.

5. EKSPERIMENTALNO DELO

Literatura in splet na mnogo mestih poročata o profiliranju sistema s pomočjo paketov TCP, ki imajo izvorna vrata nastavljena na vrednost 0. Vendar pa nismo zasledili nobene literature o tem, kako konkretno se profiliranje izvede. Zanimalo nas je, v katerih poljih paketa TCP, ki ga vrne sistem kot odgovor na drežljaj z nastavljenimi izvornimi vrati 0, natanko se pojavi razlika. Zasnovali in izvedli smo eksperiment, kot je opisano v nadaljevanju.

5.1 Opis eksperimenta

Zamislili smo si, da bomo postavili nekaj različnih spletnih strežnikov in nato na vsakega izmed njih poslali zahtevek TCP z nastavljenimi izvornimi vrati na vrednost 0 ter opazovali odgovor s pomočjo orodja Wireshark. Cilj eksperi-

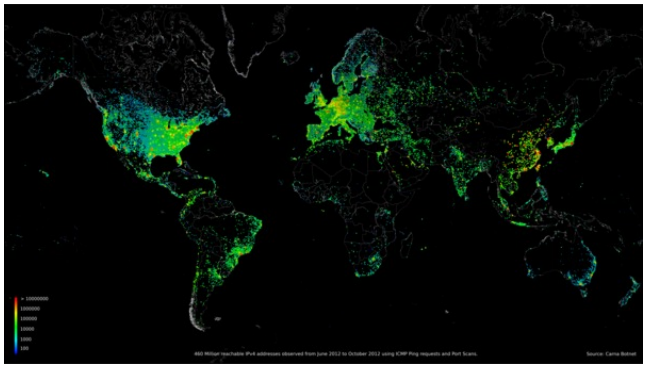


Figure 7: Zemljevid prikazuje odzivnost na ICMP pakete in s tem ranljivost sistemov na DDos napade po svetu leta 2012[1].

menta je bil ugotoviti, v čem se odgovori razlikujejo med seboj; torej na podlagi česa bi nepridipravi potencialno lahko profilirali strežnik.

5.2 Izvedba eksperimenta

Na fizični računalnik z operacijskim sistemom Windows 8.1 Pro smo namestili tri najpopularnješe spletne strežnike:

1. *Apache HTTP Server Version 2.4* - konfigurirali smo ga tako, da je poslušal na vratih 8880.
2. *Nginx 1.9.0* - konfigurirali smo ga tako, da je poslušal na vratih 80.
3. *Glassfish 4.1* - konfigurirali smo ga tako, da je poslušal na vratih 4848.

Nameravali smo napisati skripto v programskem jeziku python, ki bi strežniku poslala paket TCP z nastavljenimi izvornimi vrati 0. To se je izkazalo za izredno zahtevno opravilo, saj python nima dobre podpore za prirejanje vrednosti paketom TCP. Odlično je podprto delo z visokonivojskimi zahtevki HTTP na eni strani, nizkonivojskimi vtiči (ang. socket) na drugi strani, vse kar je vmes, pa je potrebno implementirati od začetka. Preučili smo knjižnice requests, urllib, socket, construct in se nazadnje odločili za alternativno rešitev, orodje **scapy**. Scapy ni podprt na operacijskem sistemu Windows, zato smo kreirali Virtualbox navidezni stroj z operacijskim sistemom Linux Ubutnu 14.04 in ga namestili tam. Navideznemu stroju smo dovolili bridged dostop do spleta namesto NAT, saj virtualizirani NAT dostop že sam po sebi ne dovoli pretakanja paketov z nastavljeno vrednostjo izvornih (niti ponornih vrat na 0.

Na Windows računalniku smo poiskali naslov IP (10.95.48.146) na katerega smo nato iz Ubuntu virtualnega računalnika poslali omenjene tri zahtevke TCP, na vsak strežnik po enega. Apache strežniku smo poslali na ciljna vrata 8880, Nginxu na vrata 80 in Glassfishu na vrata 4848:

```
sr(IP(dst="10.95.48.146")/TCP(sport=0,
dport=8880, flags="S")/"JAZ SEM ZLOBNI
BACEK JON.", timeout=3)
```

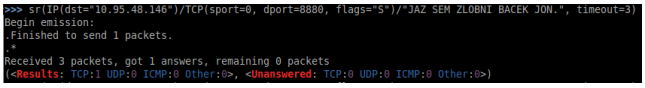


Figure 8: Uporabimo orodje scapy za pošiljanje modificiranega zahtevka TCP na server. V tem primeru pošiljamo na strežnik, ki posluša na vratih 8880, torej Apache.

Celotno korespondenco TCP smo zajeli z orodjem Wireshark (Slika 9) in jo z vsemi podrobnostmi izvozili v tekstovno datoteko, za vsak strežnik posebej. Dobili smo tri datoteke, ki smo jih s programom za spremljanje sprememb Meld primerjali med seboj (Slika 10).

Pregledali smo spremembe na posameznem nivoju omrežne plasti in ugotovili sledeče:

5.2.1 plast protokola TCP

Razlika je bila, seveda, v izvornih vratih, saj je vsak strežnik odgovoril iz svojih. Naslednja razlika se je pojavila v identifikatorju toka TCP (ang. TCP stream number), ki je prav tako pričakovana, saj ima vsaka seja TCP paketov svojo enolično identifikacijo. Tretja, zadnja razlika pa se je pojavila v vrednosti kontrolne vsote, kar zopet ni nič nepričakovanega.

5.2.2 plast protokola IP

Podobno kot na plasti TCP smo zaznali spremembo v identifikatorju in kontrolni vsoti. Drugih razlik ni bilo.

5.2.3 plast protokola Ethernet

Pri vseh treh odgovorih je bila vsebina popolnoma identična.

5.2.4 plast protokola Frame

Opazili smo razlike v zapisanih časih (frame.time, frame.time_epoch, frame.time_delta, frame.time_relative), kar nas ni presentilo. Ostala polja so bila identična.

Skupno za vse plasti ugotovimo, da nismo zaznali **nobenih pomembnih razlik** v odgovoru, na podlagi katerih bi bilo možno ugotoviti za kateri strežnik gre.

5.3 Interpretacija rezultatov in predlogi za nadaljnje raziskave

Eksperiment je pokazal, da trije popolnoma različni spletni strežniki vrnejo nediskriminativen odgovor na drežljaj s podtaknjeno vrednosto izvornih vrat paketa TCP na 0. Sodeč po rezultatih tega eksperimenta, nam pošiljanje tovrstnih paketov torej ne pomaga pri profiliranju sistema.

Zasnovali smo eksperiment, s katerim bi ugotovili, če bi se razlika morebiti pojavila v odvisnosti od operacijskega sistema (in ne strežnika). Postaviti bi morali tri navidezne stroje in na vsakega namestiti drug operacijski sistem (npr. Linux Ubintu na prvega, Microsoft Windows 8.1 na drugega in OS X 10 na tretjega). Nato bi na vseh namestili enako verzijo strežnika Apache, ki bi poslušal na vratih 0. Če bi pri takšnem eksperimentu odkrili, da dobimo diskriminativno različne odgovore na drežljaj, bi to pomenilo, da razliko povzroči operacijski sistem in ne strežnik, kot smo predpostavili v našem eksperimentu.

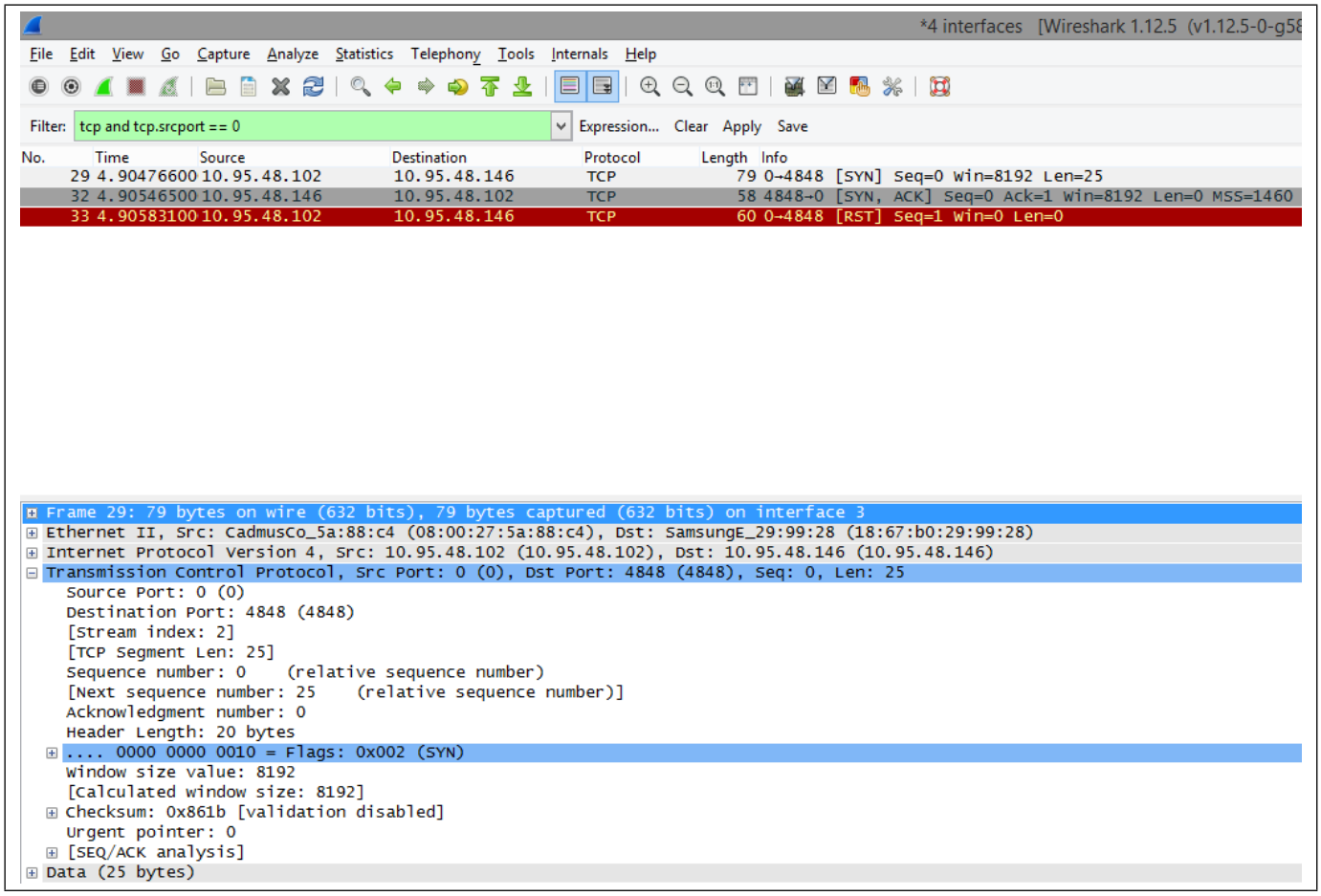


Figure 9: Zajamemo ves omrežni promet s programom Wireshark in ga filtriramo glede na izvorna vrata paketov TCP. Ko najdemo ustrezni začetni zahtevek, sledimo toku paketov (opcija Follow TCP Stream programa Wireshark) in dobimo tudi odgovor, ki ga je vrnil strežnik.

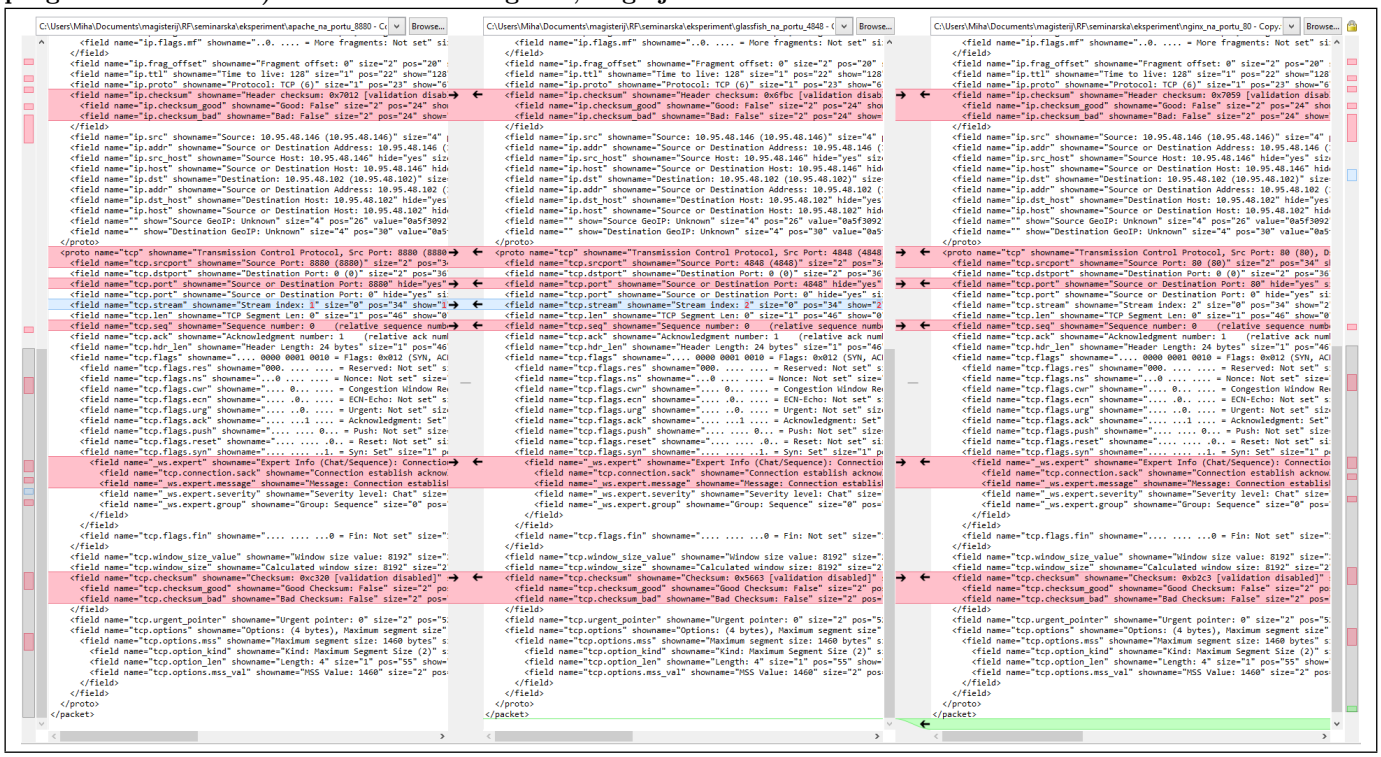


Figure 10: Meld je orodje za primerjavo datotek. Podpira primerjavo do treh datotek naenkrat, zato lahko primerjamo odgovore z vseh treh strežnikov hkrati.

Obstaja možnost, da virtualni stroji ne omogočajo dovolj dobre simulacije omrežnega dogajanja v takšnih prepovedanih razmerah (izvorna vrata 0 so prepovedana po določilih RFC). O tem bi se prepričali tako, da bi za eksperiment uporabili tri fizične računalnike.

Vprašanje, kako natanko lahko s paketi TCP, ki imajo nastavljeno vrednost izvornih vrat na 0, profiliramo sistem, po našem eksperimentu torej ostaja neodgovorjeno.

6. ZAKLJUČEK

Čeprav preiskovanje omrežja s pomočjo paketov TCP, ki imajo izvorna vrata nastavljena na 0, samo po sebi ni škodljivo, je smiselno biti seznjanjen s tem, saj pogosto predstavlja zgodnjo fazo kiber napada. Na konkretnem primeru smo pokazali, da obstajajo postopki, ki znajo detektirati takšna preiskovanja na podlagi zajema prometa v darknetu in v zaledju strežnikov DNS. Pravzaprav jih znajo celo dobro opisati (ali pa kar poimenovati točen DLL!), kajti dandanes obstajajo redno posodobljene zbirke z opisi najpogostejših tovrstnih zlonamernih programov.

Pregled sorodne literature je pokazal, da porast kiber napadov spremlja tudi porast v razvoju pristopov preučevanje napadov in pripravljenje sistemov na njihovo obrambo ter zbirk o znanih napadih. Z njihovo uporabo se lahko prepreči marsikateri napad, le na uporabniku je, da to tudi stori.

V opomin, da nepridipravi v resničnem življenju dejansko posegajo po metodi skeniranja omrežja s pomočjo paketov TCP, ki imajo izvorna vrata nastavljena na vrednost 0, nam služijo primeri, ki smo jih navedli v zadnjem delu članka. Kot dodatna spodbuda za pravilno konfiguriran sistem v omrežju naj bodo tudi opisane posledice pri primerih napadov, ki lahko služijo kot budnica tako običajnim uporabnikov omrežja kakor tudi državnim organom, organizacijam in podjetjem.

7. REFERENCES

[1] Internet census 2012 - port scanning /0 using insecure embedded devices. `http://internetcensus2012.bitbucket.org/paper.html`, 2012. [Spletni vir; dostopano 11.5.2015].

[2] Dshield/internet storm center. `http://www.dshield.org/`, 2013-2015. [Spletni vir; dostopano 11.5.2015].

[3] R. Alshammari and A. N. Zincir-Heywood. Can encrypted traffic be identified without port numbers, {IP} addresses and payload inspection? *Computer Networks*, 55(6):1326 – 1350, 2011.

[4] E. Bou-Harb, N.-E. Lakhdari, H. Binsalleeh, and M. Debbabi. Multidimensional investigation of source port 0 probing. *DFRWS[best student paper]*, 2014.

[5] E. Bou-Harb, N.-E. Lakhdari, H. Binsalleeh, and M. Debbabi. Multidimensional investigation of source port 0 probing. *DFRWS[presentation]*, 2014.

[6] M. Eto, K. Sonoda, D. Inoue, K. Yoshioka, and K. Nakao. A proposal of malware distinction method based on scan patterns using spectrum analysis. In C.-S. Leung, M. Lee, and J. H. Chan, editors, *ICONIP (2)*, volume 5864 of *Lecture Notes in Computer Science*, pages 565–572. Springer, 2009.

[7] S. Gorman and D. Yadron. Iran hacks energy firms. `http://www.wsj.com/articles/SB10001424127887323336104578501601108021968`, 2013. [Spletni vir; dostopano 11.5.2015].

[8] Y. Jin and F. Cao. Identifying and tracking suspicious activities through ip gray space analysis," umn. Technical report, 2006.

[9] Y. Jin, Z.-L. Zhang, K. Xu, F. Cao, and S. Sahu. Identifying and tracking suspicious activities through ip gray space analysis. In *Proceedings of the 3rd Annual ACM Workshop on Mining Network Data*, MineNet '07, pages 7–12, New York, NY, USA, 2007. ACM.

[10] T. Jones. Ddos attacks on port 0 – does it mean what you think it does? `http://blogs.emulex.com/visibility/2013/08/27/ddos-attacks-on-port-0-does-it-mean-what-you-think-it-` 2013. [Spletni vir; dostopano 11.5.2015].

[11] M. Konte, N. Feamster, and J. Jung. Dynamics of online scam hosting infrastructure. In *Proceedings of the 10th International Conference on Passive and Active Network Measurement*, PAM '09, pages 219–228, Berlin, Heidelberg, 2009. Springer-Verlag.

[12] Z. Li, A. Goyal, Y. Chen, and V. Paxson. Towards situational awareness of large-scale botnet probing events. *Trans. Info. For. Sec.*, 6(1):175–188, Mar. 2011.

[13] A. Moise and J. Brodkin. Ansi c12. 22, ieee 1703, and mc12. 22 transport over ip. 2011.

[14] K. Poulsen. New york times internal network hacked. `http://www.securityfocus.com/news/340`, 2002. [Spletni vir; dostopano 11.5.2015].

[15] D. Whyte, E. Kranakis, and P. C. van Oorschot. Dns-based detection of scanning worms in an enterprise network. In *IN PROC. OF THE 12TH ANNUAL NETWORK AND DISTRIBUTED SYSTEM SECURITY SYMPOSIUM*, pages 181–195, 2005.

[16] Wikipedia. 2011 playstation network outage. `http://en.wikipedia.org/wiki/2011_PlayStation_Network_outage`. [Spletni vir; dostopano 11.5.2015].

[17] C. Williams. Massive increase in reconnaissance activity – precursor to attack? `http://blogs.cisco.com/security/massive-increase-in-reconnaissance-activity-precursor-t` 2013. [Spletni vir; dostopano 11.5.2015].

[18] Z. Zorz. Wordpress sites targeted by mass brute-force attack. `http://www.net-security.org/secworld.php?id=14752`, 2013. [Spletni vir; dostopano 11.5.2015].

Detecting NAT gateways and differenting host devices behind NAT for purposes of digital forensic investigations

An overview of existing techniques

Manca Bizjak
Fakulteta za računalništvo in informatiko
1000 Ljubljana
Slovenia
manca.bizjak@guest.arnes.si

Matic Tribušon
Fakulteta za računalništvo in informatiko
1000 Ljubljana
Slovenia
matic.tribuson@gmail.com

ABSTRACT

One of the of the most common misconceptions emerging from the popular culture is the assumption that IP addresses can uniquely identify the source from which network traffic originates. While it is not impossible, the field of network forensics must adopt more general techniques that allow reliable identification of end-machines, for instance, if they are suspected of involvement in a criminal act. An example of scenario where identification solely with the use of IP addresses could lead to erroneous conclusions, are networks with NAT routers, widely used in residential, home and enterprise networks. We present a structured overview of approaches for identification of NAT devices and differentiation of host machines behind NAT and discuss their performance, reliability and generality.

Keywords

NAT, traffic flows, IP address identification, network forensics, digital evidence

1. INTRODUCTION

Digital forensics is becoming increasingly important as a consequence of extensive use of electronic devices. Many crimes indirectly involve electronic devices in one way or another, which resulted in fast development of digital forensics.

In the past few decades electronic devices have started to connect to one another as well as to the Internet, which is why it is not surprising that the number of crimes involving internet or smaller intra company networks has increased. This indicates that digital forensic investigators must frequently handle not one but several interconnected computers in order to obtain a complete picture. Similar to other crimes, it is expected that the law is enforced and that the criminals take responsibility for their actions. However, in the world of cyber criminal it may prove hard to find the perpetrator. Therefore, the field of network digital forensics has evolved to better understand and examine the digital evidence related to crime involving computer networks.

Cyber criminals often use advanced techniques to disguise themselves and the machines they are using on the Internet. We know that machines on the internet are identified by their IP addresses. At present day, the majority of users still use IPv4, which has a serious shortage of IP address namespace, which means that multiple devices have to use the same global IP address to be able to access the Internet. The latter is often achieved with the use of Network Address Translation (NAT). Devices that are capable of doing NAT are very popular nowadays and almost every household with Internet access has one. Because of NAT devices other user devices cannot be uniquely identified by their IP address which could pose a challenge when searching for cyber criminals. Prerequisite for relating a person to a cyber crime involving interconnected devices is the ability to uniquely identify each device connected to the Internet. As described earlier, IP address cannot ensure this anymore. If a malicious activity is caused by a device behind NAT, we cannot identify this specific device based on the IP address as its IP address is the same as the IP address of the NAT device as well as IP addresses of other devices behind the same NAT device. This causes trouble for digital forensic investigators since they can not seperate different devices on the Internet. When dealing with NATed networks, we have to resort to different approaches that are described, discussed and compared in this paper.

The remainder of this text is organized as follows: Section 2 provides a short overview of NAT and its problematics. Section 3 outlines the approaches for either detecting the presence of NAT gateways, counting or identifying host machines behind NAT. Section 4 provides a comparison of these approaches and highlights their strengths and potential drawbacks, ending with a short discussion and concluding remarks.

2. NAT

Network address translation (NAT) is a technique which allows several host machines to share a common public IP address. The fundamental concept of NAT is translating a private IP address into a global one, allowing several internal host machines to appear as a single device to the outside world [7]. A device responsible for translating IP addresses of all outcoming as well as incoming packets is called a NAT

router.

Most firewalls can perform NAT, enabling network administrators to connect multiple hosts to the Internet via one public IP address. Furthermore, many Internet service providers (ISPs) usually offer wireless NAT gateways to their customers (which we will call dedicated NAT gateways and are usually simple home routers that can also perform address translation), allowing them to easily connect several host devices to the Internet, while only needing one public IP address. A regular laptop with active Internet connection sharing can also act as a NAT gateway [5]. As we can see, NAT comes in variety of forms. An example of a simple network using NAT router is shown on Figure 1.

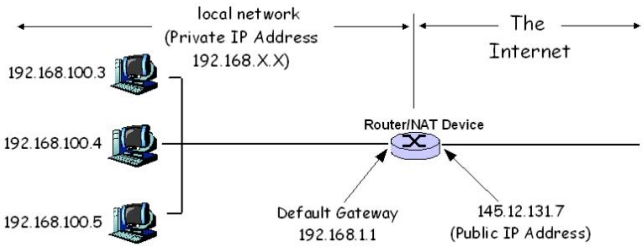


Figure 1: Simple NAT network infrastructure

Since it's first implementation in 1990s, NAT has become essential in alleviating the IPv4 address exhaustion [7]. It provided a short-term solution to overcoming the IPv4 problematic by allowing for more economical use of IPv4 addresses. The utilization of NAT was thus a more convenient alternative to address this problem with respect to much more expensive option of replacing/upgrading the existing IPv4 hardware. Moreover, NAT is able to provide some level of security, since it protects internal hosts from being directly addressed from the outside world.
However, looking from a different perspective, the very same feature introduces a new potential problem - what if a host machine behind NAT is the source of malicious internet activity? In this case NAT router provides a way for concealment of the malicious host. While IP addresses can be used as unique identifiers from the inside of private network, the address translation results in all outgoing packets' source IP being set to the IP of NAT router. This is an important observation, since at a first glance the IP addresses seem like a good way of providing a unique, global identification of an end-machine. Nowadays this reasoning seems outdated and the very use of NAT devices proves that identifying end machines with certainty only through the means of IP addresses is not only insufficient, but could lead to erroneous conclusions and should be therefore used with caution. For instance, when the backbone network is performing access control based on the IP addresses, it cannot identify the host machines behind NAT since their original IP addresses are replaced by the IP address of a NAT device. Thus, identification of NAT devices and differentiating machines behind them becomes a problem that needs to be overcome not only for the sole purpose of providing normal functioning of several network services, but from a digital forensics' standpoint as well. To justify this concern, we emphasize that several data point towards the fact that a high degree ($\approx 90\%$) of lines connecting both residential as well as business networks to the Internet, utilize NAT gateways. Therefore, we emphasize the dilemma whether a public IP address can be used as a unique public identifier at all, since it constantly introduces ambiguity of IP addresses.

As many authors have previously demonstrated, the significant increase in the usage of NAT devices over the past decade has led to several new ways of spreading different families of malware on the Internet. As NAT devices have become a rather convenient way to hide the source of malicious behaviour, there has been much effort to develop techniques for detecting the presence of NAT devices as well as differentiating (if not uniquely identifying) the host machines behind them.
However, many NAT devices mantain very limited, if any, information about the traffic that passes through them in the form of logs [2]. The latter introduces another issue, since identifying which computer from such network was potentially involved in a criminal activity, proves even more difficult. Furthermore, even if we succeed, we may stumble upon an issue of associating a certain individual with the computer involved in a criminal act, in cases where NAT devices are used to connect several hosts, physically located in public, poorly monitored places (such as libraries or cafes) to the Internet.

There are several existing approaches aimed at identifying NAT devices, as well as host machines behind them, varying drastically in many aspects, including the type of data used to make predictions about NAT devices and NATed hosts, accuracy and generality. Specifically, we review the early approach that utilizes IP packet information, passive fingerprinting approach, proposed in Maier et. al. [5], machine learning approach proposed in Gocken et. al. [3], proxy authentication proposed in Ishikawa et. al. [4] and some of their derivatives. We point out the main differences between these approaches, potential issues, performance and compare them against one another. We decided to only review the identification methods that can be applied to the existing hardware infrastructure, since solutions based on replacement or upgrading of NAT routers would introduce additional costs, proportional to the number of replaced NAT devices.

Before we start with the description of methods, let us classify the approaches according to characteristics of the results their analyses are able to provide as follows:
a) *quantitative* - providing concrete, factual data, for instance in the form of absolute numbers or percentage (e.g. percentage of analyzed networks utilizing NAT),
b) *qualitative* - providing insight into categories of host machines behind NAT gateways, concerning several possible setups, e.g. virtual machines, game consoles, smart phones.

Additionally, we divide these approaches into two semantically seperate groups: 1) methods that only use network layer information in their analyses (traffic flows) and 2) methods that rely on arbitrary application level protocol data in addition to network layer data. We will use both notations in order to classify the approaches in the remainder of this text.

3. DETECTING THE PRESENCE OF NAT

3.1 Network data based

Early approaches used for NAT gateway identification, dating back to the early 2000s, rely on the information extracted from the IP packet headers, specifically, the IP Identification (ID) and Time To Live (TTL).
Identification of NAT devices and NATed hosts based on the IP TTL field arises from the following observations: a) There is a general rule that every network router (including NAT devices) decrements the value of IP TTL field by one, as an observed packet passes through them, b) operating systems use well-defined initial TTL values in outgoing packets. For instance, Windows operating systems use the initial value 128, while Mac and Linux operating systems use initial TTL value 64. Therefore, by knowing the location (in terms of hop distance) of the monitoring point, we can divide end machines into two categories: 1.) devices directly connected to the Internet (TTL in the time of observation $= TTL_{initial} - 1$ if the hop distance between monitoring point and host device is 1),
2.) routers (dedicated NAT gateways) or hosts performing network address translation, with TTL in the time of observation $= TTL_{initial} - 2$. However, note that the potential shortcoming of this setup is reliance on the knowledge of the location of the monitoring point.

We can apply the extracted TTL values to counting different host devices behind NAT. The aforementioned difference in ranges of TTL values, varying from one OS to another, allows us to rougly distinguish between different OSes. However, this approach is unable to differentiate between hosts within the same OS family. We emphasize that according to the recent statistics, available in [1], Windows OSes are by far the most commonly used OS family, which is why TTL alone cannot help us to count NATed hosts accurately. Additionally, we also cannot differentiate between a Linux and Mac operating system since they both use 64 as the initial TTL value.

3.2 The Machine learning approach

The machine learning approach, which was extensively reviewed and discussed in [3], tries to identify NAT devices and hosts behind NAT devices only by analyzing traffic flows. It does not require any knowledge about the analysed network's topology or any application layer data. This makes it much more flexible as users do not have to know anything specific about the network they are analysing. From a forensic standpoint, this is a very important feature as investigators do not necessarily know any details about the networks they are about to investigate. Machine learning approach is also not limited only to non SSL/TLS traffic as it does not need any application data (that SSL or TLS encrypt).
In general the input of the machine learning approach are traffic flows. A program was developed that was used to extract useful data from all given traffic flows. Various traffic flow attributes are later used in the decision process. Of course, not all attributes are equal when making a decision whether a device is behind NAT or not. Therefore, it is very important to identify the most important attributes and to assign them appropriate weights.

The choice of machine learning technique is crucial for the success of this approach. Let us first summarize the conceptual background of the two approaches that the authors implemented and compared - Naive Bayes and C4.5. Both techniques classify a certain entity based on its attributes or features.

Naive Bayes is a statistical classifier based on Bayes' theorem that provides probability of a given class. It assumes the independence of the presence or absence of a specific attribute or feature. The equation 1 describes the probability model of Naive Bayes.

$$P(C|F_1, F_2, ..., F_n) = \frac{1}{Z} P(C) \prod_{i=1}^{5} P(F_i|C) \quad (1)$$

Where $P(C|F_1, F_2, ..., F_n)$ is the probabilistic model over the dependent class variable C with a small number of outcomes or classes, conditional on several feature/attribue variables $F_1, F_2, ...F_n$ and Z is a scaling factor dependent only on $F_1, F_2, ..., F_n$.

Unlike Naive Bayes, C4.5 is an algorithm that generates a decision tree using information gain - it uses the concept of information entropy given by the equation 2.

$$H(X) = -\sum_{i=1}^{n} p(x_i) log_2(p(x_i)) \quad (2)$$

C4.5 needs a training data set from which it generates a hierarchical decision tree. Every entity from training data set has to be an instance of the already classified samples. Each sample in the set is a vector where each element in the vector represents a feature of the sample. C4.5 then divides the data into smaller subsets by using the fact that each feature or attribute of the data set can help us make a decision regarding specific class affiliation. The feature or attribute with the highest information gain is used to make the decision of the division. The goal of dividing the set into smaller subsets is data purity. During tree construction a split that provides the largest decrease in impurity is chosen. Therfore, this method is localy optimal and it does not necessarly provide the best decision globalwise. Impurity at a given moment can be calculated using equation 3.

$$I_m = -\sum_{j=1}^{n} \frac{N_{mj}}{N_m} \sum_{i=1}^{k} p_{mj}^{i} log p_{mj}^{i} \quad (3)$$

C4.5 learning technique was very good at choosing proper decision making attributes. The testing and analysis showed that the most helpful attributes to detect different NAT behaviours are:

- The average number of bytes in a subflow in the forward direction
- Total bytes in the backward direction
- The mean size of packets sent in the forward direction
- The maximum amount of time that the flows was active before going idle
- The size of the smallest packet sent in the forward direction

- The size of the biggest packet sent in the backward direction
- The standard deviation from the mean of the packet sent in the backward direction.

Both learning techniques were tested on the same data sets and were compared based on two metrics - Detection Rate (DR) and False Positive Rate (FPR) described by equations 4 and 5, where TP is the number of correctly classified NAT traffic flows, TN is the number of correctly classified non-NAT traffic flows, FP is the number of incorrectly classified non-NAT traffic flows and FN is the number of incorrectly classified NAT traffic flows.

$$DR = \frac{TP}{TP + FN} \tag{4}$$

$$FPR = \frac{FP}{FP + TN} \tag{5}$$

Note that high DR value and low FPR value are desired.

Testing was performed on traffic flows obtained from networks of two different organizations. Traffic flows include encrypted and non-encrypted traffic. Both Naive Bayes and C4.5 were trained on a portion of the network flow data sets from both organizations. The training data set was randomly selected. All remaining data was used as testing data. Table 1 shows the number of traffic flows from each network used for training and testing purposes.

		Traffic flows		
		NAT	non-NAT	TOTAL
Org. 1	Training	9126	9126	18252
	Testing	3042	156199	159241
	Total	12168	165325	177493
Org. 2	Training	9126	9126	18252
	Testing	90116	35348	125464
	Total	99242	44474	143716

Table 1: Traffic flows used for training and testing

C4.5 produced much better results and showed similar metric values on both observed networks. This is the desired outcome since we want the method to be independent of network topology. A comparison of the results produced by Naive Bayes and C4.5 is presented in Table 2.

		Class NAT		Class non-NAT	
		DR	FPR	DR	FPR
Org. 1	C4.5	98.7%	3.7%	96.3%	1.3%
	Naive Bayes	15%	13%	98%	98%
Org. 2	C4.5	98%	2.4%	97.6%	2%
	Naive Bayes	34%	10%	89%	66%

Table 2: Comparison of Naive Bayes and C4.5

Results show that Naive Bayes does not perform well, since high detection rate (DR) also results in high false positive rate (FPR) and vice versa. This may be a consequence of the independence assumptions that Bayes makes.
C4.5 on the other hand performed very well on the given traffic flows. It also proved to be very consistent, regardless of the network, as the results are very similar on both tested networks.

3.3 Application-level data based

The following group of techniques performs identification of NAT devices and NATed machines with the help of application layer data. Specifically, the common ground for all of them are the information extracted from the HTTP packet headers, which is why they are labelled as the *deep packet inspection* approaches. Some might argue about the generality of these approaches since they entirely rely on the assumption that host machines behind NAT utilize the world wide web. However, since the Web is by far the most popular internet service at present day, we must note that assuming a host machine behind NAT uses it (for either legitimate use or illegal online activity), is reasonable.
The cookie authentication and proxy authentication approaches, described in Sections 3.3.1 and 3.3.2 concentrate on the identification of host machines behind NAT gateways. The passive fingerprinting approach, reviewed in 3.3.3, proposed a method for differentiation (rather than identification) and counting of NATed devices and additionally describes a method of detecting the presence of NAT itself.

3.3.1 Cookie authentication

The following approach uses HTTP cookies in a combination with HTTP proxy server authentication. A HTTP proxy server is usually located between an internal network and the Internet and acts as an intermediary between clients from the inside of the internal network and internet web servers [6]. HTTP proxies keep cached copies of web resources and can serve them to clients upon their requests, which usually results not only in improved performance but also in reduced load of outside HTTP servers. HTTP proxies can implement authentication in order to control HTTP access, which is commonly used in business and university networks. There are several ways allowing us to take advantage of the proxy authentication, two of which we describe in this text. They both use cookies - the first one is cookie authentication approach and the other is proxy authentication approach, which can be seen as an improvement of the first one.
Cookies are a mechanism commonly used as a means of introducing state into the otherwise stateless HTTP protocol. A cookie is a piece of information that is sent from a web server to the client and is kept locally in the web browser. While cookies are usually used for maintaining state and tracking user activity, they can also be used for authentication purposes. The latter allows identification of host machines behind NAT. Here, a proxy server performs cookie authentication, which can be summarized as follows:

1.) User's web browser sends a HTTP request to the target web server.

2.) The web server issues a cookie, stores it locally and sends it back to client. Thus, the HTTP server has authenticated the client.

3.) Client recieves the cookie and stores it locally. The web browser will include the cookie in all subsequent HTTP requests to the target server.

4.) Before serving requested resources, web server checks whether the client's cookie, sent in the HTTP request header, exists and whether it is still valid. Since cookies usually have a limited lifetime, the described authentication procedure must be repeated after a certain amount of time has passed.

The cookie authentication process is similar on the HTTP proxies. Since a cookie is issued for each web browser, identification of individual NATed hosts is possible. However, this approach does not precisely point to a specific host, rather than a specific client running on a host.

3.3.2 Proxy authentication

The following approach proposes an identification method of internal host devices behind a NAT gateway with proxy authentication mechanism, thus examining the *proxy-authentication* field from the HTTP packet headers. The *proxy-authentication* field is mainly used to restrict access to online resources to authorized users only. However, this field was intended to authenticate users (and thus contains the authenticated username), not individual machines, which is why the system administrator cannot identify several host machines simultaneously used by the same user. To solve this issue authors proposed that proxy servers additionally examine the *realm* field from the *proxy-authentication* header. When a proxy server requests proxy authentication, it sends an authentication request message to the client. The realm is included in the headers of all subsequent HTTP communication between a client and a proxy. The value of this field is obtained by combining a time stamp and a random string and is allocated to uniquely identify a host machine after the authentication process has successfully completed. However, this does not completely solve the issue of identification of NATed hosts, since by default there is no information that would tie the realm field with a specific host machine. To overcome this issue, the authors suggest associating MAC addresses of individual host machines with their realm values. They collected MAC addresses of NATed devices through a Java applet, which was run in the client's web browser. Therefore, in order for this approach to work, all hosts must have Java Runtime Environment (JRE) installed.

Another potential design issue of this approach could be the presumption that all HTTP messages exchanged between a client's web browser and proxy server always contain the *proxy-authentication* header, even after successful authentication. However, some browsers may not immediately include this header (for instance when opening a new tab), but only after the proxy server has requested (another) authentication. This means that proxy server could request authentication for each newly opened window or tab even after the user was successfully authenticated. In order to circumvent this problem, the proxy server in the proposed method issues a cookie that is used to keep the valid realm identifier once the authentication has succeeded.

For more detailed explanation of the proxy authentication approach, the reader can refer to [4], where the authors additionally tested their approach against the basic cookie authentication approach in an experimental system. However, the two approaches were only compared in terms of performance. The results, shown in Table 3 indicate that improved proxy authentication performs significantly better than cookie authentication.

Identification method	Mean access time (ms)
proxy authentication	172.73
cookie authentication	343.86

Table 3: Performance evaluation of cookie authentication versus proxy authentication. The proxy authentication is about twice as fast as the cookie authentication.

3.3.3 The passive fingerprinting approach

The approach summarized here was proposed in 2011 by Maier et. al. and is explained in great detail in [5]. The authors obtained a data set comprising of network traffic from more than 20000 DSL lines and identified the NAT-specific behavior, indicating that the observed device is a NAT gateway. According to the authors of the paper, their study was the first in the field to perform the identification of NAT gateways not only by means of examining network and transport layer packets (IP and TCP/UDP), but application level data as well (note, however, that examination of application level data was previously applied only for the purpose of differentiation between NATed hosts, as mentioned in Sections 3.3.1 and 3.3.2). Moreover, one of the attempted goals of their study was to provide concrete, quantitative data pointing towards how many different NATed devices they can detect, thus illustrating the error potential in the IP-to-end-host mappings. We will use the naming convention used in [3] and therefore refer to the described technique as passive fingerprinting.

The approach derives from the observation that dedicated NAT gateways, although mainly directly connected to the Internet in order to contact services such as DNS or NTP, generally do not use HTTP. This allows us to differentiate NAT devices from the user devices behind NAT gateways or regular hosts that additionally perform network address translation (they generally use HTTP). However, we are more interested in differentiating NATed hosts among themselves, since our goal is, for instance, identifying (at least) the characteristics of the host behind NAT believed to be a source of malicious activity. Such evidence could prove invaluable in a forensic investigation. As we previously argued, this cannot be accurately done solely with the use of IP packet information, although it can provide us with a starting estimate. For this reason, this method inspects HTTP header fields, containing information about the host OS and browser version, namely the *user-agent* header field, in addition to inspecting IP TTL. Thus, the approach is able to differentiate between MacOS and Linux hosts, as well as host machines using the same OS family. According to [5], the HTTP *user-agent* header field is present in about 90% of HTTP request messages, which makes this approach general enough for application in the majority of cases.

For the purpose of analysis, the authors developed a C program that allows creating simple statistics for obtained traffic data (for instance about the usage of communication protocols) and parsing HTTP *user-agent* field where it is

present. They analysed this information in order to differentiate not only PCs, but mobile, hand-held devices and game consoles, capable of accessing Web, as well.
The described approach always tends to underestimate the number of NATed hosts, which arises from the fact that it is not able to distinguish between NATed hosts using identical version of OS and Web browser. Therefore, this approach would yield inaccurate results if it was applied to the analysis of networks with such characteristics (i.e. enterprise, university networks). However, the approach can also overestimate the number of different hosts, since it identifies a computer running 2 different operating systems as two separate hosts.

4. CONCLUSION

Detecting NAT gateways and differentiating (counting) NATed hosts

Among the approaches aimed at detecting the presence of NAT gateways, we reviewed the TTL-based approach, passive fingerprinting and machine learning approach. While passive fingerprinting builds upon TTL-based approach and performs application level data extraction and analysis, the machine learning only uses network layer data (traffic flows). Passive fingerprinting and the machine learning approach were compared in [3] on the same data sets. Performance-wise the machine learning approach proved to be better than passive fingerprinting, as well as all other state-of-the-art methods they tested. Other methods had serious consistency problems, meaning they performed well on some specific network topologies and failed when tested on others (low DR and/or high FPR). This suggests that NAT behaviour can be different from network to network, which is why machine learning can make the difference - recall that the passive fingerprinting was only accurate when deployed in heterogeneous, residential networks.
Machine learning approach might be more complex than other methods described in this article but since it was the only method that also performed very well when tested on various completely different networks, the implementation complexity pays off well. It is also worth mentioning that this approach does not require any previous knowledge about the network, for instance the location measured as hop distance from the monitoring point (passive fingerprinting and TTL-based approach), nor does it require knowledge about any application layer data. The latter is a big advantage, since it yields same results even when the traffic on the network is encrypted.

Uniquely identifying NATed hosts

Both of the approaches aimed at correctly associating a specific host behind NAT gateway use authentication procedure and are simple, but rather primitive and inaccurate as well. These approaches are not concerned with identification of NAT itself, and can therefore be applied after the presence of NAT was already detected (for instance, after successful application of network data based approach). While it has already been shown that proxy authentication approach is better in terms of performance than cookie authentication, in order for it to accurately work, all of the host devices behind NAT must support Java browser plugins and must thereby have JRE installed. This is a serious limitation, since it is unrealistic to presume that every PC, mobile, hand-held device or game console could fulfill this requirement. Therefore, proxy authentication is mainly suitable for identifying PCs behind NAT, and the cookie authentication allows for identification of any device capable of surfing the Web. Another limitation of proxy authentication is the fact that it is necessary to configure HTTP proxies in order for the approach to work. We conclude that the proxy authentication approach is much less general than cookie authentication approach, and this loss of generality outweighs better performance and (in some scenarios) slightly more accurate results.

Concluding remarks

Let us end this discussion with the observation that the intent of this text was to provide a review of potentially interesting approaches for detecting the presence of NAT as well as differentiating NATed hosts and compare them superficially, rather than to implement each (or either) of the described approaches. For every described approach, either a relatively complex experimental network setup or large amounts of otherwise obtained network traffic data are required. Passive fingerprinting was built upon significant amount of packet-level traces, obtained from a large ISP. The machine learning approach used two data sets from the networks of two different organizations. We, however, did not have the luxury of such traffic flow information, and it would be difficult to draw any conclusions even if we were implementing such (either our own or one of the described) approaches. Furthermore, if we were to implement any of these approaches anyway, we would choose the machine learning approach, as it seems the most general and accurate. On the other hand, it is also the most sophisticated, which is why we would expect that our implementation of machine learning techniques would probably yield dubious results.

5. REFERENCES

[1] Operating System Market Share. `http://www.netmarketshare.com/operating-system-market-share.aspx`.

[2] E. Casey. *Digital Evidence and Computer Crime: Forensic Science, Computers, and the Internet with Cdrom*. Academic Press, Inc., Orlando, FL, USA, 1st edition, 2000.

[3] Y. Gokcen, V. Foroushani, and A. Heywood. Can we identify nat behavior by analyzing traffic flows? In *Security and Privacy Workshops (SPW), 2014 IEEE*, pages 132–139, May 2014.

[4] Y. Ishikawa, N. Yamai, K. Okayama, and M. Nakamura. An identification method of pcs behind nat router with proxy authentication on http communication. In *Applications and the Internet (SAINT), 2011 IEEE/IPSJ 11th International Symposium on*, pages 445–450, July 2011.

[5] G. Maier, F. Schneider, and A. Feldmann. Nat usage in residential broadband networks. In *Proceedings of the 12th International Conference on Passive and Active Measurement*, PAM'11, pages 32–41, Berlin, Heidelberg, 2011. Springer-Verlag.

[6] WhatIs. What is proxy server. `http://whatis.techtarget.com/definition/proxy-server`, 2015. [Online; accessed 30-April-2015].

[7] Wikipedia. Network address translation — wikipedia,

the free encyclopedia.
http://en.wikipedia.org/w/index.php?title=Network_address_translation&oldid=661536326, 2015. [Online; accessed 6-May-2015].

Steganografija v LTE (Long Term Evolution) sistemih

Seminarska naloga pri predmetu Računalniška forenzika

Sandi Mikuš
Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
sandi.mikus@gmail.com

Nina Mejač
Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
Fakulteta za matematiko in fiziko
nina.mejac@gmail.com

Maja Grujić
Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
maja.grujicm@gmail.com

POVZETEK

Seminarska naloga govori o tehnologiji LTE, njenem delovanju in omrežni steganografiji.
Predstavili bomo steganografsko metodo LaTEsteg, ki je bila zasnovana za sisteme LTE. Metoda omogoča uporabniku prenos podatkov, ki je neviden nepooblaščenim osebam, katere se ne zavedajo skrite komunikacije. V seminarski nalogi je opisano njeno delovanje, učinkovitost in varnost.

Keywords

steganografija, LTE, LaTEsteg

1. UVOD

Digitalna forenzika je veja forenzične znanosti, ki zajema preiskovanje gradiva, najdenega v digitalnih napravah. Najbolj pogosta vloga digitalne forenzične preiskave je, da potrdi ali ovrže hipotezo pred kazenskim ali civilnim sodiščem. S tehničnega vidika je digitalna forenzika razdeljena na številne podkategorije, odvisno od tipa digitalne naprave, ki je vpletena v preiskavo. To so na primer računalniška forenzika, mobilna forenzika in omrežna forenzika. Omrežna forenzika skrbi za nadzor in analizo omrežnega prometa. Ker je internet vse večji dejavnik v našem vsakdanu, je potrebno pozornost usmeriti tudi v varovanje podatkov, ki jih prenašamo po njem. Podatke ob prenosu pogosto zavarujemo z enkripcijo, kjer obe entiteti v pogovoru podatke kriptirata, preden jih pošljeta v omrežje. Tehnike kriptiranja so se v zadnjih letih močno razširile. Alternativni način varovanja podatkov je steganografija. V tej seminarski nalogi je predstavljena steganografska metoda LaTEsteg.

2. STEGANOGRAFIJA

Steganografija je definirana kot umetnost in znanost skrivanja podatkov, ki pošilja skrita sporočila po nevarnem kanalu tako, da je obstoj teh sporočil neopažen. Samo osebe, ki vedo za vstavljeno informacijo in imajo ustrezen ključ, bodo lahko dekodirale in videle prejeto informacijo. Steganografija so v bistvu vse metode, ki podatke skrijejo v nek prenosni medij.
Nasprotno od tipičnih steganografskih metod, ki izkoriščajo digitalne medije (slike, audio, video) za skrivanje podatkov, omrežna steganografija izkorišča kontrolne elemente in osnovne funkcionalnosti komunikacijskih protokolov. Takšne metode so težje detektirane in eliminirane. S steganografijo lahko zagotovimo varnost, zasebnost in anonimnost, po drugi strani pa je omrežna steganografija naraščajoča grožnja za omrežno varnost, saj se njene prednosti lahko izkoristi za različne napade ali grožnje, kot npr. izdajanje zaupnih informacij.
Da minimiziramo potencialno nevarnost, je pomembno takšne metode identificirati in kot protiukrep razviti metode za detektiranje. To zahteva razumevanje omrežnih protokolov in njihovo uporabo v steganografiji. Metode, ki se uporabljajo za zaznavo steganografije se označujejo s pojmom stegoanaliza.
Večina metod je osnovana na najbolj popularnih in pogosto uporabljenih komunikacijskih protokolih, kot na primer TCP/IP ali VoIP, ki jih lahko kombiniramo s standardnimi omrežji, kot je WiFi. Vsaka funkcija komunikacijskega protokola je lahko izkoriščena za gradnjo steganografske metode.
Za tehnike omrežne steganografije je značilno, da lahko spreminjajo nekatere funkcije komunikacijskih protokolov. Sprememba se nanaša na funkcije, ki skrbijo za nepopolnosti komunikacijskih kanalov (napake, zakasnitve) ali pa na funkcije, ki skrbijo za definiranje tipa informacije, ki se prenaša. Steganografske metode lahko v fizični in povezavni (angl. Data Link) plasti izkoriščajo fizične lastnosti komunikacijskih kanalov in/ali posnemajo njihove nepravilnosti. Sem spadata na primer WiPad, ki temelji na standardu WiFi in LaTEsteg, ki skrite podatke vstavlja v polnilo paketov, poslanih preko omrežja LTE. Tipično metoda uporablja spremembo enega omrežnega protokola, lahko pa uporablja tudi protokole iz več kot ene omrežne plasti. Takšna tehnika je na primer PadSteg, ki izkorišča Ethernet ARP in TCP protokol.

3. OMREŽJE LTE

Tehnologija LTE je trenutno zelo popularna v brezžičnih omrežjih, saj je najhitrejše komercialno mobilno omrežje. Z rastjo priljubnosti in z omogočanjem zelo hitrega brezžičnega prenosa podatkov, postajajo sistemi LTE popolni prenosniki za steganografijo.
Visoke hitrosti prenosa podatkov so dosegljive s pomočjo dodatnega radijskega spektra na povezavo. To je doseženo z večjim številom anten in bolj učinkovitim kodiranjem. Za izgradnjo sistema LTE so potrebni naslednji elementi:

- antene v radio postajah imenovane "ENodeB",
- transportno omrežje, ki vključuje mikrovalovne in optične povezave, ter usmerjevalnike IP,
- povezava do internetnega protokola IP Network tj. prehod (angl. Gateway),
- krmilnik za opravljanje mobilnosti (MME – Mobility Management Entity),
- baza podatkov (HSS – Home Subscriber Server), ki vsebuje informacije o posameznih naročninah,
- sistem za vodenje politike, ki zagotavlja, da so storitve, na katere so naročeni naročniki, ustrezno dostavljene,
- multimedijski podsistem IP (angl. IP Multimedia Subsystem), ki omogoča prenos zvoka in ostalih multimedijskih storitev preko omrežja LTE.

3.1 Delovanje omrežja LTE

Ko uporabnik pošlje sporočilo, datoteko, video itd., MME vzpostavi povezavo in nadzor obveščanja s terminalom. Podatki so z naprave poslani v paketih IP do radijske postaje ENodeB preko antene. Ta pošlje podatke preko transportnega omrežja do prehoda, ki je sestavljen iz več nivojev. Prehod usmerja in pošilja pakete naprej v omrežje in ohranja povezavo z napravo, medtem ko se uporabnik svobodno premika. Med prenašanjem paketov sistem vodenja politike beleži njihovo število in upošteva pravila uporabniškega naročniškega paketa.

3.2 Povezavna plast omrežja LTE

Sistem LTE temelji na prenosu paketov z uporabo protokola IP. Eden glavnih izzivov za povezavno plast omrežja LTE je zagotoviti zahtevano raven zanesljivosti za tokove IP podatkov, njihov širok spekter storitev in hitrosti prenosa podatkov. V ta namen, je bila povezavna plast LTE razdeljena na tri podplasti, ki so delno prepletene. Te plasti so PDCP (angl. Packet Data Convergence Protocol), RLC (angl. Radio Link Control) in MAC (angl. Medium Access Control) podplasti. S pomočjo PDCP, RLC in MAC je paket IP pred pošiljanjem ustrezno formatiran.

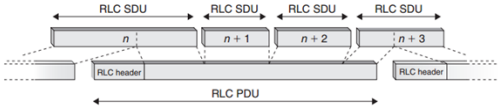
3.2.1 Plast PDCP

Plast PDCP skrbi predvsem za kompresijo glave paketa IP na strani pošiljatelja, kar zmanjša število prenesenih bitov, ter dekompresijo paketa na strani prejemnika. Poleg kompresije je plast PDCP odgovorna za integriteto in kodiranje podatkov na strani pošiljatelja in dekodiranje na strani prejemnika.

3.2.2 Plast RLC

Plast RLC skrbi za dostavo podatkov višjim slojem v pravilne vrstnem redu in brez napak. Sprejemnik RLC spremlja dohodno zaporedje paketov in identificira manjkajoče PDU enote (angl. Protocol Data Unit). Poročila o statusu so nato poslana nazaj do oddajnika RLC, od katerega zahteva ponovno pošiljanje manjkajočih podatkov. Če RLC deluje na opisani način, pravimo, da deluje v načinu AM (angl. Acknowledge Mode). Kadar sprejemnik RLC ne zahteva ponovnega pošiljanja manjkajočih PDU-jev, pravimo, da deluje v načinu UM (angl. Unacknowledge Mode).

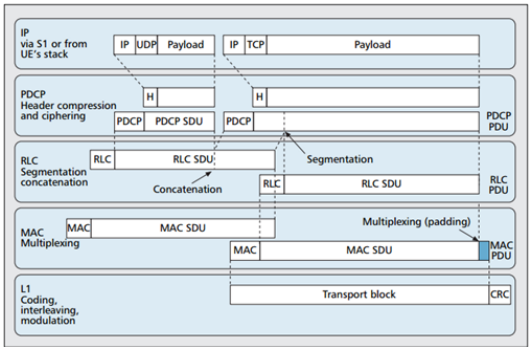
Plast RLC je odgovorna tudi za segmentacijo in konkatenacijo paketkov IP iz plasti PDCP. Razporejevalnik določi količino podatkov, ki jih bomo segmentirali oz. konkatenirali, da dobimo PDU za RLC. Zato se velikost PDU v plasti RLC dinamično spreminja.



Slika 1: Segmentacija in konkatenacija plasti RLC.

3.2.3 Plast MAC

Plast MAC je v osnovi odgovorna za reguliranje dostopa do skupnega medija, reševanje konfliktov med vozlišči, ter popravljanje napak, do katerih pride na fizični plasti.



Slika 2: Tok podatkov paketa IP skozi povezavno plast do fizične plasti.

4. SOčASNA POVEZAVA (ANGL. DUPLEXING CONNECTION)

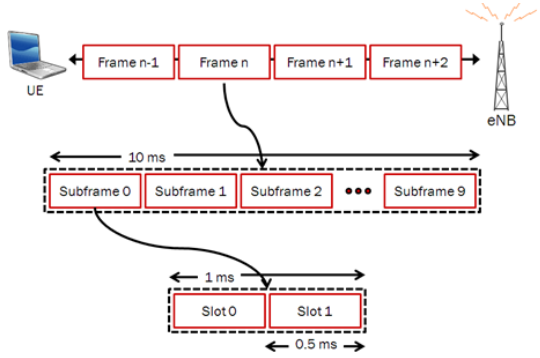
Sočasna povezava je proces, s katerim dosežemo dvosmerno komunikacijo po komunikacijskem kanalu. Poznamo dve obliki sočasne povezave: polovična (angl. half duplex connection) in polna (angl. full duplex connection) sočsna povezava. Pri polovični sočasni povezavi si dve entiteti, ki komunicirata po skupnem kanalu, izmenjavata vlogi pošiljatelja. Torej ko ena entiteta govori, druga posluša. Na takšen način deluje npr. dvosmerni radio. Polna sočasna povezava omogoča sočasno dvosmerno komunikacijo. Obe entiteti, ki komunicirata, lahko istočasno prejemata in pošiljata sporočila. Na takšen način delujejo npr. telefoni. Polna sočasna

povezava je bolj zaželen način komuniciranja, a je dražji in kompleksenjši.
Dve osnovni obliki polne sočasne povezave sta FDD (angl. Frequency-division duplexing) in TDD (angl. Time-division duplexing). Pri FDD sočasna dvosmerna komunikacija poteka z uporabo različnih frekvenc za oddajanje in prejemanje podatkov. Za učinkovito delovanje načina FDD mora biti razlika med frekvenčnima pasovoma oddajanja in prejemanja dovolj velika, da se prejemnikov in pošiljateljev signal ne motita med seboj. Način TDD uporablja za sočasno dvosmerno komunikacijo le en frekvenčni kanal, ki je razdeljen na časovne reže. Te so izmenično namenjene za prejemanje in oddajanje. Časovne reže so lahko enake dolžine, ali imajo enak prenos v smeri od uporabnika (ang. uplink) in v smeri proti uporabniku (angl. downlink). Prednost načina TDD je, da uporablja le en frekvenčni kanal, slabost pa, da je pri implementaciji potrebno zelo natančno časovno usklajevanje na pošiljateljevi in prejemnikovi strani. Če ne bi bilo sinhronizacije, bi prihajalo do prekrivanja časovnih rež.

4.1 Okvir FDD v omrežju LTE

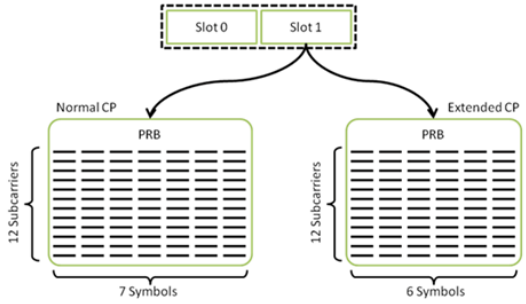
Osnovna enota za alokacijo prenešenih podatkov je blok vira (RB - angl. Resource block). Omrežje uporabniku ne dodeli enega bloka, ampak par RB. V odvisnosti od velikosti virov, ki so dodeljeni uporabniku, bazna postaja postavi podatke v primerne bloke virov. Informacija o lokalizaciji teh blokov je poslana preko drugega fizičnega kanala, tako da jih uporabnikov sprejemnik lahko locira in prebere.

V časovni domeni je vsak FDD radijski okvir $frame$ dolžine 10 ms in vsak podokvir dožine 1 ms. Vsak podokvir je sestavljen iz dveh časovnih rež T_{slot} dolžine 0.5 ms, kar pomeni, da 20 časovnih rež tvori en okvir. Blok vira zajema dve časovni reži T_{slot}, torej en podokvir (Slika 4).

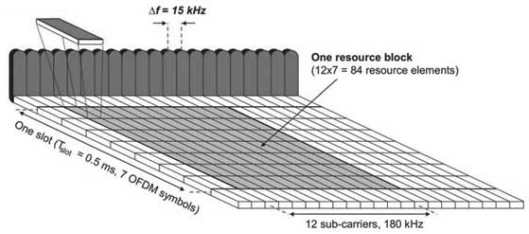


Slika 3: Okvir v časovni domeni.

V frekvenčni domeni reža vsebuje natanko 12 podnosilcev s skupno širino 180 kHz. Vsak podnosilec je 15 kHZ.



Slika 4: Okvir v frekvenčni domeni.



Slika 5: Okvir v časovni in frekvenčni domeni.

Velikost podatkov, ki jih uporabnik lahko pošilja z uporabo virov, ki so mu dodeljeni je dobro definirana. Dokument standardizacije projekta 3GPP (angl. 3rd Generation Partnership Project) vsebuje seznam vseh modulacij in kodirnih shem (MCS - Modulation and Coding Scheme). Uporablja se 28 od 31 definiranih predlogov. Vsaka shema MCS z indeksom I_{MCS} ima dodeljen parameter I_{TBS}. Ta parameter definira velikost podatkovnega bloka, ki je lahko poslan skozi kanal, odvisno od virov, ki so dodeljeni uporabniku. Tako definiran podatkovni blok imenujemo transportni blok (TB).

5. LATESTEG

Omrežje LTE postaja popolni prenosni medij za steganografijo. Raziskava podjetja Mobidia prikazuje, da prenos mobilnih podatkov preko omrežja LTE predstavlja v Južni Koreji kar 98%, v Združenih državah Amerike 92% in Združenem Kraljestvu 53% prenosa podatkov.
Ideja LaTEsteg metode je, da se za steganografijo v kombinaciji s sistemom LTE, uporablja polnilo (angl. padding), saj so bili steganografski sistemi na osnovi polnila v kombinaciji z LTE sistemom testirani in ovrednoteni kot zmogljivi in učinkoviti. LaTEsteg uporablja FDD način delovanja sistema LTE. Metoda omogoča uporabnikom, da povečajo prenos podatkov, ki so nevidni neavtorizirani entiteti, ki se ne zaveda skrite komunikacije.
Med običajno operacijo sistema LTE, se polnila paketov zapolnijo z ničlami. Takoj po prejetju paketa je to zaporedje ničel zavrnjeno s strani prejemnika, saj predstavlja nepotrebne bite brez pomena. Glavni princip LaTEsteg metode je kreiranje skritega prenosnega kanala tako, da se polnilo ne zapolni z ničlami, temveč z neko informacijo.

5.1 Delovanje metode LaTEsteg

Metoda LaTEsteg deluje na podlagi naslednjih predpostavk:

- sistem LTE deluje v UM ali AM načinu,
- v sloju PDCP ni kompresije glave paketa IP,
- segmentacija je na sloju RLC uporabljena le, kadar je velikost PDU večja od transportnega bloka (TB - transport block),
- konkatenacija na sloju RLC je uporabljena le, če dodajanje celotne enote SDU (brez fragmentacije), ne preseže razpoložljive velikosti TB.

5.2 Definicija oznak

- L_{IP} – velikost paketa IP, ki ga prenašamo,
- L_{H-PDCP} – velikost glave, ki jo dodamo enoti PDCP SDU na plasti PDCP (ponavadi 2 bajta),
- L_{H-RLC} – velikost glave v bajtih, ki jo dodamo v plasti RLC:

$$L_{H-RLC} = \begin{cases} 2, & k = 1 \\ 2.5 + 1.5k, & k = 3, 5, 7, \dots \\ 2 + 1.5k, & k = 2, 4, 6, \dots \end{cases},$$

  kjer je k število enot RLC SDU v eni enoti RLC PDU,

- L_{H-MAC} – velikost glave v bajtih, ki jo dodamo v plasti MAC in je sestavljena iz MAC pod-glav, ki so kreirane za posamezno enoto MAC SDU vsebovano v enoti MAC PDU, in iz polnila:

$$L_{H-MAC} = \begin{cases} 1, & n = 1, \\ & L_{PAD} = 0 \\ L_{SH-MAC} \times (n-1) + 1, & n > 1, \\ & L_{PAD} = 0 \\ L_{SH-MAC} \times n + 1, & n \geq 1, \\ & L_{PAD} \neq 0 \end{cases},$$

  kjer je n število enot MAC SDU vključenih v enoto MAC PDU in L_{SH-MAC} velikost pod-glav, ki je odvisna od trenutne velikosti enote MAC SDU

$$L_{SH-MAC} = \begin{cases} 2\ bajta, & L_{MAC-SDU} \leq 128\ bajtov \\ 3\ bajte, & L_{MAC-SDU} > 128\ bajtov \end{cases},$$

- L_{PAD} – velikost polnila v MAC PDU enoti (v bajtih),
- BR – kapaciteta skrivnega kanala v bajtih,
- TB_{SIZE} – velikost transportnega bloka, ki je odvisna od virov dodeljenih uporabniku ter od uporabljene sheme MCS.

5.3 Delovanje metode LaTEsteg glede na velikost paketa IP

Od velikosti paketa IP, je odvisno, kako ga formatiramo, preden je poslan po radijskem kanalu in dostavljen prejemniku. Imamo dve možnosti:

1. $L_{IP} + L_H \leq TB_{SIZE}$, kjer je $L_H = L_{H-PDCP} + L_{H-RLC} + L_{H-MAC}$ za n = 1, k = 1, $L_{PAD} = 0$

V tem primeru velikost paketa skupaj z dodanimi glavami ne preseže razpoložljive velikosti transportnega bloka in fragmentacija IP paketa ni potrebna.

Maksimalno število paketov IP, ki jih lahko vključimo v določen transportni blok, je:

$N_{MAX} = \left\lfloor \frac{TB_{SIZE} - L_{H-MAC} - L_{H-RLC}}{L_{IP} + L_{H-PDCP}} \right\rfloor$.

Ločimo dva primera

- Paketi brez polnila $L_{IP} + L_H = TB_{SIZE}$,
  kjer je
  $L_H = N_{MAX} \times L_{H-PDCP} + L_{H-RLC} + L_{H-MAC}$,
  n = 1, k = N_{MAX}, $L_{PAD} = 0$,
  N_{MAX} je število sočasno prenesenih paketov.

  Ker je velikost polnila enaka nič, v polnilo ne moremo dodati podatkov, ki jih želimo poslati v omrežje po skrivnem kanalu.

- Paketi s polnilom $L_{IP} + L_H < TB_{SIZE}$,
  kjer je
  $L_H = N_{MAX} \times L_{H-PDCP} + L_{H-RLC} + L_{H-MAC}$,
  n = 1, k = N_{MAX}, $L_{PAD} \neq 0$,

  pa velikost polnila ni enaka 0, temveč:

  $L_{PAD} = TB_{SIZE}$ - ($N_{MAX} \times (L_{IP} + L_{H-PDCP}) + L_{H-RLC} + L_{H-MAC}$) in N_{MAX} število sočasno prenesenih paketov.

2. $L_{IP} + L_H > TB_{SIZE}$

V tem primeru velikost paketa skupaj z dodanimi glavami preseže razpoložljivo velikost transportnega bloka. Potrebna je segmentacija poslanega paketa na plasti RLC. Paket IP skupaj z glavo plasti PDCP je potrebno razdeliti na q delov. Posledično je potrebnih za pošiljanje enega paketa q transportnih blokov:

$q = \left\lfloor \frac{L_{IP} + L_{H-PDCP}}{TB_{SIZE} + L_{H-MAC} + L_{H-RLC}} \right\rfloor$

Ob prenosu q transportnih blokov lahko pride do dveh primerov:

- Paketi brez polnila $L_{IP} + L_H = \mathrm{q} \times TB_{SIZE}$,

  kjer je $L_H = L_{H-PDCP} + \mathrm{q} \times L_{H-RLC} + \mathrm{q} \times L_{H-MAC}$, n = 1, k = 1, $L_{PAD} = 0$.

  V tem primeru je potrebnih natanko q transportnih blokov za prenos IP paketa.

- Paketi s polnilom $L_{IP} + L_H < \mathrm{q} \times TB_{SIZE}$,

  kjer je $L_H = L_{H-PDCP} + \mathrm{q} \times L_{H-RLC} + \mathrm{q} \times L_{H-MAC}$,

  $$\begin{cases} n = 1, k = 1, L_{PAD} = 0, & \text{prvih (q - 1) transportnih blokov} \\ n = 1, k = 1, L_{PAD} \neq 0 & \text{q - ti transportni blok} \end{cases}.$$

  Kar pomeni, da zadnji blok paketa IP ni povsem napolnjen s podatki, torej ostane nekaj prostora za polnilo. Velikost polnila v bajtih se izračuna:

  $$L_{PAD} = \frac{TB_{SIZE} - L - L_{H-RLC} - L_{H-MAC}}{q},$$

  kjer je L število bajtov velikosti paketa IP skupaj z glavo PDCP, ki so preneseni v q-tem transportnem bloku:

  $\mathrm{L} = L_{IP} + L_{H-PDCP}$ - (q - 1) $\times$ TB_{SIZE} - L_{H-RLC} - L_{H-MAC}

  Učinkovitost steganografskega sistema določuje poleg velikosti polnila še kapaciteta skrivnega kanala, ki jo izračunamo po formuli:

  $$\mathrm{BR} = \frac{L_{PAD}}{T_{frame}},$$

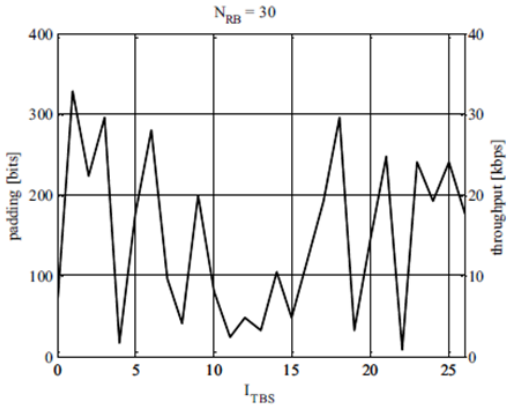
  kjer je L_{PAD} velikost polnila in T_{frame} trajanje prenašanja okvirja.

6. UčINKOVITOST METODE LATESTEG

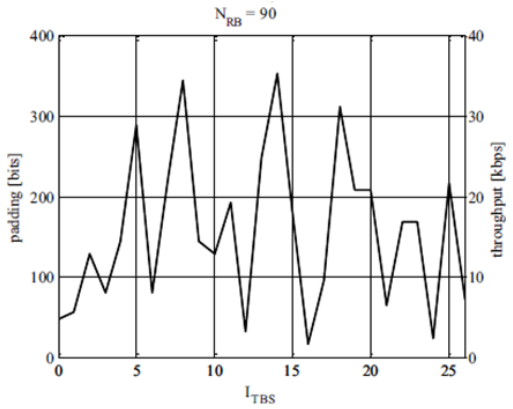
Ker se v omrežju najpogosteje pojavljajo paketi IP velikosti 40 in 1500 bajtov, je analiza steganografske metode opravljena na paketih teh dveh velikosti. Na velikost polnila vplivajo zunanji faktorji ter pogoji v omrežju. Posledično učinkovitost steganografskega sistema ni konstantna. Na kapaciteto skrivnega kanala vplivajo naslednji dejavniki:

- velikost paketa IP,
- velikost dodane glave na vsaki plasti,
- segmentacija na plasti RLC,
- velikost transportnega bloka, ki je odvisna od uporabe MCS, pogojev v radijskem okolju in virov dodeljenih uporabniku.

Spodnji štirje grafi (slike 6, 7, 8, 9) prikazujejo, kako izbor MCS in velikost dodeljenih virov uporabniku vplivajo na kapaciteto skrivnega kanala ter velikost polnila. Na prvih dveh grafih (slike 6, 7) je opravljena analiza na paketih IP velikosti 40 bajtov, na drugih dveh (slike 8, 9) pa 1500 bajtov.



Slika 6: IP velikosti 40 bajtov in velikost dodeljenih virov 30.

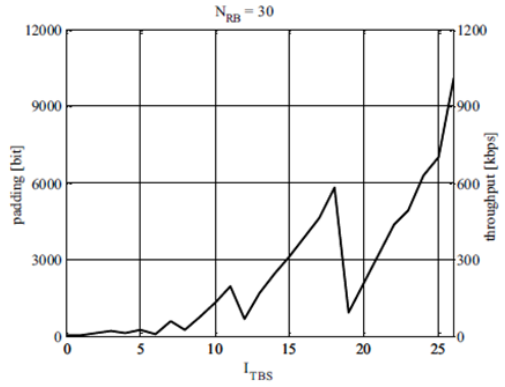


Slika 7: IP velikosti 40 bajtov in velikost dodeljenih virov 90.

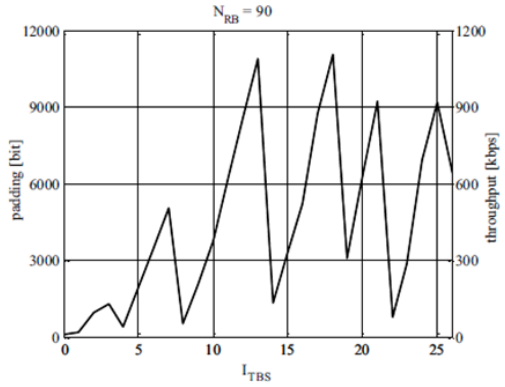
Na grafih 6 in 8 je velikost dodeljenih virov enaka 30, na grafih 7 in 9 pa 90.
Graf 10 prikazuje velikost polnila in kapaciteto skrivnega kanala kot funkcijo virov dodeljenih uporabniku, ter izbiro MCS (I_{TBS} = 0, 9, 15, 26) za pakete IP velikosti 1500 bajtov.
Iz grafov je razvidno, da velika količina dodeljenih virov ne zagotavlja velike kapacitete skrivnega kanala in večjega polnila, saj močno variira. Velikost polnila pa se spreminja z izboljšavo pogojev v radijskem okolju in z uporabo določenega MCS-ja, kar vpliva na velikost transportnega bloka. To pomeni, da imajo na učinkovitost LaTEsteg metode vpliv trenutni pogoji v radijskem kanalu.

7. REZULTATI SIMULACIJ

Za preverjanje in potrditev teoretičnih rezultatov so bile izvedene številne simulacije. S temi simulacijami je bil preverjen vpliv pogojev v radijskem kanalu na kvaliteto skrivnega



Slika 8: IP velikosti 1500 bajtov in velikost dodeljenih virov 30.

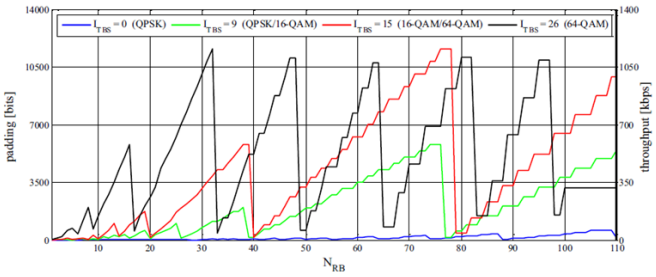


Slika 9: IP velikosti 1500 bajtov in velikost dodeljenih virov 90.

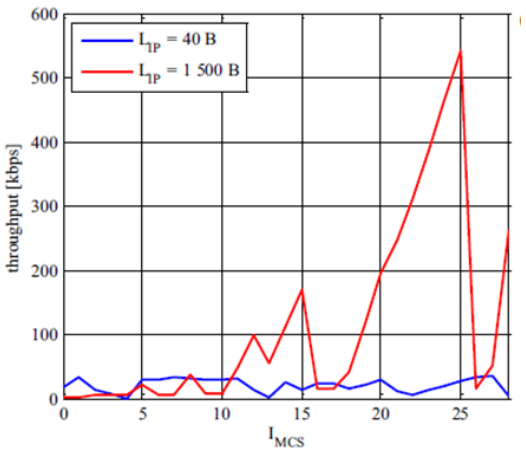
prenosa, varnost skrivnega kanala in izračunan strošek operacij steganografskega sistema. Grafa 11 in 12 prikazujeta vpliv uporabljenega MCS (I_{MCS}) na kapaciteto skrivnega kanala kot funkcijo velikosti paketa IP in razpoložljivih virov. V primeru grafa 11 je velikost razpoložljivih virov 20, v primeru grafa 12 pa 70. Iz teh dveh grafov lahko razberemo, da na kapaciteto skrivnega kanala ob določitvi velikosti paketa IP ter dodeljenih virov vpliva izbor MCS.

Grafa 13 in 14 prikazujeta vpliv pogojev v radijskem kanalu (E_b/N_0) na kapaciteto skrivnega kanala ob prenosu paketov IP velikost 1500 bajtov kot funkcijo izbranega MCS in dodeljenih virov. Na grafu 13 je število dodeljenih virov enako $N_{RB} = 20$, na grafu 14 pa $N_{RB} = 70$. Razvidno je, da izbor MCS vpliva na kapaciteto skrivnega kanala. V primeru grafa na sliki 14 lahko razberemo, da višji indeks izbranega MCS še ne pomeni večje kapacitete.

Velik pomen na kvaliteto skrivnega prenosa ima količina šuma, ki je v kanalu prisoten. Kvaliteta prenosa se meri s številom pravilno prejetih bitov v razmerju z vsemi poslanimi biti. S slabšanjem pogojev v radijskem kanalu, število pravilno prejetih bitov pada in s tem tudi kvaliteta in kapaciteta skrivnega kanala. Z uporabo nižje modulacije in višjim

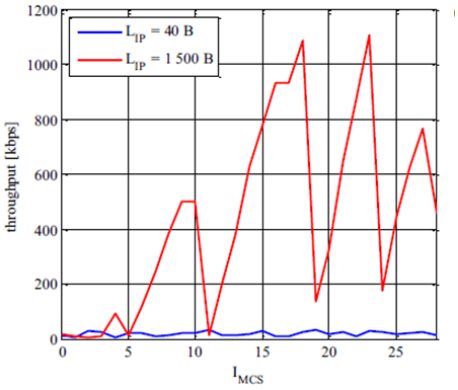


Slika 10: Velikost polnila in kapaciteta skrivnega kanala kot funkcija razpoložljivih virov za izbran MCS.

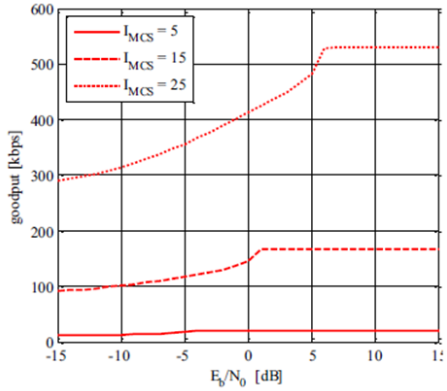


Slika 11: Vpliv uporabljenega MCS na kapaciteto skrivnega kanala za velikost razpoložljivih virov 20.

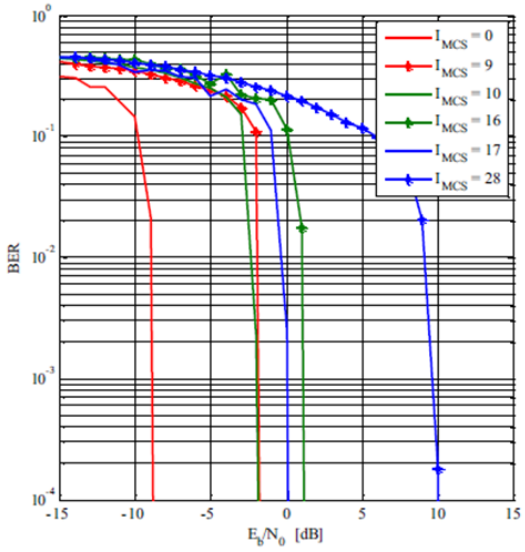
številom odvečnih bitov ima šum v kanalu manjši vpliv na poslani signal. S tem dosežemo večjo verjetnost detekcije in poprave napačno prejetih bitov. V nekaterih primerih se lahko celo izognemo vsem napakam. Za MCS z večjim parametrom I_{MCS} pada število prejetih bitov z napako počasneje glede na stopnjo prisotnega šuma, kot za parametre z nižjim I_{MCS}. To pa zato, ker MCS z nižjim indeksom uporabljajo nižjo modulacijo in večji delež odvečnih bitov, kar po prejšnji ugotovitvi zagotavlja detekcijo in popravo večjega števila napačnih bitov.



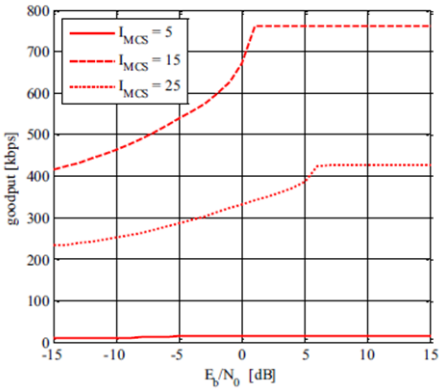
Slika 12: Vpliv uporabljenega MCS na kapaciteto skrivnega kanala za velikost razpoložljivih virov 70.



Slika 13: Vpliv pogojev v radijskem kanalu (E_b/N_0) na kapaciteto skrivnega kanala ob prenosu paketov IP velikost 1500 bajtov za velikost dodeljenih virov 20.

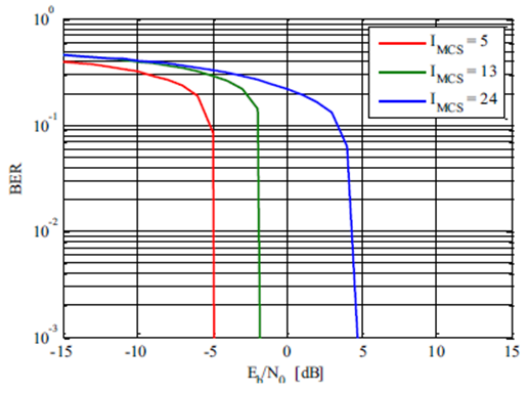


Slika 15: Vpliv šuma v kanalu (E_b/N_0) na stopnjo napačnih bitov (BER – bit error rate) v skrivnem kanalu glede na izbran MCS = 0,9,10,16,17,28.

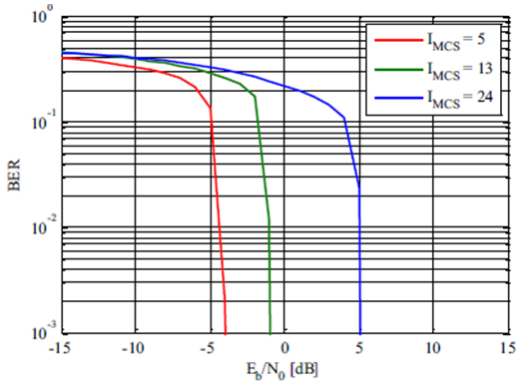


Slika 14: Vpliv pogojev v radijskem kanalu (E_b/N_0) na kapaciteto skrivnega kanala ob prenosu paketov IP velikost 1500 bajtov za velikost dodeljenih virov 70.

Graf 15 prikazuje vpliv šuma v kanalu (E_b/N_0) na stopnjo napačnih bitov (BER – bit error rate) v skrivnem kanalu glede na izbran MCS. Za nizke indekse MCS raste BER za enake stopnje prisotnosti šuma v kanalu. Razlog je različno število odvečnih bitov, kar ima vpliv na zmožnost detekcije in popravljanja napak.
Ob oblikovanju steganografskega sistema je potrebno paziti, da sistem nima vpliva na osnovne operacije v omrežju. Zaželeno je, da ima sistem minimalne dodatne stroške zaradi skrivnega kanala. Opisan steganografski sistem prinese minimalne dodatne stroške, saj so podatki v prenašajočem okvirju skriti v tistem delu, ki je drugače spregledan. Posledično skrivni prenos podatkov nima vpliva na običajne operacije omrežja ter ne povzroča novih napak.



Slika 16: Stopnja napačnih bitov (BER) za običajne operacije v omrežju.



Slika 17: Stopnja napačnih bitov (BER) za običajne operacije v omrežju ob prisotnosti steganografskega sistema.

Grafa 16 in 17 prikazujeta stopnjo napačnih bitov (BER) kot funkcijo šuma v kanalu (E_b/N_0) ter na grafu 16 običajnih operacij v omrežju oz. na grafu 17 običajnih operacij v omrežju ob prisotnosti steganografksega sistema.
V steganografski metodi LaTEsteg je maksimalna dosežena hitrost skrivnega prenosa podatkov enaka 1,162 Mb/s. Učinkovitost steganografskega sistema je odvisna od številnih faktorjev, na katere uporabnik ne more vplivati. Te faktorji so velikost paketa IP, izbran MCS, ter število dodeljenih virov. Zaradi naštetih dejavnikov lahko pride do kapacitete skrivnega kanala, ki je enaka nič.
Prednost metode LaTEsteg je, da uporaba sistema ne vpliva na delovanje sistema LTE. Zato ni dodatnih stroškov za uporabo skrivnega pošiljanja podatkov, zaradi česar je sistem zelo varen. Kakršnekoli anomalije ne porodijo suma o uporabi skrivnega kanala med uporabniki, kar naredi steganografski sistem varen in učinkovit.

8. ZAKLJUčEK

V tem članku smo se spoznali s steganografijo v sistemih LTE. Najprej smo opisali tehnologijo LTE, ter delovanje povezavne plasti sistema LTE, ki je razdeljena na tri podplasti: PDCP, RLC in MAC. Glede na delovanje povezavne plasti, je bilo postavljenih veliko domnev o učinkovitosti delovanja metode LaTEsteg, ki jo določa velikost polnila in kapaciteta skrivnega kanala. Analiza LaTEstega je bila opravljena na paketih IP velikosti 40 in 1500 bajtov, kjer je bilo ugotovljeno, da učinkovitost ni konstantna, saj na velikost polnila in kapaciteto skrivnega kanala vpliva več dejavnikov. Dejavniki so med drugim zunanji faktorji, pogoji v omrežju, velikost paketa IP, formatiranje in segmentacija paketa, ter izbira indeksa MCS. Rezultati opazovanj so podrobno predstavljeni v prejšnjih poglavjih.
Sistem je zelo varen, saj njegova uporaba ne vpliva na delovanje sistema LTE in zato uporabniki ne posumijo, da se prenašajo dodatne informacije.
Z določenimi modifikacijami je možno opisani steganografski sistem razširiti na druga omrežja, ki uporabljajo polnilo. Torej je ta metoda primerna za prenašanje skritih podatkov tudi za druga omrežja.

9. LITERATURA

[1] Frame Structures in LTE-TDD and LTE-FDD. `http://lteuniversity.com/get_trained/expert_opinion1/b/lauroortigoza/archive/2012/08/07/frame-structures-in-lte-tdd-and-lte-fdd.aspx/`. [Dostopno 3.5.2015].

[2] LTE Quick Guide. `http://www.tutorialspoint.com/lte/lte_quick_guide.htm/`. [Dostopno 3.5.2015].

[3] TDD FDD Duplex Schemes. `http://www.radio-electronics.com/info/cellulartelecomms/cellular_concepts/tdd-fdd-time-frequency-division-duplex.php`. [Dostopno 3.5.2015].

[4] What Is Half-Duplex And Full-Duplex Operation, And How Does It Affect Your Router? `http://www.makeuseof.com/tag/what-is-half-duplex-and-full-duplex-operation-and-how-does-it-affect-your-router/`. [Dostopno 3.5.2015].

[5] E. Dahlman, S. Parkvall, J. Skold, and P. Beming. *3G evolution: HSPA and LTE for mobile broadband*. Academic press, 2010.

[6] R. Gawade, P. Shetye, V. Bhosale, and P. Sawantdesai. Data hiding using steganography for network security.

[7] I. Grabska and K. Szczypiorski. Steganography in long term evolution systems. In *Security and Privacy Workshops (SPW), 2014 IEEE*, pages 92–99. IEEE, 2014.

[8] T. Innovations. Lte in a nutshell. *White Paper*, 2010.

[9] A. Larmo, M. Lindstrom, M. Meyer, G. Pelletier, J. Torsner, and H. Wiemann. The lte link-layer design. *Communications Magazine, IEEE*, 47(4):52–59, 2009.

[10] J. Lubacz, W. Mazurczyk, and K. Szczypiorski. Principles and overview of network steganography. *arXiv preprint arXiv:1207.0917*, 2012.

[11] Mobidia. LTE Data Usage. `http://www.mobidia.com/blog/lte-mobile-data-usage-0/`. [Dostopno 3.4.2015].

[12] I. Poole. OFDM Orthogonal Frequency Division Multiplexing Tutorial. `http://www.radio-electronics.com/info/rf-technology-design/ofdm/ofdm-basics-tutorial.php/`. [Dostopno 3.4.2015].

[13] B. T. Scheme. Lte: the evolution of mobile broadband. *IEEE Communications magazine*, page 45, 2009.

Overview of PeerShark and other software for detecting peer-to-peer botnets

[Digital forensic]

Luka Krsnik
Faculty of computer and information science
University of Ljubljana
Vecna pot 113
1000 Ljubljana, Slovenija
krsnik.luka92@gmail.com

Ozbolt Menegatti
Faculty of computer and information science
University of Ljubljana
Vecna pot 113
1000 Ljubljana, Slovenija
ozbolt.menegatti@gmail.com

Manca Zerovnik
Faculty of computer and information science
University of Ljubljana
Vecna pot 113
1000 Ljubljana, Slovenija
manca.zerovnik3@gmail.com

ABSTRACT

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

Theory

Keywords

digital forensic, peer-to-peer botnet, tracking conversations

1. INTRODUCTION

As introduced in the title of the article, we are going to talk about software for detecting peer-to-peer (P2P) botnets named PeerShark. We can not dig in the content, unless we clarify the meaning of each word written in the address first.

1.1 Botnet

Botnet is a derivative that consists of two parts of two different words. First part is "bot", which comes from robot, and the second is "net", which stands for network [10]. Bots are usually computer programs, which wait for commands from their master, and execute them. Word botnet comes in mind, because all of these bots are connected to the network, which is essential, because commands from master comes through it [12, 9]. Botnets are not necessarily illegal. They might be used in IRC (Internet Relay Chat), to set channel modes on bots, featuring IRC channels without unwanted users [10]. However usually when we talk about botnets we imply on illegal ones, that are formed from compromised computers, controlled by master, and used for i.e. DDoS (distributed denial-of-service) attack or spamming. The malicious software that is normally used for this, are viruses or trojans. They communicate via IRC network (through IRC channels) [12].

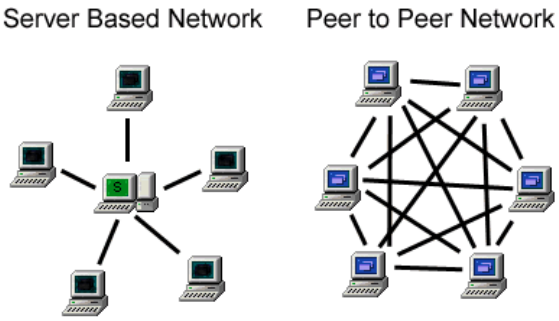


Figure 1: Server based network vs P2P network

1.2 P2P

Most of internet services are based on client-server architecture. However there exist other ways for data transmission. One possibility is P2P (Figure 1). This stands for connection between two peers (two computers), where each of the peers simultaneously acts as a server and a client, thus requiring no third party servers for communication. This way, peers are usually equal among each other (not necessarily). Nowadays these type of connection is mostly used for file sharing and video streaming [12].

1.3 P2P botnets

Originally harmful botnets were using IRC for their Command and Control (C & C) functionality. However this type of system was centralized (Figure 2), which in practice meant, that isolation of C & C server meant the end of botnet. Therefor botnets adapted and (some of them) started using P2P (Figure 3). One of the main reasons for this is the resistance of P2P connections to break down if some of the bots are disabled. Even if large amount of bots are taken down the botnet might still survive.

Now let us concentrate on how the bots usually work. The life of a bot might be separated in four phases [10]:

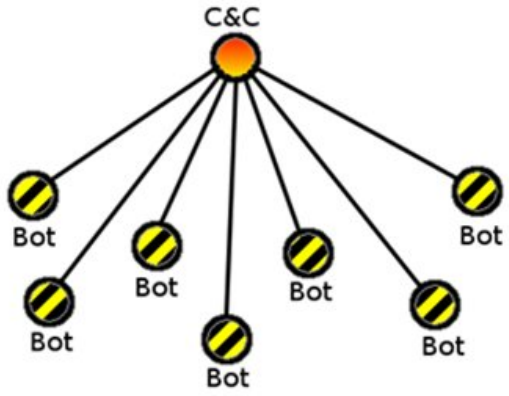


Figure 2: Scheme of botnet with centralized C & C

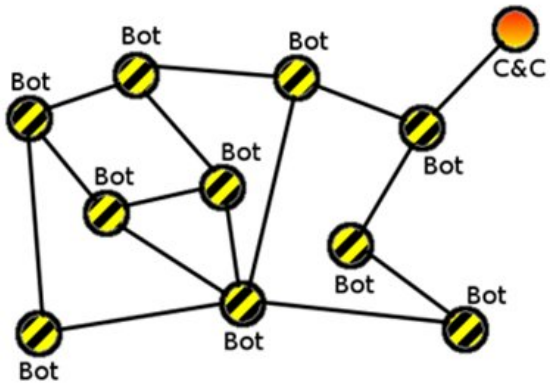


Figure 3: Scheme of P2P botnet

1. Spreading phase - A phase where bots seek to infect as many devices as possible. Bots were known to spread in various ways, one might or might not expect, i. e. drive by downloads, e-mails, USB sticks, etc.
2. Rally phase - After spreading bots have to somehow acknowledge the existence of each other. They do this by connecting to a peer list and thus join the P2P network.
3. Waiting phase - Phase in which bots wait for C & C instructions.
4. Execution phase - Last phase, where bots actually do what they were meant to in the first place. They might do a DDoS attack, send SPAM mails, etc.

Nowadays bots are getting more and more innovative, to avoid detection. They tend to have as little communication as possible, to evade Intrusion detection systems and firewalls and have been known to invade things like smart refrigerators, where one would not expect them.

2. RELATED WORK

PeerShark is the main topic of this paper, but let us review some of the previous attempts of addressing the same or similar problems.

2.1 Point of view

We can view this problem from many different perspectives. We can stand in the hearth of the network - the backbone, we can try and locate bot activity from within ISP's networks, or we can just be on the edge of our internal company network and monitor from there. Different papers, that will be discussed in next subsections all use one of these views on the problem.

2.2 Complexity

What we mean by term complexity is from what point do we wish to tackle this problem. Do we only have a connection to a network and we need to obtain packets and analyze them, or have packets been in some form or another already been salvaged. Then we might only need to examine packets and streams for whom we know beforehand are p2p traffic, or do we first filter non-p2p traffic out in order to do the last step of identifying malicious p2p activity (bots) from benign (BitTorrent, Skype and alike).

2.3 Success rate

Most every paper uses its own matrix for measuring success rate, even though usually those matrices are based on true positive and false negative rates. Second point to consider is input data. Some research teams use data, obtained from real networks, where others use data from internal academic networks. Also the size of input data is variable from project to project.

None the less, most papers boast about above 90% true positive rate and below 5% negative. Even though some researchers report numbers as high as 100% successful rate and below 0.5% false positives, those numbers are usually obtained within specific circumstances and are not comparable between each other.

Third fact to take into account is the content of subsection 2.1.

2.4 Other papers

Let us look at some work done before Peershark.

2.4.1 Sen et al. [13]

Approach in this paper uses application level signatures. This approach gives us a good fit for known botnets. The information, that such a signature contain is usage statistics of a protocol (UDP vs. TCP), longevity of connections (packets vs streams) and packet types (weather packet is transporting data or just using network to send signaling/control packets). All these properties can be accessed without deep packet inspection and as such it gives us faster and cost effective implementations.

Te authors point out, that the type of data can be determined by only a few packets send by two endpoints. Problems, that are not addressed with such approach is detecting "future" applications, being malicious or benign.

2.4.2 Iliofotou et al. [6]

Instead of using static data and using machine learning to create signatures for applications, approach in this paper uses graphs to indentify subnetworks of bots from the point

of view of an ISP. This approach identifies protocol from first 16B of a packet and using this data creates a Traffic Dispersion Graph [7]. Form this graph authors perform clustering and cluster merging based on the IP-s of the edges. From this point on, machine learning approaches are used to determine, whether a given cluster of subgraph is a p2p botnet or not. End result is graphically clear view of networks of p2p bots with high accuracy.

2.4.3 BotMiner [3]

This paper is a classical example of more engineering approach, using many available tools on the internet. It consists of two different analysis, where end result is combinations of the two.

C-monitor is a classical network communication log. From this log authors try to create clusters, which are based on multiple parameters:

- flows per hour
- packets per flow
- average number of Bytes per packer
- bytes per second exchanged

After discretization of these parameters, authors obtain clusters.

A-monitor on the other hand logs malicious info, like port scanning, spamming, binary downloading etc. . This monitor is built on top of popular IDS Snort, where application signatures were written by authors themselves. With given data, clustering algorithms are used again to obtain second set of clusters.

Cross A-C colleration gives us likely IPs with malicious activities.

2.4.4 BotTrack [6]

Here we have a novel approach to problem, using Google's PageRank. But first, authors use Cisco's NetFlow monitors to obtain network logs. From these, graph is constructed. To determine how strongly some nodes are connected, authors use PageRank. The idea is, that if we know that one node is a bot, then nodes interacting with it are more likely to be bots. On the other hand, if a node is connected to non-malicious bot (e.g. yahoo.com), then it is less likely to be malicious.

After graph is obtained, classical clustering is performed based on node flow direction (bad data out/bad data in) and given a threshold, a cluster is considered harmful or not.

2.4.5 Entelecheia [5]

Here, authors focus on social behaviour of p2p botnets during waiting stages. Typical behaviour here is constant but minimum communication. Also at the time, port switching is already used heavily by botmasters, so port numbers are excluded from this approach, using only IPs.

Between constant communication endpoints so-called superflow is marked. Using these superflows, authors build graphs, filter out high-communication superflows and after this point on, approach is simmilar to previous papers - clusterin and machine learning on clusters.

Approach here is considered to be future-proof, since no application signatures or port numbers are used.

2.4.6 PeerRush [11]

This recent attempt at the problem is trying to detect p2p communications at large, while still trying to detext botnets specifically.

Newer ideas here are:

- Failed connections are a property of most p2p apps,
- IP direct without DNS query is also frequent in p2p communications,
- Destination diversity compared to classical communications.

Second step is to diversify benign and malicious p2p traffic. Method builds on is similar to approach from 2.4.1. This newer approaches are usually better because of much better training data for machine learning, that researchers can obtain from latest botnets, such as Zeus.

2.4.7 BotGrep [8]

This article is mentioned here because of its interesting, graph-base approach to detecting p2p traffic. Similarly to previous papers, here graph of communication is constructed. On this graph are performed random walk. For more structured subgraph (classical for non p2p communications) walking over same nodes frequently is observed. Meanwhile for less structured, p2p subgraphs, a random walk amongst vertices rarely steps twce on the same vertex. All in all, approch in this paper is very mathematical and a nice novelty.

2.4.8 Zhang, Junjie, et al. [14]

In general this two-phase approach is not very different from previous. But one interesting idea is worth noting. Overlap of p2p bots is bigger the overlap of two classical p2p applications.

This happens because two distinct bots have long connections with a number of bots, where overlap between this two groups of bots is relatively big. Given two instances of bit-torrent applications within a smaller local network, the probability of one (who is downloading newest Game Of Thrones episode) and the other (downloading latest Coldplay album) haveing lots of common connected peers is much smaller.

Given the point of view, where we sit on the edge of a local network, this fact gives us good indication, if and where there are bots within our network.

2.5 On the papers

Last thought on papers, that we studied (many are reviewed in subsection 2.4) is, that most are easily digestible, comparing them to papers from other scientific fields. Also, a hint of marketing approaches can be sensed in the wording.

3. PEERSHARK

In [9] is presented a new approach for detecting P2P botnets named PeerShark. It is based on tracking conversation. The proposed approach can differentiate network traffic from malicious and benign and categorize P2P applications running on a host in two groups - regular or botnet - with 95 % accuracy.

3.1 Basic approach

PeerShark is based on conversation tracking. It does not observe network packets payload, it does not relay on any information about signatures or blacklists of IPs. The main activity of conversation tracking is to observe network conversation by tracking UDP/TCP headers, amount of payload and time between packets being send. A special behavior of malicious conversations have been detected, hence at the end supervised machine learning algorithms for classification are used to categorize application whether in regular or botnet class.

3.2 Contributions

PeerShark successfully improve flow-based approaches for P2P botnet detection. It can detect advanced botnets with randomized port and botnets which operate over TCP and also UDP. Also the PeerShark does not rely on Deep Packet Inspection, hence the payload encryption doesn't affect the approach. PeerShark reveal the malicious network traffic in their rally or waiting stages, when network administrator could not detect it.

3.3 Main concepts

Primary task of detecting botnet is to detect it before execution stage. When execution stage happen, it is not hard for network administrator to differentiate malicious traffic from regular, because often the amount of communication becomes big. Some of the attacks remain stealthy. Stealthy botnet is the one that PeerShark successfully detect. Behavior of botnet conversations have been observed. It have been figured out, that the bot-peers need to communicate among near pears to exchange states and commands. But their conversations should not be very frequent to avoid the firewalls/IDS. Their behavior is stealthy and different from regular. Another important approach is that PeerShak does not incorrectly categorize network traffic in flows as flow-based approach which make flows based on port number which is randomly changed on hosts. One conversation between two peers is separated in more flows. That does not happen in PeerShark which was noticed as a significant benefit - Peershark track conversations between two peers and not flows with the same port and protocol.
PeerShark sucesfully categorize P2P applications in regular or botnet. Everything is based on the behavior of applications conversations. Malicious applications stay in contanct with each other until the attack. They maintain communication with same peers for long time which is not usual for benign application which often connect with certain peer only for the short time of sharing files. Communication of bot will be low at volume, often there are only short commands, in contrast regular peers often exchange multimedia files. Every P2P application has its own specific control messages (for searching files, connect/disconnect to network, searching for files etc.). Patterns in those have been detected and on this basis aditional features for categorization have been extracted. Four most significant features which are used for PeerShark categorization are [9]:

1. The duration of conversation
2. The number of packets exchanged in the conversation
3. The volume of data exchanges in the conversation
4. The median value of the inter-arrival time of packets in that conversation

3.4 Implementation

The implementation is separated in four modules: packet filtering module, conversation creation module, conversation aggregation module and classification module.
The input of PeerShark are network logs in the form of raw packets. They are filtered in packet filtering module. All packets which does not have a valid IPv4 Header, TCP or UDP header and payload greater than zero, are filtered out. From remaining packets four features are extracted as an input for next module: source IP, destination IP, payload size and timestamp.
Filtered packet information is input for the conversation creation module. Initial flowgap value is used to create conversations. Conversation is made if a pair of IPs in packet match with another packet IPs and if their timestamps lies in the flowgap from beginning or end of conversation. Conversation is iterative modified.
Those conversations are fed as input in conversation aggregation module. Algorithm is very similar as in previous module. The conversations are aggregated with higher flowgap value specified by user for example a network administrator. It can be set on 2 hours, 30 hours, etc. This is very useful to detect different malicious traffic. The result of this module are features which are used in classification module and were previously explained: number of packets, conversation volume, conversation duration and the median if inter-arrival time of packets in the conversation.
The classification model uses Bayesian networks, Decision trees and Boosted REP trees. Those are all a supervised machine learning algorithms.

Modules are graphically represented in Figure 4.

3.5 Evaluation and results

PeerShark was tested on two datasets obtained by authors of [11] from University if Georgia. For benign P2P Data 50,000 conversations of eMule and uTorrent were used. For Malicious P2P data 50,000 conversations of Storm and Waledac were used. The number of known malicious host for Waledac is 3 and for Storm 13. But it is not known for other IPs if they are malicious or benign. To create a 'ground truth' all conversation who has at least one malicious IP were treated as malicious.

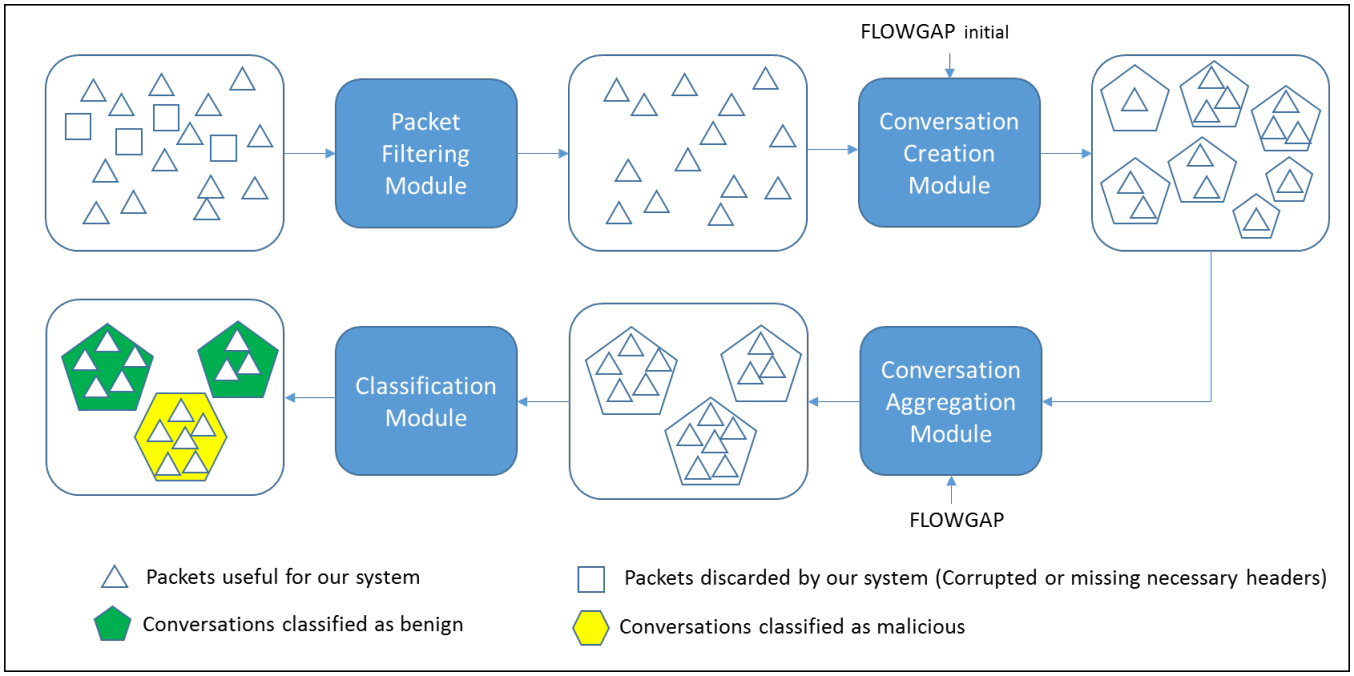


Figure 4: Implementation of PeerShark [9]

To avoid over-fitting in classification multiple machine learning algorithms were applied with 10 fold cross-validation. In [9] only three of them are presented: Decision trees, Boosted REP trees and Bayesian Networks.
Decision trees are simple classification algorithm which works very well. The J48 implementation of decision trees were used. Despite trees complex structure there is a high probability of over-fitting. High result on training data would not be as well high in real world examples.
For robustness and better performance in generalize cases Boosted REP (Reduced-error pruning) trees were used. Maximum depth of trees is limited to 8. That causes worse performance on train set but the model is more generalized. To improve accuracy of each classifier Boosting presented in [2] were used. At the end the AdBoost meta-classifier of Weka [4] with 10 REP trees were used.
The third described approach for classify data, are Bayesian networks [1]. Their main advantage is that they work well with missing data and outliers in the data.
Total accuracy of described algorithms is shown in figure 5. It is mesaured on the data prevously described. We can estimate that total accuracy of Peershark is above 95 %.

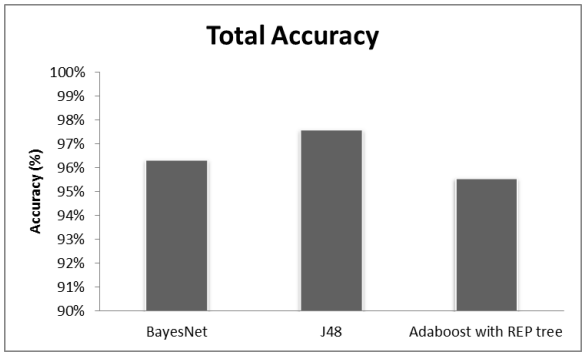


Figure 5: Total accuracy of PeerShark [9]

3.6 Possible problems

Peershark is meeting some problems in its performance. One of them is that PeerShark is multi-classifier which always classify a P2P application in one of the four classes - the same as in training phase. That doesn't work well with unknown or new P2P applications. The solution proposed for this problem is to add an 'unknown' class label. But the solution is obviously not perfect.
Next problem is that if two P2P applications are running between same hosts and one of them is benign and the other one is malicious, Peershark will not differentiate those two applications. They will be considered as one conversation. The answer on this problem is that the probability of this event is very low.
Another problem are of course smarter botnets which may arise in future. Botnets with noise in communications, botnets with benign-like activities. PeerShark could not detect such botnets because its main mechanism is to observe characteristics of conversations. If healthy and malicious conversations would behave alike the mechanism would classify them wrong. The proposed solution is that such botnets are more likely to be detected by network masters, because the volume of payload will not be low.
Next botnets can change their 'high-duration' conversation and despite that PeerShark would not detect them. But utility of botnet without communication between peer is questionable.
Another doubt about PeerShark is that if peer A is communicating with peer B with healthy application and B would have an unhealthy conversation with peer C, all A, B and C will be treated as malicious. It was discovered that this

behavior is not really bad, because it is very likely, that the healthy one will be infected in future.
The last weakness which was introduced is Packet Filtering Module. It was discovered that some of the useful information is lost when all packets with payload zero are excluded from future treatment.

3.7 Future work

In future the authors of [9] want to improve PeerShark. Especially the success of classifying benign P2P application which is still lower than success of classifying malicious P2P applications. Only two benign applications were used in study. They want to expand their work in more datasets and applications. The problems addressed in previous subsection are intended to be solved by an already developing approach where flow-based and conversation-based approaches are mixed together.

4. CONCLUSION

The proposed software for P2P botnet detection is using new conversation based approach. It was shown, that approach is successful and useful.

5. ADDITIONAL AUTHORS

6. REFERENCES

[1] R. R. Bouckaert. Bayesian network classifiers in weka for version 3-5-7. *Artificial Intelligence Tools*, 11(3):369–387, 2008.

[2] Y. Freund, R. E. Schapire, et al. Experiments with a new boosting algorithm. In *ICML*, volume 96, pages 148–156, 1996.

[3] G. Gu, R. Perdisci, J. Zhang, W. Lee, et al. Botminer: Clustering analysis of network traffic for protocol-and structure-independent botnet detection. In *USENIX Security Symposium*, volume 5, pages 139–154, 2008.

[4] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

[5] H. Hang, X. Wei, M. Faloutsos, and T. Eliassi-Rad. Entelecheia: Detecting p2p botnets in their waiting stage. In *IFIP Networking Conference, 2013*, pages 1–9. IEEE, 2013.

[6] M. Iliofotou, H.-c. Kim, M. Faloutsos, M. Mitzenmacher, P. Pappu, and G. Varghese. Graph-based p2p traffic classification at the internet backbone. In *INFOCOM Workshops 2009, IEEE*, pages 1–6. IEEE, 2009.

[7] M. Iliofotou, P. Pappu, M. Faloutsos, M. Mitzenmacher, S. Singh, and G. Varghese. Network monitoring using traffic dispersion graphs (tdgs). In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 315–320. ACM, 2007.

[8] S. Nagaraja, P. Mittal, C.-Y. Hong, M. Caesar, and N. Borisov. Botgrep: Finding p2p bots with structured graph analysis. In *USENIX Security Symposium*, pages 95–110, 2010.

[9] P. Narang, S. Ray, C. Hota, and V. Venkatakrishnan. Peershark: detecting peer-to-peer botnets by tracking conversations. In *Security and Privacy Workshops (SPW), 2014 IEEE*, pages 108–115. IEEE, 2014.

[10] R. Puri. Bots & botnet: An overview. *SANS Institute*, 3:58, 2003.

[11] B. Rahbarinia, R. Perdisci, A. Lanzi, and K. Li. Peerrush: Mining for unwanted p2p traffic. *Journal of Information Security and Applications*, 19(3):194–208, 2014.

[12] R. Schoof and R. Koning. Detecting peer-to-peer botnets. *University of Amsterdam*, 2007.

[13] S. Sen, O. Spatscheck, and D. Wang. Accurate, scalable in-network identification of p2p traffic using application signatures. In *Proceedings of the 13th international conference on World Wide Web*, pages 512–521. ACM, 2004.

[14] J. Zhang, R. Perdisci, W. Lee, U. Sarfraz, and X. Luo. Detecting stealthy p2p botnets using statistical traffic fingerprints. In *Dependable Systems & Networks (DSN), 2011 IEEE/IFIP 41st International Conference on*, pages 121–132. IEEE, 2011.

Cloud Forensics: iCloud

Jaka Cikač
Faculty of Computer and Information Science
University of Ljubljana
Slovenia
jaka.cikac@gmail.com

Anže Schwarzmann
Faculty of Computer and Information Science
University of Ljubljana
Slovenia
anze.schwarzmann@gmail.com

ABSTRACT

Cloud forensics has been a hot topic since the increase in popularity of cloud storage. There are already numerous cloud storage providers, one of them being Apple with their iCloud service. In this work we update the research carried out by Oestreicher in 2013 using iCloud 2013 and Mac OS X Maverics. The aim of that research was to develop a forensically robust method for iCloud data acquisition. This was done by checking the MD5 hash values and timestamps on the synchronized data via iCloud. We carry out similar research using a newer version of iCloud (2015) along with the new Mac OS X Yosemite (2015). We show that in two years time not much has changed and we obtain similar results. There are however some differences, one of them being Apple's newest preinstalled Photos application, which produces matching timestamps as well as MD5 hash values as data is synchronized. Timestamps and MD5 hash values still mismatch for the preinstalled applications (except for Photos and Calendar) as we compare the original and downloaded data.

Categories and Subject Descriptors

D.4 [**Operating System**]: *Security and protection, Storage management, Communications Management*

Keywords

cloud data, cloud forensics, iCloud, forensic soundness, MD5, metadata

1. INTRODUCTION

Digital cloud forensics has been a hot topic in the previous years since cloud storage usage has jumped with popular services like Dropbox, Amazon, Apple's iCloud, Google Drive and others. Most cloud services offer synchronization features between (mobile and desktop) devices, easy access with a web interface and much more. When digital forensic experts are interested in retrieving data and presenting it at court as evidence, there are a few problems. Some have already been explored to some extent but need further investigation as cloud service providers update their services.

The first issue with cloud data is that we don't necessarily have the files synced to the device in investigation, this requires different approaches to data retrieval. The second issue is assuring the integrity of the retrieved data, as it is synced between the desktop computer and the servers. Data might be used on some mobile devices as well. How does this affect the metadata such as timestamps and checksum calculated? Will it remain unchanged?

Many researchers[1] have already investigated this problem and have found that there are differences between cloud providers and differences between the types of data (or which application data) is being dealt with. The purpose of this article is to further investigate how Apple's iCloud handles metadata and checksums as data is synced across devices. This has already been done in previous research, where the author has demonstrated and extensively tested the synchronization of data between two Mac OS X Maverics (2013) machines with iCloud (2013).

In this article we have used the newer version of both OS X Yosemite (2015) and iCloud (stable release, 30th of April, 2015) to see what happens with data integrity and timestamps of data that is being synchronized. Metadata is sometimes the most important part of a forensic analysis since it can be used to determine who was where and did what at certain times.

1.1 Cloud Forensics: An Overview

Many researchers have already explored digital forensics of cloud data. One of such conducted research was done on Dropbox, Google Drive, and Microsoft SkyDrive. Authors of [5] have used known forensic tools such as FTK, EnCase and XRY to analyze metadata and timestamps of data acquired from the mentioned cloud service providers. They have found that data integrity was not an issue. That was not the case with metadata and timestamp as these were changed by each cloud service provider. These methods are however not relevant for iCloud data acquisition and examination, since no virtual image of the hosting server can be created. iCloud data is downloaded from the internet, when requested by investigators.

[1]Darren Quick, Kim-Kwang Raymond Choo, Josiah Dykstra, Alan T Sherman, Hyunji Chung, Jungheum Park, Sangjin Lee, Cheulhoon Kang, Ben Martini, ...

A popular cloud service provider is Amazon. Research of [2] have deemed them the most digital forensics friendly as Amazon "exports the data requested to an external hard drive, maintains chain-of-custody, and ships the drive directly to the requester. Along with the drive, Amazon also includes date and time of the transfer, location on the storage device, MD5 checksum and number of bytes" [2]. Researchers have determined that data from Amazon has remained unchanged and their hash values were matching [5].

Other researchers have focused more on the residual artifacts left behind by file synchronization rather than on the data itself [1]. They have found that useful artifacts are left behind, especially when using the web browser, where data resides in the temporary internet files and internet history areas. This is not the primary focus of this paper and we are therefore not interested in such residual data.

Some cloud service (more specifically storage) providers have native applications that are installed to the client devices. These applications store various files on the device and enable data synchronization to the device. This is also how iCloud operates and enables digital investigators to obtain the data and their metadata directly from client devices.

1.2 About iCloud

Apple's iCloud has grown increasingly more popular along with Apple's wide range of devices since the initial iCloud announcement in 2011. Since then Apple has added numerous features to iCloud along with extensive integration of it's services into every product available by Apple. Further more, Apple has ensured that synchronization between their mobile and desktop devices is as seamless as possible. This was done with the introduction of iCloud Drive in 2014 which is integrated into the operating system Mac OS X[2]. The iCloud service offers cloud storage, iCloud Drive, numerous applications (such as Find My Mac / iPhone, online collaboration suite iWork - which includes Pages, Keynote and Numbers; Photo Stream and others[3]), a web interface, device backups, iTunes Match and API availability that allows developers to develop their own applications that use iCloud Drive as a synchronization and storage medium between different devices. Apple has also released iCloud for Windows, which enables Windows users to access iCloud Drive data from a Windows machine.

As reported by Apple, there are more than 320 million iCloud accounts, that have already generated over 900 million iMessages, over 125 billion photo uploads and more[4]. iCloud is an enormous resource for digital evidence.

When it comes to data retrieval for forensic purposes, the data is encrypted on the Apple's servers, however Apple keeps the master key that is used if so required by government agencies[3]. When it comes to data security Apple is serious about preventing unauthorized access to user data and has installed many precautions in place to do so[5].

[2]Another addition to boost cloud storage usage was the low pricing for the amount of storage.

[3]iCloud keychain, Email, Contants, Calendars, Find My Friends, Back to my Mac, Notes

[4]This data is from 2013, since Apple has not released an updated report on iCloud users statistics. Link: Apple reports 320 m iCloud account.

2. METHODOLOGY

Research conducted in this paper aims to update the research of [4], therefore the methodology is quite similar. Virtual machines have been created using virtualization software. On each of the virtual machine (original-side version and examiner-side version) applications in question (namely Pages, Numbers, and Keynote) were installed along with a clean install of Mac OS X Yosemite 10.10 (2015). Several actions were taken to produce data in a newly registered iCloud and iCloud Drive account. After synchronization of data on the examiner-side virtual machine an in-depth examination was performed.

For every data synchronized, MD5 hashes were computed to check the integrity of data. Data locations stored on the host device were recorded. Contents of data was inspected to ensure, that the content did not change and metadata was checked for timestamps. The acquired results were then compared to that of the original research done by [4].

2.1 Initial configuration and data acquisition

The setup was done on a Macbook Pro Retina late 2013. A virtual machine was setup using VMware Fusion Professional version 7.1.1. After a clean install of OS X Yosemite 10.10.3 on a virtual machine (to represent the original machine for examination - VM1) along with additional applications (Pages, Numbers and Keynote - software versions are listed in Table 1) a snapshot (1) was made and used as a clone for another virtual machine (to represent the examiner side version - VM2). A new iCloud account was created for the purpose of this research and used to sign in on VM1. After the iCloud and iCloud Drive setup was complete, data was added and various actions taken for different application as listed in Table 1. When synchronization with Apple servers was completed, VM1 was powered off and another snapshot (2) was taken. This completed all actions required on the original-side virtual machine.

Examiner-side virtual machine VM2 was then started and another snapshot (3) was taken. Next, the log-in procedure to iCloud and enabling the iCloud Drive along with synchronization of pre-installed applications data (eg. Contants, Reminders, ...) was completed and another snapshot (4) taken. In order to make certain applications synchronize (namely Pages, Keynote, Numbers and Photos) it was required to open them. After synchronization took place the examiner virtual machine was shut down and a final snapshot (5) was taken. This is a similar approach as it was taken in [4].

3. ANALYSIS

After successful data acquisition, both virtual machines were powered off and analysis was performed. The first step was discovering file locations as the user is signed into iCloud in Mac OS X. A tool Visual Differ v. 1.6.4 was used to compare

[5]There is still a possibility of a man in the middle attack. This is what Elcomsoft exploits in order to decrypt the downloaded iPhone backup stored on the iCloud, thus avoiding the encryption key.

Table 1: Applications used.

Application	*Action taken*
Contacts (9.0 (1579))	Created a new contact with name and phone information.
Mail (8.2 (2098))	E-mail from Apple received and e-mail sent.
Calendar (8.0 (2034.9))	Created a new event.
Reminders (3.0 (356.4))	Created a new reminder.
Safari (8.0.6 (10600.6.3))	Opened Safari web browser, visited URL: publicdomain-pictures.net, bookmarked and downloaded a photo, which was saved to Photos.
Photos (1.0 (209.52.0))	Opened while saving picture from Safari, dragged to iCloud Photo Sharing.
Pages (5.5.3 (2152))	New document created and saved to iCloud Drive Pages applications folder.
Numbers (3.5.4 (2150))	New document created and saved to iCloud Drive Numbers folder.
Keynote (6.5.3 (2151))	New document created and saved to iCloud Drive Keynote folder.

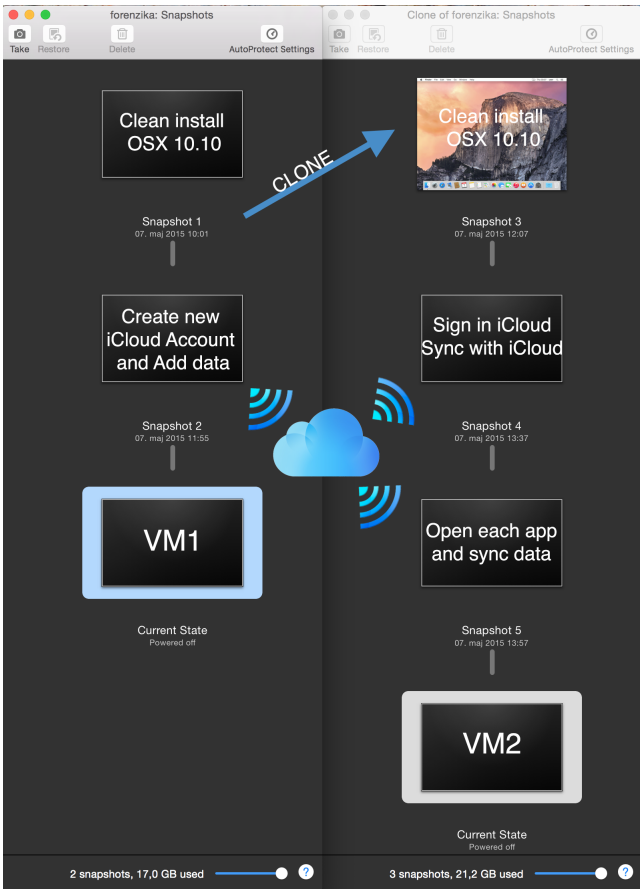


Figure 1: Snapshots made for file change comparison.

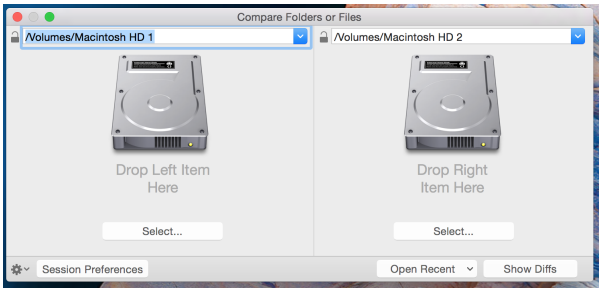


Figure 2: Virtual hard drive locations entered into Visual Differ v. 1.6.4.

which files and their respective file paths have been added, removed or deleted between snapshot 1 and snapshot 2. Visual Differ is able to visualize such differences and conveniently color codes changes by simply entering (virtual) hard drive locations as seen on Figure 2. This allows us to discover iCloud file locations on the host device.

Since VMWare Fusion does not store snapshots as virtual hard drives required by Visual Differ, a conversion needs to be done from *.vmdk* (used by VMWare Fusion) and *.dmg* (used by Mac OS X to represent virtual hard drives). This procedure is described in [4], it can however be done without the use of AccessData's Forensic Toolkit (FTK) Imager. It can be done using a Terminal command[6] to obtain a *.raw* file, which can be manually converted to *.dmg* simply by changing the extension name[7]. To ensure read only of the virtual hard drives we can also choose to lock the generated images. Images need to be mounted (double-clicking them will do that in OS X) and everything is ready for examination.

3.1 Discovering file locations

After we successfully run Visual Differ with two snapshots for comparison we can use a feature that displays mismatched files, which enables us to quickly find added, changed or deleted files. Newly added files are colored blue and changed files are colored in red. Results of comparing snapshot 1 with snapshot 2 are displayed on Figure 3. The corresponding file locations are listed in Table 2[8].

There are almost no differences between iCloud 2013 in combination with Mac OS X Maverics and iCloud 2015 in combination with Mac OS X Yosemite when it comes to file paths. The only difference is the new application Photos (2015), which is meant to replace iPhoto (2002- 2015) and even here the main difference is in the name of the library (Photos Library instead of iPhoto Library).

3.2 Data integrity - MD5 and timestamps

After discovering file locations we were able to calculate the MD5 hash values of synchronized data. This was done

[6]qemu-img convert -f vmdk -O raw Documents/Virtual Machines.localized/forenzika.vmwarevm/Virtual Disk-000002.vmdk Documents/snapshot2.raw

[7]Note that one can convert directly to *.dmg* using quemu-img convert, thus omitting one step.

[8]Note that Reminders are stored as Calendar application data, thus having the same file path.

Table 2: File paths to data stored on the device.

Application	*File paths*
Numbers	/Users/user/Library/Mobile Documents/com~apple~Numbers/Documents/
Pages	/Users/user/Library/Mobile Documents/com~apple~Pages/Documents/
Keynote	/Users/user/Library/Mobile Documents/com~apple~Keynote/Documents/
Safari	/Users/user/Library/Safari/
Contacts	/Users/user/Library/Application Support/AddressBook/Sources/
Mail-inbox	/Users/user/Library/Mail/V2/AosIMAP-forenzika/INBOX.mbox/
Mail-sent	/Users/user/Library/Mail/V2/AosIMAP-forenzika/Sent Messages.mbox/
Reminders	/Users/user/Library/Calendars/
Calendar	/Users/user/Library/Calendars/
Photos	/Users/user/Pictures/Photos Library.photoslibrary/

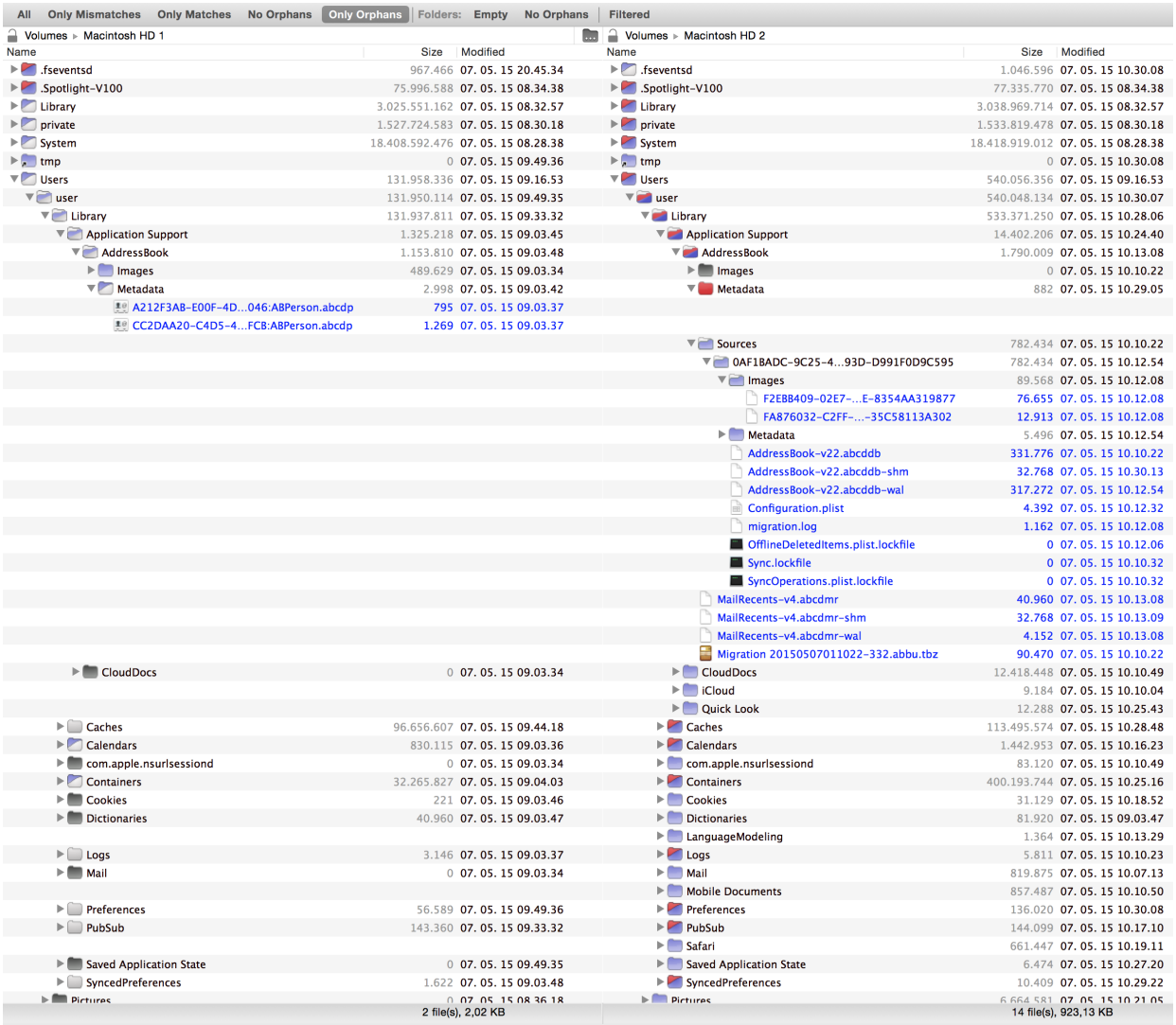


Figure 3: Displaying differences of snapshot 1 and snapshot 2.

using a Terminal command *md5sum*[9]. To obtain accurate timestamps for modified, accessed and created metadata, another Terminal command was used, namely *mdls*, which displays all metadata that the file contains. This command displays times in UTC time[10]. We have collected all MD5 hash values and timestamps in GMT[11] time format in Figure 5. This allows us to compare the obtained table with a table produced by the original researcher [4]. Both tables are color-coded with the same colors[12] for easy visual comparison.

MD5 analysis of original-side and examiner-side data shows mixed results, just as in the original research done by [4]. There are some slight differences however. We found that there are matching hash values with Mail-inbox, Calendar, Pages, Numbers and Keynote, Photos (note that Pages, Numbers and Keynote are non-preinstalled applications and preinstalled applications are Mail-inbox, Calendar, Photos). First difference with [4] is found with the Calendar and Photos applications, which now have a matching MD5 value, despite belonging to the preinstalled group. Reminders, Safari, Contacts and Mail-sent have mismatching MD5 values (note that these are all preinstalled applications). Therefore there are some changes between our research and the one done by [4].

3.3 Metadata analysis

Due to differences between preinstalled and non-preinstalled applications we divide this part of analysis into two groups.

- **Preinstalled applications**: Contacts, Mail-inbox, Mail-sent, Calendar, Reminders, Safari, Photos.
- **Non-preinstalled applications**: Pages, Keynote, Numbers.

3.3.1 Non-preinstalled applications

Timestamp analysis for non-preinstalled applications is visible on Figure 5. Note that Photos application is preinstalled in Mac OS X Yosemite as opposed to Mac OS X Maverics (as in [4]). In our results accessed, modified and created timestamp are not matching in any of these applications. [4] had matching modified and created timestamps for Pages and Numbers. Another difference is that in our results all timestamps have examiner-side time variation (depicted in orange)[13] as opposed to [4] where some timestamps had original-side time variation (depicted in yellow).

To summarize the differences, our results have less matching timestamps and all of them have examiner-side time variations at the time when applications were opened and files synchronized.

3.3.2 Preinstalled applications

Results of [4] showed that all timestamps of preinstalled applications were not matching. They all corresponded to the time when they were downloaded to the examiner machine.

[9]example: md5sum snapshot2.dmg
[10]UTC +0000
[11]GMT +0200
[12]As much as we were able to reproduce the original color.
[13]This means that the examiner-side timestamp is later in time as the original side.

Our results are different for Photos, which is now a pre-installed application. Surprisingly all timestamps (along with the MD5 hash value) were matching. For the rest of the preinstalled applications timestamps were mismatching. All timestamps corresponded to the time when they were downloaded to the examiner machine except for Mail-inbox, which had timestamps of when the e-mail was received on the original machine.

4. CONCLUSION

This work aims to update the research carried out by [4]. Results show that not much has changed when it comes to timestamp analysis and MD5 hash values analysis in two years time (from 2013 to 2015). Non-preinstalled applications still have matching MD5 hash values, their timestamps however remain modified to the time of download on the examination machine.

Preinstalled applications (except for Photos and Calendar) still have modified timestamps and their hash values are not matching, as has been the case in [4].

Apple's newest application Photos might indicate that there is effort to assure that timestamps and MD5 values do not change as they are synchronized between devices, but this is just an assumption and further research will be required as Apple updates the iCloud service along with Mac OS X and it's applications.

This method still remains forensically sound as far as data acquisition goes (as pointed out in [4]). Since MD5 values for non-preinstalled applications were matching data integrity remains established. This is however not the case with pre-installed applications (except for Photos and Calendar) and further research is required to find an answer as to why there are mismatched MD5 values and timestamps.

5. REFERENCES

[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] J. Dykstra and A. T. Sherman. Acquiring forensic evidence from infrastructure-as-a-service cloud computing: Exploring and evaluating tools, trust, and techniques. *Digital Investigation*, 9:S90–S98, 2012.

[3] C. Foresman. Apple holds the master decryption key when it comes to icloud security, privacy, Apr. 2012.

[4] K. Oestreicher. A forensically robust method for acquisition of icloud data. *Digital Investigation*, 11:S106–S113, 2014.

[5] D. Quick and K.-K. R. Choo. Forensic collection of cloud storage data: Does the act of collection result in changes to the data or its metadata? *Digital Investigation*, 10(3):266–277, 2013.

Color Key:	Original	MD5 Mismatch	Examiner-Side Time Variation
	Examiner Download	Match	Original-Side Time Variation

Application Name/File Path	Created	Accessed	Modified	MD5
Contacts				
/Users/user/Library/Application Support/AddressBook/Sources/BDC8A8D1-4D5E-46D3-8746-CF9DC4D39AC4/AddressBook-v22.abcddb-wal	12/5/2013 3:38:28 PM	12/5/2013 3:40:41 PM	12/5/2013 3:40:41 PM	f9dea3d92ec684cbbb1fd62ccd787312
/Users/user/Library/Application Support/AddressBook/Sources/D2907400-687E-4FDE-B8F4-4B27C0E49878/AddressBook-v22.abcddb-wal	12/5/2013 3:49:39 PM	12/5/2013 3:51:31 PM	12/5/2013 3:51:24 PM	a8051b97be968ffd9c54dd0866b135a4
Mail - Inbox				
/Users/user/Library/Mail/V2/AosIMAP-cloudstone2000/INBOX.mbox/BA27AA0A-C69C-4AC7-B8A8-C517FF666DB3/Data/Messages/4.emlx	12/5/2013 3:39:41 PM	12/5/2013 3:39:41 PM	12/5/2013 3:39:41 PM	6cc2cb1018d246483bc343361172d548
/Users/user/Library/Mail/V2/AosIMAP-cloudstone2000/INBOX.mbox/FF972C17-0FCE-4B58-B948-E6FACAF3CBCE/Data/Messages/3.emlx	12/5/2013 3:51:09 PM	12/5/2013 3:51:09 PM	12/5/2013 3:51:09 PM	6cc2cb1018d246483bc343361172d548
Mail - Sent				
/Users/user/Library/Mail/V2/AosIMAP-cloudstone2000/Sent Messages.mbox/BA27AA0A-C69C-4AC7-B8A8-C517FF666DB3/Data/Messages/5.emlx	12/5/2013 3:39:41 PM	12/5/2013 3:39:41 PM	12/5/2013 3:39:41 PM	fe10c6d7a33829c2af5d6f800edddf7f
/Users/user/Library/Mail/V2/AosIMAP-cloudstone2000/Sent Messages.mbox/FF972C17-0FCE-4B58-B948-E6FACAF3CBCE/Data/Messages/4.emlx	12/5/2013 3:51:12 PM	12/5/2013 3:51:12 PM	12/5/2013 3:51:12 PM	89a5aea52c9afda1164106a816a87336
Calendar (Individual Events)				
/Users/user/Library/Calendars/F87CD4FA-5B1D-4E6E-B5D7-3F0AB61E5C50.caldav/A8F3EBFA-F7FC-4026-86E3-5DD05FD96E9E.calendar/Events/93F37727-B90B-4B04-90CF-36082DE392F3.ics	12/5/2013 3:40:50 PM	12/5/2013 3:40:50 PM	12/5/2013 3:41:01 PM	5a23a8f5a48ba07ff801950c0e6e9ee2
/Users/user/Library/Calendars/74FA4B91-AA8C-4F7C-A84D-8028C1588220.caldav/E4F71911-C68F-488F-A6FC-817A31EB4DBF.calendar/Events/93F37727-B90B-4B04-90CF-36082DE392F3.ics	12/5/2013 3:49:49 PM	12/5/2013 3:49:49 PM	12/5/2013 3:49:49 PM	5921e1be2cb51bf9d65ddb41c16e1310
Reminders				
/Users/user/Library/Calendars/F87CD4FA-5B1D-4E6E-B5D7-3F0AB61E5C50.caldav/5EA1CEE1-7ECB-47B3-9F5E-86EF7BC8BD71.calendar/Events/AD6FD050-9ECE-43D2-978F-C33DEB8B83D5.ics	12/5/2013 3:41:21 PM	12/5/2013 3:41:21 PM	12/5/2013 3:41:22 PM	4591c02173038908a63e9c8ec4af42dd
/Users/user/Library/Calendars/74FA4B91-AA8C-4F7C-A84D-8028C1588220.caldav/9E61D143-8996-465D-8751-F65BAC2C0C9D.calendar/Events/AD6FD050-9ECE-43D2-978F-C33DEB8B83D5.ics	12/5/2013 3:49:50 PM	12/5/2013 3:49:50 PM	12/5/2013 3:49:50 PM	4591c02173038908a63e9c8ec4af42dd
Safari Bookmarks				
/Users/user/Library/Safari/Bookmarks.plist	12/5/2013 3:42:40 PM	12/5/2013 3:42:40 PM	12/5/2013 3:42:40 PM	3528cc9800bb14b8b8fdfd0d036ada06
/Users/user/Library/Safari/Bookmarks.plist	12/5/2013 3:49:43 PM	12/5/2013 3:49:43 PM	12/5/2013 3:49:43 PM	91df269b4e48b4d8dca41369c1f06429
iPhoto				
/Users/user/Pictures/iPhoto Library.photolibrary/Masters/2013/12/05/20131205-154255/red-rosebud.jpg	12/5/2013 3:42:47 PM	12/5/2013 3:42:47 PM	12/5/2013 3:42:47 PM	0641687b2f509ab4fdc6ca7b506077a1
/Users/user/Pictures/iPhoto Library.photolibrary/Masters/2013/12/05/20131205-155248/red-rosebud.jpg	12/5/2013 3:42:47 PM	12/5/2013 3:52:48 PM	12/5/2013 3:42:47 PM	0641687b2f509ab4fdc6ca7b506077a1
Pages				
/Users/user/Library/Mobile Documents/com~apple~Pages/Documents/Pages Document.pages/Index.zip	12/5/2013 3:44:20 PM	12/5/2013 3:44:22 PM	12/5/2013 3:44:20 PM	a7237ea6a0e8ca28c7a372db8cf5c08e
/Users/user/Library/Mobile Documents/com~apple~Pages/Documents/Pages Document.pages/Index.zip	12/5/2013 3:44:20 PM	12/5/2013 3:44:20 PM	12/5/2013 3:44:20 PM	a7237ea6a0e8ca28c7a372db8cf5c08e
Numbers				
/Users/user/Library/Mobile Documents/com~apple~Numbers/Documents/Numbers Document.numbers/Index.zip	12/5/2013 3:45:00 PM	12/5/2013 3:45:03 PM	12/5/2013 3:45:00 PM	3154d40656fc4042be20e99e65525755
/Users/user/Library/Mobile Documents/com~apple~Numbers/Documents/Numbers Document.numbers/Index.zip	12/5/2013 3:45:00 PM	12/5/2013 3:45:00 PM	12/5/2013 3:45:00 PM	3154d40656fc4042be20e99e65525755
Keynote				
/Users/user/Library/Mobile Documents/com~apple~Keynote/Documents/Keynote Document.key/Index.zip	12/5/2013 3:45:41 PM	12/5/2013 3:45:45 PM	12/5/2013 3:45:42 PM	2e95e189841ae61771b2ecd98bfb32ed
/Users/user/Library/Mobile Documents/com~apple~Keynote/Documents/Keynote Document.key/Index.zip	12/5/2013 3:45:42 PM	12/5/2013 3:45:42 PM	12/5/2013 3:45:42 PM	2e95e189841ae61771b2ecd98bfb32ed

Figure 4: Table of data locations, MD5 values and timestamps, composed by [4].

Application Name/File Path	Created	Accessed	Modified	MD5
Contacts:				
/Users/user/Library/Application Support/AddressBook/Sources/0AF1BADC-9C25-48CB-B93D-D991F0D9C595/AddressBook-v22.abcddb-wal	7.5.2015 10:10:22	7.5.2015 10:12:54	7.5.2015 10:12:54	dc7d7dbb3272b96f6cf8bf40f0b06ead
/Users/user/Library/Application Support/AddressBook/Sources/7E6DA095-C6D0-44A6-AC15-8CFF9B81F183/AddressBook-v22.abcddb-wal	7.5.2015 12:33:27	7.5.2015 12:35:13	7.5.2015 12:35:13	183e2245b9783315746bf5df2070d335
Mail-inbox:				
/Users/user/Library/Mail/V2/AosIMAP-forenzika/INBOX.mbox/59D3FBAF-77DC-47AD-9C8F-D2137801D00D/Data/Messages/3.partial.emlx	7.5.2015 21:44:40	7.5.2015 21:44:41	7.5.2015 21:44:41	2716417c0d529ae6279a2a2749cba886
/Users/user/Library/Mail/V2/AosIMAP-forenzika/INBOX.mbox/4447907E-6E9E-40A9-BD37-1A244FA285BC/Data/Messages/1.partial.emlx	7.5.2015 12:37:00	7.5.2015 12:37:01	7.5.2015 12:37:01	2716417c0d529ae6279a2a2749cba886
Mail-sent:				
Users/user/Library/Mail/V2/AosIMAP-forenzika/Sent Messages.mbox/59D3FBAF-77DC-47AD-9C8F-D2137801D00D/Data/Messages/5.emlx	7.5.2015 21:44:38	7.5.2015 21:44:41	7.5.2015 21:44:41	75956b747a56bbb0eabacd3b6af15739
/Users/user/Library/Mail/V2/AosIMAP-forenzika/Sent Messages.mbox/4447907E-6E9E-40A9-BD37-1A244FA285BC/Data/Messages/3.emlx	7.5.2015 21:48:24	7.5.2015 21:48:24	7.5.2015 21:48:24	5d7aa94d62c2da82eee7895684399685
Calender:				
/Users/user/Library/Calendars/1DBF8361-D60B-41BB-BA9B-5D02ED7BCC80.caldav/199EEFD7-1A6E-43CC-AF4A-48C573791BF5.calendar/Events/C8FE68F8-3989-4CC7-84FA-51FB137724DC.ics	7.5.2015 10:16:31	7.5.2015 10:16:31	7.5.2015 10:16:31	fdcd9f33f5103e11e41f72ce80c9a7cf
/Users/user/Library/Calendars/C31C62DA-0A20-4681-A957-C9F6E40E6BD3.caldav/51DF1D80-3CFC-4459-8776-15600479E330.calendar/Events/C8FE68F8-3989-4CC7-84FA-51FB137724DC.ics	7.5.2015 12:33:39	7.5.2015 12:33:39	7.5.2015 12:33:39	fdcd9f33f5103e11e41f72ce80c9a7cf
Reminders:				
/Users/user/Library/Calendars/1DBF8361-D60B-41BB-BA9B-5D02ED7BCC80.caldav/9D17C549-63EF-40D2-A391-352D392CDD1E.calendar/Events/D9F9257B-077D-4B80-AE2E-0DEB38F869D4.ics	7.5.2015 10:16:47	7.5.2015 21:45:31	7.5.2015 21:45:31	8c11f1bcb0c8c3533198612f71b79166
/Users/user/Library/Calendars/C31C62DA-0A20-4681-A957-C9F6E40E6BD3.caldav/209E1162-9870-41CA-8B21-88FE7D0B0812.calendar/Events/D9F9257B-077D-4B80-AE2E-0DEB38F869D4.ics	7.5.2015 12:33:40	7.5.2015 21:48:16	7.5.2015 21:48:16	8be73a709e2cc2425947fe447d77b537
Safari bookmarks:				
/Users/user/Library/Safari/Bookmarks.plist	7.5.2015 10:19:05	7.5.2015 10:19:05	7.5.2015 10:19:05	76e22eb8ffaad4252c9cddd8c57d05c9
/Users/user/Library/Safari/Bookmarks.plist	7.5.2015 12:33:31	7.5.2015 12:33:31	7.5.2015 12:33:31	db2e2b570c73ce48976866b8e3f0e817
Photos:				
/Users/user/Pictures/Photos Library.photoslibrary/Masters/2015/05/07/20150507-082148/1-1231002924wqjW.jpg	7.5.2015 10:21:47	7.5.2015 10:21:47	7.5.2015 10:21:47	99b355f2ab18132af2da9848b3b3369a
/Volumes/Macintosh HD 2/Users/user/Pictures/Photos Library.photoslibrary/Masters/2015/05/07/20150507-103427/1-1231002924wqjW.jpg	7.5.2015 10:21:47	7.5.2015 10:21:47	7.5.2015 10:21:47	99b355f2ab18132af2da9848b3b3369a
Pages:				
/Volumes/Macintosh HD 1/Users/user/Library/Mobile Documents/com~apple~Pages/Documents/Untitled.pages/Index.zip	7.5.2015 10:24:39	7.5.2015 10:24:39	7.5.2015 10:24:39	1838742ce557ef994a95c8864a1e1a0b
/Volumes/Macintosh HD 2/Users/user/Library/Mobile Documents/com~apple~Pages/Documents/Untitled.pages/Index.zip	7.5.2015 12:33:58	7.5.2015 12:33:59	7.5.2015 12:33:59	1838742ce557ef994a95c8864a1e1a0b
Numbers:				
/Users/user/Library/Mobile Documents/com~apple~Numbers/Documents/Untitled.numbers/Index.zip	7.5.2015 10:25:09	7.5.2015 10:25:09	7.5.2015 10:25:09	1582b5d63a794c545bbb85e2b92ed70b
/Users/user/Library/Mobile Documents/com~apple~Numbers/Documents/Untitled.numbers/Index.zip	7.5.2015 12:33:55	7.5.2015 12:33:57	7.5.2015 12:33:57	1582b5d63a794c545bbb85e2b92ed70b
Keynote:				
/Users/user/Library/Mobile Documents/com~apple~Keynote/Documents/Untitled.key/Index.zip	7.5.2015 10:25:39	7.5.2015 10:25:39	7.5.2015 10:25:39	ade6c3230206c8cf5e004875df2055ac
/Users/user/Library/Mobile Documents/com~apple~Keynote/Documents/Untitled.key/Index.zip	7.5.2015 12:34:08	7.5.2015 12:34:10	7.5.2015 12:34:10	ade6c3230206c8cf5e004875df2055ac

Figure 5: Table of data locations, MD5 values and timestamps, composed by authors.

Del III

Digitalna forenzika na sistemih

Towards a Forencsic-Aware Database Solution:

Using a secured Database Replication Protocol and Transaction Management for Digital Investigations

Matic Volk
Fakulteta za računalništvo in informatiko
Večna pot 113
1000 Ljubljana
mv0281@student.uni-lj.si

Nejc Zupan
Fakulteta za računalništvo in informatiko
Večna pot 113
1000 Ljubljana
nz9595@student.uni-lj.si

Povzetek

Podatkovne baze so danes zelo razširjene in vsebujejo tako veliko količino informacij, da bi bilo življenje brez njih nepredstavljivo. Ker pogosto vsebujejo osebne podatke in na splošno informacije, ki so privatne narave, podatkovne baze vsebujejo sisteme za zaščito. Klub vsemu, pa lahko pride do zlorabe, kraje ali prirejanja podatkov. Podatki pridobljeni v forenzični analizi so pomembni v naslednih primerih. Večino podjetji skrbi za podatke svojih uporabnikov in je inforamcija ali so bili podatki razkriti, oziroma na kakršen koli način kršena privatnost uporabnikov velikega pomena. V ta namen se potrebuje orodja s katerimi se lahko preveri, oziroma dokaže pristnost podatkov in ali je prišlo do zlorabe sistema. Drug primer je analiza napadov in ugotavljanje šibkosti sistema, ki jih je napadalec izkoristil, ter bodoča preprečitev.

V ta namen bomo predstavili rešitev forenzično zavednega sistema za upravljanje podatkovnih baz, ki uporablja interne strukture za podvajanje in transakcije kot osnovo za preiskavo in za rekonstrukcijo podatkov v forenzični analizi. Velika prednost predstavljene metode je, da za analizo ne potrebujemo pregledovati dodatnih dnevniških zapisov. Prav tako metoda zagotavlja pristnost dokazov in utrdi dokazno verigo skrbništva. Za lažjo oceno je predstavljen tudi prototip izvedbe v MySQL podatkovni bazi in celovita ocena varnosti, glede na najbolj relevantne napade.

Ključne besede

InnoDB, MySQL, digitalna forenzika, podvajanje, transakcijski sistem, sistem za upravljanje podatkovnih baz

1. UVOD

Naloge sistema za upravljanje s podatkovno bazo [1] so shranjevanje podatkov, interakcija z uporabnikom, podpora poizvedovalnega jezika, administracija baze in mnoge druge.

[1]DBSM - Database Management System

Naloge na katere se osredotoča obravnavan članek [4] so upravljanje s transakcijami, povrnitev prejšnjega stanja (*rollback*), okrevanje po sesutju in tudi bolj napredne kot so podvajanje in podpora gručne arhitekture. Njihov glavni namen je zagotavljanje celovitosti in konsistentnosti podatkov. Te metode za samo delovanje uporabljajo podatkovne strukture in protokole, ki so namenjene za uporabo izključno znotraj sistema.

Glede na razširjenost podatkovnih baz in veliko količino vsebovanih podatkov so te pogosto vključene v forenzično preiskavo in dober vir informacij. Standardizirane forenzične metode omogočajo forenziku lažjo pridobitev informacij brez preučevanja celotnega aplikacijskega sloja.

Kljub uporabi forenzičnih metod pri preiskavi podatkovnih baz je bilo malo poudarka namenjenega notranjim podatkovnim strukturam. Za ilustracijo, da pogosto potrebujemo zagotovilo o pristnosti podatkov so navedena naslednja vprašanja:

- Ali so bili podatki spremenjeni v določenem časovnem obdobju in kdaj natančno?
- Ali so bili podatko spremenjeni brez uporabe SQL vmesnika?
- Kateri zahtevki so bili izvedeni v določenem času?
- Kako so se podatki spreminjali skozi čas?
- Katere transakcije so bile povrnjene v prejšnje stanje?

Članek, ki je glavni vir obravnave predstavlja prototip, ki naredi sistem za upravljanje podatkovnih baz forenzično zavednega. Za delovanje uporablja notranje podatkovne strukture, uporabljene za namene podvajanja, transakcij in okrevanja v primeru sesutja baze. Ti zapisi so uporabljeni za pravilno delovanje sistema in niso berljivi uporabnikom. Prednost predstavljenega sistema je, da za forenzično analizo in vzpostavitev revizijske sledi ni potrebna ročna obdelava administracijskih dnevniških zapisov. Sistem omogoča naknadno zaznavo nedovoljenega spreminjanja podatkov, ter zagotavlja pristnost in integriteto rekonstruiranih podatkov.

Ker sistem temelji le na transakcijskih zapisih in zapisih za podvojevanje je realizacija mogoča na vseh sistemih za upravljanje podatkovnih baz, ki so skladni z ACID modelom.

Predstavljen pa je prototip na primeru MySQL podatkovne baze.

2. SORODNA PODROčJA

Povečanje uporabe računalniških sistemov na področju preiskav povečuje potrebe računalniške forenzike na področju informacijske varnosti. V tem poglavju bodo predstavljena sorodna področja forenzike baz podatkov in varnostnega beleženja [4].

2.1 Forenzika baze podatkov

Že desetletja so beležke na nivoju operacijskega sistema glavni vir pri zbiranju dokazov v digitalni forenziki, vendar je ta metoda na nivoju podatkovnih baz nepopularna, čeprav predstavlja pomemben del podjetniških sredstev. Leta 2009 je Martin Olivier naredil pregled tedanjega stanja v primerjavi z stanjem v letu 2004 in prišel do zaključka, da se uporaba forenzike na nivoju podatkovnih baz ni izboljšala. V zadnjih letih se je število literature s področja forenzike podatkovnih baz povečala. Martin Olivier je napisal članek o vsebini metapodatkov v podatkovnih bazah in njihovo uporabo v forenzični analizi [9]. Oluwasola Mary Fasan opisuje rekonstrukcijo podatkovne baze za forenzične namene. Trenutno podano stanje podatkovne baze in beležka sprememb s pomočjo rekonstrukcijskega algoritma omogočajo pridobitev podatkov, ki so se hranili v preteklosti [8].

2.2 Uporaba notranjega sistema podatkovne baze za forenzično preiskavo

V enem od del je prikazana trajna ohranitev podatkov na nivoju datotečnega sistema z uporabo MySQL ali PostgreSQL [13]. Ta tema je podrobneje opisana v člankih [6, 12], ki opisujejo ohranitev podatkov znotraj notranje strukture upravljalnega sistema podatkovne baze. V [10] je predstavljena obnovitev shranjenih podatkov, čeprav niso več dostopni s strani SQL vmesnika. Pred kratkim so se razvile nove metode raziskovanja notranjih beležk. V [11] so avtorji razvili forenzičen pristop, ki temelji na podatkih shranjenih znotraj *redo* beležk. Prikazali so učinkovito metodo za izluščitev enostavnih INSERT, DELETE in UPDATE ukazov, skupaj z izbrisanimi informacijami. Poleg tega so analizirali strukturo *redo* beležk.

2.3 Uporaba mehanizmov za kreiranje različic in arhiviranja podatkovnih baz

Kreiranje različic in arhiviranje se uporablja za varnostne kopije, vendar je velika razlika med konceptoma. Varnostne kopije ponavadi niso ustvarjene za učinkovito pridobivanje preteklih stanj enega zapisa, ampak so kopije preteklega stanja in omogočajo pridobitev celotnega stanja. Ta metoda se uporablja predvsem v stopnjujočih (incremental) varnostnih kopijah. Raziskovalci so razvili veliko število orodij za ustvarjanje različic podatkov, ki ohranjajo celotno zgodovino s povpraševanji (queries) [7], toda arhivi in začasne podatkovne baze se redko uporabljajo v praksi. Mehanizem za kreiranje različic je ponavadi implementiran na aplikacijskem nivoju ali kot vmesni programi (middleware). Pristop v [4] se za probleme kreiranja različic in arhiviranja razlikuje od glavnega namena tega mehanizma, glede na to, da je cilj preverjanje podatkov v podatkovni bazi in ne obnovitev preteklih stanj. Zelo pomembno je, da ta pristop ne potrebuje vseh podatkov za zagotovitev preverljivega stanja.

2.4 Integriteta beležk in varno beleženje

V [2] je opisana metoda, ki skrbi za vnaprejšnjo tajnost vnosov v beležke, kar napadalcem onemogoča branje zabeležk. Poleg tega metoda detektira spremembe in brisanje zabeležk. Ta metoda ni primerna za raziskave notranjih podatkovnih struktur, saj metoda, ki jo opisuje članek potrebuje beležke, da lahko pridobi informacije pomembne za forenzične namene. Ne glede na celovitost beležk, je Marson razvil pristop za varno beleženje [5]. Ker so beležke zelo pomembne za forenzične raziskave, avtorji raziskujejo problem avtentikacije lokalno zabeleženih beležk. V primeru vdora morajo mehanizmi zavarovati zbrana sporočila proti možnim manipulacijam s strani vsiljivca. Avtorji [5] predlagajo dve glavni zahtevi: Mehanizmi morajo biti vnaprejšnje varni in sledljivi, da omogočajo preverljivost celovitosti in vrstni red zabeležk.

3. POSTOPEK VERIžNIH PRIč

Vsak upravljalni sistem podatkovnih baz mora vsebovati varne transakcije, replikacije in povrnitev v prejšnje stanje zato, da lahko zagotovi atomarnost in doslednost. Relevantni podatki so skriti v notranjih podatkovnih strukturah. Znotraj teh struktur je mogoče dodati mehanizem, ki skrbi za avtentičnost podatkov in tako neopaženo manipulira z notranjimi podatkovnimi strukturami, da veljajo kot priče pri forenzičnem raziskovanju.

3.1 Notranje podatkovne strukture

Vsi večji sistemi za upravljanje s podatkovnimi bazami uporabljajo mehanizme za ponovno pridobitev nekonsistentnih stanj, notranjih napak ali povrnitev baze v prejšnje stanje. Njihova implementacija omogoča nadaljnje razvijanje programske opreme v smeri podpisanih beležnih datotek. Podatkovne baze že sedaj uporabljajo razne beležke za beleženje stanj in dogajanja v podatkovni bazi. S tem, ko so beležke podpisane, zagotavljajo integriteto in avtentičnost shranjenih podatkov. MySQL transakcijske beležke beležijo razveljavitve in obnovitve. Namen obnovitvenih beležk je posodobitev sprememb, ki so se zgodile le v spominu in ne v trajni tabeli zapisov. Obnovitvene beležke obnovijo nekonsistentnost stanja, ki ga je povzročilo sesutje. Razveljavitvene beležke se uporabljajo za razveljavitev nedokončane transakcije ali nedokončane vrnitve v prejšnje stanje. Različni upravljalni sistemi uporabljajo različne načine shranjevanja beležk zato, da zagotavljajo avtentičnost zapisanih podatkov. MySQL podvajanje uporablja binarno beleženje. Posodobitve in spremembe se obravnavajo kot dogodki, ki so uporabljeni na podrejenim (slave) kot predpostavka, da je nadrejeni (master) "neumen" [4]. Oracle uporablja le obnovitvene beležke, ki avtomatsko skrbijo za potrditve brez uporabnika. Upravljalni sistemi ostalih podatkovnih baz uporabljajo podobne mehanizme varovanja podatkov.

3.2 Transakcijski mehanizem

Transakcijski mehanizem je v glavnem namenjen za vračanje stanja in razveljavitve, kar zagotavlja atomarnost. Vsebuje vse potrebne informacije za obnovitev prejšnjih različic podatkov, ampak ne hrani informacij o dogodkih, ki ne morejo biti razveljavljeni. Taki dogodki so spremembe strukture tabel, spremembe velikih objektov ali spremembe jezika definiranih podatkovnih struktur.

3.3 Protokol za podvajanje podatkov (Replication protocol)

Podatkovni podvojitveni mehanizem hrani podatke, da zrcali celotno bazo iz nadrejenega (master) v več podrejenih (slave) vozlišč za implementacijo redundantnih skladišč. Logična struktura hrani enake podatke kot sama baza in poleg tega še metapodatke in velike strukture kot so informacije o seji ali časovne značke. Postopek verižnih prič sloni na osnovi transakcijskega in replikacijskega mehanizma.

3.4 Avtentičnost

Glede na to, da podvojitveni in transakcijski mehanizem hranita podatke, ki so pomembni za forenzične preiskave je potrebno zagotoviti avtentičnost in integriteto teh podatkov. Paziti je potrebno tudi na neskladnost podvojenih podatkov. Za časovno analizo dogodkov se naštetim mehanizmom doda časovno značko in tako omogoči rekonstrukcijo zaporedja sprememb v podatkovnem sistemu. Za ohranitev integritete in avtentičnosti med podvojenimi podatki se uporablja verižne razpršilne funkcije nad beležkami.

3.5 Opis mehanizma za forenzično raziskavo

Formalno okolje deluje neodvisno od vira forenzičnih informacij tako, da se ga lahko uporablja na poljubnem notranjem mehanizmu in beležki. Predvidevamo da upravljalni

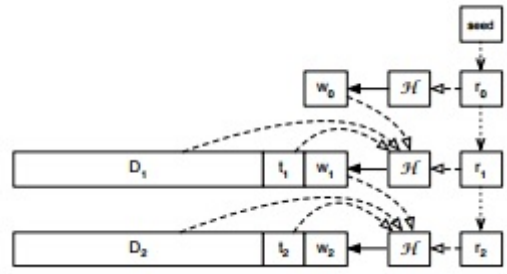


Figure 1: Razpršilna funkcija nad podatkovnimi delčki [4]

sistem podatkovne baze deluje na sistemu S. Naj bo n število podatkovnih delcev našega vira in D_i predstavlja $i-ti$ podatkovni delček shranjen na sistemu S ob času t_i. R predstavlja kriptografski naključni generator, ki je inicializiran z skrivnim semenom s, ki je zaupan zunanji instituciji T, sistemu S pa je neznan. Generator R mora generirati naključna števila, ki jih napadalec ne zmore pridobiti glede na prejšnji rezultat ali glede na seme s. Vrednost $i-te$ iteracije generatorja R je predstavljena s r_i. Naj H predstavlja enosmerno kriptografsko razpršilno funkcijo. Predpostavimo da sta H in R varna. Za zagotovitev avtentičnosti shranjenih podatkov v notranjem sistemu je vsak podatkovni delček opremljen s časovno značko in pričo w_i, ki hrani ključ (hash) podatka D_i, skupaj s trenutno generirano številko r_i, časovno značko t_i in pričo prejšnje iteracije w_{i-1} [4].

$$w_i = H(w_{i-1}||D_i||t_i||r_i) \quad (1)$$

Skupina vseh n prič w_i se imenujejo preverljive podatkovne beležke. Začetno fazo začne zunanja institucija T, ki izbere seme s in generira prvo vrednost r_0 in tako definirajo prvo pričo w_0, ki je definirana kot [4]:

$$w_0 = H(r_0) \quad (2)$$

Postopek generiranja verižnih prič je prikazan na sliki1.

3.6 Preverjanje avtentičnosti podatkov

Vsaka sprememba v datoteki se na eni strani zapiše v podatkovno skladišče, na drugi strani pa v notranje podatkovne strukture, ki se uporabljajo za upravljanje transakcij in podvajanje. Poleg tega se vsaka sprememba beleži in jo je mogoče izračunati s pomočjo prejšnjega stanja (backup) kot je prikazano na sliki 2. Sistem za preverjanje avtentičnosti

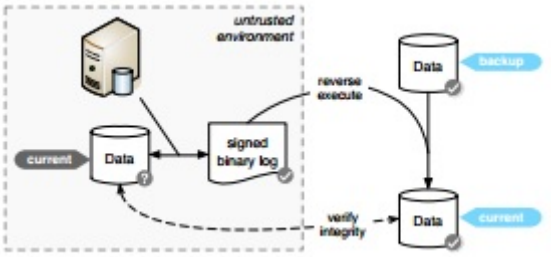


Figure 2: Preverjanje avtentičnosti podatkov [4]

bere podatkovno beležko od zadaj naprej in tako preverja pričo podatkovnega nabora z uporabo skrivnega začetnega vektorja naključnega generatorja. Če priča ni veljavna, je bil preverljivi podatkovni nabor manipuliran. Po zaključenem postopku preverjanja se preveri ali je trenutno stanje enako izračunanemu. To odkrije vse spremembe, ki so bile narejene direktno v datotečnem sistemu. Nenadomestljive razpršene vrednosti pokažejo manipulacije notranjega podatkovnega mehanizma in so lahko uporabljene za detekcijo. Ta potek dela preveri avtentičnost upravljalnega sistema podatkovne baze in avtomatsko prikaže nepooblaščene spremembe, ki jih lahko uporabimo v nadalnjih raziskavah. Pokaže se točna pozicija manipulacije ter spremembe, ki so se zgodile zaradi manipulacije.

3.7 Varnostne zahteve

Vnaprejšnja tajnost je funkcija, ki preprečuje uspešnemu napadalcu, da bi spremenil dokaze dogodkov, ki so se zgodili preden je bil sistem ogrožen. Vnaprejšna tajnost se uporablja v kriptografskih aplikacijah. Protokol, ki deluje na osnovi dogovora s ključi ima vgrajeno vnaprejšnjo tajnost ter s pomočjo nabora ključev dodeljuje sejne ključe. S tem sistemom onemogoča, da bi lahko sklepali na prejšnje sejne ključe, če je bil eden od njih najden.

Beleženje informacij mora biti varno hranjeno, da lahko služi kot forenzični dokaz na sodiščih. Razvili so se nekateri mehanizmi za arhiviranje beležk. Eden od njih je verižni mehanizem, ki je bil opisan v prejšnjem poglavju. CSPRNG je kratica za [**?**], ki mora zagotoviti naslednjim potrebam [4]:

- CSPRNG mora biti dovolj dolg in mora opraviti statistične teste naključnosti,
- CSPRNG mora zadostiti *next-bit* test, ki preverja kolikšna je uspešnost predvidenja naslednjega bita naključno generirane sekvence(ta mora biti manjša od 50%),

- CSPRNG zagotavlja, da ni mogoče rekostruirati sekvenco naključnih številk ne glede na odkrita stanja CSPRNG-ja.

Kriptografska razpršilna funkcija vključuje odpornost na ugotavljanje vhoda glede na izhod kot tudi, odpornost na trke. S razpršilno funkcijo zagotavljamo, da priče ne morejo biti skonstruirane s strani nasprotnika.

3.8 Ravnanje z zaprto kodnimi upravjalnimi sistemi

Veliko upravljalnih sistemov ne objavlja izvorne kode, zaradi takih in drugačnih interesov, zato je mehanizem za forenzične raziskave omejen na delovanje na podvojenih podatkih. Glavna razlika je v podpisovanju podvajanja in generiranju ustreznih prič, ki ni opravljeno na nadrejenem vozlišču (master node). Podvajalni promet je prestrežen in preusmerjen v transportni ali sejni nivo. Tok podatkov je razdeljen na koščke fiksne dolžine odvisne od podpisa. Priče podpisanega prometa so preverjane in ponovljene na podrejenem zaupnem okolju. Podrejeni je v originalnem stanju kot je bil v času zbiranja prometa. Podrejeni sprejme spremembe pridobljene skozi promet in primerja končno stanje s stanjem poljubnega podrejenega z nezaupanega okolja. Odstopanja med stanji se obravnavajo kot dokazi o spremembah. **Prototip deluje na točno takem upravljalnem sistemu in pripadajoči** infrastrukturi.

3.9 Primerjanje notranjih podatkovnih struktur

Kot je že bilo omenjeno so v glavnem dva glavna mehanizma, ki ustrezata za izvajanje digitalnih raziskav. Transakcijski protokol in replikacijski protokol s pomočjo različnih metod omogočata pridobitev različnih podatkov in vsak mehanizem ima svoje prednosti. Replikacijski mehanizem zbira podatke, ki so potrebni za dvojnike nadrejene (master) podatkovne baze s pomočjo podrejenih (slaves). Torej ni mogoče izpustiti nobenega objekta, ker bi sicer manjkal. Nasprotno od replikacijskega mehanizma se pri transakcijskem mehanizmu shranjujejo le manipulacijske poizvedbe nad podatki, ki omogočajo vračanje v prejšnje stanje (rollback). Veliko manipulacijskih poizvedb se zanaša na vsebino s katero je bila poizvedba izvedena, kot so:

- časovne označene poizvedbe,
- posodobitve, ki uporabljajo naključna števila,
- unikatne lastnosti ali
- avto naraščajoče lastnosti poizvedb.

Replikacijski mehanizem shranjuje vse te informacije, da jih nato posreduje podrejenim, ampak večino teh podatkov ni relevantnih za vračanje stanja (rollback), zato jih transakcijsko upravljanje ne hrani. Čeprav se v različnih podatkovnih bazah hrani enak podatek, so vidne razlike v metapodatkih. Naprimer različni upravljalni sistemi različno vstavljajo podatke v podatkovno bazo in različno struktuirajo podatke za hitrejše izvajanje in tako nastanejo vidne razlike na strukturi skladiščka. Način shranjevanja je transparenten, ampak ko želimo uporabiti napredne tehnihe za forenzične namene pride do raznih odstopanj, kar povzroča probleme. Replikacijski mehanizem ne podvaja metapodatkov na podrejenem sistemu, medtem ko transakcijski mehanizem vsebuje metapodatke za povrnitev stanja. Za delovanje mehanizma za forenzično preiskavo je potrebno notranje mehanizme ustrezno prilagoditi. Pomembno je najti vmesnik, ki bo ustrezal notranjim podatkom. Pri replikacijskem protokolu je to enostavno. Glede na to, da se spremembe prenašajo na podrejena vozlišča, je potrebno zagotoviti vsakemu upravljalnemu sistemu vmesnik, ki skrbi za zbiranje podatkov. Zaprtokodne rešitve omogočajo preslikave podatkov na podrejeno vozlišče in shranitev skupaj s pričo. Transakcijsko upravljanje je popolnoma zaprt mehanizem. Standardne baze ne potrebujejo zunanjih dejavnikov za upravljanje sistema, nimajo vmesnikov in ne spreminjajo datotečnega sistema. Za implementacijo prototipa je potrebno pridobiti izvorno kodo upravljalnega sistema podatkovne baze, kar povzroča težavo pri zaprtokodnih produktih. Zanesljivost podatkov v primeru sesutja sistema je pri transakcijskem mehanizmu bolj zanesliva kot pri replikacijskem mehanizmu, čeprav replikacijski sistem ne potrebuje takojšnjega prenosa informacije o zbranih spremembah.

4. IZVEDBA

V tem razdelku bo predstavljena izvedba razširitve, ki sistem za upravljanje podatkovnih baz [1] naredi forenzično zavednega. Razširitev je namenjena odprtokodni različici MySQL podatkovne baze, ter za svoje delovanje uporablja transakcijski in podvojitveni sistem. Pri tem predpostavlja, da sta MySQL transakcijski in podvojitveni sistem implementirana pravilno.

4.1 MySQL podvajanje

MySQL omogoča asinhron način podvajanja v *master-slave* načinu. Pri tem ima *master* strežnik vlogo vzdrževanja dnevniških zapisov imenovanih *binary log*, ki beležijo vse spremembe, ki na kakršenkoli način spreminjajo zapise v bazi. Pri tem se shrani tudi nekaj metapodatkov o spremembah. *Slave* se poveže na *master* iz katerega prenaša shranjene zapise na lokalni strežnih, kar je prikazano na sliki 3.

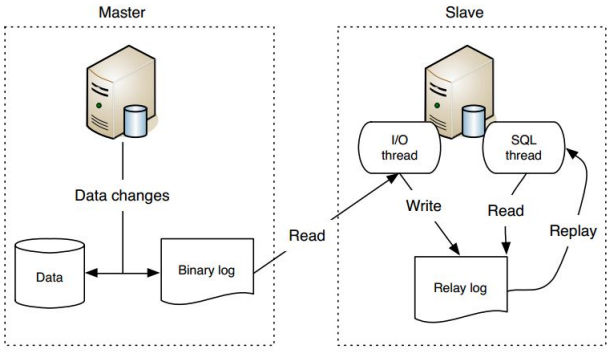


Figure 3: MySQL podvajanje (master - slave način)

Tako se vsi zapisi izvajajo iz lokalnega strežnika. Pri procesu podvajanja se na *slave* strežniku hranita parametra *current*

log name in *current log position*. Tako se v primeru prekinjene povezave ob naslednji vzpostavitvi povezave pridobijo vsi zapisi od lokacije *current log position* naprej. MySQL podpira *statement based* in *row-based* načina podvajanja.

4.2 Format binarnih dnevniških zapisov

Vsaka datoteka, ki vsebuje binarne dnevniške zapise se začne z blokom 4 bajtov ($\xfe\x62\x69\x6e$), ki se uporabljajo za identifikacijo datotek z zapisi, ter za preverjanje popolnosti datotek.

Za tem sledijo dnevniški zapisi, ki so sestavljeni iz glave, ki je fiksne dolžine in telesa, ki vsebuje vsebino glede na tip zapisa. Vsak tip zapisa ima svojo implementacijo metod za zapis glave in telesa. MySQL vsebuje paket *mysqlbinlog*, ki omogoča branje binarnih dnevniških zapisov in izvozi vsebino v SQL format, ki je uporabljen za podvajanje. Za lažje ocenjevanje prototipa, ki ga opisuje [4] je ta baziran na izvedbi omenjenega paketa.

4.3 Transakcijski dnevniški zapis

Poleg zgoraj opisanih dnevniških zapisov, ki se uporabljajo v namene podvajanja, se beležijo tudi drugi zapisi, ki vsebujejo veliko podatov o sistemu za upravljanje podatkovnih baz. Transakcijski zapisi se shranjujejo pri InnoDB [3] pomnilniškem pogonu, ki nudi podporo tujim ključem in *ACID* lastnosti skladne z SQL standardom. Ti zapisi se uporabljajo za zagotavljanje transakcij, obnovitev v primeru sesutja in zagotavljanje zanesljivosti sistema.

Za vsako spremembo se shrani vsaj en zapis, ki vsebuje glavo zapisa z IDjem transakcije, pozicijo zapisa in pa telo, ki vsebuje originalni del zapisa.

4.4 Izvedba prototipa

Pri implementaciji prototipa [4] so veliko pozornosti posvetili neodvisnosti, saj so želeli minimizirati spremembe v jedru sistema DBMS in s tem zagotoviti lažje vzdrževanje. Ključnega pomena je tudi, da spremembe ne vplivajo na funkcionlnost sistema za upravljanje podatkovne baze.

Za navigiranje skozi dnevniške zapise, MySQL uporablja reference. Prototip to funkcionalnost izkoristi in med dva zaporedna vnosa doda mrtvi prostor. To dosežemo tako, da povečamo dolžino zapisa, ki se uporablja za preračun lokacije naslednjega vnosa.

Mrtvi prostor ali *slack space* je prostor, ki ga datotečni sistem zasede, vendar ga sistem za upravljanje podatkovnih baz ne uporablja, prav tako pa ne spremeni funkcionalnosti sistema.

4.5 InnoDB transakcijski dnevniški zapis

InnoDB za vsak zapis ustvari žalogovnik"dolg 512 bajtov, prikazan na sliki 4.

V primeru, da vnos presega omenjeno dolžino, se ta razdeli v dva zalogovnika. V InnoDB pomnilniškem pogonu vsak transakcijki zapis vsebuje generično glavo zapisa, kraj izvora v datotečnem sistemu in kopijo prepisanih podatkov.

Opisana rešitev razširja privzeti zapis in uvaja dodaten podpis za vsak vnos. V ta namen je potrebno posodobiti dolžino zapisanega bloka. Ker vsak blok vsebuje svojo dolžino, InnoDB ignorira dodane podpise, ki so na sliki 5 označeni z S. S tem ne spremenimo nobene funkcionalnosti, uvedemo pa sistem, ki nam omogoča shranjevanje dodatnih parametrov, ki jih uporabljamo za preverjanje ustreznosti.

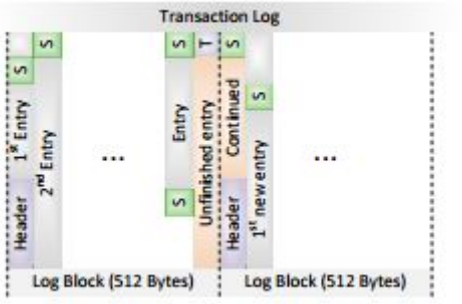


Figure 4: Podpisi dodani v blok

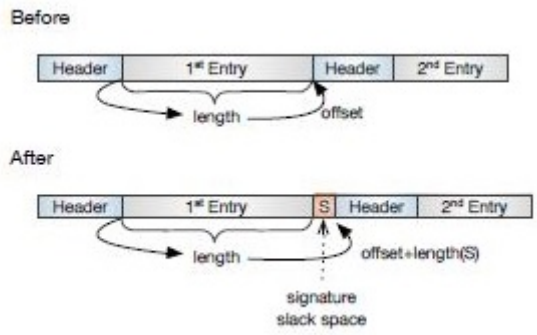


Figure 5: Dodan podpis v InnoDB transakcijskih zapisih

4.6 MySQL binarni dnevniški zapisi

Podobno, kot v primeru transakcijskih zapisov, lahko tudi v binarnih dnevniških zapisih uvedemo sistem, ki nam bo omogočal dodajanje dodatnih zapisov za namene preverjanja ustreznosti podatkov. Predstavljena rešitev to izkorišča tudi pri binarnih dnevniških zapisih. V glavi sporočila je vsebovan podatek o dolžini vnešenega sporočila in odmik do naslednjega. Slika 6 prikazuje, kako je rešitev implementirana v prototipu [4].

Rešitev je zelo podobna tisti pri modifikaciji InnoDB transakcijskih zapisov, vendar še enostavnejša, saj je za to potrebna le preprosta sprememba pri odmiku naslednjega zapisa. Dodan podpis je skupaj z dodelano verzijo orodja *mysqlbinlog* uporabljen za preverjanje integritete zapisanih vnosov.

5. VREDNOTENJE

V poglavju bodo najprej predstavljeni scenarij napadov v nadaljevanju pa vpliv opisanega sistema na varnost.

5.1 Scenarij napadov

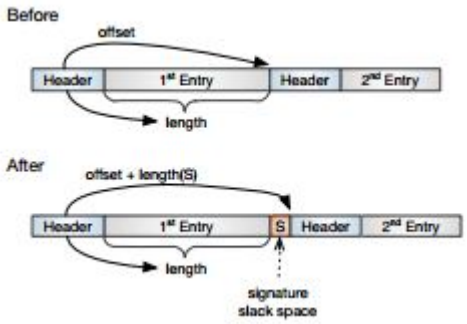


Figure 6: Dodan podpis v MySQL binarnih zapisih

V raziskavi sta identificirana sva modela napadov, sredstva in opis kako predstavljen prototip vpliva nanj.

5.1.1 Model napadov

Sledita dva potencialna tipa napadov, ki sta pogosta v velikih sistemih in podatkovnih skladiščih, ki zagotavljajo neprekinjeno delovanje.

Zlonamerni administrator datotečnega sistema. V tem primeru ima napadalec bralni in pisalni dostop do datotečnega sistema v podatkovnem skladišču in do datotek v sistemu za upravljanje podatkovnih baz. Podatke lahko spreminja brez uporabe vmesnikov sistema za upravljanje podatkovnih baz. Pri tem:

- lahko ureja poljubne datoteke, predvsem tiste, ki vsebujejo tabele in notranje podatkovne strukture. Te spremembe niso nadzorovane s strani sistema za upravljanje podatkovnih baz.
- lahko ureja dnevniške zapise operacijskega sistema
- nima dostopa do poizvedovalnega vmesnika
- nima dostopa do RAM pomnilnika

5.1.2 Zlonamerni administrator podatkovne baze

Napadalec ima pravice administratorja podatkovne baze kar pomeni, da ima pravice za nadzor in urejanje podatkovne baze. Pri tem:

- lahko administrira podatkovno bazo
- ima le bralne pravice na datotečnem sistemu
- lahko spreminja uporabniške pravice podatkovne baze in poti zapisovanja dnevniških zapisov

5.2 Sredstva in ocena varnosti

Predstavljena so sredstva, ki jih lahko ogrozi kateri od opisanih napadalcev. Obravnavana so le sredstva, ki so povezana z lastnostmi opisanega prototipa.

5.2.1 DBMS instanca

Predstavlja dejanski sistem za upravljanje podatkovne baze, preko katerega lahko bazo izklopimo in zaženemo. Kljub temu, da bi napadalec imel dostop do omenjenih pravic, lahko akcije zaznamo in pravilno ukrepamo.

5.2.2 Konfiguracije

Konfiguracije predstavljajo sredstvo preko katerega lahko napadalec bere in celo spreminja nastavitve podatkovne baze med samim delovanjem. Napadalec bi lahko spremenil velikost notranjih podatkovnih struktur, nastavitve podvajanja in beleženja dnevniških zapisov. Kljub neopaženi spremembi nastavitev bi sistem za preverjanje avtentičnosti podatkov spremembe zaznal, saj te ne bi bile vsebovane v beležkah za zagotavljanje avtentičnosti.

5.2.3 SQL vmesnik

SQL vmesnik dovoljuje izvedbo SQL poizvedb, kot tudi vse ostale veljavne DBMS poizvedbe in bazne procedure. Predlagan sistem administratorju ne more preprečit izvedbo veljavnih poizvedb tudi, če so zlonamerne, lahko pa zagotovi, da poizvedbe ostanejo zabeležene. V nasprotnem primeru, bi lahko administrator izbrisal poizvedbe iz beležke izvedenih poizvedb.

5.2.4 Izvorna koda

Sredstvo za napad je lahko tudi sama izvorna koda DBMS sistema. Možnost izvedbe programskega popravka, ponovne prevedbe kode bi omogočila dodajanje ali odstranitev katere od funkcionalnosti. Taka sprememba bi za delovanje potrebovala ponovni zagon podatkovne baze, kar bi sprožilo alarm. Napadalec bi izvorno kodo DBMS sistema spremenil tako, da bi odstranil funkcionalnosti predlaganega sistema ali beležil vse zgenerirane naključne ključe. Za preprečitev takega napada bi potrebovali osnovo, ki bi dovoljevala uporabo le pravilno podpisane kode. Poleg tega, bi ponovni zagon zahteval novo seme za generator naključnih števil, ki pa ga dodeli zaupanja vredna tretja oseba.

5.2.5 Notranje podatkovne strukture

Notranje podatkovne strukture so sredstva, ki jih uporablja DBMS sistem za zagotavljanje pravilnega delovanja. Potencialni napadalec bi lahko strukture spreminjal, dodajal vnose ali jih iz njih brisal. Brez predlaganega sistema bi bile take spremembe neopazene, zdaj pa bo sistem za preverjanje avtentičnosti zaznal spremembe, saj se podatki ne bodo ujemali s tistimi v zapisih za preverjanje avtentičnosti. Napadalec bi lahko podatke spremenil le začasno in bi napad ostal nezaznan. V ta namen pa bi lahko preverjanje avtentičnosti izvajali naključno in tako zmanjšali to verjetnost.

5.2.6 Generator naključnih števil

Generator naključnih števil je kritični element predlaganega sistema, saj na njem sloni celotna varnostna arhitektura. Zajemanje vseh zgeneriranih naključnih vrednosti bi napadalcu omogočilo generiranje novih naključnih vrednosti, s čimer izgubimo nadzor nad varnostnim sistemom. Kljub temu, pa bi stari podatki ostali varni. Da preprečimo še to nevarnost bi lahko uporabili strojni generator naključnih števil. Tako napadalcu onemogočimo odkritje postopka, saj bi za to bila potrebna fizična prisotnost in izvedba obratnega inžinirstva nad vezjem .

6. ZAKLJUčEK

V tem članku je opisan pristop, ki z uporabo sistema za zagotavljanje transakcij in podvajanja naredi sistem za upravljanje podatkovnih baz forenzično zavednega. Notranje podatkovne strukture, ki jih predstavljeni sistem uporablja za forenzično rekonstrukcijo so v osnovi namenjene le za zagotavljanje pravilnega delovanja sistema in človeku niso berljive. Predstavljen sistem tako z minimalnim posegom v izvorno kodo doda forenzično zavednost sistemu in za delovanje ne potrebuje nobenih dodatnih administracijskih zapisov. Poleg opisanega prototipa članek vsebuje tudi možne scenarije napadov in opis sredstev preko katerih so ti možni. Za vsakega od sredstev je tudi opisana ocena varnosti, ki podrobno obrazloži kako sistem reši pomankljivost ali pripomore k njeni zaznavi.

7. VIRI

[1] Dbms functions. `http://databasemanagement.wikia.com/wiki/DBMS_Functions`, 2009. Pogledano 8.5.2015.

[2] J. K. B. Schneier. Secure audit logs to support computer forensics. *ACM Transactions on Information and System Security (*, 2(2):159–176, 1999.

[3] P. Frühwirt, M. Huber, M. Mulazzani, and E. Weippl. Innodb database forensics. In *Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on*, pages 1028–1036, April 2010.

[4] P. Frühwirt, P. Kieseberg, K. Krombholz, and E. Weippl. Towards a forensic-aware database solution: Using a secured database replication protocol and transaction management for digital investigations. *Digital Investigation*, 11(4):336–348, 2014.

[5] B. P. G. A. Marson. Practical secure logging: Seekable sequential key generators. *Computer Security - ESORICS 2013, Vol. 8134 of Lecture Notes in Computer Science, Springer Berlin Heidelberg*, 8134:111–128, 2013.

[6] P. S. G. Miklau, B. N. Levine. Securing history: Privacy and accountability in database systems. *CIDR*, pages 387–396, 2007.

[7] R. T. S. G. Ozsoyoglu. Temporal and real-time databases. *Knowledge and Data Engineering*, 7(4):513–532, 1995.

[8] M. O. F. Martin S. Olivier. Reconstruction in database forensics. *Advances in Digital Forensics*, 7(7):273–287, 2011.

[9] M. S. Olivier. On metadata context in database forensics. *Digital Investigation*, 5(3-4):115–123, 2009.

[10] M. H. M. M. E. R. W. P. Frühwirt. Innodb database forensics. *Advanced Information Networking and Applications (AINA)*, pages 1028–1036, 2010.

[11] S. S. M. H. E. W. P. Frühwirt, P. Kieseberg. Innodb database forensics: Reconstructing data manipulation queries from redo logs. *5th International Workshop on Digital Forensic*, 5, 2012.

[12] B. N. L. P. Stahlberg, G. Miklau. Threats to privacy in the forensic analysis of database systems. *Proceedings of the 2007 ACM SIGMOD international conference on Managment of data*, pages 91–102, 2007.

[13] P. Stahlberg. Forensic analysis of database systems–final report. *: Fall seminar report, Department of Computer Science*, 2005.

Prednosti in slabosti žive in mrtve forenzične analize

Andrej Česen
Univerza v Ljubljani Fakulteta
za računalništvo in informatiko
Večna pot 113
1000 Ljubljana, Slovenija
ac2681@student.uni-lj.si

Luka Šarc
Univerza v Ljubljani Fakulteta
za računalništvo in informatiko
Večna pot 113
1000 Ljubljana, Slovenija
ls5713@student.uni-lj.si

Melanija Vezočnik
Univerza v Ljubljani Fakulteta
za računalništvo in informatiko
Večna pot 113
1000 Ljubljana, Slovenija
mv6082@student.uni-lj.si

POVZETEK

Članek obravnava različne pristope forenzičnega preiskovanja, ki se izvaja na osumljenčevem računalniku. Podrobneje sta predstavljeni metodi žive in mrtve analize, njune prednosti ter slabosti. Mrtva analiza se uporablja predvsem zaradi dogovora, da se s takim načinom forenzične preiskave ohranja celovitost podatkov na podatkovnem nosilcu, živa analiza pa nam služi v primerih, ko je za preiskavo kritično, da zajamemo trenutno stanje sistema. Da bi ugotovili, če do sprememb pri mrtvi analizi prihaja na praktičnem primeru, smo poskusili izvesti podoben eksperiment kot v referenčnem članku [7] - z zgoščevalno funkcijo SHA1 smo izračunali zgoščene vrednosti osumljenčevega diska pred in po izvajanju mrtve analize ter opazovali morebitne spremembe, ki jih tovrstna analiza naredi na disku.

Ključne besede

računalniška forenzika, mrtva analiza, živa analiza, prenosni mediji, zagonski CD/DVD/USB, živi CD/DVD/USB

1. UVOD

Forenzičnim preiskovalcem je pri preiskavah na voljo veliko različnih forenzičnih orodij in pristopov za izvajanje pregleda osumljenčeve naprave na prizorišču zločina. Orodja se lahko nahajajo na prenosnih medijih, kot so zgoščenke (*ang. Compact Disc - CD*), digitalne video plošče (*ang. Digital Video Disc - DVD*) in pomnilniški ključi (*ang. Universal Serial Bus - USB*). Orodja so lahko samostoječi operacijski sistemi (največkrat distribucije operacijskega sistema Linux), ki vsebujejo posamezne forenzične aplikacije in kjer se prenosni mediji uporabijo za zagon operacijskega sistema na osumljenčevemu računalniku. Forenzična orodja pa so lahko tudi različni programi, ki jih forenzik izvaja neposredno na operacijskem sistemu, ki teče na osumljenčevem računalniku. Analiza te vrste prinaša številne prednosti, med drugim:

- Hitro preiskavo osumljenčevega računalnika na prizorišču zločina.
- Preiskavo sistema brez zamudnih postopkov, kot je pridobitev trdega diska, kjer mora forenzični preiskovalec najprej računalnik ugasniti, odstraniti vse povezovalne kable ter odstraniti trdi disk.
- Pridobitev slike pomnilnika z naključnim dostopom (*ang. Random-Access Memory - RAM*) operacijskega sistema [7].

Po drugi strani uporaba forenzičnih orodij na prizorišču zločina prinaša tudi nekatere slabosti, med drugim lahko preiskovalno orodje med izvajanjem spremeni podatke na podatkovnem nosilcu osumljenčevega računalnika, tako da digitalni dokazi lahko postanejo vprašljivi in neveljavni na sodišču, če niso pravilno analizirani s strani forenzika. Preiskovanje torej lahko privede do necelovitosti podatkov in morebitne izgube pomembnih podatkov ali celo dokazov.

Prenosne medije za forenzično preiskavo naprave lahko uporabljamo za izvedbo žive (*ang. live*) in mrtve (*ang. dead*) analize. Živa analiza za preiskovanje uporablja osumljenčev operacijski sistem in njegove vire. Nasprotno mrtva analiza za preiskovanje uporablja operacijski sistem iz prenosnega medija in ne osumljenčevega. Posledično pri izvedbi žive analize poleg informacij na obstojnem pomnilniku, dobimo tudi informacije o trenutnem stanju sistema, ki se sicer izgubijo ob izklopu naprave. V nadaljevanju članka bomo podrobneje pogledali lastnosti obeh vrst analiz. Pogledali si bomo kaj so prednosti in slabosti posamezne analize ter kakšna konkretnejša orodja oziroma metodologije se uporabljajo za izvajanje obeh vrst analiz. V zaključku članka je predstavljen poskus, ki smo ga izvedli, in je podoben poskusu iz referenčnega članka [6]. Pri poskusu smo ugotavljali, ali orodja za izvajanje mrtve analize pustijo na obstojnem pomnilniku kakšne sledi v obliki spremenjenih podatkov na disku.

2. ŽIVA ANALIZA

Živa analiza je postopek forenzične preiskave osumljenčeve naprave, pri kateri preiskovalec izkoristi nameščeni operacijski sistem in računalniške vire, ki jih naprava ponuja. Pri tem uporablja različna forenzična okolja, ki se nahajajo na prenosnih medijih kot so USB ključi, CD-ji ali DVD-ji. Živa analiza torej ne uporablja tradicionalnega pristopa, pri katerem forenzik na ustrezen način zaustavi osumljenčevo napravo, jo označi in šele v kontroliranemu okolju izvaja različne

tehnike preiskovanja podatkov, niti ne uporablja mrtve analize, katero bomo spoznali v naslednjih poglavjih. Pri živi analizi preiskovalec že na prizorišču zločina preišče osumljenčeve naprave. To prinaša številne prednosti in hkrati tudi slabosti. V nadaljevanju bomo govorili kdaj in zakaj se je pametno odločiti za tak postopek preiskovanja in o posledicah, ki jih prinaša [7].

2.1 Prednosti

Poglavje o prednostih žive analize je povzeto po [3, 7, 9]. Živa analiza ima številne prednosti v primerjavi z mrtvo analizo. Ker pri mrtvi analizi napravo izklopimo, lahko pri tem izgubimo pomembne podatke o trenutnem stanju naprave. Nasprotno lahko pri živi analizi zajamemo trenutno stanje naprave in operacijskega sistema, ki se na njej izvaja, kar je ena izmed največjih prednosti. Zajamemo lahko podatke o:

- tekočih procesih,
- aktivnih omrežnih povezavah,
- aktivnih oziroma prijavljenih uporabnikih,
- odprtih sistemskih knjižnicah,
- odprtih datotekah in
- priključenih in dekriptiranih datotečnih sistemih, ki jih je sicer zelo težko ali celo nemogoče dekriptirati ob nepoznanem kriptirnem ključu in kriptirni tehniki.

Pri tem je pomembno podatke zajemati v vrstnem redu spremenljivosti. Nekateri tipi podatkov, kot so na primer trenutne aktivne omrežne povezave, se spreminjajo veliko pogosteje kot trenutni prijavljeni uporabniki. Zaradi neobstojnosti (*ang. volatile*) je zelo zaželen in uporaben pristop zajetja posnetka (*ang. dump*) pomnilnika z naključnim dostopom. Toda takšna operacija lahko traja več minut, kar pa že lahko vpliva na pogosto spremenljive kratkotrajne tipe podatkov. Zato je zelo pomembno, da se pri analizi tekočega operacijskega sistema upošteva prioritete preiskovalca in se, če je situaciji primerno, najprej opravi zajem za primer pomembnih kratkotrajnih podatkov in se šele nato izvede zajem posnetka celotnega pomnilnika z naključnim dostopom.

Živa analiza lahko omogoča praktično vse forenzične pristope, ki jih ponujata statična in mrtva analiza, a z nekaj omejitvami. Omejitve bomo postopoma spoznavali v nadaljevanju članka pri opisu slabosti in različnih pristopov k izvajanju žive analize.

Ena od glavnih prednosti žive analize je ta, da je poceni. Izvajanje mrtve analize je v določenih primerih lahko enostavno predrago. Primer te vrste je napad na strežnik. Takrat morebitna preiskava zahteva zajetje podatkov iz strežnika, nepredstavljivo pa je za več ur izklapljati strežnik, ki nudi storitve, od katerih je odvisnih veliko uporabnikov. To je še posebej pomembno pri lažno pozitivnih opozorilih, poslanih iz sistema za preprečevanje vdorov IDS (*ang. Intrusion Detection System*). V teh primerih torej mrtva analiza nikakor ne pride v poštev, medtem ko živa analiza omogoča preiskovanje brez izpadov storitev.

Živa analiza je primerna tudi takrat, ko želimo, da se osumljenec ne zaveda preiskave ali ko želimo izvesti neke vrste triažo ali predhodno preiskavo. Na primer, osumljenca želimo čim prej soočiti z dokazom o njegovem napačnem ravnanju in na ta način skušamo že na prizorišču zločina pridobiti njegovo priznanje. Živa analiza torej tako deluje kot nekakšno prisilno sredstvo oziroma kot orodje za izvajanje pritiska na osumljenca za pridobitev čimprejšnjega priznanja krivde, to pa lahko precej poenostavi preiskavo.

2.2 Slabosti

Poglavje o slabostih žive analize je povzeto po [3, 4, 5, 7]. Živa analiza poleg prednosti s sabo prinaša tudi slabosti. Večini slabosti bi sicer težko pripisali tako slab prizvok, je pa pomembno, da se jih preiskovalec zaveda in temu primerno izvaja analizo podatkov in obravnava rezultate.

Prvi problem se pojavi že pri zaganjanju preiskovalnega forenzičnega okolja. Kljub omembi, da okolja nalagamo iz prenosnih medijev in so prirejena na način, da ne uporabljajo trdega diska preiskovane naprave, to ni vedno popolnoma res. Določeni programi enostavno potrebujejo nekaj prostora za svoje delovanje tudi na disku. Pri tem pa lahko prepišemo določene izbrisane vnose v datotečnem sistemu, ki bi lahko bili za primer pomembni, in s tem posežemo v sam dokaz. Vseeno je to velikokrat izguba, ki smo jo pripravljeni sprejeti za namene dostopa do trenutnega stanja sistema.

Naslednja težava tiči v sami sliki diska. Praktično nemogoče je pridobiti sliko diska, ki bi se po izračunani zgoščeni vrednosti ujemala z zgoščeno vrednostjo diska. Razprševalne funkcije uporabljamo za računanje zgoščenih vrednosti slik in diskov ter z njihovo pomočjo dokazujemo, da nismo spreminjali vsebine diska pri sami preiskavi. Tega pa ne moremo doseči z zajemom slike diska na aktivnem sistemu, ker med preiskovanjem operacijski sistem naprave še vedno normalno deluje. V tem primeru se na disku ves čas dogajajo spremembe, četudi majhne, in vključujejo na primer le spremembe v beležkah (*ang. log*) operacijskega sistema. Vsekakor je že majhno razlikovanje odločilno za popolnoma različne zgoščene vrednosti. Omenjena problema lahko vplivata na preiskavo, nista pa kritična, v kolikor se jih preiskovalec zaveda in poseduje znanje za ustrezno rokovanje s pomanjkljivostma.

Slabost žive analize najdemo tudi pri virih lažnih podatkov. Nekatera forenzična okolja za izvajanje žive analize namreč uporabljajo deljenje knjižnice sistema, ki jih nekoliko bolj pretkan napadalec lahko priredi na način, pri katerem filtrira podatke. Tako se vsi zahtevki preiskovalnega okolja, ki gredo skozi sistemske ukaze operacijskega sistema in to knjižnico, pred končnim rezultatom filtrirajo in priredijo sam rezultat na način, ki ga je definiral napadalec. To slabost sicer izkoriščajo predvsem pisci različne škodoželjne programske opreme, ni pa izvzeto, da to lahko stori kdorkoli. Na takšen način napadalec zelo kvalitetno zakrije svojo aktivnost na napravi. Še en problem tiči tudi v tem, da je možno filtre ustvariti na več načinov. Preiskovalna okolja jih sicer na različne načine poskušajo rešiti in zaobiti, a pri tem zaradi velikega števila različnih implementacij filtrov niso vedno uspešna.

2.3 Pristopi in okolja

Poglavje je povzeto po [4, 5]. Obstaja več pristopov in načinov oziroma okolij, s katerimi lahko preiskovalci izvajajo živo analizo. Ogledali si bomo nekaj različnih pristopov.

2.3.1 Standardni uporabniški vmesnik

Najpreprostejši in popolnoma trivialen pristop je uporaba standardnega uporabniškega vmesnika, ki ga ponuja operacijski sistem. Pri tem lahko uporabimo tako grafične vmesnike (*ang. Graphical User Interface - GUI*) ali različne konzolne vmesnike (*ang. Command Line Interface - CLI*). Takšen pristop ne zahteva večjih posegov v operacijski sistem. Preiskovalec lahko zelo hitro in učinkovito pridobi podatke o aktivnih omrežnih povezavah, aktivnih uporabnikih ipd. Na nekaterih operacijskih sistemih je možno celo pridobiti posnetek bralno-pisalnega pomnilnika.

Pravice dostopa lahko predstavljajo težave za ta pristop. Preiskovalec potrebuje ustrezna sistemska dovoljenja, da lahko pridobi podatke. V kolikor preiskovalec nima ustrezne avtorizacije, se ta pristop zelo hitro konča. Pristop s standardnim uporabniškim vmesnikom predpostavlja pravilno delovanje operacijskega sistema. Če je sistem na kakršenkoli način pokvarjen, lahko uniči dokazno gradivo in krši integriteto sistema. Primer so že omenjeni morebitni obstoječi filtri na napravi. Še največja težava pa je, da preiskovalec z uporabo uporabniškega vmesnika ustvarja nepopravljive spremembe na operacijskem sistemu in stanju samega sistema.

2.3.2 Uvožena orodja

Pristop z uvoženimi orodji se osredotoča na integriteto sistema. Uvaža orodja iz prenosnih medijev preiskovalca. Primer delovanja je, da preiskovalec namesto privzetega programa, ki se izvede ob ukazu `ps` v lupini `bash`, izvede program `ps` s svojega prenosnega medija, za katerega verjame, da je pravilen. S tem se izogne uporabi sistemskega ukaza, ki bi lahko bil žrtev napadenega sistema. S prenosnim medijem lahko preiskovalec pridobi na široki paleti orodij in okolij za analizo sistema.

Omenjen pristop od preiskovalca še vedno zahteva ustrezne pravice dostopa do operacijskega sistema. Čeprav je osredotočen na integriteto sistema, je to seveda še vedno zelo težko izvedljivo pri tekočem sistemu. Pristop prav tako ne zmore vedno reševati problema morebitnega pokvarjenega operacijskega sistema. V nekaterih primerih namreč nikoli ni možno izvesti določenih ukazov ali programov v sistemu brez uporabe sistemskih knjižnic, za katere pa smo že omenili, da so lahko žrtve napada. Pristop z uvoženimi orodji bi bil torej primeren, ko je forenzik prepričan, da sistemske knjižice niso prizadete ali pa, ko je prepričan, da njegova uporabljena orodja že v osnovi ne uporabljajo sistemskih knjižnic.

2.3.3 Modificiran sistem

Predpogoj pristopa z modificiranim sistemom je prirejena implementacijo sistema na način, da ta omogoča zajemanje forenzičnih podatkov. Za to obstaja več načinov, najbolj pogost pa je z uporabo sistema honeypot. Tak sistem omogoča opazovanje (*ang. monitoring*) vseh dogodkov in procesov in zajetje morebitnega napada od začetka do konca. Primer je orodje Sebek podjetja The Honeynet Project, ki na primer omogoča sledenje vsem sistemskim klicem [1, 2].

Seveda ima tudi ta pristop svoje slabosti, najbolj očitna pa je ta, da mora biti tak sistem pred nastavljen. Zelo pogosta težava je tudi v zaznavnosti takega sistema. Napadalcem namreč velikokrat uspe sistem za opazovanje (*ang. monitoring system*) zaznati, ga zaobiti ali celo izključiti oziroma popolnoma onesposobiti.

2.3.4 Dodatna strojna oprema

Uporaba dodatne strojne opreme preiskovalcem omogoči priklop na različna vodila naprav, preko katerih je enostavno pridobivati podatke. Preiskovalec lahko celo komunicira z DMA (*ang. Direct Memory Access*) krmilnikom in popolnoma zaobide uporabo centralne procesne enote (*CPE*) kot sredstvo za dostop do pomnilnika.

Problem pristopa je zopet ta, da je potrebno imeti tak sistem nastavljen pred zagonom same naprave oziroma operacijskega sistema. Naslednja težava je, da so nekateri operacijski sistemi sposobni dodatno strojno opremo zaznati. S tem pa poskušajo po svojih pravilih z njo sodelovati in posegati v njeno delovanje. V najslabšem primeru lahko pride celo do zaustavitve dodatne strojne opreme ali do nepravilnega delovanja. Problemi se pojavljajo tudi pri pridobivanju slike pomnilnika. Preiskovalec s to metodo pridobi zgolj sliko, ki pa jo mora zatem še interpretirati. To pa zahteva veliko več truda kot nek preprost vpogled v stanje sistema, na primer preko uporabniškega vmesnika operacijskega sistema. Težava je tudi v tem, da lahko spreten napadalec vpliva na CPE in preko vodil PCI (*ang. Peripheral Component Interconnect*) pošilja drugačne kopije podatkov kot jih vidi CPE. To privede do neustreznih slik pomnilnika in prikriti napad na sistem. Tudi samo zajemanje preko zunanje strojne opreme lahko privede do hujših napak v delovanju sistema in posledično do samodejne zaustavitve delovanja.

3. MRTVA ANALIZA

Digitalna forenzika se je tradicionalno izvajala s statično analizo kopije oziroma slike obstojnega podatkovnega nosilca. Skozi čas so se začeli uveljavljati pristopi, ki izvajajo preiskovanje neposredno na osumljenčevem računalniku. Z izrazom mrtva analiza se označuje vse načine forenzičnega preiskovanja, pri katerem se uporabi zaupanja vreden operacijski sistem (oziroma okolje) namesto naloženega operacijskega sistema na osumljenčevem računalniku. Za to je potreben izklop oziroma ponovni zagon računalniškega sistema, ki pa ima destruktivne posledice do podatkov v pomnilniku in ostalih neobstojnih podatkovnih nosilcih.

Medije CD/DVD, ki se uporabljajo za mrtvo analizo, označujemo s skupnim terminom zagonski CD/DVD-ji (*ang. boot CD/DVD*), srečamo pa jih tudi pod imenom živi CD/DVD-ji (*ang. live CD/DVD*) [7]. Podobno govorimo tudi o zagonskih oziroma živih USB ključih. Zagonski medij vsebuje operacijski sistem s pred naloženimi forenzičnimi aplikacijami, ki ga lahko poganjamo neposredno iz bralnega medija in za svoje delovanje ne potrebuje trdega diska oziroma drugega obstojnega podatkovnega nosilca. S tem se doseže tudi enostavna prenosljivost na različne računalniške platforme.

3.1 Prednosti

Mrtva analiza podobno kot živa omogoča preiskovanje sistema že na mestu samem, brez potrebe po razstavljanju računalnika in ustvarjanju slike podatkovnega nosilca. V primeru starejšega sistema se lahko zgodi, da preiskovalec nima na razpolago potrebne opreme za zajem podatkov, za kar pa je možno uporabiti taisti sistem. S takim pristopom se lahko pohitri celoten proces forenzične preiskave in v nekaterih primerih tudi pravočasno napelje na pravilne sledi.

Za sprejemljivost dokazov na sodišču je zelo pomembna integriteta le-teh. Digitalni dokazi so še posebej dovzetni za spremembe, zato je pomembno minimizirati verjetnost spreminjanja podatkov. Živa analiza se izvaja na zagnanem sistemu in je posledično bolj tvegana v smislu ohranjanja integritete podatkov v primerjavi z mrtvo analizo, kjer poganjamo ločen (preverjen) operacijski sistem, ki za svoje delovanje ne potrebuje obstojnega podatkovnega nosilca. Podatkovne medije se lahko priklopi v izključno bralnem načinu in s tem prepreči spreminjanje dokazov. Do nekaterih sprememb podatkov v manjšem obsegu sicer vseeno prihaja, vendar so te večinoma predvidljive in zato na vplivajo na verodostojnost dokazov [7].

Z večanjem prostora na podatkovnih nosilcih se povečuje tudi čas, potreben za izdelavo slik za namene statične analize. Ta proces lahko traja po več ur [8], s tem pa se hkrati podaljšuje tudi nedostopnost preiskovanega sistema. Še posebej izdelave kopij bolj zapletenih oblik diskov, kot so diskovna polja RAID (*ang. Redundant Array of Independent Disks*) in omrežni diski NAS (*ang. Network Attached Storage*), terjajo veliko več časa. Z mrtvo analizo dosežemo podoben učinek preiskovanja kot s statično, toda v občutno krajšem času, saj ni potrebno izdelati slik podatkovnih nosilcev.

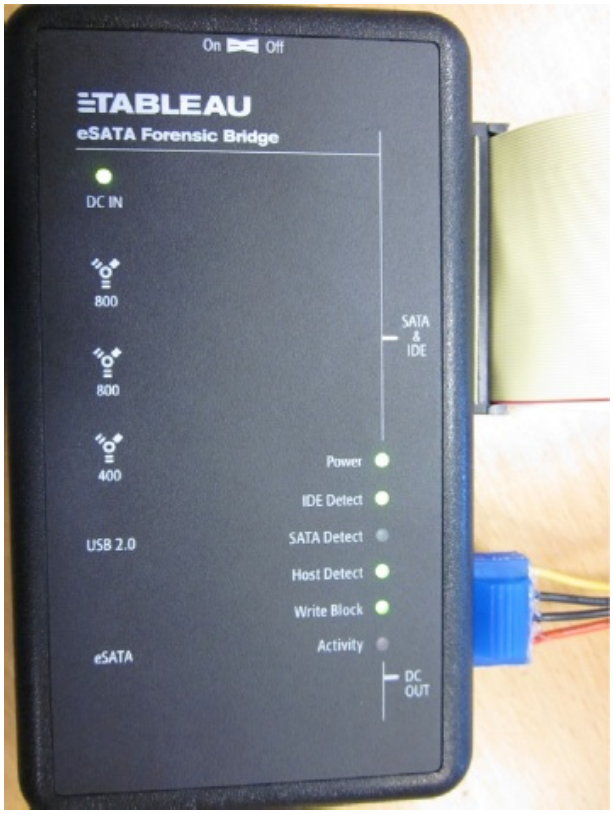
3.2 Slabosti

Za izvedbo mrtve analize je potrebna zaustavitev računalniškega sistema. Računalnik se lahko izklopi običajno, preko operacijskega sistema, lahko pa se tudi prekine napajanje z izklopom napajalnega kabla. S prekinitvijo napajanja se onemogoči izvedbo skript, ki bi ob normalnem izklopu pobrisala morebitne dokaze z računalnika, lahko pa ima za posledico nekonsistentno stanje diska in nesinhroniziran pisalni predpomnilnik [8]. Oba načina zaustavitve sistema terjata izgubo podatkov z neobstojnih podatkovnih nosilcev in s tem povezanih morebitnih dokazov, kar pa je največja slabost pristopa mrtve analize. Če je podatkovni medij šifriran, se ob izklopu sistema izgubijo tudi morebitni že dešifrirani podatki in šifrirni ključi.

Kljub omogočanju priklopa podatkovnega nosilca na ekskluzivno bralen način v okoljih Linux, operacijski sistem ne zagotavlja, da se na medij ne bo vršilo pisalnih operacij [6]. Ta lastnost lahko naredi dokazno gradivo pridobljeno z mrtvo analizo nesprejemljivo za sodišče, zato je pomembno določiti, za kakšne vrste sprememb gre. Zaradi hitrega razvoja zagonskih medijev je dokumentiranost obsega sprememb zelo skopa [7].

3.3 Okolja

Nedavna raziskava je zbrala tri popularne distribucije Linux za forenzično preiskavo - Knoppix v7.0, Helix 3 Pro 2009R3, Kali Linux v1.0 - in na ponovljiv način testirala, kakšne spremembe so se pojavile na podatkovnem nosilcu po uporabi forenzičnih orodij v vnaprej definiranem (realnem) primeru uporabe [7].



Slika 1: Preprečevalnik pisanja po disku.

V eksperimentu so uporabili osumljenčev računalnik, preprečevalnik pisanja po disku (*ang. hardware write blocker*) in forenzično delovno postajo (*ang. forensic workstation*). Osumljenčev računalnik je prenosnik IBM Thinkpad T42 z operacijskim sistemom Windows XP in trdim diskom velikosti 40 GB, formatiranim z datotečnim sistemom NTFS. Preprečevalnik pisanja po disku je naprava med forenzično delovno postajo in osumljenčevim računalnikom, ki blokira ukaze za pisanje podatkov in dovoljuje samo ukaze za branje podatkov. Prikazan je na sliki 1.

Uporabili so preprečevalnik eSATA Forensic Bridge. Uporabljali so ga za pregledovanje sprememb na disku tako, da so sliko diska naredili pred analizo na osumljenčevem računalniku in po njej. Izkazalo se je, da je do sprememb prišlo pri vseh treh distribucijah. V vseh primerih so se spremembe odražale le v zadnjem času dostopa do specifičnih datotek, novih datotek pa ni bilo ustvarjenih. Spremembe so torej minimalne in v večini primerov ne bi zmanjšale verodostojnost pridobljenega dokaznega gradiva z metodo mrtve analize. Ugotavljajo, da bi bilo potrebno opraviti strožje testiranje zagonskih medijev in podrobneje dokumentirati spremembe pri forenzičnem preiskovanju za lažje dokazovanje integritete dokazov.

3.3.1 Helix3 Pro

Helix3 Pro je komercialno forenzično orodje, namenjeno forenzični preiskavi diskov. Nahaja se na zgoščenki, ki se lahko uporablja kot zagonska ali živa. Pri preiskovanju slik diskov ne ohranja enake zgoščene vrednosti SHA1 slike diska pred in po uporabi orodja, četudi disk logično priklopimo v načinu samo za branje (*ang. read-only*). Spremenijo se časi dostopanja pri treh sistemskih datotekah, kar ni veliko, zato je orodje vseeno zanesljivo.

3.3.2 Kali

Distribucija Kali poleg izvajanja digitalne forenzike lahko testira tudi vdore. Temelji na distribuciji BlackTrack, zato obe distribuciji nudita podobna orodja za izvajanje digitalne forenzike. Pri preiskovanju slik diskov ne ohranja enake zgoščene vrednosti SHA1 slike diska pred in po uporabi orodja, saj je pri štirih sistemskih datotekah spremenila čase dostopanja.

3.3.3 Knoppix

Knoppix je splošno namenska zagonska distribucija Linux, ki temelji na distribuciji Debian. Veliko se uporablja v digitalni forenziki za izvajanje mrtve analize. Pri poganjanju programov, se po nalaganju s prenosnega medija transparentno izvede dekompresija in nalaganje programa v glavni pomnilnik.

Omogoča ustvarjanje novih datotek v grafični lupini kljub temu, da je bil disk priklopljen v bralnem načinu. S stališča zagotavljanja integritete dokaznega gradiva je to zelo slaba lastnost, saj je potrebna še večja pazljivost forenzika, da ne pride do sprememb na podatkovnem nosilcu. Med preiskovanjem podatkovnega nosilca so se spremenili časi dostopa trem sistemskim datotekam in eni uporabniški datoteki. Glede na naključnost spremenjene datoteke obstaja verjetnost, da se lahko spremeni katerakoli datoteka na sistemu. Na podlagi te ugotovitve orodje ni primerno za forenzično preiskavo.

4. PREIZKUS OHRANJANJA CELOVITOSTI DISKA S SISTEMOMA KNOPPIX IN KALI LINUX

V zadnjem delu raziskovanja smo se odločili izvesti test iz članka [7] in preveriti, ali lahko dosežemo enake rezultate s čim bolj podobno metodologijo, kot je opisana v članku. Žal smo se pri izvajanju testa morali precej omejiti. V izvornem članku so bile uporabljene tri distribucije Linux okolja - Knoppix v7.0, Helix 3 Pro 2009R3 in Kali Linux v1.0. Že tu je nastopil prvi problem, saj je Helix Pro distribucija dostopna le v plačljivi različici, zato smo se morali testiranju na tej distribuciji odreči. Avtorji članka so za pregledovanje diska uporabljali forenzično delovno postajo in preprečevalnik pisanja po disku. Ker sami te opreme nismo imeli, smo se odločili za nekoliko prirejen testni scenarij.

Odločili smo se za izvedbo testa v virtualnem okolju - uporabljen je bil program Virtual Box v4.3.24, ki je tekel na forenzični delovni postaji z nameščenim operacijskim sistemom Windows 8.1. Uporabili smo virtualno sliko diska velikosti 11GB, formata .vmdk, z naloženim operacijskim sistemom Windows XP na datotečnem sistemu NTFS. To je predstavljalo disk našega osumljenca. Dodelili smo mu virtualno napravo, ki je predstavljala osumljenčev računalnik. Disk smo dodali na standarden način, ki ga omogoča okolje Virtual Box - v nastavitvah virtualne naprave, smo v zavihku "Storage" dodali obstoječi osumljenčev disk. Da bi pridobili referenčne slike virtualnih diskov v forenzični napravi, smo disk pred mrtvo analizo na osumljenčevi napravi priklopili v virtualno forenzično postajo z nameščenim operacijskim sistemom Debian. V ta sistem smo predhodno dodali nastavitev, ki onemogoča samodejni priklop osumljenčevega diska med zagonom samega operacijskega sistema. To smo dosegli z dodano nastavitveno vrstico v datoteki /etc/fstab:

```
UUID=7A5C17F65C17AC3F none ntfs ro,noauto.
```

Preden smo torej izvedli kakršenkoli test na distribucijah za izvajanje mrtve analize, smo v forenzičnem sistemu Debian naredili sliko diska. Enako smo naredili tudi po analizi diska z distribucijo za izvajanje mrtve analize. Tako smo lahko v primeru sprememb v slikah diskov pred in po izvajanju mrtve analizi primerjali morebitne spremembe. Slike smo ustvarili z naslednjim ukazom:

```
dd if=/dev/sdb of=slika.raw
```

Nad kreiranimi slikami osumljenčevega virtualnega diska v forenzičnem operacijskem sistemu smo izvedli še ukaz `sha1sum` za izračun zgoščene vrednosti z zgoščevalno funkcijo SHA1. Na ta način smo takoj pridobili prvi vpogled v to, ali se je disk med mrtvo analizo spremenil. Ta postopek smo kasneje ponovili za vsako testiranje forenzičnih orodij za izvajanje mrtve analize. Obe distribuciji, Knoppix v7.4 in Kali Linux v1.1, smo zagnali iz živega diska DVD (ki se v okolju Virtual Box doda na enak način kot disk, le da vstavimo enoto CD/DVD). Izvedli smo torej naslednje korake:

1. V forenzični virtualni napravi smo virtualni disk vstavili vanjo, naredili sliko diska z ukazom `dd` in izračunali zgoščeno vrednost slike diska z zgoščevalno funkcijo SHA1.
2. Virtualni disk smo nato izklopili iz forenzične naprave in ga vstavili v osumljenčevo virtualno napravo. Zagnali smo to napravo, ki se je zaradi vstavljenega živega DVD-ja zagnala v trenutno testirano distribucijo operacijskega sistema Linux, ki je namenjena izvajanju mrtve analize.
3. Particijo osumljenčevega diska smo priklopili v distribucijo za izvajanje mrtve analize z ukazom:

   ```
   mount -t ntfs-3g -o ro /dev/sdb1 /mnt/sdb1
   ```

4. Na particiji smo poskušali shranjevati in brisati datoteke - preko konzole kot standardni uporabnik, preko konzole v privilegiranem načinu (*ang. superuser*) in preko GUI vmesnika.
5. Nato smo sistem ugasnili, virtualni disk izklopili, ga vstavili nazaj v forenzično virtualno napravo, ponovno naredili sliko diska in izračunali zgoščeno vrednost SHA1 na enak način kot pred testiranjem.
6. Osumljenčev disk smo zopet vstavili nazaj v njegovo virtualno napravo, zagnali operacijski sistem iz živega DVD-ja in na enak način kot v koraku 3 priklopili disk v operacijski sistem. Namesto koraka 4 smo tokrat

poskušali odpirati nekaj datotek in s tem povzročiti spremembe na disku (spremembe v zadnjih dostopnih časih). Ko smo končali, smo ponovili korak 5.

Prišli smo do zelo presenetljivih rezultatov. Zgoščene vrednosti SHA1 so bile v vseh primerih med seboj ekvivalentne in enake prvotni vrednosti, ki je bila izračunana pred testiranji. Vsebina diska se ni spremenila niti med priklapljanjem diska niti med poskusom ustvarjanja/brisanja datotek in niti med poskusi spreminjanja zadnjih dostopnih časov z odpiranjem datotek. Prav tako v nobenem primeru nismo uspeli ustvariti datoteke na enostaven način preko GUI-ja, kot je to uspelo avtorjem članka v operacijskem sistemu Knoppix [6]. Ker smo bili že zelo skeptični do izračunanih zgoščenih vrednosti, smo na koncu priklopili disk osumljenca kar v forenzični operacijski sistem Debian, tokrat z vsemi pravicami (tako branja kot pisanja), in izbrisali nekaj datotek. Ponovno smo ustvarili sliko diska in tokrat le dobili drugačno zgoščeno vrednost SHA1. Rezultati vseh zgoščenih vrednosti so vidni v Tabeli 1:

zgoščena vrednost	ime slike
3c8547593b0c36 42e2011d1c1154 3e28f977b2be	suspect_test.raw
3c8547593b0c36 42e2011d1c1154 3e28f977b2be	suspect_knoppix_ro_priklop.raw
3c8547593b0c36 42e2011d1c1154 3e28f977b2be	suspect_knoppix_ro_odpiranje.raw
3c8547593b0c36 42e2011d1c1154 3e28f977b2be	suspect_kali_ro_priklop.raw
3c8547593b0c36 42e2011d1c1154 3e28f977b2be	suspect_kali_ro_odpiranje.raw
d4e0228597111e 2eadcc75a1d706 7d32dc4b343d	suspect_debian_rw_brisanje.raw

Tabela 1: Izračunane zgoščene vrednosti SHA1 slik diskov pred (prvi primer) in po izvedbi testiranj.

Rezultati so nas presenetili in so v popolnem nasprotju z rezultati iz referenčnega članka in posledično s pričakovanimi rezultati. V referenčnem članku so v vseh primerih izračunali različne zgoščene vrednosti SHA1. Kljub vsemu trudu posnemanja metodologije, se je naša metoda precej razlikovala od originalne. V prihodnosti bi bilo potrebno izvesti testne scenarije na pravi napravi ter s pravim trdim diskom in pravim forenzičnim orodjem (preprečevalnik pisanja po disku), torej brez uporabe virtualnih približkov. Šele s tako analizo bi lahko zanesljivo potrdili oziroma ovrgli naše rezultate in sploh samo ustreznost prilagojene metodologije. V tem trenutku pa lahko rezultate razglasimo zgolj kot nezadostne in neustrezne za oblikovanje korektnih sklepov.

5. ZAKLJUčEK

Zagonski medij za forenzično preiskavo okolja lahko uporabljamo za izvedbo žive in mrtve analize. Živa analiza za preiskovanje uporablja osumljenčev operacijski sistem in njegove vire. Na drugi strani mrtva analiza za preiskovanje uporablja operacijski sistem iz prenosnega medija in ne osumljenčevega ter z njim povezanih virov. Pri mrtvi analizi napravo izklopimo, zato nekatere podatke o njenem trenutnem stanju izgubimo. Oba pristopa omogočata izvedbo statične analize, ki je zajetje slike diska.

Največja prednost, ki jo prinaša živa analiza, je zajetje trenutnega stanja naprave in operacijskega sistema, ki se na napravi izvaja. Nadalje je živa analiza v primerjavi z mrtvo poceni. Na primer, pri napadu na strežnik preiskava zahteva zajetje strežniških podatkov in je včasih nepredstavljivo za nekaj ur izklapljati strežnik, kot bi to bilo potrebno pri mrtvi analizi. Živa analiza prinaša tudi nekatere slabosti. Te so zaganjanje preiskovalnega forenzičnega okolja, pridobivanje slike diska in viri lažnih podatkov. Obstaja več pristopov za izvajanje žive analize. Ti so standardni uporabniški vmesnik, uvožena orodja, modificiran sistem in dodatna strojna oprema.

Prednost mrtve analize je, da podobno kot živa omogoča preiskovanje sistema že na prizorišču zločina in nima potrebe po razstavljanju računalnika in ustvarjanju slike diska. Nadaljnje je mrtva analiza v primerjavi z živo analizo manj tvegana v smislu spreminjanja podatkov, saj pri njej poganjamo ločen operacijski sistem, ki za svoje delovanje ne potrebuje obstojnega podatkovnega nosilca. Prednost mrtve analize je tudi krajši čas nedelovanja sistema v primerjavi s statično analizo. Slabost mrtve analize je, da je za njeno izvedbo potrebna zaustavitev računalniškega sistema. Poleg tega operacijski sistem Linux ne zagotavlja, da se na podatkovnem disku, nastavljenem na ekskluzivno bralen način, ne bo vršilo pisalnih operacij.

Ločimo več orodij za izvedbo mrtve analize. Primeri so Helix3 Pro, Knoppix in Kali Linux. Glede na rezultate eksperimenta referenčnega članka, nobeno izmed prej omenjenih okolij ne ohranja enake zgoščene vrednosti SHA1 slike diska pred in po uporabi orodij. Tudi mi smo izvedli podoben eksperiment, v katerem smo za orodji Kali Linux in Knoppix preverili, če dobimo enake rezultate. Rezultati so bili v popolnem nasprotju z rezultati iz referenčnega članka in posledično s pričakovanimi rezultati. Vse zgoščene vrednosti SHA1 so se ujemale.

Četudi smo poskusili čim bolj posnemati metodologije iz referenčnega članka, se je naša metoda precej razlikovala. V prihodnosti bi bilo potrebno izvesti testne scenarije na pravi napravi ter pravim trdim diskom in s pravim forenzičnim orodjem, ne pa zgolj z virtualnimi približki. Šele s tako analizo bi lahko zanesljivo potrdili oziroma ovrgli naše rezultate in opravili podrobnejšo analizo.

6. LITERATURA

[1] Honeypot. `http://www.honeynet.org/`. Zadnjič dostopano 11. 5. 2015.

[2] Honeypot (computing). `http://en.wikipedia.org/wiki/Honeypot_%28computing%29`. Zadnjič dostopano 11. 5. 2015.

[3] F. Adelstein. Live forensics: Diagnosing your system without killing it first. *Commun. ACM*, 49(2):63–66, Feb. 2006.

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

[5] B. Hay, M. Bishop, and K. Nance. Live analysis: Progress and challenges. *Security Privacy, IEEE*, 7(2):30–37, March 2009.

[6] S. Maxim. Linux for computer forensic investigators: pitfalls of mounting file systemst. `http://www.computer-forensics-lab.org/pdf/Linux_for_computer_forensic_investigators.pdf`. Zadnjič dostopano 11. 5. 2015.

[7] A. F. A. L. Mohamed, A. Marrington, F. Iqbal, and I. Baggili. Testing the forensic soundness of forensic examination environments on bootable media. *Digital Investigation*, 11, Supplement 2(0):S22 – S29, 2014. Fourteenth Annual {DFRWS} Conference.

[8] S. Mrdovic, A. Huseinovic, and E. Zajko. Combining static and live digital forensic analysis in virtual environment. In *Information, Communication and Automation Technologies, 2009. ICAT 2009. XXII International Symposium on*, pages 1–6, Oct 2009.

[9] P.-H. Yen, C.-H. Yang, and T.-N. Ahm. Design and implementation of a live-analysis digital forensic system. In *Conference: Proceedings of the 2009 International Conference on Hybrid Information Technology*, 2009. {ICHIT} 2009.

VMI-PL: Jezik za nadzor navideznih platform z vpogledom v navidezne stroje

Jan Berčič
Fakulteta za matematiko in fiziko in
Fakulteta za računalništvo in informatiko
Univerza v Ljubljani
jb6413@student.uni-lj.si

POVZETEK

V računalniški forenziki imamo vse večkrat opravka s sistemi, ki na tak ali drugačen način tečejo v navideznih strojih, npr. v oblaku ali drugačnih okoliščinah z deljeno strojno opremo. Navidezne platforme omogočajo nevidno oz. skoraj nevidno opazovanje delovanja sistema, kar lahko močno olajša pridobivanje dokazov ter hkrati ne spreminja dokaznega gradiva (t.j. obnašanja programske opreme v opazovanem stroju).

Kategorije

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

Pojmi

Jeziki, Merjenje, Pravni aspekti, Preverjanje

Ključne besede

VMI-PL, navidezni stroji, instrumentacija, introspekcija, programski jezik, varnost, virtualizacija

1. UVOD

Pri zbiranju forenzičnih dokazov smo večkrat soočeni s problemom opazovanja obnašanja sistema, ki ga hkrati ne smemo spreminjati. Z običajnimi metodami opazovanja programske opreme sistem pod drobnogledom nujno tudi spremenimo, saj v svrho diagnostike spreminjamo okolje, v procese vstavljamo nadzorno kodo, spreminjamo jedro sistema (oz. z njegovo pomočjo opazovanje izvajamo), itd. V primeru navideznih strojev pa se nam ponudi priložnost, da tarčni stroj opazujemo na tak način, da to njegovim programom ni opazljivo (oz. je zelo težko in nedeterministično).

Avtorji članka [1] so razvili domensko specifičen jezik, VMI-PL[1], namenjen prav opazovanju navideznih strojev, ki ga v tem članku predstavijo; vpeljejo tudi novo klasifikacijo tehnik instrumentacije navideznih strojev, ki sloni na glavnih paradigmah opazovanja, omenjenih spodaj. Njihov jezik deluje na podlagi običajnih osnovnih primitivov, kot so npr. branje spomina stroja, pisanje v spomin, zajemanje tokov podatkov v stroj in iz njega ter opazovanje nižjenivojskih dogodkov procesorja (predvsem prehodi med uporabniškim in jedrnim kontekstom). Na podlagi teh primitivov so potem zgrajene pomensko bogatejše višjenivojske funkcije v smislu sledenja sistemskim klicem, življenskemu ciklu procesov ipd.

[1]Virtual Machine Introspection Probe Language

2. OBIČAJNI PRISTOPI

Programe lahko opazujemo na več načinov, ki se razlikujejo predvsem po vsiljivosti, količini informacij, ki jih lahko z njimi pridobimo, ter logični lokaciji — t.j. kje so glede na opazovani program (v njem, na istem računalniku, zunaj računalnika).

2.1 Vstavljanje/spreminjanje kode

Najbolj vsiljiv način instrumentacije je, da v program preprosto dodamo lastno kodo. Le-ta lahko z delujočim procesom naredi marsikaj, npr. s prepisovanjem kode spreminja delovanje, vodi evidenco o odprtih datotekah, na lahek način doda možnost, da prestrezamo in spreminjamo vse tokove podatkov, s katerimi program komunicira s svetom, itd. Ob primernem nivoju nadzora nad gostiteljskim sistemom (skrbniški dostop) je tudi zelo lahko izvedljiva, sploh na sistemih, temelječih na mikrojedru Mach in podobnih. Pri teh namreč lahko iz enega procesa v drugem „od zunaj“ ustvarjamo nove niti, alociramo pomnilnik, ipd.

Poleg dejstva, da nadebudni program tovrstno modifikacijo trivialno odkrije in s tem potencialno prilagodi svoje obnašanje, je ta metoda manj dopustna tudi zato, ker seveda direktno spreminjamo dokazno gradivo. Morebitne zlonamerne učinke kode smo namreč lahko povzročili kar s svojim vstavkom, ki potem na nepredvidljive načine reagira s programom.

2.2 Razhroščevalniki

Vsem znan pristop k odkrivanju napak v programu z znano, ali pa tudi neznano, kodo, je takisto zelo vsiljiv in za opazovani program lažje zaznaven. Na tak način seveda dokazno gradivo (t.j. živ sistem s sumljivimi programi) tudi uničimo, zato je uporaben predvsem kot pomoč pri „offline“ analizi delovanja programa, ko posamezne majhne dele kode spreminjamo, med delovanjem pa nas zanima predvsem obnašanje glede na identificirane spremenljivke. Po kompleksnosti

imamo razpon od osnovnih razhroščevalnikov, kot je `gdb`, do zelo zapletenih orodij tipa IDA, ki so sposobna poleg vseh običajnih operacij med delovanjem programa računati tudi graf klicev funkcij ter vodenje (in upoštevanje pri prikazu) metapodatkov o različnih strukturah v programu.

2.3 Sledilniki izvajanja

So za opazovani program že manj intruzivni, saj so izvedeni povsem zunaj procesa, vendar še vedno zaznavni. Pri njih povečini že začnemo izgubljati informacije o opazovanjem programu (podatkovne strukture), lahko pa hitreje ugotovimo, če se pot izvajanja v programu kaj in koliko spremeni glede na spremenjene zunanje parametre. Sledilniki, kot je npr. OProfile na sistemu Linux, imajo zelo majhno ceno izvajanja, ker lahko uporabljajo strojne števce za izvajanje ukazov.

Na drugi strani imamo sledilnike, ki imajo osnovno paradigmo ekvivalentno jeziku VMI-PL, kot je npr. ogrodje `dtrace` na sistemih Solaris, OS X, FreeBSD in nekaterih drugih. Pri tem takisto definiramo odzive na posamezne dogodke, ki so lahko semantično precej bogati, saj vključujejo sistemsko znanje o procesu/-ih (življenski cikel procesa, naložene knjižnice, sistemski klici). Ogrodje je izpeljano v jedru sistema, opazovanega programa ne spreminja, je pa še vedno lastno stroju, ki sistem izvaja.

2.4 Navidezni stroji in semantična luknja

Na tem nivoju smo od programa najbolj oddaljeni. To je dvorezen meč: po eni strani je za opazovani sistem v takem okolju najtežje ugotoviti, če je opazovan ali ne, po drugi strani pa izgubimo vse višjenivojske informacije. Ne poznamo več povezave med „programi“ in posameznimi nitmi izvajanja na nivoju procesorja, za kakršnokoli ugotavljanje uporabniku prijaznih informacij (npr. čas izvajanja procesa, da o kakršnihkoli imenih naloženih izvedljivih datotek ali gonilnikov niti ne govorimo) je potrebno zelo intimno poznavanje jedra sistema, če je le-to seveda katero od standardnih in ne povsem prilagojeno s strani osumljenca, ipd. Temu razkoraku med metapodatki, ki jih lahko predloži opazovalni sistem zunaj stroja in metapodatki, ki jih o sebi lahko da sistem sam, pravimo *semantična luknja*[2]

Avtorji predlagajo razdelitev načinov takega „opazovanja od zunaj“ na te kategorije:

Podatkovno usmerjeno opazovanje izvajamo z branjem spomina (delovnega spomina, diskov) in pisanjem vanj. Na ta način lahko pridemo do kode, ki se v sistemu izvaja, ter podatkov, ki jih obdeluje. Zaradi zelo revnih informacij o strukturi obeh virov je to lahko analitično zelo zahtevno dokazno gradivo.

Dogodkovno usmerjeno opazovanje temelji na prestrezanju dogodkov, ki jih navidezni stroj lahko sporoči. To so predvsem prehodi med uporabniškim in sistemskim kontekstom[3] ter preklopi med nitmi izvajanja. Odvisno od podpore navideznega stroja so dogodki lahko tudi spremembe registrov procesorja (uporabni so predvsem tisti, za katere je potreben sistemski kontekst) ter strojne prekinitve — dogodki, ki jih sporočajo ostale navidezne naprave v sistemu.

Tokovno usmerjeno opazovanje je opazovanje tokov informacij, ki si jih navidezni stroj izmenjuje s svetom. To so lahko tokovi skozi omrežne vmesnike, komunikacija s periferno strojno opremo, ipd.

Težave s semantično luknjo odtehta dejstvo, da s to tehniko nikakor ne spreminjamo opazovanega sistema; seveda ob predpostavki, da uporabljena virtualizacijska rešitev podpira potrebne funkcije.

3. VMI-PL

VMI-PL je dogodkovno voden, sintaktično pa modeliran podobno, kot recimo jezik `awk`: ima začetni blok, v katerem izrazimo nastavitve, sledijo pa mu bloki, ki se izvajajo ob posameznih dogodkih. V teh blokih imamo na voljo funkcije za pregledovanje spomina navideznega stroja (naslovi so primerno prevedeni za spominsko shemo trenutnega procesa) ter pisanje vanj.

```
Configuration {
  <configuration-item>
}
<event-based-probe>(<parameters>) {
  <data-based-probe>(<parameters>)
  <filter>(<parameters>) {
    <data-based-probe>(<parameters>)
  }
}
```

Slika 1: Primer sintakse VMI-PL

Avtorji so se jezik odločili implementirati kot razširitev za Linuxov navidezni stroj KVM. Implementacija je razdeljena na dva dela:

- izvajalno okolje za VMI-PL, ki je, zanimivo, spisano v Pythonu, ter
- KVM modul, ki implementira vse potrebne posege v KVM, da je nadzor stroja mogoč.

Izvajalno okolje je običajen uporabniški program, ki prevede dano VMI-PL skripto ter jo pošlje navideznemu stroju kot konfiguracijo, s katero se nastavijo vse potrebne sonde in ostali parametri. Oba dela sta povezana z Linuxovim psevdo-mrežnim vmesnikom `netlink`.

Implementirane sonde so treh vrst, v skladu s klasifikacijo zgoraj, vsaka pa ima lahko povezane filtre za primer, da nas ne zanimajo vsi dogodki, ki jih sonda lahko generira.

Podatkovne sonde, s katerimi pregledujemo spomin navideznega stroja.

Dogodkovne sonde, s katerimi lahko lovimo sistemske dogodke, kot so prehodi iz uporabniškega v sistemski kontekst, ipd.

[2] Avtorji članka temu originalno pravijo *semantic gap*.
[3] Angl. *user-space* in *kernel-space*.

Tokovne sonde, s katerimi lahko opazujemo tokove mrežnih in drugih komunikacijskih kanalov s svetom.

3.1 Zmogljivost

Za primerjavo zmogljivosti se avtorji osredotočijo na ogrodje VProbes podjetja VMware. V obeh primerih (t.j. VProbes in VMI-PL) so v gostujočem sistemu izvedejo neko predvidljivo aktivnost ter izmerijo čas izvajanja v treh ali več scenarijih, med katerimi so seveda vedno:

- izvajanje z izklopljenim instrumentacijskim ogrodjem,
- izvajanje z vklopljenim ogrodjem, vendar brez aktivnih sond,
- izvajanje z vklopljenim ogrodjem in aktivno vsaj eno sondo.

Med drugim seveda pridejo do očitnega rezultata, da kakršnakoli vklopljena instrumentacija upočasni delovanje navideznega stroja (npr. spremljanje življenjskega cikla procesov ima povprečno upočasnitev 21,3%). Pomemben rezultat pa je, da ima njihova implementacija VMI-PL opazno nižjo ceno izvajanja in posledično manjši vpliv na gosta, kot VProbes.

Vsekakor manj očitno je, zakaj točno do teh razlik pride — je za to kriva implementacija VProbes, VMwarov navidezni stroj ali pa morebiti okornejši programski vmesnik VProbes. Avtorji o tem, razumljivo, ne zgubljajo besed, saj je VMwarova rešitev zaprta in torej na ta način neprimerljiva s čimerkoli.

4. LITERATURA

[1] F. Westphal, S. Axelsson, C. Neuhaus, and A. Polze. Vmi-pl: A monitoring language for virtual platforms using virtual machine introspection. *Digital Investigation*, 11:85–94, 2014.

Del IV

Druga področja digitalne forenzike

Impacts of increasing volume of digital forensics data: A survey and future research challenges

Bojana Todorović[*] BT0599@student.uni-lj.si
Peter Kragelj[†] peter.kragelj@gmail.com
Borja Bovcon[‡] gojace@gmail.com

ABSTRACT
Rapid development of digital media devices and its availability has contributed to the development of cyber-crime. In order for police to cope with the increase of cyber-crime new methods for digital forensic need to be developed. In this paper we will present and discuss methods of digital forensic analysis and current problems in this field.

Keywords
digital forensic, volume data challenge, proposed solutions, data mining, triage, data reduction, intelligence analysis, data analysis, artificial intelligence, methodology

1. INTRODUCTION
Popularity of digital devices and relatively low prices of storage media are main reason for collection huge amount of data. With growing volume of data, also grow number of cases for forensic analysis. During the years, has been variety of research undertaken in relation to the volume challenge [31].

There are multiple proposed approaches for solving increasing volume of forensic data. First step is process of triage on the crime scene. Than there are methods like data mining, data reduction and subsets that are trying to decrease volume of data that examiners have to check. Intelligence analysis can provide examiner with new more specific hypothesis that can shorten time needed for investigation, but can also have long-term effect providing early warning of threats that might force high level decisions to allocate resources to different types of crime. There are also proposed solutions for research speed up from parallel computing, visualization, digital forensics as a service or even artificial intelligence [31].

[*]Survey
[†]Proposed solutions
[‡]Discussion

2. SURVEY
Survey is general look over parts, features or contents. During more than 15 years was written many papers on problem with forensic data, such as volume of data. We divide research on two parts: volume of data form 1999 - 2009 and volume of data from 2010 - 2014. According to the existing literature on volume of data in digital forensic examination, huge problem is amount of data and digital forensic is relatively new discipline. Those factors are 'stumbling stone' in digital forensic investigation.

2.1 Volume of data 1999 - 2009
Information plays important role in today's society. Having right information at the right moment is priceless. Development of technology has led to huge amount of data, and every year is increasing which make more difficult for, i.e. proper forensic investigation.

McKemmish (1999) [26] wrote that, accurate and timely processing of large volumes of data provides the challenge for the forensic investigators. Problem with increasing information stored on computers mainly happening because of increased use of multimedia, combined with the rapid expansion of the Internet. Palmer (2002) [30] did research the need for "incorporating scientifically based approaches to conducting forensic analysis in the digital world rather than developing digital technologies and then adapting them to benefit from forensic analysis techniques." The digital forensic information must be relevant and credible. Challenges of large investigation becomes serious, for example just keeping track of it and maintaining records for the purposes of continuity of custody, wrote Sommer (2004) [42]. Hard-disk sizes continue to increase as prices fall.

Eexponential growth of high-capacity storage devices vs. linear growth of the I/O transfer is gap known as the "megahertz wars" according to Roussev and Richard (2004) [38]. Also, they point out the increased sophistication of digital forensics analysis, for example "the identification and extraction of data hidden using steganography, and speech recognition technology to analyze voice message." Ferraro and Russell (2004) [13], pointed three issues, and last discussed issue "is the burgeoning problem of storing and returning evidence years after its seizure".

"The computer forensics discipline is somewhat unique as it serves several different communities," military, law enforcement, private sector, public sector, ... Rogers and

Seigfried (2004) [36]. They did survey on 60 persons, and found out that most common issues are issues facing the discipline are: education/training/certification, technology, encryption, data acquisition, tools and so on. One year later, Brown et al. (2005) [7], wrote on the challenges in digital forensic and he stated that finding and discovering "forensically interesting, suspicious, or useful patterns" is analogous to "needle-in-the-haystack". Mainly, because a digital forensic investigation can involve a large number of data/evidence derived from a variety of sources as images, network packets, videos, music, etc. Data rates are doubling every nine months – twice as fast as Moore's Law and no proper data mining techniques were applied to solve this problem, Beebe and Clark (2005) [5]. Shannon (2004) [40], "forensic practitioners are often faced with enormous of data, sub-optimal searching strategies, and incomplete information".

Adelstein (2006) [1] point out using live digital forensic instead of traditional digital forensic, because with the increase in the quantity of data it will soon become impossible to acquire all disk data related to a case. According to Alink at al. (2006) [2], a typical digital forensic investigation involves four phases: media capture (forensic disk duplicate), feature extraction (parsing file systems, chat logs, ...), analysis (browsing, querying, correlating) and reporting (writing down findings for court). Data processed in this way are huge, so that diversity on a typical hard disk is overwhelming.

Case (2008) [9] stated that average case size has tripled in 3 years (according to FBI statistics). Tools are mostly focused on specific task and that makes time-consuming for investigator to have wider view on investigation. First generation of computer forensic tools rapidly becoming outdated, they are not keeping step with increased complexity and data volumes of modern investigations, Ayers (2009) [3]. Forensics need tools for reconstruction, analysis, clustering, data mining, and sense-making. Such tools frequently require the development of new scientific techniques, Garfinkel et al. (2009) [17].

2.2 Volume of data 2010 - 2014

"Golden Age of Digital Forensic"is quickly coming to the end. Garfinkel (2010) [18] give an brief overview of digital forensic through roughly forty years. The early days of digital forensic were mainly preformed ad hoc, due to small amount of data. Then came the Golden Age of digital forensic, roughly from 1999 – 2007, marked by a rapid growth in digital forensics research and professionalization. Today, digital forensic is facing a crisis. Last decade's progress is becoming irrelevant, mainly because of: the growing size of storage devices, the increasing prevalence of embedded flash storage, propagation of operating systems and and file formats, universal encryption. Garfinkel (2012) [16] wrote "somewhat surprisingly, it has been difficult to apply many 'big data' solutions from other fields to digital forensics". Problem in digital forensic tool development is the amount of data that must be processed and the never-ending battle with storage and performance bottlenecks.

Jones et al. (2012) [22] proposed random sampling of files and applying statistical estimation to the results in investigations involving child abuse material, and has been able to reduce backlogs from three months to 24h. As an example, the London Metropolitan Police Service Digital Electronics and Forensic Service (MPS DEFS) currently receives over 38,000 digital devices per annum to be examined by a team of approximately 80. Some of these devices will have a storage capacity of gigabytes of terabytes, Overill et al. (2013) [29]. Another example is the average amount of data per case, as experienced by FBI's 15 Regional Computer Forensic Laboratories, has grown 6.65 times (from 84 GB to 550 GB) in eight years (2003 – 2011, RCFL), pointed Roussev (2013) [37]. Further, more computing enables the cheap processing and storing of more data, which indirectly demands proportionally more compute resources to be deployed for forensic purposes. Shaw and Browne (2013) [41], suggest, for big data problem, solution in term of digital forensic triage. The term is poorly defined and means different thing to different people. Mostly, triage refers to the practice of eliminating digital devices from the process of forensic analysis, or simply prioritising the order in which devices are examined.

Casey (2013) [11] mention the trend, on demands for digital forensic laboratories, will continue. Also, the amount of data received for each examination is increasing significantly. Handling hundreds of thousands of files is a major challenge in today's digital forensic, in order to deal with this investigators often apply hash functions for automated input identification, Breitinger et al. (2014) [6].

2.3 Growth of media

As survey of data showed, forensic's data are constantly increasing and partially because of increased storage media. Moore's Law is observetion that the number of transistors per square inch on integrated circuits had doubled every year since the integrated circuit was invented . Some studies have shown physical limitations could be reached by 2017. Roussev et al. (2013) outline the acquisition rates based on maximum sustained throughput of hard droves to approximate that, in 2003 a 200 GB hard drive should take about 1 h to image at 58 MB/s, and in 2013 a 3 TB hard drive to take almost 7 h at 123 MB/s, but they then state that in reality a 3 TB hard drive actually took over 11 h to image [37]. In recent years, use of mobile portable devices (mobile phones and tablets) is in increase, as well increase in criminal cases, and those devices are special case which are look for special care. Due to large variety of manufactures and models it is make difficult to copy and examine all data.

In Figure 1, is presented general trend in growth volume of Hard Drives in MB, RAM in MB and average FBI cases size. It is graphical representation of survey section. "A logarithmic scale base 2 is used as the increase in hard drive size is greater than the RAM and FBI sizes, skewing the chart when a standard scale is used" [31].

Today, average user have opportunity to use Internet and collect large amount of data. As the size of media increases, users are storing more, including; every e-mail, document, spreadsheet, picture, and video they have, and as storage is easily purchased and inexpensive, there is little need to organise their data [31]. As we saw in survey section, fact is digital forensic data is growing and it will continue.

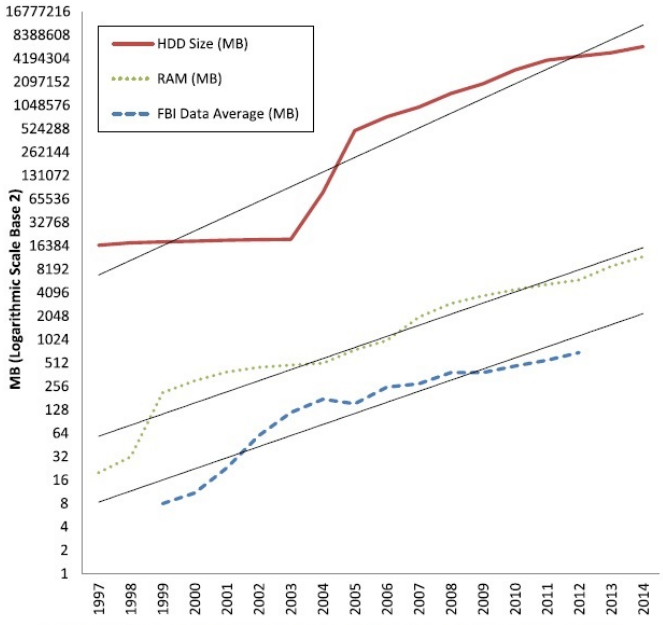


Figure 1: General trend in HDD (MB), RAM (MB) and FBI case size (MB) between 1997 and 2014. [31].

3. PROPOSED SOLUTIONS

With the growing sizes of databases, law enforcement and intelligence agencies face the challenge of analyzing large volumes of data involved in criminal and terrorist activities [23]

Dramatic rise in the numbers of digital crimes committed have led to the development of a whole slew of computer forensic tools. These tools ensure that digital evidence is acquired and preserved properly and that accuracy of results regarding the processing of digital evidence is maintained [25]. Such tools exist in the form of computer software and have been developed to assist digital investigators conduct a digital investigation.

For a very long time, the standard digital forensics procedure has been to seize all the physical media to analyse images acquired from the media. This practice, however, has been criticized because it is thought that a defendant's privacy could be excessively infringed and seizing of all physical media can disrupt corporate business activities. It is the right of a defendant to be secure from indiscreet search and seizure by governmental authorities. Owing to the high risk of privacy infringement and business disruption, restrictions on the search and seizure of digital evidence have been increasingly tightened.

The invisible characteristic of digital information, unlike any tangible object, results in law enforcement officers reviewing the contents of files on any digital media during the search and seizure process, inevitably resulting in privacy infringement against defendants. The reality is also that, owing to limitations in terms of labor, technology, and the equipment being utilized during the search and seizure process, strict limitations are imposed to directly select only incriminating data at the crime scene.

Therefore, it is regarded as reasonable for the law enforcement officers that excessive restrictions on the search and seizure of digital information could downgrade the efficiency of investigations and may hinder the finding of important evidence.

Because of this, a most reasonable method that can balance the two contradicting values - protection of human rights and achievement of an effective investigation - is urgently needed. Up to the present time, few research efforts have dealt with on this problem in the digital forensics field [20].

3.1 Data mining

Data mining is the process of extracting useful information from large datasets or databases [31]. It is a process that uses various statistical and pattern-recognition techniques to discover patterns and relationships in data [27].

It is also a step in wider process of Knowledge Discovery in Databases (KDD - Figure 2) - a process of understanding data, including methods and techniques, which are addressed by mapping low-level data which is too large to easily understand, using specific data mining methods relating to pattern discovery and extraction.

The KDD process contains nine following steps [12]:

1. learning the domain,
2. creating a target datasets,
3. data cleaning and preprocessing,
4. data reduction and projection,
5. choosing the function of data mining,
6. choosing the data mining algorithm,
7. data mining,
8. interpretation and
9. using the knowledge.

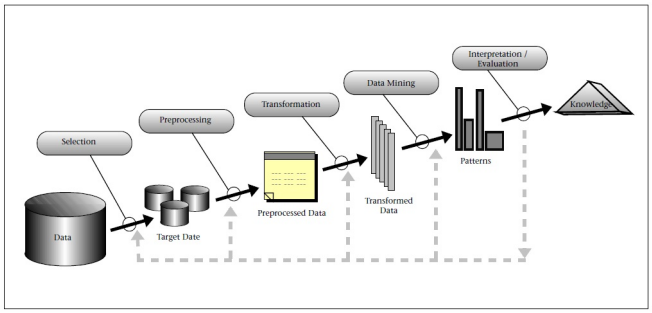


Figure 2: An overview of the steps that compose the KDD process [12].

The formal methodology of data mining includes following basic steps [19]:

- Determine the nature and structure of the representation of the data sets.
- Decide how to quantify the data; compare how well different representations fit the data
- Choose an algorithmic process to optimize the scoring function
- Decide what principles of data management are required to implement the algorithms efficiently.

Data mining has several applications in digital forensics [31].

These include:

- identifying correlations in forensic data (association),
- discovering and sorting forensic data into groups based on similarity (classification),
- locating groups of latent facts (clustering), and
- discovering patterns in data that may lead to useful predictions (forecasting).

While this technique is ideal for association, classification, clustering and forecasting, it is also particularly useful for visualization.

Crime data mining is classified as follows [23].

1. Entity extraction has been used to automatically identify person, login ID, Password, IP of the system, and personal properties from reports or logs.
2. Clustering techniques such as "concept space" have been used to automatically associate different objects (such as persons, organizations, hardware systems) in crime records.
3. Deviation detection has been applied in fraud detection, network intrusion detection, and other crime analyses that involve tracing abnormal activities.
4. Association rule has been applied to finding associations and sequential patterns between web transactions are based on the Apriori Algorithm. Mining results shows motive, pattern and counts of similar types of attacks happened during a period.

Data mining and information sharing techniques are also principal components of the White House's national strategy for homeland security [27].

3.2 Data reduction and subsets

Data reduction is a step prior to data mining [31] and represents a process that minimizes the amount of data that needs to be examined and analyzed in a forensic investigation. It is an automatic or semi-automatic process that can dramatically eliminate redundant data and reduces cost of investigation. Typical techniques of data reduction include data compression, filtering and data deduplication [34].

Compression is the process of encoding information with fewer bits than the uncoded information would use. Compression programs can be used to hide or disguise sensitive data [45].

Filtering may involve removing duplicate files, searching for keywords, or grouping data based on file types. Filtering the data for duplicates, date, and keywords, and transforming the data to a reviewable format can be performed either by examiner using forensic software or other processing tools, or with an electronic discovery database [10].

Deduplication can reduce the load on counsel, who typically will review documents for relevance and privilege prior to production. Deduplication essentially identifies, using some algorithm, that a file presented to the processing tool is a copy of a file already in the data set for review. Once a file is identified as a duplicate of another file already in review, the new file can be suppressed - although the tool should not "forget" that it found the duplicate [10].

3.3 Triage

Triage in computer forensics represents a targeted review of all available media to determine which items contain the most useful evidence and require additional processing [10]. It is normally performed on-scene and can be used to identify the "richest sources" of digital evidence at the scene, allowing the detected media's in-depth examination to be focused and expedited.

Triage is an extremely high-level examination, designed to be fast, not thorough. It is not designed to normally remove media from needing further examination, and only allows for a shallow, focused view of found evidence that helps in the priorization of all the exhibits [39].

On-scene triage inspections may provide to the data necessary for investigators subsequently:

1. Assess the severity of a crime and prioritizing it accordingly.
2. Assess the offender's possible danger to society [35]
3. Obtain actionable intelligence in exigent circumstances (e.g., missing person, military operations, risk of evidence destruction).
4. Identify the richest sources of digital evidence pertaining to an investigation.
5. Identify victims that are or may be at acute risk identify potential charges related to the current situation.
6. Determine whether a certain item requires deeper inspection, such as recovery of deleted information or decoding of encrypted data.

Although information obtained from on-scene triage inspections may resolve certain questions in a case, it is more often just the starting point in an investigation.

Even though many triage models have been widely developed to discover timely probative information and utilize

it in investigations, most of them simply focus on the necessity to quickly find information to provide investigative leads in a relatively short time [35] or the need for useful intelligence in order to prioritize among the huge amount of digital evidence for backlog reduction [8] however, they do not deal with legal requirements such as the demand for privacy protection.

Erin et al. [24] dealt with selective search and seizure from the legal perspective. They proposed a risk sensitive digital evidence collection model to collect digital information selectively from live systems, and emphasized the need for selective search and seizure following the 4th Amendment protection. In the same vein, Turner [43] proposed the Digital Evidence Bag (DEB) methodology to effectively manage selectively chosen data by imaging it.

One example of triage process is The computer forensics field triage process model (CFFTPM) [35].

3.3.1 *The computer forensics field triage process model (CFFTPM)*

The phases of CFFTPM include: planning, triage, usage/user profiles, chronology/timeline, Internet activity, and case specific evidence (see Figure 3). These six phases constitute a high level of categorization and each phase has several subtasks and considerations that vary according to the specifics of the case, file system and operating system under investigation.

Once the appropriate planning has been completed, the investigative process moves to those phases that deal more directly with the actual suspect or crime scene (depending upon the case). Since time is a crucial factor in the CFTTPM, it is extremely important that some sort of initial prioritization be undertaken. An effective and time-tested approach is to follow the medical triage model.

"A process for sorting injured people into groups based on their need for or likely benefit from immediate medical treatment. Triage is used in hospital emergency rooms, on battlefields, and at disaster sites when limited medical resources must be allocated."

For our purposes triage can be distilled down to: A process in which things are ranked in terms of importance or priority. Essentially, those items, pieces of evidence or potential containers of evidence that are the most important or the most volatile need to be dealt with first.

The triage phase is fundamental to the process model and along with proper planning it is the foundation upon which the other phases are built [35].

3.4 Intelligence analysis and digital analysis

Intelligence, narrowed down to law enforcement use, could be described as information that is acquired, exploited and protected by the activities of law enforcement institutions to decide upon and support criminal investigations [44].

Intelligence always involves a degree of interpretation resulting in an inevitable degree of speculation and risk. The

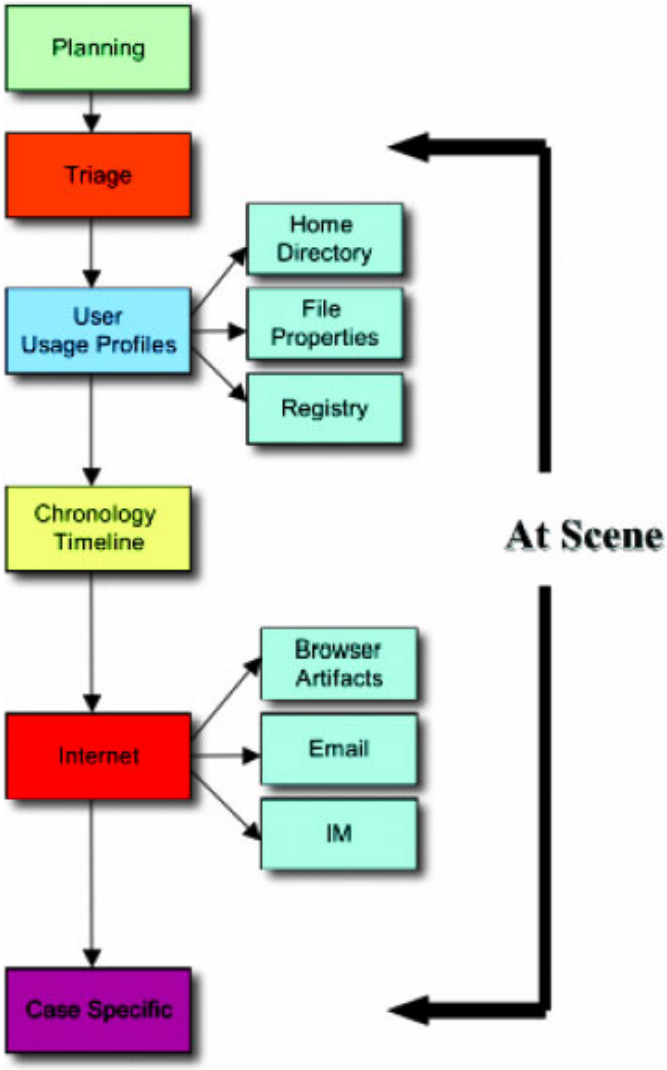


Figure 3: CFFTPM Phases [35].

amount of speculation and risk is dependent upon the quality and quantity of information.

Intelligence is usually divided in two main areas:

3.4.1 *Operational intelligence*

Operational intelligence typically provides an investigative team with hypotheses and inferences concerning specific elements of illegal operations of any sort. These will include hypotheses and inferences about specific criminal networks, individuals or groups involved in unlawful activities, discussing their methods, capabilities, vulnerabilities, limitations and intentions that could be used for effective law enforcement action [44]. It usually aims to achieve a specific law enforcement outcome. This might be an arrest, seizure, forfeiture of assets or money gained from criminal activities, or the disruption of a criminal group [21].

Operational analytical support includes:

- Identifying links between suspects, and their involvement in crimes and criminal activity;
- Identifying key investigative or information gaps;
- Preparing profiles of known or suspected criminals.

3.4.2 Strategic intelligence

Strategic intelligence focuses on the long-term aims of law enforcement agencies. It typically reviews current and emerging trends changes in the crime environment, threats to public safety and order, opportunities for controlling action and the development of counter programmes and likely avenues for change to policies, programmes and legislation [44]. It is intended to inform higher level decision makers, and the benefits are realized over the long term.

The intention is to provide early warning of threats and to support senior decision makers in preparing their organizations to deal with emerging criminal issues. This might include allocating resources to different types of crime, or increasing training in a particular crime-fighting technique [21].

Strategic analysis includes the identification of:

- Modus operandi - criminals habits of working;
- Crime trends and patterns;
- Emerging threats;
- The potential impact of external factors such as technology, demographics or economics on crime.

Although these research efforts propose various ideas and techniques, more discussions are needed to help law enforcement officers solve problems they could face at crime scenes during the selective search and seizure process.

Decisions made by law enforcement officers at a crime scene include those as to what extent of any data encountered are related with the criminal activity as well as various measures for ruling out irrelevant data. The problem is that if the decisions are deemed unreasonable, the admissibility of the evidence can be denied by the criminal courts.

Therefore, objective standards are required to determine whether the decisions are reasonable, and those standards should be established with full understanding of the situation. Standards that lack a holistic consideration of the crime scene are merely meaningless and unrealistic demands [20].

3.5 Other proposed solutions

There have been also other proposed solutions to the issue of increasing volume of data seized for analysis. For example distributed and parallel processing, visualisation, digital forensic as a service (DFaaS), and the use of artificial intelligence techniques.

3.5.1 Parallel and distributed processing

Parallel and distributed processing offers a potential to speed up analysis of forensic data. Task like indexing, and searching can be divided across multiple workstations and so reaserch time is reduced [33].

There have been some developments in relation to processing of forensic data, for example Access Data Distributed Network Attack for password recovery. But there also other areas that would also benefit from parallel processing, such as data carving, and timeline reports [31].

Because of all the advantages and necessity, parallel processing and multi-treading is one of major requirements for the second generation of forensic tools [3].

3.5.2 Artificial intelligence

One of the concepts for the future of digital forensics is that digital forensic systems will be able to utilise artificial intelligence to assimilate the contents of digital media, and use inference rules to produce information that can guide an analyst when conducting analysis. Rules can be established to interpret differing data structures, combined with artificial intelligence's ability to learn from each case it is used on, and linkage to a global network of information has potential to assist examiners [31].

3.5.3 Visualisation

Application of non-hierarchical and hierarchical visualisation techniques to folder tree structures of hard disk drives, and conducted experiments comparing visualisation with text searches, concluding that visualisation methods have potential benefits to forensic analysis.

Furthermore timestamp information is a common denominator for the large variety of structured and unstructured data common to digital forensic holdings, and can be used to create visualisations of digital forensic data. Timestamp information from multiple forensic tools can be used to produce timeline visualisations [31].

3.5.4 Digital forensic as a service (DFaaS)

Traditional forensic labs generally make specialists responsible for a variety of administrative tasks, and hence have less time to perform specialised analysis.

In the DFaaS implementation one would have a team of support personnel, responsible for administration of; applications, databases, storage, infrastructure, and other systems. This would free up specialists to perform specialised tasks.

In relation to DFaaS being applied in a real world situation, it is stated that it would be beneficial for digital information to be available within the first few days to investigators, but in traditional systems, this is not the case [46].

4. DISCUSSION

This section consists of two parts. In the first part, we will take a look at general situation regarding digital forensic as a science. We will point out some of the gaps and problems that increasing amount of data is causing and present some

of the opportunities for development in this field. In the second part of this chapter, we will take a look at how digital forensic is progressing in Slovenia and point out some interesting statistical numbers that Slovenian police department annually releases in their brochure [47].

4.1 General

In section 3 we have mentioned quite a few techniques which address the data volume challenge in computer forensic, but up to date not many frameworks that reduce or analyse the large volume of data have been developed or published. Some of the research fields that might be able to cope with the data volume challenge are *Knowledge discovery*, *Knowledge management*, *Data mining* and *Intelligence analysis*.

There are a few significant gaps between digital forensic data and some of the previously listed research fields.
First we shall take a look at a problem connected with data mining. In year 2001 *Beebe* and *Clark* [4] have described their research, which focuses on whether data mining methodologies can be used in digital forensic, but not much progress has been made in this direction since then. There still is a significant gap in relation to applying data mining methodologies to digital forensic data. For digital forensic data we would like a methodology which can be applied to real-world data and which help us achieve desired results. Some of the results that we are after are: a reduction of time in process of data analysis; an appropriate method for archiving the collected data; an appropriate method for gaining knowledge from the collected data. Data mining may provide an overall increase in knowledge, but it is probably not an overall solution to all of the issues raised by the data volume challenge.
Another significant gap relates to the use of intelligence gained during forensic analysis. With the use of intelligence analysis, we could aggregate a large volume of information into common data holdings and provide rapid searches to link associated information and assist in investigations. For example we may find a useful information for a current investigation from data that was stored on a phone which was seized in another, seemingly unrelated, investigation. There has been very little discussion of a methodology to utilise the intelligence gained from one investigation to assist with other investigations, nor to build a body of knowledge inherent in historical cases.
There could be another gap that needs to be analysed. This is the gap between processing times of data. To analyse this, we should gather informations about average processing times of average evidence amounts per year using common processing techniques, recording timeframes of processing and extrapolating this to hard drive sizes of the time. Such information could reveal whether processing techniques are improving analysis timeframes.

In 1999 *McKemmish* [32] addressed another problem related to digital forensics. He stated, that the law is too slow to address issues relating to digital forensic analysis. That means, that any new methodologies must abide to the legal environment to which they are applied and slow pace of changes in law makes it difficult to implement unique solutions. This problem indirectly creates a so called research gap between effectiveness of the various approaches and legal acceptance of used approaches. In 2001 *Spafford* [15] stated that beside technical and legal problems we have to cope with social and procedural problems too. Procedural problem relates to the procedural guidelines of collecting and examining the data. Social problem relates to storing the date for long periods of time - we could use those resources to address other issues instead of storage concerns.

There is a lot of talk about growing backlogs of cases awaiting analysis. Most of the agencies do not allow to release their actual figures which additionally complicates accurate understanding of the problem. This real-world figures, along with the information relating to device volume, number of devices, timeframes, imaging times, processing times, backlogs, number of cases with data in cloud storage, review times, report volumes and successful presentations in a legal context, could significantly help researchers to understand where to focus their work.

Even though there are many concepts that try to improve triage, imaging, processing and analysis, there is little discussion about scaling these to the volume of current case requirements. Some of the methods are out-dated, meaning that they are meant to be used on a very small amount of data. When we try to extrapolate this out-dated methods to current case sizes it results in a very time consuming processing. It would be appropriate to improve and scale this methods to apply to current real-world data volumes and also make greater use of real-world data corpus. To do so, it would be beneficial to test some of the common forensic tools across a common forensic corpus. Gathered results would then help us determine if there have been any improvements in technology and tools.

Since the digital forensic field is still relatively new, it gives us opportunities for developing a framework with which digital forensic practitioners would be able to apply data mining and intelligence analysis techniques to digital forensic data and by doing so they could reduce the time spent for gathering understanding of seized evidential data. There is also a need for developing digital forensic data mining and intelligence analysis techniques and methodologies. Developed techniques and methodologies need to be tested on test data and real-world data, so we can determine which methodology is the most appropriate in which scenario. An example of data reduction methodology is *Digital Forensic Data Reduction Framework* which is shown in figure 4. Data reduction techniques have opportunity to influence various different stages of forensic analysis and they also have an ability to provide benefits to other areas.

4.2 Digital forensic in Slovenian Police Department

In this section we will take a look how digital forensic is evolving in Slovenia. In Slovenia, digital forensic and investigation of digital crime systematically works since year 1999. Center for digital investigation, which operates within Criminal police Directorate at the General Police was established in 2009 along with four regional departments of digital investigation which are operating in the sectors of Criminal

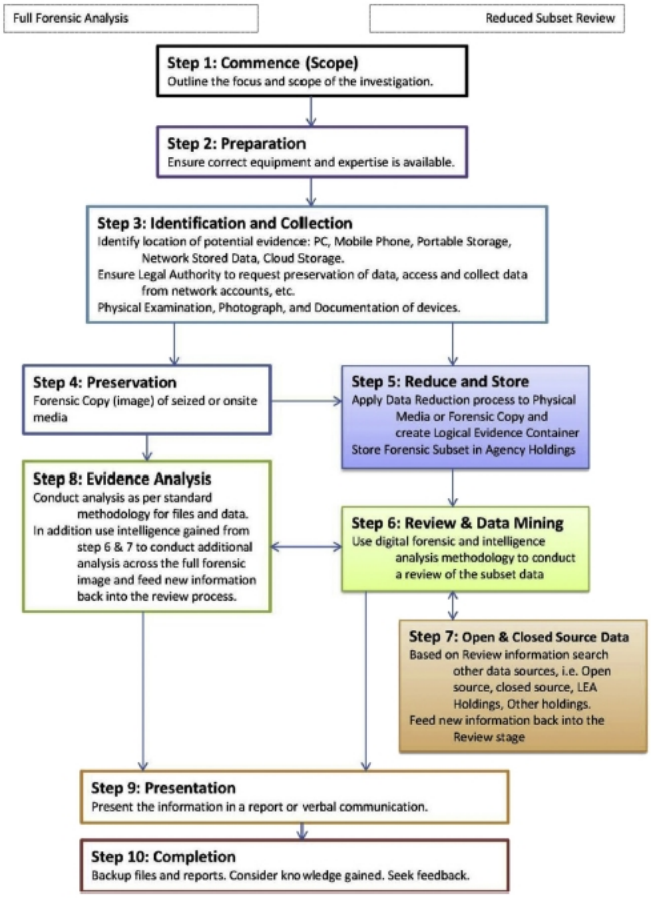


Figure 4: Flowgraph of Digital Forensic Data Reduction Framework.

Police at Police Headquarters Koper, Ljubljana, Celje and Maribor.

Center for digital investigation in Slovenia covers three main areas of work:

- the investigation of computer crime (abuse of personal data, violation of authors' rights, an intrusion in information systems, production and acquisition of software intended for the offense, ...);
- the investigation of seized electronic devices and electronic data;
- professional assistance to other areas (criminal offenses regarding child pornography, online frauds, online racial hatred, tax evasion, corruption, abuse of online marketing and paying, ...)

Because of rapid development of IT areas and new ways and types of digital crime, Slovenian Police Department is trying to keep up with regularly updating of computer hardware, software and training of employees. In year 2013 there were 244 digital crimes in Slovenia and 89 suspects were convicted. In year 2014 there were 168 digital crimes and 145 suspects were convicted. This means that there was a decrees of 38.6% in digital crime. If we take a look a few years back, there was approximately 60 to 110% increase in digital crime each year until now (for example in 2010 there were 107% more digital offenses than in year 2009).

There is also an increasing amount of (pre-trial) seized electronic devices which need to be analysed and investigated. The increase of seized electronic devices is approximately 30 to 35% per year. In year 2014 there were 5228 digital devices seized and 3833 of them were analysed and investigated. The reason for 34% increase in seized and analysed digital devices is an enhanced staffing units for digital investigation and the development and availability of technology for the masses. Digital devices are increasingly being used in the commission of criminal activities or they carry some preserved evidence.

To acquire additional knowledge regarding digital forensic, Slovenia Police cooperate with external institutions in Slovenia as well as foreign institutions abroad. In year 2013 there were 104 executed trainings of employees in the field of computer science and 54 executed trainings in 2014. Even though the number of executed trainings decreased, there were more participants overall in 2014 (8276 participants in 2013 and 14808 participants in 2014).

A big portion of cyber-crime usually cover more than one country. This means that in the same crime investigation different police and judicial authorities may cooperate. The Slovenian police have cooperated in numerous international operations, under which they have dealt with online abuse in the field of child pornography, fraud and hacking. The basis of such joint operations is resulting from the Code of Criminal Procedure and the Convention of Cybercrime, which was legally ratified by Slovenia in 2004. In the said Convention and the Law on ratification, there are defined general principles of international cooperation, exchange of information and mutual assistance between countries.

5. CONCLUSIONS

In digital forensic science is real challenge how to deal with data, how to pick right information in less time as possible. Many reports show that are still unsolved problems which make investigation more time consuming than it should be. Any criminal case needs to be solved with great precision and accuracy in reasonable period of time. Currently, research showed that there is a need for greater impact in digital forensic science.

One of the most promising methods right now is data mining, although we should use it in combination with other methods for better results. The right methodology for data mining, with respect to social standards and law regulations, still needs to be defined. Another promising method is intelligence analysis. We could use this method to find connection of evidence from different seemingly unconnected

crimes. Triage processes are well discussed and they could help us cope with increasing volume of data and devices. In order for triage processes to be successfully developed we would need more actual informations and backlogs of cases.

Digital forensic is progressing in Slovenia as well. By looking at statistics of Slovenian Police department we can see that each year more and more Police employees are required to participate in training abroad and within external institutions in Slovenia. They are also regularly updating and upgrading system hardware and software and every year there are more advertised vacancies for professional digital forensic employees.

6. REFERENCES

[1] F. Adelstein. Live forensics: diagnosing your system without killing it first. *Communications of the ACM*, 49(2):63–66, 2006.

[2] W. Alink, R. Bhoedjang, P. A. Boncz, and A. P. de Vries. Xiraf–xml-based indexing and querying for digital forensics. *digital investigation*, 3:50–58, 2006.

[3] D. Ayers. A second generation computer forensic analysis system. *Digital Investigation*, 6:S34–S42, Sept. 2009.

[4] Beebe and Clark. Dealing with terabyte data sets in digital investigations. pages 3–16, 2005.

[5] N. Beebe and J. Clark. Dealing with terabyte data sets in digital investigations. In *Advances in Digital Forensics*, pages 3–16. Springer, 2005.

[6] F. Breitinger, H. Baier, and D. White. On the database lookup problem of approximate matching. *Digital Investigation*, 11:S1–S9, 2014.

[7] R. Brown, B. Pham, and O. de Vel. Design of a digital forensics image mining system. In *Knowledge-Based Intelligent Information and Engineering Systems*, pages 395–404. Springer, 2005.

[8] G. Cantrell, D. Dampier, Y. S. Dandass, N. Niu, and C. Bogen. Research toward a Partially-Automated, and Crime Specific Digital Triage Process Model. *Computer and Information Science*, 5(2):p29, Feb. 2012.

[9] A. Case, A. Cristina, L. Marziale, G. G. Richard, and V. Roussev. Face: Automated digital evidence discovery and correlation. *digital investigation*, 5:S65–S75, 2008.

[10] E. Casey. *Handbook of Digital Forensics and Investigation*. Academic Press, 2009.

[11] E. Casey, G. Katz, and J. Lewthwaite. Honing digital forensic processes. *Digital Investigation*, 10(2):138–147, 2013.

[12] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. The KDD process for extracting useful knowledge from volumes of data. *Communications of the ACM*, 39(11):27–34, Nov. 1996.

[13] M. M. Ferraro and A. Russell. Current issues confronting well-established computer-assisted child exploitation and computer crime task forces. *Digital Investigation*, 1(1):7–15, 2004.

[14] M. Frank. Kriminalisticna preiskava racunalniške kriminalitete slovenske policije v sodelovanju z ameriškim FBI – informacija z novinarske konference, 2015.

[15] P. GA. Road map for digital forensic research. 2001.

[16] S. Garfinkel. Lessons learned writing digital forensics tools and managing a 30tb digital evidence corpus. *Digital Investigation*, 9:S80–S89, 2012.

[17] S. Garfinkel, P. Farrell, V. Roussev, and G. Dinolt. Bringing science to digital forensics with standardized forensic corpora. *digital investigation*, 6:S2–S11, 2009.

[18] S. L. Garfinkel. Digital forensics research: The next 10 years. *digital investigation*, 7:S64–S73, 2010.

[19] D. Hand, H. Mannila, and P. Smyth. *Principles of Data Mining*. 2001.

[20] I. Hong, H. Yu, S. Lee, and K. Lee. A new triage model conforming to the needs of selective search and seizure of electronic evidence. *Digital Investigation*, 10(2):175–192, Sept. 2013.

[21] INTERPOL. Criminal Intelligence analysis, 2015.

[22] B. Jones, S. Pleno, and M. Wilkinson. The use of random sampling in investigations involving child abuse material. *Digital Investigation*, 9:S99–S107, 2012.

[23] K. K. Sindhu and B. B. Meshram. Digital Forensics and Cyber Crime Datamining. *Journal of Information Security*, 03(03):196–201, July 2012.

[24] E. E. Kenneally and C. L. Brown. Risk sensitive digital evidence collection. *Digital Investigation*, 2(2):101–119, June 2005.

[25] A. Marcella, Jr., and D. Menendez. *Cyber Forensics: A Field Manual for Collecting, Examining, and Preserving Evidence of Computer Crimes, Second Edition*. 2007.

[26] R. McKemmish. *What is forensic computing?* Australian Institute of Criminology, 1999.

[27] J. Mena. *Investigative Data Mining for Security and Criminal Detection*. Butterworth-Heinemann, 2003.

[28] R. P. Mislan, E. Casey, and G. C. Kessler. The growing need for on-scene triage of mobile devices. *Digital Investigation*, 6(3-4):112–124, May 2010.

[29] R. E. Overill, J. A. Silomon, and K. A. Roscoe. Triage template pipelines in digital forensic investigations. *Digital Investigation*, 10(2):168–174, 2013.

[30] G. L. Palmer. Forensic analysis in the digital world. *International Journal of Digital Evidence*, 1(1):1–6, 2002.

[31] D. Quick and K.-K. R. Choo. Impacts of increasing volume of digital forensic data: A survey and future research challenges. *Digital Investigation*, 11(4):273–294, Dec. 2014.

[32] M. R. What is forensic computing? Trends and issues in crime and criminal justice. pages 1–6, 1999.

[33] G. Richard and V. Roussev. Breaking the performance wall: The case for distributed digital forensics. In *Proceedings of the 2004 Digital Forensics Research Workshop (DFRWS 2004), Baltimore, Maryland*, 2004.

[34] M. Rogers and K. C. Seigfried-Spellar, editors. *Digital Forensics and Cyber Crime: 4th International Conference, ICDF2C 2012, Lafayette, IN, USA, October 25-26, 2012, Revised Selected Papers*. Springer, 2013.

[35] M. K. Rogers, J. Goldman, R. Mislan, T. Wedge, and

S. Debrota. Computer Forensics Field Triage Process Model, June 2006.

[36] M. K. Rogers and K. Seigfried. The future of computer forensics: a needs analysis survey. *Computers & Security*, 23(1):12–16, 2004.

[37] V. Roussev, C. Quates, and R. Martell. Real-time digital forensics and triage. *Digital Investigation*, 10(2):158–167, 2013.

[38] V. Roussev and G. G. Richard III. Breaking the performance wall: The case for distributed digital forensics. In *Proceedings of the 2004 digital forensics research workshop*, volume 94, 2004.

[39] K. Ruan, editor. *Cybercrime and Cloud Forensics: Applications for Investigation Processes: Applications for Investigation Processes*. IGI Global, 2012.

[40] M. Shannon. Forensic relative strength scoring: Ascii and entropy scoring. *International Journal of Digital Evidence*, 2(4):1–19, 2004.

[41] A. Shaw and A. Browne. A practical and robust approach to coping with large volumes of data submitted for digital forensic examination. *Digital Investigation*, 10(2):116–128, 2013.

[42] P. Sommer. The challenges of large computer evidence cases. *Digital Investigation*, 1(1):16–17, 2004.

[43] P. Turner. Selective and intelligent imaging using digital evidence bags. *Digital Investigation*, 3:59–64, Sept. 2006.

[44] UNODC. *Criminal Intelligence Manual for Analysts*. United Nation Office on Drugs and Crime, 2011.

[45] J. Vacca and K. Rudolph. *System Forensics, Investigation, and Response*. Jones & Bartlett Publishers, 2010.

[46] R. van Baar, H. van Beek, and E. van Eijk. Digital Forensics as a Service: A game changer. *Digital Investigation*, 11:S54–S62, May 2014.

[47] R. S. M. za notranje zadeve POLICIJA. Porocilo o delu policije za leto 2014, 2015.

Video Evidence as Used in Court of Law and its Source Identification

Faculty of Computer and Information Science
University of Ljubljana
Computer Forensics
May 2015

Dušan Kalanj
Student at Faculty of Computer and Information Science
Ljubljana, Slovenia
dk8010@student.uni-lj.si

Gašper Kolenc
Student at Faculty of Computer and Information Science
Berlin, Germany
gk2334@student.uni-lj.si

Alja Kunovar
Student at Faculty of Computer and Information Science
Ljubljana, Slovenia
ak5051@student.uni-lj.si

ABSTRACT

In the court of law, image and video is used more and more throughout new court cases as a means to help establish a valid case. With gaining more weight, it is also becoming more important to assure proper source of said material and to establish its authenticity.

In this paper we begin with a broad overview of how video and image evidence is nowadays used in the court of law. Later, we describe several techniques that are currently being used for image and video source identification. We conclude the paper with a description of the source identification method by Chen et. al [1] and the improvements that should be introduced into surveillance systems that employ wireless IP cameras.

Categories and Subject Descriptors

Applied Computing [**Computer Forensics**]: Evidence collection, storage and analysis—*video and image source analysis*

Keywords

Video and image evidence, video and image forensics, source identification, wireless video streaming

General Terms

Field overview

1. INTRODUCTION

In this day an age we are surrounded by digital information in practically all aspects of our lives. The analogue is continuously being replaced by the digital, the consequences of which can be actively observed. In law, the changes are just as big as in any other field. To cope with the shift, new methods and equipment had to be developed to adequately process all the digital evidence that suddenly became available.

One of the most important conditions that has to be satisfied when dealing with digital evidence in court is the source identification of the evidence, especially when dealing with videos and images, as they may be, in many cases, found to be incontrovertible evidence of a crime or of a malevolent action. The aim of source identification is to identify the type, brand, model or the specific device that produced the digital asset. Just as important as the source identification is the verification of authenticity of the evidence, meaning that the evidence has not been tampered with in any way.

If the described conditions are met, digital evidence, particularly images and videos, can be of the utmost value in court. Next section describes the importance of such evidence through a few examples and the procedures that have to be followed when handling with the said evidence.

2. VIDEO AND IMAGE USED IN THE COURT OF LAW

In this section we will describe how images and videos are used in court as evidence. Due to the surge in digital recording, photos and surveillance video have become an integral part of the majority of court proceedings and investigations. Today, surveillance technology has advanced so much, that an object or event can be viewed and recorded from various angles, thus giving the court influential tool in detecting and prosecuting crimes. Furthermore, the verdict is reached much faster if courts are presented with any photos or video surveillance than if no visual digital evidence is presented.

2.1 Court cases

To further present the usefulness of video evidence in court of law, we present several court cases where video evidence was successfully used to reach a verdict.

- Video surveillance was one of key factors in used in establishing a case against a bar patron responsible for a violent assault in Kamloops, Canada in 2012. The accused, Yousef Almotairi, was believed to have stabbed a person in the eye with a pool cue in a bar. Almotairi was found guilty in the British Columbia Supreme Court as the attack was caught on camera and the video footage was considered reliable evidence during the trial to convict him [2].
- In New Mexico, there was recently a case where a state trooper was accused of having sex with a female on the hood of a car. The defendant claimed and was presumed to be in a remote location at the time. Video surveillance that was recently installed has however proven otherwise and the state trooper was brought to justice [3].
- Haresh Mehta, a 20 year old student, has been continuously harrased. Even with numerous complaints to the police, his offender was never brought to justice. That is until he recorded the offender using Google Glass during one of the acts of violence against himself. The footage was allowed to carry out a private prosecution against the bully [4].

2.2 Procedures used in court for video evidence handling

Because of the importance of digital evidence certain essentials in the law of evidence must be considered when obtaining and dealing with images and videos [5]:

- Conservation. The obligation to correctly store and maintain memory-cards, disks and other impermanent storage gadgets on which photos or videos are documented.
- Genuineness. The digitally recorded documentation is an honest and precise likeness of what the supporter of the data claims it to be.
- Policy. There should be a structured agency policy for evidence compilation and maintenance that incorporated digital and electronic evidence
- Admissibility. Proving that the alleged evidence is applicable and to what degree it influences the probability of that fact.

Adhering to typical sequence of evidence protocol such as where the video originated from, how it was recorded, who maintained the equipment, leads to last indent described above, admissibility. All appropriate steps and precautions in obtaining and utilizing evidence have to be considered by the law enforcement in the procedure. Digital evidence is correctly processed in following steps [6]:

- Assessment. Computer forensic examiners should assess digital evidence thoroughly with respect to the scope of the case to determine the course of action to take
- Acquisition. Digital evidence, by its very nature, is fragile and can be altered, damaged, or destroyed by improper handling or examination. Examination is best conducted on a copy of the original evidence. The original evidence should be acquired in a manner that protects and preserves the integrity of the evidence.
- Examination. The purpose of the examination process is to extract and analyse digital evidence. Extraction refers to the recovery of data from its media. Analysis refers to the interpretation of the recovered data and putting it in a logical and useful format.
- Documenting and reporting. Actions and observations should be documented throughout the forensic processing of evidence. This will conclude with the preparation of a written report of the findings.

3. MULTIMEDIA FORENSICS

Multimedia forensics is a science that tries to give an assessment on such digital content through acquisition and examination of information that could be useful to support a specific investigation [7]. It has to be able to develop instruments to deal with the various devices that can produce digital content and with processing tools that allow users to manipulate data. There are two basic approaches to multimedia forensics: active and passive [8] [9].

3.1 Active approach

With this approach additional information is embedded in or added to the original data. The former technique is called digital watermarking. Applying a watermark to an image can be done by inserting a slight modification to the original data with a certain key, which is then used to extract the watermark in the reverse process. The process is depicted on figure 1. The second technique that adds additional information to the original data uses a digital signature for authentication and verification purposes. Active methods are only effective if certain conditions are satisfied, the most important being:

- Use of a specially equipped and trustworthy camera that inserts the watermark or computes the signature.
- Use of a secure watermarking/signature algorithm.
- Use of a widely accepted watermarking/signature standard.
- Extracting of watermarks and signatures must be performed on original, unaltered data.

All of these requirements are almost always hard to satisfy, that is why a passive approach to multimedia forensics is usually the only option.

3.2 Passive approach

Methods that follow a passive approach to multimedia forensics do not require any prior information for verifying whether an image is authentic or not, but rely on the analysis of some intrinsic features that are present inside the observed data. These features can be found in the modification of the image structure (anomalous pixel values) and also in inconsistencies within the image content itself such as anomalies in the illumination direction [7]. Obviously that kind of investigation is harder and more time consuming than the previous, however it is usually the only option. That is also why we will only describe passive methods for image source identification in this paper. But in order to grasp how exactly image source identification is performed, we first need to explain how an image is acquired and processed by a device.

4. PHOTO ACQUISITION PROCESS AND CHARACTERISTICS

The process of image acquisition is depicted on figure 2 and usually passes through the following five components [10]:

- Lens system: composed of a lens and the mechanisms to control exposure, focusing and image stabilization to collect and control the light from the scene.
- Filters: includes the infra-red and anti-aliasing filters to ensure maximum visible quality.
- Image sensor: an image sensor is an array of rows and columns of small elements that are composed of a photodiode and a capacitor and together they represent a pixel. When an image is taken from real life, light is focused by the lenses on the pixels that generate an analogue signal proportional to the intensity of light, which is then converted to a digital signal and processed by the DIP [7].
- CFA: since the sensor pixels are not sensitive to colour, the rays from the scene are first filtered with a specific colour mosaic (CFA - Colour filter array), which assigns only one of three or four colours to each pixel. CFA patterns can use RGB (3 colours) or YMCG (4 colours) matrices.
- DIP: the output from the sensor with a Bayer (RGB) filter is a mosaic of red, green and blue pixels of different intensities. Each pixel contains the information of only one colour. The digital image processor implements interpolation (demosaicing) algorithms, which differ between manufacturers, to recover the missing information of the other two colours of each pixel. The signal then undergoes additional signal-processing such as white balancing, noise reduction, image sharpening, gamma correction and other operations to produce a good quality image. In the end the data is stored in the camera memory in a specified image format such as JPEG, PNG, etc [11].

5. DIGITAL FINGERPRINTS

It is now easier to understand that the characteristics of each operation and the properties of every element, from the framed scene to the final image file, influence the digital data. During image acquisition and any post-processing operation the elements described in the previous section leave a distinctive imprint on the data, as a sort of digital fingerprint [7]. Said fingerprints can be roughly divided into three categories [12]:

- In-camera fingerprints: as stated before, each element in the image acquisition process modifies the input and leaves a certain fingerprint on the data. Each camera component, such as a CFA and colour interpolation, employs a particular set of algorithms with an appropriate set of parameters to modify the input scene. In these processing stages, each camera uses a different algorithm (that may be proprietary to the camera manufacturer, brand, or model) and leaves intrinsic fingerprint traces on the output data.
- Postcamera fingerprints: each processing (resizing, rotating, stretching, etc.) applied to digital media further modifies their properties, leaving peculiar traces. The two fingerprints that we mentioned so far are independent of the content of the image, while the next one is directly involved in the scene itself.
- Scene fingerprints: the world has specific properties depending on the content which characterize the reproduced scene. One of the most informative properties is the lighting and its direction.

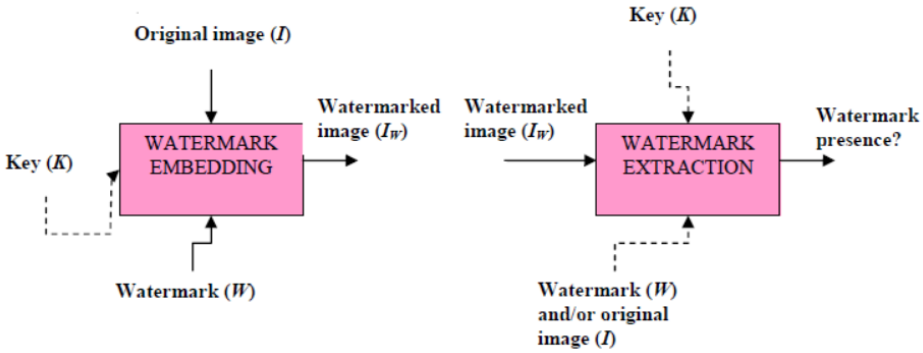


Figure 1: Watermarking process and its extraction [7]

In general, multimedia forensics tries to detect the existence of such fingerprints, but because postcamera and scene fingerprints are not unique to the acquisition device, only the in-camera fingerprints can be used for source identification. All three categories can be exploited for integrity verification [7]. Now that we have briefly explained the possible approaches to multimedia forensics, the image acquisition process and the possible intrinsic fingerprints, we can address the topic of image source identification.

6. IMAGE SOURCE IDENTIFICATION

Digital images can be stored in a variety of formats (JPEG, PNG, etc.) that can tell a lot about the image, usually through metadata. The metadata describe the source of the image and usually includes the camera type, resolution, focus, settings and other features. Although such metadata provide a significant amount of information they can be easily edited, deleted and falsified, therefore it is important to provide a reliable source identification regardless of the type of metadata [13].

In general, image source identification tries to detect the existence of a certain fingerprint (described in the previous chapter) that is introduced by an image acquisition device. Because of camera lens, characteristics such as chromatic aberration are introduced. Chromatic aberration is a type of distortion in which there is a failure of a lens to focus all colours to the same convergence point and can be visible as coloured edges [14]. Further characteristics are introduced by the sensor, such as sensor defects and sensor noise. Another characteristic used as forensic method are dependencies between adjacent pixels, which are a result of colour interpolation techniques. Those techniques are used because of sensors' inability to differentiate between colors [7].

Based on device-dependent characteristics described above, the task of image source identification can be divided in three subtasks [15]:

- detection of the device type (digital camera, scanner, etc..)
- detection of device model
- detection of the device itself

6.1 Building a training set

In order to link a device to a certain image, a training set must first be built from the media captured by the device under investigation. This is done by capturing a number of images (cca. 50) with the device under analysis. These digital images make it possible to extract the device sensor's fingerprint. The images should not be too noisy, textured, or contain multiple edges. The best camera reference images are light, not overexposed, not extremely textured images. Good examples of such images are pictures of light walls, sky, white papers, etc. Moreover, digital images should be original and not modified by software [16]. Examples of good and bad images can be seen on figures 3 and 4.

Figure 3: Examples of good reference pictures [16].

Figure 4: Examples of bad reference pictures [16].

The process is pretty similar when creating a reference video. Instead of a number of digital reference images, a single reference video file is used to calculate the camera's sensor fingerprint. This video should be at least 25 seconds in duration and it is highly recommended that the camcorder reference video content has the same properties as those of digital photos, as described above [16].

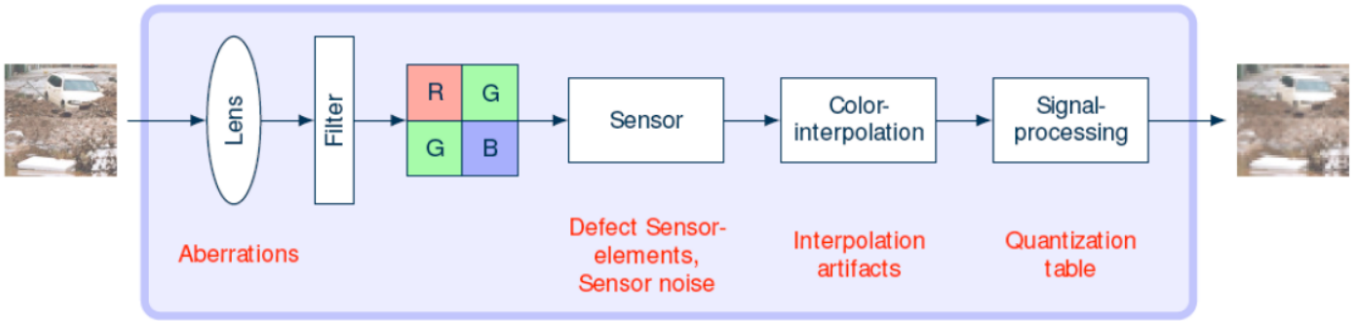


Figure 2: Image Acquisition Process [15]

6.2 Source identification methods

On the acquired training set, certain fingerprints that were imprinted on the image or video by the color filter array (CFA) interpolation, sensor imperfections and lens aberration are looked for. Those fingerprints are then compared to the ones on the images from the evidence. In the next section, the methods for identifying inherent fingerprints will be briefly described.

6.2.1 Colour filter array and demosaicking

Single-sensor digital cameras (most common) capture images by covering the sensor surface with a color filter array (CFA) such that each sensor pixel only samples one of three primary color values. To render a full-color image, an interpolation process, commonly referred to as CFA demosaicking, is required to estimate the other two missing color values at each pixel [17]. The techniques for acquisition device identification try to find the color filter array pattern and the color interpolation algorithm employed by the DIP of the device that acquired the image. Because each manufacturer employs a specific demosaicking algorithm, these techniques can only be used to identify the device brand. The model is more difficult to detect because the manufacturers usually implement similar kinds of interpolation methods across their devices [18].

6.2.2 Lens aberration

Due to the design and manufacturing process, lenses produce different kinds of aberrations in images. Generally, two of them are investigated to solve the problem of source device identification: lens radial distortion and chromatic aberration [7].

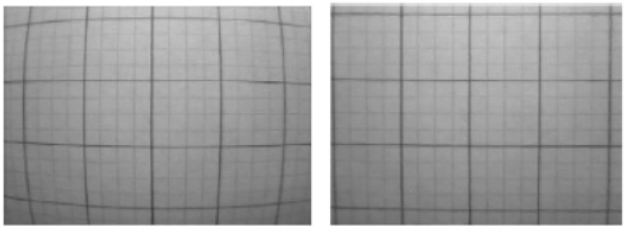


Figure 5: Example of a radial distortion [10].

The radial distortion causes straight lines in the object space rendered as curved lines on camera sensor. A similar phenomenon happens when one looks through a drinking glass. Source identification methods try to find certain distortion parameters that constitute the fingerprint. However, this method fails if there are no straight lines in the image and also if two cameras of the same model are compared. Besides, it is also possible to correct the radial distortion on an image using a certain software [19]. An example of the distortion is depicted on figure 5.

The second type of aberration used for source identification is the chromatic aberration. It is a type of distortion in which there is a failure of a lens to focus all colors to the same convergence point and it leads to various forms of color imperfections in the image [14]. The task of source identification methods is to estimate certain distorted parameters and compare them with the training set. The phenomenon can be observed on figure 6.

Figure 6: Example of a chromatic aberration [14].

6.2.3 Imaging sensor imperfections

Before the actual image is recorded and transferred from the digital device, various noise sources degrade the image. Source identification methods try to identify and extract systematic errors that are produced by the imaging sensor and appear on all images independent of the scene content. There are several distinct sources of noise produced by the camera. When a pictured of an absolutely evenly lit scene is taken, the image will exhibit small changes in intensity among individual pixels. These errors include sensor pixel defects and pattern noise. The latter has two major components, namely fixed pattern noise and photo response non-uniformity noise (PRNU) [20].

Defective pixels act as a fingerprint, which are present in all images obtained by a specific image sensor, and hence could be used for device identification. However, with increased manufacturing standards, the presence of defects is decreasing. Furthermore, defective pixels may be corrected after image acquisition in the post-processing stage in the camera, which makes this method of identification rather void [15].

For a reliable camera identification, the idea is to estimate the pattern noise. The fixed pattern noise (FPN) refers to pixel-to-pixel differences when the sensor array is not exposed to light (so called dark current) and also depends on exposure and temperature. The FPN is used for source camera identification but it is an additive noise and some middle to high-end consumer cameras suppress this noise by subtracting a dark frame from every image they take [21].

If the above method fails, photo-response non-uniformity noise (PRNU) is usually searched for. Its most important component is the pixel non-uniformity (PNU) which is defined as different sensitivity of pixels to light. That is due to the inhomogeneity and impurities in silicon wafers and imperfections introduced by the sensor manufacturing process. As a result, when the illumination on a number of pixels is exactly the same for all pixels, the output signals

from these pixels will be slightly different, creating a pattern with some pixels outputting systematically lower or higher signals than their neighbours. It is also possible to suppress this kind of noise using a process called flat fielding, but consumer digital cameras do not flat-field their images due to its complexity [22].

6.3 Issues of existing methods

As mentioned before, techniques that use the CFA pattern are only capable of telling the device brand and they also fail when the image is compressed or modified. Similarly, methods that try to find a certain lens aberrations pattern are also incapable of telling the specific device and can only identify the device model. The pattern noise methods report the most reliable results of all the above. However, all these solutions suffer from significant performance degradation if a streamed video is under inspection. Those are usually captured by wireless cameras and contain blocking and blurring [1]. This problem is occurring in surveillance systems where wireless IP cameras are usually employed. Another issue that surveillance systems are facing is the spoofing attack in which the attacker masquerades as the entity under attack and fills the communication with falsified data [23]. Next section describes how Chen et al. [1] defines the wireless camera spoofing attack, and provides a systematic detection method, which builds on sensor pattern noise and deals with such attacks.

7. SOURCE IDENTIFICATION IN LOSSY WIRELESS NETWORKS

With wireless IP cameras, packet loss and unpredictable transmission delays are inevitable. This is even more apparent if the network does not meet the required minimum specification in regards to bandwidth. As a result, blocking and blurring are frequently observed in the received frames and are a major obstacle with source identification in lossy networks. The phenomena can be observed on figure 7.

The already discussed source identification methods suffer greatly from significant performance hits or even fail to work under these conditions. In paper [1], contributors Chen, Pande, Zeng, Mohapatra and their colleagues, introduce a new systematic methodology for identification of video sources, which achieves a far greater performance as the afore mentioned methods.

The method in question performs well not only with conventional videos, but also works well with wirelessly streams videos that include blurring and / or blocking. As a bonus, this method is also applicable as a defence against wireless camera spoofing attacks. To achieve better results, the method builds upon sensor pattern noise extraction method.

7.1 Problems of lossy wireless networks and their effects on conventional identification methods

Video blocking is an event that occurs, if data of some blocks are missing. As a result, borders of said blocks form a sort of a grid which in turn suppresses the real sensor pattern noise. This is due to the fact that videos are compressed based on 8x8 or 16x16 pixel blocks and therefore form a grid.

Figure 7: Frames with blocking and blurring.

This affects both sensor pattern noise recognition inside the blocking areas where noise is weakened, as well as on their borders where the signal gets stronger.

The other apparent problem with lossy wireless networks is video blurring. Blurring can occur because of various reasons, usually it is due to high compression rate, limited lens resolution or fast motion of the subject being caught on camera. The underlying problem is similar as with video blocking, where data of a block gets lost, with the difference being that with blurring only high frequency component is lost.

Because of the underlying reason of both these occurrences is based on the same problem - that is data loss, this method treats both of them uniformly.

7.2 Basic idea

Methodology from paper [1] treats frames with blocking differently as normal frames, as it rules out blocking areas but still uses the rest of the frame for pattern noise extraction. Therein lies the advantage in comparison to other methods, as frames with blocking are not entirely disregarded but are still somehow used up to their best potential.

As blocking recognition in frames is vital for this methodology to work properly, it introduces its own method for blocking detection. As its main purpose is dealing with video that include packet loss, it uses a non-reference method for blocking detection and is wavelet based as to largely reduce the computational overhead.

7.3 Blocking recognition algorithm

As previously stated, the blocking recognition algorithm developed for source identification in lossy wireless networks is based on wavelet transforms. By performing first-level wavelet transform on an original image without blocking and the same image with blocking present, the researches in paper [1] present two observations:

- Image with blocking produces more elements in the diagonal subband d(i,j) with zero or close to zero values.

- By adding absolute values of d(i,j) by rows, the paper [1] demonstrates periodic like characteristics like the teeth of a saw on the video with blocking.

It is very apparent that the first observation stated is much easier to exploit than the second one. But it is worth noting, that big chunks of objects in plain color produce the same result as videos with blocking, which also happen to be the best sources for extracting sensor pattern noise. Fortunately researches from [1] came to a conclusion, that the second observation mentioned is unique only to video blocking and therefore makes a better choice for blocking recognition.

7.4 Application of blocking recognition algorithm with the basic idea

It is proposed, that a frame should be divided into 32x32 pixel squares, where each square is individually parsed by the blocking detection algorithm. If it is examined that video blocking exists the frame is discarded, otherwise it is adopted for noise extraction.

To further improve the algorithm and make it more IP spoofing-proof, three additional improvements are implemented:

- Some of the algorithm functions, such as wavelet transformations and local variance estimations can be parallelized and thus when the algorithm is being run on a multi-core computer, speed boosts accordingly. This methodology becomes very scalable with this improvement and is more viable to be run on a live video feed.

- By incorporating selective frame processing, where I-frames contain more information than P- and B-frames, only noise from I-frames is processed. In comparison, using 40 I-frames gives similar results as using 200 mixed frames.

- Authors also propose incorporation of wireless channel signatures with sensor fingerprints. Metrics should include packet loss ratio, jitter, average signal strength, signal strength variance and the percentage of blocking frames.

8. CONCLUSIONS

In this paper, image and video have been presented from digital forensics' point of view. We began with a short overview of how such evidence is treated during an examination and in court. The paper then focused primarily on image and video source identification methods. Some of the methods were mentioned and some were depicted with more detail. Their comparison shows that none except the method by Chen et al. [1] are suitable for source identification in lossy wireless networks. The excepted method exhibits excellent performance even in the presence of video blocking and blurring, which are present in wireless networks. The same authors also proposed improvements such as wireless channel signatures and selective frame processing to accelerate the method.

9. REFERENCES

[1] Chen, S., Pande, A., Zeng, K., Mohapatra, P.: Live Video Forensics: Source Identification in Lossy Wireless Networks. Information Forensics and Security Vol. 10 No. 1, (2014).

[2] Wallace, J.: Clear Video Evidence the Building Block of a Court Case. Retreived May 8, 2015, http://avigilon.com/connected/security/clear-video-evidence-the-building-block-of-a\-court-case-says-legal-expert-2

[3] Dwyer, T.P.: Legal considerations in the use of digital video in criminal cases. Retreived May 8, 2015, http://www.policeone.com/police-products/investigation/evidence-management/articles/4450086-Legal-considerations-in-the-use-of\-digital-video-in-criminal-cases/

[4] Bullied man uses video from sunglasses to mount private court case. Retreived May 8, 2015, http://www.theguardian.com/uk-news/2014/oct/07/bullied-student-video-sunglasses-private-court\-case

[5] Using Video Surveillance as Evidence in Court. Retreived May 8, 2015, http://securitybros.com/using-video-surveillance-as-evidence-in-court/

[6] Ashcroft, J., Daniels, D.J., Hart, S.V.: Forensic Examination of Digital Evidence: A Guide for Law Enforcement. U.S. Department of Justice, National Institute of Justice.

[7] Caldelli, R., Amerini, I., Picchioni, F., De Rosa, A., Uccheddu, F.: Multimedia Forensic Techniques for Acquisition Device Identification and Digital Image Authentication. Handbook of Research, (2010).

[8] Maura, B.: Multimedia Forensics: introduction and motivations. Multimedia Security. University of Siena.

[9] Batagelj, B.: Računalniški vid v digitalni forenziki. Laboratory for computer vision. University of Ljubljana. (2015).

[10] Wu, Ja-Ling.: Overview of Digital Image Forensics - Image Source Identification. National Taiwan University.

[11] Khanna, N., Mikkilineni, A.K., Delp, E.J.: Forensic Camera Classification: Verification of Sensor Pattern Noise Approach. Forensic Science Communications Vol. 11 No. 1, (2009).

[12] Swaminathan, A., Wu, M., Ray Liu, K.J.: Digital Image Forensics via Intrinsic Fingerprints. IEEE Transaction on Information Forensics and Security. Vol. 3 No. 1, (2008).

[13] Exchangeable image file format. In Wikipedia. Retreived May 30, 2015, http://en.wikipedia.org/wiki/Exchangeable_image_file_format

[14] Chromatic aberration. In Wikipedia. Retreived May 9, 2015, http://en.wikipedia.org/wiki/Chromatic_aberration

[15] Geradts, Z., Gloe, T.: Identification of images. Future of Identity in the Information Society, (2009).

[16] Verifeyed software manual. Retreived May 30, 2015, http://verifeyed.com/documentation/

[17] Lu, W., Tay, Y.P.: Color filter array demosaicking: new method and performance measures. IEEE Trans Image Process. Vol. 12 No. 10, (2003).

[18] Sencar, H.T., Memon, N., Avcibas, I.: Source Camera Identification Based on CFA Interpolation. Image processing. (2005).

[19] Choi, K.S., Lam, E., Wong, K.: Automatic source camera identification - using the intrinsic lens radial distortion. Optics Express. Vol. 14 No. 24, (2006).

[20] Sankey, J.: Noise in Digital Cameras. Retreived May 8, 2015, http://www.johnsankey.ca/noise.html

[21] Lukas, J., Fridrich, J., Goljan, M.: Digital camera identification from sensor pattern noise. IEEE Transactions on Information Forensics and Security. Vol. 1, No. 2, (2006).

[22] Peng, F., Shi, J., Long, M.: Comparison and Analysis of the Performance of PRNU Extraction Methods in Source Camera Identification. Journal of Computational Information Systems Vol. 9 No. 14, (2013).

[23] Spoofing Attack. In Wikipedia. Retreived May 10, 2015, http://en.wikipedia.org/wiki/Spoofing_attack

Pridobivanje skritih sporočil iz steganografskih slik

Mark Hočevar
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana, Slovenija
mh2870@gmail.com

Anej Placer
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana, Slovenija
anej.placer@gmail.com

Jurij Slabanja
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana, Slovenija
jurijslabanja@gmail.com

POVZETEK

Steganografija omogoča skrivanje podatkov v različne krovne medije (ang. cover media/images) tako, da se originalni in steganografsko obdelani medij na pogled ne razlikujeta. Pogost pristop k steganografiji je skrivanje sporočil v najmanj pomembne bite medija, t.i. LSB steganografija. Pri tem lahko tudi definiramo kateri deli medija bodo uporabljeni. Če se sporočilo skrije kar po vrsti od začetka, gre za preprosto LSB steganografijo. Obstaja pa tudi taka, ki uporablja kodirne ključe, ki skrito sporočilo razpršijo po celotnem krovnem mediju.

Za odkrivanje skritih podatkov iz steganografskih medijev obstaja mdr. tehnika lociranja tovora (ang. payload location). Le-ta računa slike ostankov na podlagi originalnega in steganografsko obdelanega medija. Uporabna je, da izvemo kateri biti so bili spremenjeni, ponavadi pa ne zna sama po sebi ugotoviti zaporedja bitov, posebej če je bil uporabljen kodirni ključ. V članku pokažemo, da zaporedje vendarle lahko rekonstruiramo iz pričakovanih srednjih vrednosti slike ostankov (ang. expected mean residuals).

V članku tudi pogledamo nekaj primerov steganografske programske opreme, ki je na voljo za prosto uporabo in primerjamo njihove rezultate.

Kategorije in opis predmeta

I.4 [**Procesiranje slik in računalniški vid**]: Razno; I.3.3 [**Računalniška grafika**]: Generiranje slik; E.4 [**Kodiranje in teorija informacij**]: Razno

Splošni pojmi

Algoritmi, Teorija

Ključne besede

steganografija, LSB, skupinska pariteta, OpenStego, StegExpose, StegDetect

1. UVOD

Z uporabo steganografije je mogoče skriti sporočila v navadne slike, pri čemer tretja oseba na pogled ne more razločiti med originalno sliko in sliko s skritim sporočilom. Med bolj popularnimi algoritmi je steganografija z uporabo najmanj pomembnega bita oz. LSB (ang. Least Significant Bit steganography). Tako se imenuje, ker spreminja samo najmanj pomembne bite pikslov in je razlika z originalno sliko minimalna. Primerjavo uporabe najmanj in najbolj pomembnega bita pikslov lahko vidimo na sliki 2 in sliki 3. Pri skrivanju sporočila v najmanj pomemben bit je popačenje slike praktično nevidno. Če pa sporočilo skrijemo v najbolj pomemben bit, se razlika z originalno sliko takoj opazi. Še vedno pa je pomembno, da za določeno dolžino sporočila izberemo dovolj veliko sliko. Tudi če spreminjamo samo LSB se z dovolj zamenjavami lahko zazna, da je bila slika spremenjena, več o tem v poglavju 2.1.

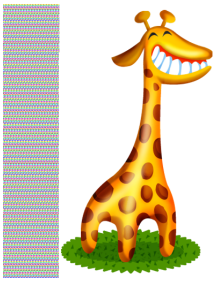
Dva pristopa k LSB steganografiji sta LSB zamenjava in LSB ujemanje (ang. LSB replacement, LSB matching). Pri zamenjavi se lihim pikslom odšteje vrednost 1, medtem ko se sodim prišteje vrednost 1, glede na bitno zaporedje sporočila, ki ga hočemo skriti. Pri ujemanju se pikslom prišteva/odšteva vrednosti ne več glede na sodost ali lihost, ampak samo glede na bitno zaporedje sporočila.

Ko zaznamo, da je bila neka slika steganografsko spremenjena, jo je potrebno obdelati, da lahko iz nje razberemo skrito sporočilo. To ni najbolj trivialen problem. Eden od načinov je, da poskušamo odkriti kodirni ključ. To pride v poštev predvsem, ko je ključ šibek in poznamo kakšne podrobnosti steganografskega sistema, s katerim je bilo sporočilo skrito v sliko. Drugi način je, da poskušamo najti položaj skritega sporočila. Ta pristop se imenuje lociranje tovora in je uporaben predvsem, ko imamo več slik enakih velikosti in se sporočilo vedno nahaja na istih mestih.

Lociranje tovora je učinkovito, ko je sporočilo skrito z uporabo preproste LSB steganografije ali steganografije s skupinsko pariteto (ang. group-parity steganography). Ponavadi je implenetirano tako, da računa sliko ostankov. Omogoča lociranje posameznih bitov skritega sporočila, vendar jih ponavadi ne zna sestaviti nazaj skupaj. Članek pokaže, da v določenih primerih vendarle lahko razberemo vrstni red bitov sporočila, s pomočjo pričakovanih srednjih vrednosti slike ostankov in posledično lahko sporočilo rekonstruiramo.

Slika 1: Izvorna slika

Slika 2: Primer skrivanja sporočila v najmanj pomembne bite piksla.



Slika 3: Primer skrivanja sporočila v najbolj pomembne bite piksla.

2. TEORIJA

2.1 Korensko pravilo steganografske kapacitete

Pojem steganografska kapaciteta označuje količino podatkov, ki jo lahko uspešno skrijemo v nek medij, npr. sliko, ne da bi bili pri tem odkriti. V popolnoma varnem steganografskem sistemu je distribucija krovnega medija identična steganografski distribuciji, kar pomeni da noben statistični test ne more odkriti skritega sporočila. Kapaciteta takega sistema linearno narašča z velikostjo krovnega medija. Žal taki sistemi obstajajo samo za umetne krovne medije. Steganografski sistemi za realne medije so vedno nepopolni, kar pomeni, da tudi kapaciteta takega sistema narašča manj kot linearno. Izkaže se, da je kapaciteta nepopolnega steganografskega sistema korenskega reda[9][10][6]. To potegne za sabo zanimivo posledico. Vzemimo za primer dve sliki, eno večjo in eno manjšo. V obe skrijemo sorazmerno velikosti slike enako veliko podatkov. Posledica korenskega pravila je, da v večji sliki lažje odkrijemo skrito sporočilo.

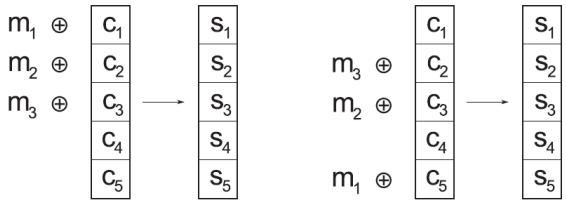
2.2 Osnovne metode steganografije

Pri uporabi preproste LSB steganografije je vsak bit sporočila zakodiran z natanko enim pikslom. Naivne metode sporočilo preprosto zakodirajo v prvih m pikslov krovne slike, kjer je m število bitov sporočila in mora biti $m <= n$, kjer je n število pikslov v sliki. To ni najboljša možnost, saj je ves steganografski šum zgoščen na prvem delu slike in je relativno lahko odkriti skrito sporočilo.

Za razrešitev tega problema so bili uvedeni kodirni ključi. Kodirni ključ je funkcija, ki razprši sporočilo po celi sliki oz. določi kateri piksli v sliki bodo uporabljeni za skrivanje posameznega bita sporočila. Primer brez uporabe kodirnega ključa v primerjavi z njegovo uporabo je prikazan na sliki 4. Z uporabo enakega ključa na isti sliki, je zelo verjetno, da bodo ponovno uporabljeni isti piksli na njej. To šibkost izrablja metoda lociranja tovora. Lokacijo sporočila je možno enostavno odkriti, če imamo poleg dostopa do slike s skritim sporočilom, tudi dostop do originalne krovne slike. Pri tem odštejemo krovno sliko od slike s skritim sporočilom, da dobimo sliko ostankov. Na vsakem mestu, kjer je piksel na sliki ostankov neničelen, je bil skrit bit sporočila.

Nekoliko bolj kompleksna oblika steganografije je steganografija s skupinsko pariteto. Pri tem je vsak bit sporočila določen s skupino k pisklov. Bolj natančno, vsota LSB teh pikslov po modulu dva določa vrednost bita sporočila. Zatorej je tudi odkrivanje sporočila tu nekoliko težje, saj moramo polega spremenjenih pikslov ugotoviti tudi kako so združeni v skupine.

Tako preprosta LSB steganografija kot tudi steganografija s skupinsko pariteto sta obe posebni obliki kodiranja matrik v sliko na podlagi Hammingovega koda. V splošnem Hammingov kod s pomočjo k elementov kodira q podatkovnih elementov oz. $Hamming(k, q)$. Iz tega lahko zaključimo, da je preprosta LSB steganografija pravzaprav $Hamming(1, 1)$, steganografija s skupinsko pariteto pa $Hamming(k, 1)$. Posledično tehnike matričnega kodiranja veljajo tudi za ti dve obliki steganografije.



Slika 4: Na levi je primer kodiranja brez ključa, na desni z uporabo ključa. Sliki prikazujeta trivialen primer, kjer je slika velika pet pikslov in sporočilo dolgo tri bite. Brez uporabe ključa je sporočilo vstavljeno v sliko po vrsti, z njegovo uporabo pa je videti nekoliko bolj naključno.

2.3 Odkrivanje sporočil

Pomembno je, da se zavedamo, da poznavanje lokacije tovora ni dovolj za konstrukcijo sporočila. Za to je potrebno najdene dele sporočila urediti po njihovem logičnem zaporedju. Glavni razlog, da zaporedja ni mogoče določiti iz najdenih lokacij tovora je predpostavka, da vsaka stego slika

vsebuje točno določen tovor velikosti m. Z omiljenjem te omejitve, tako da je velikost tovora lahko velkosti med 1 in m pokažemo, da povprečni ostanki slike vsebujejo dovolj informacij, da lahko vsebovan tovor logično uredimo in pridobimo skrito sporočilo.

2.3.1 Enostavna LSB steganografija

Naj bo R_i indikatorska spremenljivka za dogodek, da bo logični piksel i moral biti spremenjen tako, da vsebuje bit m_i skritega sporočila. Naj bo $L{\sim}f_L(l)$ naključna spremenljivka, sorazmerna z velikostjo tovora. Predpostavimo, da masovna verjetnostna funkcija $f_L(l)$ zadošča predpostavki, da je $f_L(l) > 0$ za vse $l \in \{1, ..., m\}$ in nič povsod drugod. Naj bo $F_L(l) = \sum_{i=1}^{l} f_L(i)$ kumulativna masovna funkcija za L, ki strogo narašča po l za $l \in \{1, ..., m\}$. Sedaj pokažemo, da $E[R_i] > E[R_j]$ za vse piksle i, j, kjer $i < j$. Za tovor velikosti l velja

$$p(R_i = 1|L = l) = \begin{cases} \frac{1}{2}, & \text{če } l \geq i, \\ 0, & \text{sicer.} \end{cases} \tag{1}$$

Sedaj,

$$E[R_i] = \sum_{l=1}^{m} p(R_i = 1|L = l) f_L(l) \tag{2}$$

$$= \sum_{l=i}^{m} f_L(l) \tag{3}$$

$$= \frac{1}{2}(1 - F_L(i-1)). \tag{4}$$

Ker $F_L(l)$ strogo narašča po l za $l \in \{1, ..., m\}$ sledi, da $E[R_i] > E[Rj]$ za vse piksle tovora i, j, kjer $i < j$. Da izrazimo skrito sporočilo je torej potrebno najden tovor urediti po padajočem vrstnem redu povprečnih ostankov slike. Za poseben primer, kjer je $f_L(l)$ enoten, velja, da $F_L(l) = \frac{1}{m}$ za $l \in \{1, ..., m\}$ in

$$E[R_i] = \frac{m + 1 - i}{2m}. \tag{5}$$

Povprečni ostanki slike padajo linearno kot funkcija od i.

2.3.2 Steganografija s skupinsko pariteto

Pravilen vrstni red se lahko določi tudi za dele tovora steganografije s skupinsko pariteto. Naj bo G_i skupina k pikslov, ki določajo sporočilo m_i. Drugače povedano,

$$\sum_{j \in G_i} LSB(s_j) \ mod \ 2 = m_i. \tag{6}$$

Naj bo R_i^j indikator slučajne spremenljivke za dogodek, da bo piksel j iz skupine G_i potrebno modificirati, da bo vseboval bit m_i skritega sporočila. Za tovor velikosti l velja,

$$p(R_i^j = 1|L = l) = \begin{cases} \frac{1}{2k}, & \text{če } l \geq i, \\ 0, & \text{sicer.} \end{cases} \tag{7}$$

To velja zaradi dejstva, da je potrebno modificirati le enega od k pikslov, če pariteta skupine ni skladna z bitom sporočila. Z uporabo enake izpeljave za (4) dobimo

$$E[R_i^j] = \frac{1}{2k}(1 - F_L(i-1)). \tag{8}$$

Naj bo $R_i = \sum_{j \in G_i} R_i^j$ ostanek skupine G_i. Velja

$$E[R_i] = \sum_{j \in G_i} E[R_i^j] = \frac{1}{2}(1 - F_L(i-1)). \tag{9}$$

Ker $F_L(l)$ strogo narašča po l za $l \in \{1, ..., m\}$, sledi, da $E[R_i] > E[Rj]$ za vse skupine tovora G_i, G_j, kjer $i < j$. Locirane bite tovora, ki pripadajo skupinam tovora, se lahko uredi po padajočem vrstnem redu povprečnih ostankov skupin, da razkrijejo skrito sporočilo. Podobno, če je $f_L(l)$ enotna, velja

$$E[R_i] = \frac{m + 1 - i}{2m}. \tag{10}$$

Steganografija s skupinsko pariteto je posebna vrsta matričnega vgrajevanja sporočil. Za celoten razred vgrajenih algoritmov bi bila analiza podobna, z edino razliko, da lahko urejamo le skupine po q bitov in ne individualnih.

V obeh primerih (enostavno LSB vgrajevanje in steganografija s skupinsko pariteto) ni potrebno, da velja $f_L(l) > 0$ za vse $l \in \{1, ..., m\}$. Ta predpostavka je lahko omiljena, da dopusti $f_L(l) \geq 0$. V tem primeru lahko pridobimo le delno ureditev.

2.3.3 Ocena pokritja

Zgornja analiza je pokazala, da je skrita sporočila možno pridobiti, če poznamo krovno sliko in lahko izračunamo ostanke. V takšnem scenariju ima digitalni forenzik dostop do računalnika in drugih digitalnih pripomočkov steganografa. Tudi, če naivni steganograf krovne slike z računalnika ali digitalne kamere izbriše, jih je še vedno mogoče obnoviti s tehnikami izrezovanja.

Seveda pa obstajajo primeri, ko krovne slike ni mogoče obnoviti, kar pa za forenzično analizo predstavlja problem, saj ostankov ni mogoče nemudoma izračunati. V takšnih primerih se oceni približek krovne slike. Problem je enak kot pri lokaciji tovora, ko krovna slika ni znana. Ocena pokritja je formulirana na naslednji način. Pri podani steganografski sliki s, se poišče najbolj verjetno krovno sliko:

$\hat{c} = \text{argmax}_c\, p(c|s)$.

Eden od aktualnih krovnih estimatorjev uporablja naključna polja markova (MRF) in je pri iskanju steganografskih tovorov precej uspešen. MRF model oceni krovno sliko tako, da pripiše oznako y_i vsakemu pikslu p_i. Oznaka pove ali je piksel bil spremenjen oz. kako je bil spremenjen. Ta pogoj se izrazi v obliki Gibbsove porazdelitve:

$$p(y|s;w) = \frac{1}{Z(s;w)} e^{-E(y|s;w)}, \tag{11}$$

kjer sta $w = (w_1, w_2)$ uteži, Z normalizacijski člen in E energijska funkcija oblike

$$E(y|s;w) = w_1 \sum_{i \in \nu} f_i(y_i|s) + w_2 \sum_{ij \in \varepsilon} f_{ij}(y_i, y_j|s). \tag{12}$$

ν se nanaša na piksle, ε pa predstavlja sosednje pare pikslov v štiridelni mreži. Izraza f_i in f_{ij} sta enotna in parna cena, ki sta odvisne od vgrajevalne funkcije.

Namesto LSB velja

$$f_i(y_i|s) = \begin{cases} -log(1-\rho), & \text{če } y_i = 0, \\ -log(\rho), & \text{če } y_i = 1. \end{cases} \tag{13}$$

Parameter ρ označuje delež spremenjenih pikslov in je lahko ocenjen z uporabo posebnih tehnik [8].

Z $\tilde{s}$ je označena po LSB obrnjena verzija s. Parna cena f_{ij} je definirana kot

$$f_{ij}(y_i, y_j|s) = \begin{cases} -\log(s_i, s_j), & \text{če } y_i = 0 \text{in } y_j = 0, \\ -\log(s_i, \tilde{s_j}), & \text{če } y_i = 0 \text{in } y_j = 1, \\ -\log(\tilde{s_i}, s_j), & \text{če } y_i = 1 \text{in } y_j = 0, \\ -\log(\tilde{s_i}, \tilde{s_j}), & \text{če } y_i = 1 \text{in } y_j = 1. \end{cases} \tag{14}$$

Združene verjetnosti p izvemo iz znanih krovnih slik. Pridobljene oznake y označujejo piksle, ki so bili spremenjeni. Npr. $y = 1$ označuje, da je bil LSB piksla i spremenjen. To ustreza ostanku r_i.

Za LSB ujemanje velja

$$f_i(y_i|s) = \begin{cases} -\log(1-\rho), & \text{če } y_i = 0, \\ -\log(\frac{\rho}{2}), & \text{če } 1 \leq s_i + y_i \leq 254 \\ & \text{in } y_i \neq 0, \\ -\log(\rho), & \text{če } (s_i = 1 \text{ in } y_i = -1) \\ & \text{ali } (s_i = 254 \text{ in } y_i = 1), \\ \infty, & \text{sicer} \end{cases} \tag{15}$$

in

$$f_{ij}(y_i, y_j|s) = -\log p(s_i + y_i, s_j + y_j). \tag{16}$$

Piksel i je bil spremenjen, če $y_i \in 1, -1$. Torej velja $r_i = |y_i|$ [11].

3. PRAKTIČNI PRIMERI

3.1 Enostavna LSB steganografija

3.1.1 Skrivanje

Za testiranje LSB steganografije smo uporabili odprto kodno knjižnico, katere avtor je Robin David [12]. Program deluje po principu LSB steganografije to je, za skrivanje podatkov uporablja najmanj pomemben bit vsake barve v pikslu. Program uporablja enostaven postopek, kjer skriva bite v zaporedne piksle slike. Za testiranje smo uporabili sliko žirafe [1]. V sliko smo skrili niz "Hello hidden world!!" dolg dvajset znakov. Po pričakovanjih je slika na pogled ostala nespremenjena [5]. Knjižnica omogoča tudi skrivanje slike v sliko.

Slika 5: Slika s skritim sporočilom

Da smo se prepričali, če smo sliko sploh kaj spremenili smo obe sliki odprli s programom GIMP. V programu smo obe sliki približali tako, da smo videli prvi piksel. Prebrali smo njegovo vrednost in ugotovili, da ima v izvirni sliki vrednost R:255, G:255, B:255 v sliki s sporočilom pa R:254, G:254, B:254. Kar pomeni, da je pri vseh barvah v pikslu spremenjen najmanj pomemben bit. Razlika med barvama je tako majhna, da je nismo opazili tudi, ko smo opazovali samo en piksel.

3.1.2 Iskanje sporočila

Program za skrivanje uporablja kar zaporedne piksle, to nam omogoča zelo enostavno odkrivanje skritega zapisa. Enostavno zaporedno beremo najmanj pomembne bite pikslov, ko naletimo na znak "null", s katerim se zaključujejo nizi smo uspeli prebrati celo sporočilo. Izpis vidimo na sliki 6.

```
> python unhide.py
Text value: Hello hidden world!!
>
```

Slika 6: Izpis skritega sporočila

3.2 Naključna LSB steganografija

OpenStego [1] je malo bolj napreden program za uporabo steganografije. Ima podporo za različne algoritme za skrivanje s pomočjo vtičnikov. V osnovi program podpira naključni LSB (Randomized LSB) algoritem, kar je izboljšana različica algoritma LSB, ki smo ga uporabili pri prejšnjem programu. Program podpira tudi kriptiranje sporočila. To je zelo dobro, saj s tem zelo izboljšamo možnosti, da našega sporočila ne bo mogoče prebrati s prej omenjenimi metodami odkrivanja sporočil. Prav tako kot prejšnja knjižica nam program omogoča skrivanje teksta in slik.

3.3 Odkrivanje sporočil

Preizkusili smo tudi nekaj programov za odkrivanje steganografije v slikah. Ustvarili smo šest različnih primerov:

- **Primer 1**. Tekst "Hello hidden world!!¨ skrit v sliko žirafe (slika 7) po postopku enostavnega LSB skrivanja.
- **Primer 2**. Manjša slika leva (128x128 pikslov) (slika 8) skrita v sliko žirafe (slika 7) po postopku enostavnega LSB skrivanja.
- **Primer 3**. Tekst "Hello hidden world!!¨ skrit v sliko žirafe (slika 7) po postopku naključnega LSB skrivanja s programom OpenStego.
- **Primer 4**. Tekst "Hello hidden world!!¨ zakriptiran s ključem "p" in skrit v sliko žirafe (slika 7) po postopku naključnega LSB skrivanja s programom OpenStego.
- **Primer 5**. Slika leva (slika 8) skrita v sliko žirafe (slika 7) po postopku naključnega LSB skrivanja s programom OpenStego.
- **Primer 6**. Slika leva (slika 8) zakriptiran s ključem "p" in skrita v sliko žirafe (slika 7) po postopku naključnega LSB skrivanjas programom OpenStego.

Slika 7: Krovna slika uporabljena pri vseh primerih

Slika 8: Slika leva, ki smo jo skrili v krovno sliko

3.3.1 StegExpose

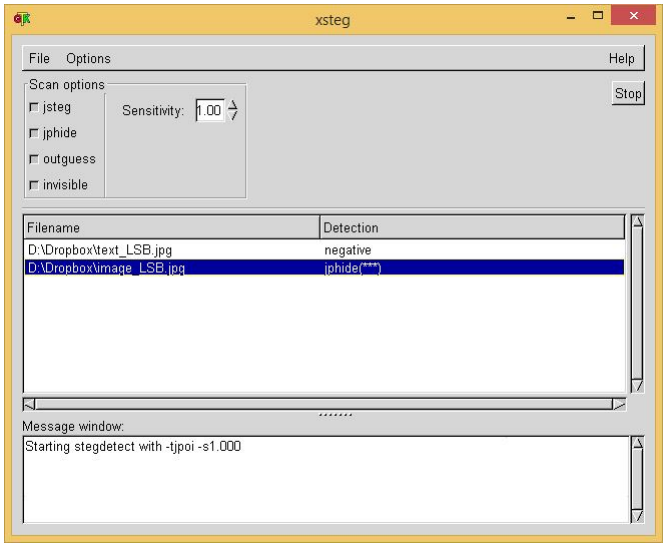
Prvi program, ki smo ga preizkusili je StegExpose [3]. Omogoča nam samo detekcijo ne pa tudi pridobivanje skritega sporočila. Deluje po principu že preverjenih metod za odkrivanje in analizo steganografije kot so Sample Pairs by Dumitrescu [5], RS Analysis by Fridrich [7], Chi square Attack by Wastfeld [13] in Primary Sets by Dumitrescu [4]. Pri naših testih program ni bil preveč učinkovit. Uspešno je ugotovil, da je slika sumljiva pri primeru 5 in 6. Se pravi pri skrivanju slike v sliko po postopku naključnega LSB in pri skrivanju zakriptirane slike v sliko po postopku naključnega LSB. Razlog za neuspešnost pri ostalih primerih je po našem mnenju, da je niz "Hello hidden world!!" prekratek, da bi ga program zaznal. V primeru dve pa je skrita slika manjša kot v primeru 5 in 6 zato, je tudi v tem primeru po vsej verjetnosti program ni zaznal zaradi premajhne velikosti.

```
> java -jar StegExpose.jar openstego/
photoNoPass.png is suspicious. Approximate amount of hidden data is 91266 bytes.
photoSimplePass.png is suspicious. Approximate amount of hidden data is 89228 bytes.
> java -jar StegExpose.jar lsb_steg/
```

Slika 9: Uporaba programa stegexpose za detekcijo steganografije

3.3.2 StegDetect

StegDetect [2] omogoča detekcijo in tudi pridobivanje skritega sporočila. Na žalost pa deluje samo nad datotekami jpg. Ker program OpenStego kot izhod lahko zapiše samo png datoteke smo ta program preizkusili samo na prvih dveh primerih. V primeru 1, detekciji teksta je bil program neuspešen. V primeru 2 pa je program uspešno zaznal in odkril skrito sliko leva (slika 8). Na sliki 10 vidimo uporabniški vmesnik programa, ki prikazuje ime datoteke in ali je bila odkrita steganografija.



Slika 10: Uporaba programa StegDetect

4. ZAKLJUČEK

Končni cilj steganografsko-analitične forenzike ni le detektiranje skritih informacij v slikah, temveč tudi njihova ekstrakcija. Koncept lokacije tovora je pripomogel k realizaciji tega cilja. Glavna predpostavka pri lokaciji tovora je, da je velikost tovora točno določena. Ta predpostavka je seveda nepotrebna in nerealna, razen v posebnih primerih, ko se zahteva točno določeno velikost tovora. Jasno je, da se od steganografa pričakuje, da bo v sliko poskril sporočila različnih velikosti. V takšnih primerih tu opisan pristop pokaže, da je sporočila vseeno mogoče izluščiti tako, da locirane dele tovora uredimo po padajočem vrstnem redu povprečnih ostankov slike. Tak pristop je primeren tudi za druge domene, kot so npr. kvantizirani JPEG koeficienti.

Čeprav je bilo prikazano, da je moč izluščiti sporočila tudi ob nepoznavanju vgradnega ključa, je jasno, da naloga forenzikov postaja vedno težja. Pri analizi morajo imeti zadostno količino steganografskih slik, da so lahko pri preiskovanju uspešni. Prav tako je v pomoč, da imajo dostop do krovnih slik, predvsem zato, ker so trenutni algoritmi ocenjevanja krovnih slik pomankljivi. Kljub vsemu temu lahko v resničnih primerih realistično pričakujemo, da bodo uspešno izluščeni le deli sporočil, kar pa digitalnim forenzikom vseeno koristi. Opisan pristop je predvsem primeren za iskanje sporočil, skritih z algoritmi, baziranimi na bločnem vgrajevanju. Bolj sofisticiran steganograf se lahko tej metodi izogne z uporabo naprednejših algoritmov, ki pri skrivanju upoštevajo velikost sporočila.

S praktičnega vidika gledano, računalniki operirajo nad bajti in ne biti. Tovor je torej sestavljen iz bajtov in urejamo skupine po osem bitov. V praksi so lahko steganografske slike generirane z različnimi ključi. V takšnih primerih moramo biti sposobni ločiti te slike po vgradnih ključih in opisan pristop uporabiti na posamezni skupini slik na enkrat.

5. LITERATURA

[1] Openstego homepage. `http://openstego.sourceforge.net/`, note = Datum dostopa: 9.5.2015.

[2] Stegdetect homepage. `http://www.outguess.org/detection.php`, note = Datum dostopa: 9.5.2015.

[3] Stegexpose homepage. `https://github.com/b3dk7/StegExpose`, note = Datum dostopa: 9.5.2015.

[4] S. Dumitrescu, X. Wu, and N. Memon. On steganalysis of random lsb embedding in continuous-tone images. In *Image Processing. 2002. Proceedings. 2002 International Conference on*, volume 3, pages 641–644. IEEE, 2002.

[5] S. Dumitrescu, X. Wu, and Z. Wang. Detection of lsb steganography via sample pair analysis. *Signal Processing, IEEE Transactions on*, 51(7):1995–2007, 2003.

[6] T. Filler, A. D. Ker, and J. Fridrich. The square root law of steganographic capacity for markov covers, 2009.

[7] J. Fridrich. *Steganography in digital media: principles, algorithms, and applications*. Cambridge University Press, 2009.

[8] J. Fridrich and M. Goljan. On estimation of secret message length in lsb steganography in spatial domain, 2004.

[9] A. Ker. A capacity result for batch steganography. *Signal Processing Letters, IEEE*, 14(8):525–528, Aug 2007.

[10] A. D. Ker. The square root law requires a linear key. In *Proceedings of the 11th ACM Workshop on Multimedia and Security*, MM&Sec '09, pages 85–92, New York, NY, USA, 2009. ACM.

[11] T.-T. Quach. Extracting hidden messages in steganographic images. *Digital Investigation*, 11, Supplement 2(0):S40 – S45, 2014. Fourteenth Annual {DFRWS} Conference.

[12] D. Robin. Lsb-steganography. `https://github.com/RobinDavid/LSB-Steganography`. Datum dostopa: 9.5.2015.

[13] A. Westfeld and A. Pfitzmann. Attacks on steganographic systems. In *Information Hiding*, pages 61–76. Springer, 2000.

PRILOGE

A. PYTHON PROGRAM ZA UPORABO LSB KNJIŽNICE

Python koda za uporabo LSB knjižice. Vsebuje skrivanje teksta v LSB in MSB del piksla ter skrivanje slike v sliko.

```
import MSBSteg as msb
import LSBSteg as lsb
import cv
#Hide
str = "Hello␣hidden␣world!!"

#Skrivanje teksta v MSB piksle slike
carrier = cv.LoadImage("g.png")
steg = msb.MSBSteg(carrier)
steg.hideText(str)
steg.saveImage("text_MSB.png")

#Skrivanje teksta v LSB piksle slike
```

```
carrier = cv.LoadImage("g.png")
steg = lsb.LSBSteg(carrier)
steg.hideText(str)
steg.saveImage("text_LSB.jpg")

#Skrivanje slike v sliko
imagetohide = cv.LoadImage("l_small.jpg")
carrier = cv.LoadImage("g_big.png")
steg.hideImage(imagetohide)
steg.saveImage("image_LSB.jpg")
```

Analiza stisnjenega RAM-a na operacijskih sistemih Mac OS X in Linux

Tomislav Slijepčevič
Fakulteta za računalništvo in informatiko
Rusjanov trg 2
Ljubljana, Slovenija
ts2287@student.uni-lj.si

Jakob Kokalj
Fakulteta za računalništvo in informatiko
Goričane 13a
Medvode, Slovenija
jakob.k91@gmail.com

ABSTRACT

V forenziki se čedalje bolj uporablja analiza računalniškega pomnilnika z namenom izboljšanja že uveljaljenih forenzičnih metod za pregledovanje podatkovnih nosilcev, saj je v neobstojnem pomnilniku veliko število informacij, ki jih na statičnih pomnilnikih ni.

Z zajetjem neobstojnega pomnilnika dobimo vpogled v delujoče stanje sistema, informacije o programih, ki se izvajajo, podatke o tekočih internetnih povezavah in še več. Še vedno pa nastopajo težave pri analizi izmenjevalnih datotek (ang. swap file), ki običajno kot particije na disku služijo kot dodatni prostor za shranjevanje informacij, do katerih lahko potem po potrebi dostopa operacijski sistem. Čeprav so lahko izmenjevalne datoteke bogat vir informacij, je njihova analiza vsebovala le iskanje znakovnih nizov in majhnih binarnih struktur.

Bolj napredna analiza izmenjevalnih datotek je otežena zaradi vse bolj pogoste uporabe enkripcije na izmenjevalnih datotekah in zajemanja medsebojno skladnih izpisov pomnilnika in izmenjevalnih datotek. Vendar novejši operacijski sistemi skušajo zmanjšati izmenjavanje na disk z uporabo stiskanja. Hramba stisnjenih strani v RAM-u poveča učinkovitost delovanja, poleg tega pa predstavlja novo možnost zajemanja digitalnih dokazov, ki so bili v preteklosti izmenjani na disk.

Ta članek razpravlja o težavnosti analize izmenjevalnih datotek v večjih detajlih, možnostih stisnjenega RAM-a na operacijskih sistemih Mac OS X in Linux ter o novih orodjih za analizo le tega.

Keywords

Analiza pomnilnika, stisnjeni RAM, izmenjevalne datoteke, digitalna forenzika

1. UVOD

Klasična digitalna forenzika se je do sedaj predvsem osredotočala na obstojne pomnilnike in sicer kako jih hraniti, preslikovati, obnoviti in analizirati. Pomnilniki so se ponavadi zajemali tako, da si pred odklopom ugasnil gostiteljski računalnik, a si s tem izgubil vsebino njegovih neobstojnih pomnilnikov (pomnilniki RAM). V neobstojnih pomnilnikih je shranjeno gostiteljevo stanje. Sestavljeno je iz informacij o izvršujočih procesih, informacij o mrežnih povezavah, zlonamerne programske opreme, podatkov iz odložišča in podatkov od operacijskega sistema ter programskih podatkovnih strukturah.

Forenzična skupnost se je začela zavedati potenciala za analizo spomina in živo forenziko. S slednjo označujemo preiskovanje prižganih naprav in gostiteljev. Tipično se na prižganih gostiteljih zažene statično prevedene programe, s čimer dobimo vpogled v trenutno stanje gostitelja. Ti programi so običajno administrativna orodja, ki med drugim prikažejo izvršujoče procese, nadzirajo aktivnost datotečnega sistema, zajamejo mrežni promet, nadzirajo spremembe v registru Windows in poskušajo zaznati zlonamerno programsko opremo.

Analiza spomina običajno vključuje zajem spomina (angl. RAM dump) in preiskovanje z orodji za analizo spomina. Spomin se zajame s kombinacijo programske in strojne opreme. Z živo forenziko ima skupno to, da obe omogočata vpogled v podatke, ki bi bili drugače forenziku nedostopni. Prav tako sta oba pristopa dosti neinvazivna, saj ne posegata toliko v gostiteljevo stanje. Pri analizi spomina se poskuša poseg na gostitelju čim bolj minimizirati in zato je dovolj, če uporabimo le orodje za prepisovanje spomina. Zaradi nedavnih napredkov na področju analize spomina je možno v laboratoriju iz zajetega spomina obnoviti večino gostiteljevega stanja, ki bi ga sicer videli z živo forenziko. Problem pri analizi spomina je preiskovanje datoteke za izmenjevanje (ang. swap), ki služi kot dodatna shramba navideznemu pomnilniku operacijskega sistemu. Datoteka je shranjena na disku in vsebuje strani navideznega pomnilnika, ki so bile izmenjane iz spomina. Do izmenjanja pride zaradi pomanjkanja spomina, do pomanjkanja pa pride, ko se sočasno izvršuje veliko procesov oz. ko imamo procese, ki so spominsko požrešni. Izmenjevalna datoteka je lahko velika in se nahaja na počasnem in obstojnem mediju, zato je težko sočasno zajeti konsistentni kopiji spomina in izmenjevalne datoteke, dokler se sistema izvaja.

Moderni operacijski sistemi imajo v navideznem sistemu novo komponento imenovano stisnjen RAM oz. stisnjen SWAP, ki nam omogoča preiskovanje gradiva, ki bi v preteklosti bilo izmenjano na disk. Predstavili bomo delovanje navideznega pomnilnika, kako se analizira spomina in nove vtičnike za ogrodje Volatility, ki samodejno prepoznajo in raztegnejo stisnjena območja spomina na operacijskem sistemu Mac OS X Mavericks in Linux.

2. ANALIZA SPOMINA NA MODERNIH SISTEMIH Z NAVIDEZNIM POMNILNIKOM

Navidezni pomnilnik je bistvena komponenta modernih operacijskih sistemov, saj zagotavlja linearni naslovni prostor za procese in precej poenostavi upravljanje pomnilnika. Operacijski sistemi pogosto uporabljajo mehanizem za klicanje virtualnega spomina in tako omogočijo velikosti dodeljenega pomnilnika tekočih procesov, da preseže fizično velikost RAM-a tako, da presežek prelijejo v izmenjevalno datoteko. V tem članku smo posvetili operacijskim sistemom, ki podpirajo tako klicanje.

Na modernih napravah je navidezni pomnilnik implementiran s kombinacijo strojne in programske opreme, kjer večina sodobnih CPU-jev priskrbi strojno podporo za prevajanje med navideznimi in fizičnimi naslovi in sledenje prisotnih strani v RAM-u. Dostop do neprisotnih strani povzroči napako strani, za katero poskrbi operacijski sistem tako, da sproži eno od možnih dejanj, med drugim tudi dodelitev spomina ali izmenjavo strani iz izmenjevalne datoteke.

Izmenjavanje strani mora biti minimalno, želimo se izogniti razmetavanju [13], kjer se zgodi, da se strani ves čas vzema ali odlaga v izmenjevalne datoteke zaradi kritičnega pomanjkanja RAM-a. Zaradi razlike v pasovni širini diska in pomnilnika posledično pride do vpliva na delovanje računalnika, ki se razlikuje za več redov velikosti. Kot primer te razlike lahko pogledamo delovanje visoko zmogljivega računalnika Mac Pro, ki ga je Apple predstavil leta 2013. Pasovna širina pomnilnika je do 60 GB/s, medtem ko je pasovna širina diska do 1,2 GB/s. Moderni operacijski sistemi uporabljajo napredne algoritme za nadomeščanje strani, ki stremijo k ohranjanju strani v RAM-u za vse programe [14], ki se trenutno izvajajo na operacijskem sistemu. Kljub temu še vedno prihaja do izmenjavanja pri povečani uporabi pomnilnika, vendar ti algoritmi pripomorejo k največji učinkovitosti delovanja.

3. IZMENJEVALNE DATOTEKE KOT VIR DOKAZOV

Kot smo že omenili, so izmenjevalne datoteke lahko zanimiv vir informacij, vendar je običajno analiza vsebovala le iskanje znakovnih nizov ali pa manjših binarnih struktur. Ta iskanja so ciljala na IP naslove [16], številke kreditnih kartic, delčke spletnih strani itd.. V mnogo primerih so bili zadetki razkrivajoči, vendar je bilo skoraj nemogoče ugotoviti lastništvo podatkov v izmenjevalni datoteki brez dodatnega analiziranja jedra operacijskega sistema v navideznem pomnilniku.

Prvič, izmenjevalna datoteka je organizirana kot neurejena zbirka neobdelanih strani (ali segmentov, ki so po velikosti večji od strani). Že določanja, kateri proces je neodvisno od jedrske strukture generiral podatke v izmenjevalni datoteki, se ne da izvesti zanesljivo. Drugič, izmenjevalna datoteka se običajno ne pobriše ob ponovnem zagonu računalnika,[1] kar v izmenjevalni datoteki pusti zastarele podatke. To pomeni da lahko v izmenjevalni datoteki ostanejo strani pomnilnika, ki so se preko mnogo ponovnih zagonov sistema prepletle v izmenjevalno datoteko. Tretjič, nekateri operacijski sistemi, kot na primer Microsoft Windows, ne počistijo diskovnih blokov, ko so ti prvotno dodeljeni izmenjevalni datoteki, kar pomeni, da se lahko v izmenjevalni datoteki nahajajo podatki, ki so *popolnoma neodvisni od navideznega pomnilnika sistema.*

Dodaten zaplet pri analizi izmenjevalnih datotek je uporaba enkripcije, ki zmanjša puščanje neobstojnih, privatnih podatkov na obstojne podatkovne nosilce. Na primer pri operacijskem sistemu Mac OS X 10.7 in kasnejših verzijah so izmenjevalne datoteke že privzeto kriptirane, ne glede na to ali je vklopljena enkripcija celotnega diska File Vault 2. Poleg tega so 128-bitni AES ključi, ki se uporabljajo za enkripcijo izmenjevalnih datotek, drugačni od tistih, ki se uporabljajo za enkripcijo celotnega diska, in so ponovno generirani ob vsakem zagonu računalnika.

Poglejmo si funkcijo `swap_crypt_ctx_initialize` v jedru operacijskega sistema Mac OS X Mavericks [2]. Funkcija se izvrši ob prvi operaciji izmenjave strani pri zagonu sistema in tako zagotavlja, da bodo vse izmenjane strani v trenutni seji kriptirane z novim ključem. Podpora za enkricijo izmenjevalnih datotek je mogoča tudi na v Windowsih (Vista in naprej) ter pri sistemih Linux preko funkcije dm-crypt, vendar pri obeh sistemih ne moremo kriptiranja omogočiti kot privzeto možnost.

Če lahko pridobimo izpis fizičnega pomnilnika in kopija izmenjevalne datoteke, potem lahko lahko v teoriji uporabimo analizo pomnilnika da povežemo izmenjane strani v navideznih naslovnih prostorih procesov z njihovimi lastnimi procesi in da dekriptiramo strani (iz RAM-a dobimo ključe). V praksi to ne deluje najbolje zaradi nekonsistentnosti v izpisu fizičnega pomnilnika v primerjavi z izmenjevalno datoteko, saj sistem med zajemanjem pomnilnika še vedno deluje. Večina delujočih sistemov običajno zelo težko konsistentno zajemo tako RAM kot izmenjevalno datoteko [9, 17], izjema so sistemi, ki jih poganjamo v navideznih okoljih [15], kjer lahko uporabimo posnetek navideznega sistema, da bolj konsistentno zajamemo RAM in izmenjevalno datoteko hkrati.

Da bi videli zakaj, predpostavimo, da medtem ko poteka zajem izmenjevalne datoteke, sistem nadaljuje z izvajanjem in preslikavanjem med stranmi in njihovimi lastniki, posledično se lahko lokacije v izmenjevalni datoteki spremenijo, kar se pokaže v veliki neskladnosti. Če se uporabi priporočen pristop in najprej zajamemo pomnilnik, potem bo analitik zajel spomin s pomočjo stroj in programske opreme in potem uporabil drugo programsko opremo za zajem izmenjevalne datoteke. Na sistemih z zmerno do veliko velikostjo RAM-a bo prvotni proces trajal nekaj minut. Med tem čas sistem še vedno izvaja programe in spreminja podatke v izmenje-

[1] S spreminjanjem registrov v Microsoft Windows lahko ukažemo operacijskemu sistemu naj ob zaustavitvi sistema počisti izmenjevalno datoteko, z namenom izboljšanja varovanja, vendar žrtvujemo učinkovitost delovanja.

valni datoteki. Do takrat, ko imamo zajeta tako pomnilnik in izmenjevalno datoteko je zelo verjetno (in je bilo opaženo pri poskusih), da izmenjani podatki v jedru kazali na strani, ki so bile že prepisane.

Če program za zajemanje slepo zaupa tem podatkom, potem bo bral podatke, ki niso povezani z prevedenim navideznim naslovom. Na primer to lahko pomeni, da bo izmenjana preslikava znotraj Internet Explorerja (na operacijskem sistemu Windows) kazala na izmenjevalno datoteko, odmaknjeno na disku, kjer so zdaj podatki nekega drugega procesa. Podobne težave se pojavijo celo takrat, ko poiskuša preiskovalec zajeti pomnilnik in izmenjevalno datoteko hkrati. Ker zaenkrat še ni poznanega načina, kako bi s programom zaznali te neskladnosti, je analiza izmenjevalnih datotek zelo težka, pristnost podatkov pridobljenih s takimi analizami pa je negotova.

Ideja stiskanja RAM-a, da bi zmanjšali pomnilniško obremenitev ter tako odpravili ali minimizirali uporabo izmenjevalnih datotek, ni nova, saj so se prvi komercialni produkti kot na primer RAM Doubler pojavili že leta 1995. Tudi akademske raziskave na tem področju so stare že več kot desetletje [23, 12]. Zgodnji oviri pri uspešnem izvajanju stiskanja RAM-a sta bili počasni procesorji in slaba sistemska integracija. Nove naprave za stiskanje RAM-a pri Mac OS X Mavericks in naprave za stiskanje izmenjevalnih datotek pri sistemih Linux pa izkoristijo hitrost večjedrnih procesorjev in tesno integrirajo stisnjen RAM v njihove navidezne pomnilne sisteme. Če damo na stran zmerno količino RAM-a za komponento, ki ji bomo rekli *kompresor* (glej poglavje Analiza stisnjenega RAM-a za implementacije na Linux in Mac OS X sistemih), in za stiskanje strani pomnilnika, ki jih shranimo na disk namesto izmenjamo, lahko preskrbimo zmerno obremenitev pomnilnika brez uporabe izmenjevalnih datotek. Ko pa obremenitev preide neko kritično mejo, potem se lahko strani iz RAM-a (in kompresorja) še vedno izmenjajo na disk.

Z ustreznimi orodji je analiza stisnjenega RAM-a precej lažja kot pri izmenjevalnih datotekah. Pojava neskladnosti pri navideznem pomnilniku je minimalna, saj je zajemanje RAM-a precej hitrejše kot pri izmenjevalnh datotekah, ki se nahajajo na počasnejših, obstojnih pomnilnikih. Da bi izkoristili to priložnost, sta raziskovalca Golden G. Richard in Andrew Case razvila nekaj novih orodij za analizo stisnjenega RAM-a, ki podatke avtomatsko raztegnejo in umestijo stisnjene strani v njihov naslovni prostor procesov, ki jih pregledujemo. V naslednjem poglavju so predstavljene podrobnosti implementacije stisnjenega RAM-a na operacijskih sistemih Mac OS X in Linux in orodja, ki so vgrajena v ogrodje Volatility [22].

4. ANALIZA STISNJENEGA RAM-A

Volatility je odprtokodno ogrodje za spominsko forenziko napisano v jeziku Python. Analizirati zna posnetke spomina (angl. RAM dump) operacijskih sistemov Linux, Mac in Windows v 32 in 64-bitni inačici. Ogrodje se uporablja za hitro in natančno modeliranje strukture jedra operacijskega sistema in omogoča preslikovanje navideznih naslovov v fizične. Prav tako omogoča obnovitev navideznega pomnilniškega prostora za procese tako, da pregleda strani, ki so bodisi prisotne v posnetku spomina bodisi izmenjane. Trenutna različica ogrodja Volatility ne obdeluje izmenjanih strani, temveč samo označi njihovo odsotnost. Novi vtičniki[2] za ogrodje Volatility omogočajo analizo stisnjenega RAM-a na sistemu Mac OS X in Linux. Z vtičniki je možno analizirati strani, ki so odsotne, toda stisnjene.

4.1 Stisnjen RAM v Mac OS X

Mac OS X ima hibridno jedro, sestavljeno iz sistema BSD in komponent Mach. Navidezni pomnilniški sistem je implementiran znotraj Mach, kot je vidno na sliki 2. Vsako opravilo oz. proces ima v jedru svoj naslovni prostor, predstavljen z `vm_map`, ki vsebuje dvojno povezan seznam objektov `vm_map_entry`. Objekti v seznamu so dostopni prek `vm_map_header`. Posamezen objekt prestavlja niz sosednjih pomnilniških naslovov v linearnem naslovnem prostoru opravila in je povezan z objektom `vm_object`. Slednji upravlja s stranmi velikosti 4K, ki so v spominu ali shrambi, kot je npr. datoteka na disku ali izmenjevalna datoteka (angl. swap file). Ostranjevalnik (angl. pager) je odgovoren za jemanje strani iz shrambe, če so le-te odsotne. V Mac OS X različici Mavericks sta najpomembnejša ostranjevalnika *vnode pager* in *compressor pager*. Prvi upravlja s stranmi, ki so shranjene v datoteki na disku, slednji pa transparentno stisne oz. raztegne strani, ki bi sicer bile prenešene bodisi v izmenjevalno datoteko na disku bodisi iz izmenjevalne datoteke. Čeprav ima vsak objekt `vm_object` svoj zasebni ostranjevalnik *compressor pager*, obstaja tudi globalni ostranjevalnik v objektu `compressor_object`, ki hrani vse stisnjene strani in je zadolžen za stiskanje in raztezanje, kot tudi za vzdrževanje stisnjenih strani. Med drugim je tudi njegova naloga, da nadzoruje prostor za stisnjene strani in označuje segmente, ki bi morali biti izmenjani na disk.

4.1.1 Podpora v ogrodju Volatility

Razvijalca vtičnikov sta najprej preverila, če se lahko stisnjene strani v Mac OS X identifica in raztegne [18]. Raziskala sta shrambo stisnjenih strani, ki prebiva v globalnem objektu `compressor_object`. Ugotovila sta, da so operacije in vmesniki za upravljanje s shrambo, stiskanje in raztezanje implementirani v datotekama `vm_compressor.{h, c}` (v direktoriju `osfmk/vm/`). Za stiskanje in raztezanje strani se uporablja optimizirana implementacija algoritma WKdm v 64-bitni zbirni kodi, ki se nahaja v datotekah `WKdmDecompress_new.s`, `WKdmData_new.s` in `WKdmCompress_new.s` (direktorij `osfmk/x64_64`). Avtorja sta analizirala implementacijo WKdm v zbirniku in ugotovila, da je relativno podobna prvotni implementaciji od Wilsona in Kaplana [23].

Da bi raztezala strani v Pythonu, sta avtorja implementirala WKdm v Pythonu, ki je kompatibilen različici v zbirniku. Nato sta implementirala vtičnik `mac_compressed_swap` za Volatility, ki najprej v jedru izčrpa lokacije pomembnih podatkovnih struktur povezanih s kompresorjem in nato analizira kompresorjevo shrambo, kjer so stisnjene strani. Shramba je dinamično alocirano polje elementov tipa `c_segment`. Struktura `c_segment` je prikazana v izvlečku kode 1. Vsak `c_segment` ima en izravnalnik (angl. buffer), v katerem hrani informacije o stisnjenih straneh, in 2D polje za dostop do strani v izravnalniku.

[2]Novi vtičniki: mac_compressed_swap, mac_dump_maps, linux_compressed_swap| in linux_dump_maps.

Izpis izvorne kode 1: Struktura segmenta v kompresorjevi shrambi na Mac OS X.

```
struct c_segment {
    // ...
    int32_t      c_bytes_used;
    int32_t      c_bytes_unused;
    uint32_t     c_mysegno:19,
    // ...
                 c_ondisk:1,
                 c_was_swapped_in:1,
                 c_on_minorcompact_q:1,
                 c_on_age_q:1,
                 c_on_swappedin_q:1,
                 c_on_swapout_q:1,
                 c_on_swappedout_q:1,
                 c_on_swappedout_sparse_q:1;
    uint16_t     c_firstemptyslot;
    uint16_t     c_nextslot;
    uint32_t     c_nextoffset;
    uint32_t     c_populated_offset;
    uint32_t     c_creation_ts;
    uint32_t     c_swappedin_ts;
    union {
        int32_t      *c_buffer;
        uint64_t     c_swap_handle;
    }
    // ...
    struct c_slot *c_slots[C_SEG_SLOT_ARRAYS];
};
```

Lokacija stisnjene strani v kompresorju je opredeljena z oknom (angl. slot), ki je predstavljen s strukturo `c_slot_mapping` (izvleček kode 2). Okno sledi številki segmenta (`s_cseg`) in omogoča izračun indeksa polja struktur `c_segment` ter 10-bitnega indeksa (`s_cindx`). Slednji indeks tvori dva manjša indeksa, s katerima nato pridobimo lokacijo strani v segmentovem dvodimenzionalnem polju, kjer hrani lokacije svojih strani. Ko je stran v naslovnem prostoru procesa stisnjena oz. raztegnjena, preslikavo med naslovom strani in lokacijo strani v kompresorju vzdržuje tisti zasebni ostranjevalnik, ki je asociiran s stranjo. Naloga kompresorja ni, da sledi navideznemu naslovu, ampak da deluje le nad okni, ki sledijo lokacijam poljubnih strani v shrambi. Ostranjevalnik mora znati asociirati posamezne strani s predalom v shrambi.

Izpis izvorne kode 2: Struktura za preslikovanje oken, ki jedru omogoča sledenje lokacij stisnjenih strani.

```
struct c_slot_mapping {
    uint32_t     s_cseg:22,
                 s_cindx:10;
};
```

Vtičnik `mac_compressed_swap` se sprehodi po strukturah `c_segment` in ugotovi, kateri segmenti so prisotni. Za vsakega prisotnega se nato še sprehodi po 2D polju z okni in izračuna lokacijo ter dolžino pripadajoče strani v `c_buffer`. Če je stisnjena stran dolga natanko 4K, pomeni da WKdm stiskanje ni uspelo zmanjšati prvotne velikosti strani, kar pomeni, da je stran nespremenjena in primerna za takojšnjo analizo. Sicer je treba stran raztezati z implementacijo WKdm v Pythonu-u. Če raztezanje strani uspe, potem jo lahko analiziramo, sicer jo moramo zavreči. Moramo poudariti, da ta vtičnik ne asociira raztegnjene strani s procesom, ampak le zbere vse stisnjene strani za nadaljnjo analizo, saj izvor stisnjenih strani ni na voljo v kompresorju. Za asociiranje strani v naslovnem prostoru procesa z lokacijo v kompresorju potrebujemo preslikavo med naslovom strani in oknom v kompresorju. To počne zasebni ostranjevalnik za posamezen objekt `vm_map_entry`. Za ta namen je bil razvit vtičnik `mac_dump_maps`.

Asociiranje stisnjenih strani s posameznim procesom je kompleksnejše in zahteva globlje razumevanje podsistem Mach za ostranjevanje, ki je sestavljen iz približno 70.000 vrstic kode v jeziku C. Vtičnik `mac_dump_maps` se sprehodi po navideznem naslovnem prostoru enega ali več procesov in prekopira vse razpoložjive strani v datoteko. Volatility bi pri kopiranju strani preskočil neprisotne strani v spominu, toda vtičnik `mac_dump_maps` prestreže preskočeno stran in pogleda, če je zasebni ostranjevalnik za asociiran objekt `vm_map_entry` tipa kompresor. Vkolikor je tipa kompresor, potem vtičnik preveri, če kompresor lahko zagotovi stran. Če stran obstaja v kompresorju, potem se stran raztegne z uporabo prejšnjega vtičnika.

4.2 Stisnjen swap v Linuxu

Linux od jedra v3.11 ponuja mehanizem za ravnanje ob obremenitvi spomina, imenovan stisnjen swap (angl. compressed cache for swap pages). Omogoča transparentno stiskanje strani v spominu, ki bi bile sicer prenese na disk. Uporaba mehanizma je neobvezna, vključi pa se z nastavitvijo možnosti zswap.

4.2.1 Frontswap

Frontswap je generični API, ki implementira stisnjen swap. Zadolžen je za upravljanje izmenjanih strani, nalaganje izmenjanih strani in priključevanje ter izključevanje zaledij (angl. backends), kot je zswap. Frontswap je integriran v jedro prek funkcij `swap_writepage`| in `swap_readpage`. Ti sta poklicani ob zapisovanju in branju strani v swap. Običajno ti funkciji neposredno bereta iz diska oz. pišeta na disk.

Funkcija `swap_writepage` najprej pokliče funkcijo `try_to_free_swap`, ki preveri, če je stran treba še obdržati v shrambi swap. Vkolikor frontswap zna stran stisniti in shraniti, potem ne pride do zapisovanja na disk. Funkcija `swap_readpage` preveri prisotnost strani v stisnjenem pomnilniku, preden gre ponjo na disk. `frontswap_store` je preprosta ovojna funkcija, ki preveri, če je frontswap vključen, in pokliče funkcijo `__frontswap_store`. Slednja prejme strukturo strani, ki jo želimo shraniti in jo nato posreduje zswapu. Posreduje mu tudi tip strani in odmik znotraj pomnilnika swap. Ko zswap shrani stran, se nato še stran označi, da je bila uporabljena v frontswap. `frontswap_load` je inverzna operacija od `frontswap_store` in priskrbi zahevano stran iz stisnjenega pomnilnika swap. Stran se priskrbi tako, da se najprej iz njene strukture izvleče podatke o tipu in odmiku ter se jo nato poišče tako, da se pregleda zapise znotraj zswap.

4.2.2 Zswap

Zswap je implementacija stisnjenega swapa za Linux. Shranjuje stisnjene strani in jih vzdržuje z uporabo dodeljevalnika zbud ter rdeče-črnih dreves. zbud (direktorij `mm/zbud.`

c) je dodeljevalnik spomina in je zasnovan za stisnjene strani. Hrani dve stisnjeni strani znotraj ene fizične strani in je zadolžen za dodeljevanje, sproščanje in sledenje statusom strani.

Za forenzike sta najpomembnejši operaciji za shranjevanje in nalaganje v implementaciji zswap. Če želimo poiskati strani, moramo razumeti operacije in uporabljene podatkovne strukture. `zswap_frontswap_store` je funkcija zswap, ki implementira shranjevanja v frontswapu in je zadolžena za sledeča opravila:

- Stiskanje dane strani.
- Dodeljevanje spomina z zbud za shranjevanje stisnjene strani.
- Ustvarjanje zapisa `zswap_entry` s podatki o strani in shranjevanje zapisa v drevo stisnjenih strani.
- Posodabljanje statističnih podatkov, ki so na voljo za nadzorovanje statusa zswap.

Privzet stiskalni algoritem je LZO, ki ima funkciji za stiskanje in raztezanje implementirani v jedru. Trenutno naš vtičnik podpira samo LZO, vendar se zlahka vključi tudi druge algoritme. Za razliko od Mac OS X, ki zaradi učinkovitosti uporablja ročno optimizirano zbirno kodo WKdm, Linux uporablja implementacijo v C-ju, kar pomeni, da lahko zlahka dodamo nove algoritme.

`zswap_frontswap_load` je funkcija zswap, ki implementira nalaganje v frontswapu. Deluje tako, da poišče stran v drevesu in jo nato raztegne. Isto operacijo mora izvršiti novi vtičnik za Volatility, ko se stran raztegne v stisnjenem pomnilniku swap.

4.2.3 Podpora v Volatility

Prvi razvit vtičnik za Linux je `linux_compressed_swap`, ki izpiše statistične podatke o statusu fronswapa in zswapa ter izvleče vse stisnjene strani in jih v neraztegnjeni obliki zapiše na disk. Drugi vtičnik je predelana različica `linux_dump_maps`, ki zapiše preslikave spomina na disk vključno z raztegnjenimi stranmi.

`linux_compressed_swap` najprej pridobi podatke zswapa in fronswapa, kot je št. vseh stisnjenih strani in št. stisnjenih strani na swap datoteko. Nato razišče vsako zswap drevo v datoteki swap in poišče lokacijo stisnjene strani.

`linux_dump_maps` se sprehodi po preslikavah spomina procesov in jih zapiše na disk. Te preslikave lahko vsebujejo izvršljive datoteke, deljene knjižnice, sklad, kopico, anonimni spomin procesa, itd. Vtičnik uporablja API Volatility za iskanje začetnega in končnega naslova območij v spominu, ki pripadajo procesu. Če je stran najdena, jo izpiše na disk, sicer se poišče stran v stisnjenem pomnilniku swap.

5. OCENJEVANJE

Pri snovanju, implementaciji, testiranju in ocenjevanju orodij sta raziskovalca testirala več konfiguracij Mac OS X in

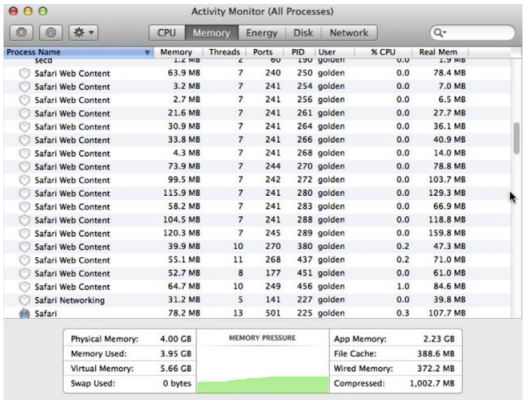


Figure 1: Aplikacija Activity Monitor pri Mac OS X Mavericks, ki prikazuje pomnilniško obremenitev in uporabo stisnjenega RAM-a. V tem primeru je odprtih več Safari oken, vidimo lahko več kot 1 GB stisnjenega RAM-a in nobenega izmenjavanja.

Linux sistemov, ki se razlikujejo po velikosti RAM-a in obremenitvi pomnilnika. Predvsem sta bila pozorna na pravilen postopek zajema in kako zajeti (oziroma nezajeti) stisnjeni podatki vplivajo na preiskavo, tako kvalitativno (dokazi le v stisnjenih regijah) kot kvantitativno (koliko podatkov je stisnjenih). Seveda se količina in relevantnost dokazov spreminja od primera do primera, vendar je po najinih opažanjih na velikem številu sistemov z velikostjo RAM-a od 2 GB do 32 GB velik kos podatkov stisnjen in nedostopen do programov, ki se ne zavedajo stisnjenega RAM-a.

To še posebej velja pri Mac OS X sistemih, kjer se stisnjen RAM uporablja bolj pogosto. Aplikacija Activity Monitor na teh sistemih nam kaže obremenitev pomnilnika in statistiko kompresorja. Na sistemih s skromno količino RAM-a je skoraj 1 GB RAM-a stisnjenega, brez izmenjavanja. V nadaljevanju sta predstavila Mac in Linux primera z 2 GB RAM-a in zmerno pomnilniško obremenitvijo. V povprečju bo več podatkov stisnjenih na sistemih z manj RAM-a, vendar sta opazila tudi veliko aktivnost kompresorja na sistemih z 32 GB RAM-a med realističnimi obremenitvami.

V primeru Mac sistema sta uporabila 64-bitni sistem Mac OS X Mavericks z 2 GB RAM-a in 124 tekočimi procesi, relativno visoko pomnilniško obremenitev. V trenutke, ko sta izpisala pomnilnik, je bilo stisnjenih približno 300 MB podatkov. Pri pregledu stisnjenih strani za vsak proces sta ugotovila, da 21 procesov ni imelo stisnjenih strani, 86 procesov je imelo med 1 in 1000 stisnjenih 4 K strani, 15 procesov je imelo med 1001 in 10000 stisnjenih strani in 2 procesa sta imela več kot 20000 stisnjenih strani.

V primeru Linux sistema sta raziskovalca uporabila 64-bitni Ubuntu sistem z 2 GB RAM-a in 115 tekočimi procesi, relativno visoko pomnilniško obremenitev. Pri izpisu pomnilnika sta opazila, da je bilo v kompresorju 160 MB stisnjenih podatkov, 46 procesov ni imelo stisnjenih strani, 66 procesov je imelo med 1 in 1000 4 K stisnjenih strani, ostali procesi pa so imeli med 2000 in 13000 stisnjenih strani.

V zgornjih testih je uporabnik brskal po spletu (n.pr. gle-

dal slike na Flickrju), urejal več tekstovnih datotek (ki niso bile shranjene na disku) in izvajal druge običajne računalniške naloge. Ko sta analizirala stisnjen pomnilnik sta našla konkretno količino dokazov, med drugim fragmente spletnih strani z slikami, ki jih je uporabnik gledal, velike kose tekstovnih datotek, ki jih je urejal, in še več. Ti podatki bi bili nedosegljivi brez novih orodij.

Ena slabost uporabe ogrodje Volatitily je učinkovitost delovanja, saj je Python slaba izbira za računsko intenzivne operacije kot je raztezanje. Na Mac sistemu sta funkciji za stiskanje in raztezanje napisani in optimizirani v 64-bitnem zbirniku, na sistemih Linux pa v C-ju. Raziskovalca sta se sredotočila na Mac OS X implementacijo WKdm kot primerjavo delovanje z našo Python implementacijo. Pri ocenjevanjih sta smatrala stiskanje in nato raztezanje ene 4 K strani kot eno operacijo. Na procesorju i7 so WKdm funkcije v zbirniku zmožne več kot 260000 operacij na sekundo. Na enakem sistemu je optimizirana implementacija v C zmogla več kot 142000 operacij na sekundo. Python implementacijo so zmogle le 390 operacij na sekundo. Kljub temu, da analiza pomnilnika ni omrežna operacija, je za računsko intenzivno forenzično analizo Python slaba izbira navkljub drugim prednostim, ki jo prinaša implementacija ogrodja Volatility v Pythonu.

6. SORODNO DELO

Opravljeno je bilo že veliko kvalitetnih raziskov na področju analize pomnilnika, vse od leta 2005, ko je DFRWS postavil nov izziv v analizi pomnilnikov. Volatility [22] se je pojavil kot vodilno ogrodje za integracijo rezultatov v analizi pomnilnika Mac, Windows, Linux in Android sistemov, čeprav je množica drugih kvalitetnih raziskav pripomogla k sodobnim zmožnostim analize [4, 3, 10, 6, 5, 7, 11, 8, 20, 21, 19, 1].

Kar se tiče analize izmenjevalnih datotek, obstaja le malo uradnega dela, predvsem zaradi težav, podrobneje opisanih v poglavju Izmenjevalne datoteke kot vir dokazov. Kot je tam pojasnjeno, je nekaj opažanj z težavnostjo analize izmenjevalnih datotek povezanih z Kornblum [9] in Walters [17]. Pred kratkim je bil na temo izmenjevalnih datotek opravljena še ena raziskava, ki analizira obnašanje kopice programov z namenom zaznati slabogramje (ang. malware). V raziskavi uporabijo navidezne sisteme za analiza izmenjevalnih datotek na delujočem sistemu [15].

Naprave za stiskanje RAM-a, ki so predstavljene v tem članku, so nove in kot prva raziskovalca sta določala forenzično vrednost teh naprav ter ustvarili orodja za njihovo analizo.

7. ZAKLJUčEK

Če je bilo prej izmenjevalne datoteke prej težko vključiti v forenzično analizo, potem te nove implementacije analize stisnjenega RAM-a v Mac OS X in sistemih Linux omogočajo dostop do podatkov z uporabo standardnih postopkov pri analizi pomnilnika. V članku sta dokumentirani obe implementaciji stiskanja na Linux in Mac OS X Mavericks sistemih in predstavljena so nova odprtokodna orodja za raztezanje in analizo stisnjenih regij pomnilnika.

Orodja, ki so vključena v ogrodje Volatility, nam omogočajo pridobitev vseh stisnjenih strani naenkrat, ali pa po procesih. Ekstrakcija vseh strani naekrat je zelo uporabna pri iskanju osebnih ali finančnih podatkov ali pa iskanju slabogramja z Yara podpisom. Pridobivanje strani po procesih nam zagotovi, da bo pri izpisu pomnilnika za določen proces na voljo največ možnih informacij, kar se tiče tega procesa. Raziskovalca sta pri poskusih opazila, da je v mnogo primerih pregledovanje stisnjenega pomnilnika ključnega pomena, če želimo popolno pregledati določen proces, saj je lahko zgledna količina naslovnega prostora stisnjena. Prikazali sta vrsto uporabnih forenzičnih podatkov, ki sta jih na preiskušanju uspela najti v stisnjenem pomnilniku in tako potrdila, da je analiza stisnjenega pomnilnika potrebna pri digitalni forenziki.

8. PRIHODNJE DELO

Pričakujemo, da bo v prihodnosti stisnjen RAM integriran tudi v druge operacijske sisteme, kot je Microsoft Windows, saj je bil sprejem implementacij na Mac in sistemih Linux pozitiven. Zaprtokodni operacijski sistemi bodo predstavljali nove izzive, saj bo potrebno za raztezanje spomina modelirati velik del navideznega pomnilnika. Podpora za analizo stisnjenih izmenjevalnih datotek na mobilnih napravah je tudi del prihodnjega dela. Na primer, na Androidu 4.4 lahko z nastavitvami omogočimo prednje izmenjevalne datoteke, za analizo stisnjenih izmenjevalnih datotek na napravah z ARM arhitekturo bo potrabne nadgradnja osnovne funkcionalnosti ogrodja Volatility.

Še ena smer prihodnjega dela je ustvarjanje javno dostopnih primerov pomnilnika z omogočenim stiskanjem. Ko sta se raziskovalca lotila tega projekta, ni bilo dostopnih primerov pomnilnika, ki bi jih lahko uporabili. Da bi podprla nove raziskave na tem področju, trenutno delata na novih vzorcih pomnilnika, ki bodo javno dostopni.

9. REFERENCES

[1] S. Andreas. Searching for processes and threads in microsoft windows memory dumps. digit investig 2006;3:10e6.

[2] Apple. Xnu source code http://www.opensource.apple.com/source/xnu/xnu-2422.1.72/; 2013.

[3] D.-G. Brendan. The vad tree: a process-eye view of physical memory. digit investig 2007;4:62e4.

[4] L. L. L. W. P. M. J. X. Carbone Martim, Cui Weidong. Mapping kernel objects to enable systematic integrity checking. in: Proceedings of the 16th acm conference on computer and communications security; 2009.

[5] R. I. G. G. Case A, Marziale L. Dynamic recreation of kernel data structures for live forensics. in: Proceedings of the 10th annual digital forensics research workshop (dfrws 2010); 2010.

[6] M. L. R. I. G. G. R. V. Case Andrew, Cristina Andew. Face: automated digital evidence discovery and correlation. in: Proceedings of the 8th annual digital forensics research workshop (dfrws 2008); 2008.

[7] N. C. R. I. G. G. Case Andrew, Marziale Lodovico. Treasure and tragedy in kmem-cache mining for live forensics investigation. in: Proceedings of the 10th annual digital forensics research workshop (dfrws 2010); 2010.

[8] C. B. D. Risks of live digital forensic analysis. commun acm 2006; 49(2):56e61.

[9] K. J. D. Using every part of the buffalo in windows memory analysis. digit investig march 2007;4(1):24e9.

[10] T. P. G. J. Dolan-Gavitt Brendan, Srivastava Abhinav. Robust signatures for kernel data structures. in proceedings of the 16th acm conference on computer and communications security. acm; 2009. pp. 566e77.

[11] A. Frank. Live forensics: diagnosing your system without killing it first. commun acm 2006;49(2):63e6.

[12] D. Fred. The compression cache: using on-line compression to extend physical memory. in: Proceedings of the usenix winter. conference; 1993. pp. 519e29.

[13] D. P. J. Thrashing: its causes and prevention. in: Proceedings of the 1968 fall joint computer conference, part i. acm; 1968a. pp. 915e22.

[14] D. P. J. The working set model for program behavior. commun acm 1968b;11(5):323e33.

[15] A. I. R. I. G. G. Javaid Salman, Zoranic Aleksander. Atomizer: a fast, scalable and lightweight heap analyzer for virtual machines in a cloud environment. in: Proceedings of the 6th layered assurance workshop (law'12); 2012.

[16] G. S. L. Digital media triage with bulk data analysis and bulk extractor. comput secur 2013;32:56e72.

[17] F. T. A. W. A. Petroni Jr Nick L, Walters Aaron. Fatkit: a framework for the extraction and analysis of digital forensic data from volatile system memory. digit investig 2006;3(4):197e210.

[18] G. G. Richard and A. Case. In lieu of swap: Analyzing compressed ram in mac os x and linux.

[19] M. L. R. I. G. G. Sylve Joe, Case Andrew. Acquisition and analysis of volatile memory from android devices. digit investig 2012;8(3):175e84.

[20] V. Timothy. The acquisition and analysis of random access memory. j digital forensic pract 2007;1(4):315e23.

[21] V. B. A. Van Baar RB, Alink Wouter. Forensic memory analysis: files mapped in memory. digit investig 2008;5:s52e7.

[22] T. volatility framework: Volatile memory artifact extraction utility framework. http://www.volatilesystems.com/default/volatility.

[23] S. Y. Wilson Paul R, Kaplan Scott F. The case for com- pressed caching in virtual memory systems. in: Usenix annual technical conference, general track; 1999. pp. 101e16.

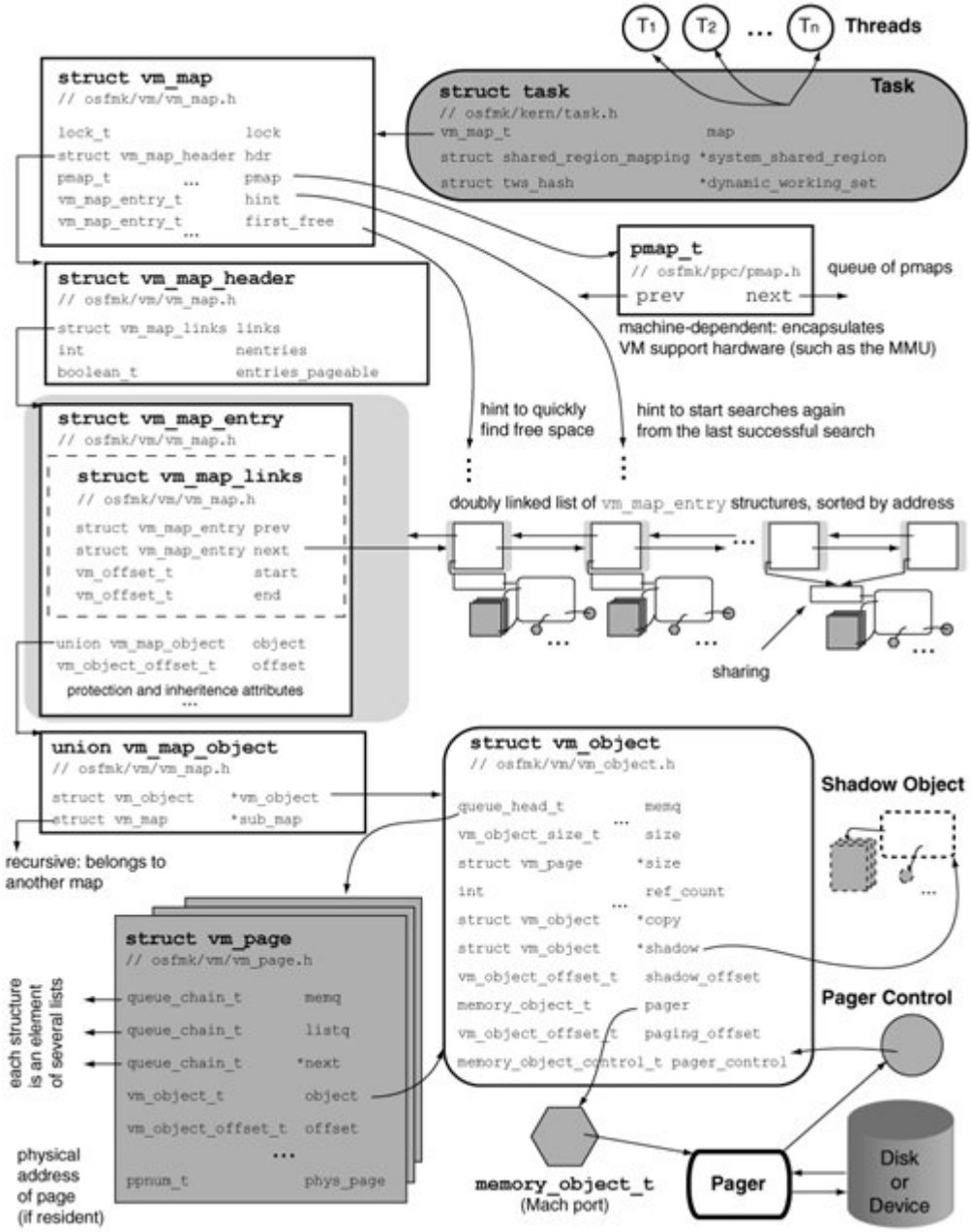


Figure 2: Pregled navideznega pomnilniškega sistema operacijskega sistema Mac OS X, ki se v OS X jedru nahaja znotraj dela Mach. Nov ostranjevalnik (ang.pager), imenovan stiskalni ostranjevalnik (ang. compressor pager), enkapsulira večino stvari za stistkanje in raztezanje strani.

Skrivanje podatkov v šifriranih video vsebinah H.264/AVC s pomočjo zamenjave kodnih besed

Danijel Mišanović
dm6563@student.uni-lj.si

Matevž Kren
matevzkren@gmail.com

POVZETEK

Digitalne video vsebine lahko shranimo v šifrirani obliki. Na ta način zagotovimo večjo varnost in ohranimo zasebnost. V namen označevanja vsebin in zaznave nedovoljenih posegov oziroma podatkovih sprememb, šifriranim video enotam v nadaljevanju dodamo podatke, ki so skriti. S tem ohranimo zaupnost vsebine brez potrebe po dešifraciji. Omenjeni postopek je zato hitrejši in boljši od podobnih tehnik skrivanja podatkov. V besedilu raziščemo tehniko skrivanja podatkov neposredno v šifrirano obliko H.264/AVC video vsebin. Opisana tehnika vključuje podatkovno šifriranje, vstavljanje podatkov in njihovo izvlečenje. Pri šifriranju in skrivanju se osredotočimo na tri lastnosti kodeka H.264/AVC, ki so intra-napovedni načini, razlika vektorjev in koeficiente ostankov, kjer iskoriščamo značilnosti zapisa nihovih kodnih besed. Neodvisno od pomena video vsebine lahko potem vstavimo oziroma skrijemo podatke z uporabo tehnike zamenjave kodnih besed. Izvlečenje skritih podatkov je sicer možno, če so podatki šifrirani ali ne. Količina podatkov, ki jih lahko skrijemo je odvisna od same dolžine in značilnosti (količine gibanja, količine tekstur) video zapisa. Glavna prednost predlaganega postopka je, da ta ne spremenijo velikosti končnih datotek.

1. UVOD IN PREGLED PODROčJA

Računalništvo v oblaku postaja vse pomembnejši tehnološki trend in ponuja visokozmogljivo računanje, procesiranje in shranjevanje video vsebin. Zaradi podvrženosti napadom, zlorabam in nezaupanju sistemskim skrbnikom je zaželjeno, da so video vsebine dostopne v šifrirani obliki [1]. Opisane pomankljivosti in predvsem uhajanje video vsebin lahko naslovimo s skrivanjem podatkov. Strežnik v oblaku lahko dodatne informacije oziroma podatke vstavi v šifrirano obliko video zapisa H.265/AVC z uporabo tehnike zamenjave kodnih besed. Na ta način lahko preverimo pristnost in zagotovimo varnost video podatkov brez nujnega razumevanja pomena vsebin.

Omenjeno tehniko lahko uporabimo tudi v namen lažjega razvrščanja in upravljanja z video zapisi, še posebaj pri primerih, ko mora biti zagotovljena zasebnost posnetih ljudi kot na primer v domeni biomedicinksih podatkov ali varnostnih posnetkov, kjer v video vsebino skrijemo podatke namenjene za temu.

Obstajajo številne tehnike skrivanja podatkov, ki pa so bila predvsem usmerjena v skrivanje podatkov znotraj slikovnih gradiv od tega jih je le nekaj, ki delujejo na šifriranih podatkih [] []. V zadnjem času se je pojavila vse večja potreba po skrivanju podatkov tudi znotraj video vsebin. Algoritmov, ki bi skrivanje izvajalo na šifriranih podatkih ni. Še najbolj podobna sta mu je algoritma [] [], ki pa skrivanje in šifriranje izvede hkrati v fazi kompresije videa.

Obstaja torej zahteva po algoritmih za skrivanje podatkov, ki v celoti delujejo v povezavi s šifriranimi podatki. Opazimo številne izzive. Največjo oviro predstavlja izračun primernega dela video zapisa in načina bitnega zapisa za skrivanje podatkov. V nadaljevanju opazimo tudi zahtevo po ohranitvi kvalitete oziroma točnosti izgleda video vsebine. Poskrbeti moramo tudi za primerno velikost končnih datotek in možnost izvlečenja skritih podatkov ne glede na to, ali so podatki šifrirani ali ne. Glede na opisana pričakovanja in potrebe so avtorji preučenega besedila predlagali tehniko skrivanja podatkov neposredno v stisnjeno in šifrirano obliko bitnega zapisa.

Najprej z preiskovanja lastnosti kodeka H.264/AVC prevedemo kodne besede intra-napovednih načinov (*IPMs* - intra-prediction mode), razlik vektorjev premikanja (*MVDs* - motion vector difference) in koeficientov ostankov, kjer izkoristimo značilnosti kodiranja *Exp-Golomb* entropije in *CAVLC* (Context-adaptive variable-length coding) [3], ki dolžino kodne besede ohranja nespremenjeno. Skrivanje podatkov je nato izvedeno s pomočjo zamenjave kodnih besed. To nam omogoče odlične rezultate v sledečih primerih:

- Skrivanje podatkov je izvedeno neposredno v šifriranih video zapisih.
- Postopek zagotavlja ustreznost in pravilnost formata zapisa in ohranja nespremenjeno velikost datotek.
- Podatke je moč izvleči, ko so le-ti šifrirani ali ne.

2. OPIS ALGORITMA IN UPORABLJENIH TEHNIK

Algoritem vključuje šifriranje, vstavljanje in izvlečenje podatkov. Lastnik video vsebine datoteko najprej šifrira. Entiteta za skrivanje podatkov (npr. računalniški sistem v oblaku) nato, sicer brez vedenja oziroma razumevanja vsebine in pomena, vstavi dodatne informacije in podatke s pomočjo zamenjave kodnih besed. Izvlečenje skritih podatkov lahko izvedemo na podatkih, ki so lahko šifrirani ali ne. Postopek šifriranja in skrivanja je predstavljen na sliki 1 a) medtem ko je postopek izvlečenja tako iz šifriranih kot nešifriranih podatkov prikazan na sliki 1 b).

A) H.264/AVC šifriranje podatkov
Hitrost je velikokrat glavna zahteva šifriranja podatkov. Iz tega naslova sledi, da šifriranje celotnih datotek največkrat ni časovno smiselno, saj delo porabi preveč računskega časa. Rešitev najdemo v šifriranju manjšega dela datoteke, ki pa še vedno zagotavlja željeno varnost. Pri tem je pomembna izbira dela podatkov, ki ga šifriramo. Tekom poglavja predstavimo postopek šifriranja, ki zagotovi časovno sprejemljivo zmogljivost, varnost in ustrezen zapis.

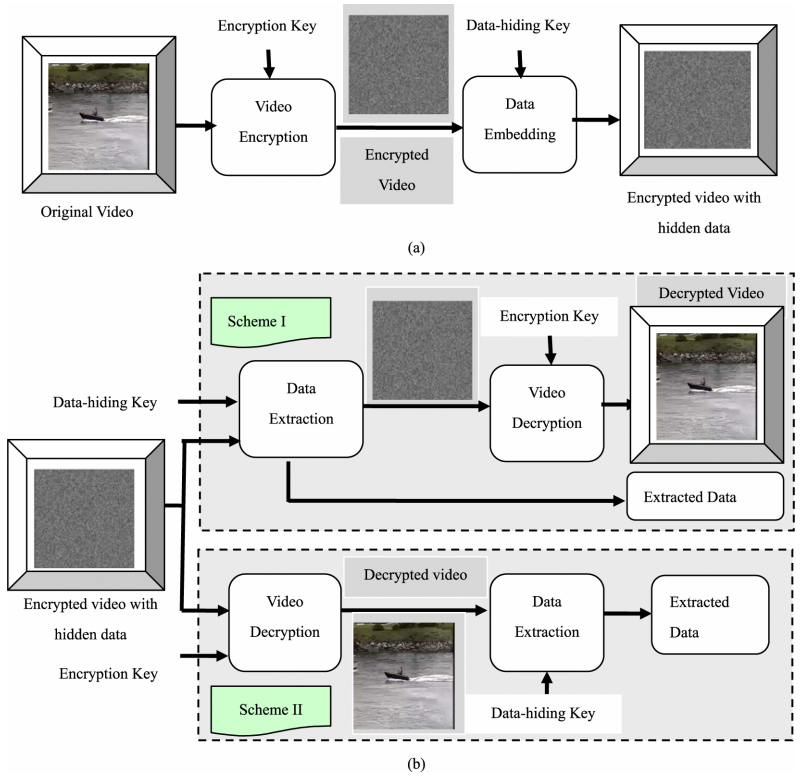
Kot že zapisan, se pri šifriranju osredotočimo na tri lastnosti kodeka H.264/AVC intra-napovedne načine (*IPMs*), razlik vektorjev premikanja (*MVDs*) in koeficiente ostankov. Iz stališča razumevanja delovanja algoritma ni pomembno kaj točno te lastnosti pomenijo oziroma si lahko več o njih preberete v sami definiciji standarda []. Na grobo si jih lahko predstavljamo kot neki parametri, ki nam opisujejo samo sliko. Ker govorimo o kompresijskem standardu so te lastnosti tiste iz katerih nato generiramo sliko oziroma okvirje do piksla natančno, kjer se za generiranje teh naslanjamo na prejšnje in prihodnje okvirje.

(i) Šifriranje kodnih besed intra-napovednih načinov (*IPMs*)

Glede na standard H.264/AVC, poznamo štiri podprte načine kodiranja - Intra_4 x 4 , Intra_16 x 16, Intra_chroma in I_PCM. Standard Intra_16 x 16 npr. je predstavljen s štirimi napovednimi načini (IPMs), ki so posredno opisani preko tipov mp (*macroblock types*) v tabeli na sliki 2, ki je povzeta po standardu. V splošnem, šifriranje kodnih besed napovednih načinov pomeni spremembo napovednega načina brez kršitev semantičnih pravil oziroma pravilnosti bitnega zapisa. Torej v tem primeru to dosežemo tako, da najprej generiramo neko psevdo-naključno zaporedje, na podlagi nekega enkripcijskega ključa, ki jo XOR-amo z zadnjimi biti kodnih besed. Omejimo se samo na zadnje bite, saj le tako zagotovimo, da ne spreminjamo CBP-jev (Code Block Pattern - Vzorec kodnih blokov)

(ii) Šifriranje razlik vektorjev premikanja (*MVDs*)

Ker želimo zaščititi tako podatke tekstur kot tudi podatke premikov, moramo šifrirati tudi vektorje premikanja. Po standardu H.264/AVC razlike vektorjev premikanja dobimo s predvidevanjem vektorjev premikanja. Uporabljeno je *Exp-Golomb* kodrianje entropije. Kodna beseda sestoji iz [M ničel] [1] [INFO], pri čemer INFO predstavlja polje dolžine M. V tabeli na sliki 3 so predstavljene vrednosti razlik vektorjev premikanja in pripadajočih Exp-Golomb kodnih besed. Opazimo da podobno kot pri šifriranju kodnih besed intra-napovednih načinov lahko z XOR-anjem psevdonaključnega zaporedja z zadnjim bitom kodne besede temu spremenimo pred znak ter s tem še dodatno spremenimo oziroma zabrišemo orginalno video vsebino. MVD 0 spustimo, pri enkripcij, da zadostimo standardu.



Slika 1: Opis postopka šifriranja in skrivanja podatkov.(a) Video šifriranje in skrivanje podatkov na strani pošiljatelja (b) Izvlečenje podatkov in prikaz video vsebine iz strani prejemnika, v dveh različnih scenarijih.

mb_type	Name of mb_type	Intra16x16 PredMode	Chroma CBP	Luma CBP	Codeword
1	I_16x16_0_0_0	0	0	0	010
2	I_16x16_1_0_0	1	0	0	011
3	I_16x16_2_0_0	2	0	0	00100
4	I_16x16_3_0_0	3	0	0	00101
5	I_16x16_0_1_0	0	1	0	00110
6	I_16x16_1_1_0	1	1	0	00111
7	I_16x16_2_1_0	2	1	0	0001000
8	I_16x16_3_1_0	3	1	0	0001001
9	I_16x16_0_2_0	0	2	0	0001010
10	I_16x16_1_2_0	1	2	0	0001011
11	I_16x16_2_2_0	2	2	0	0001100
12	I_16x16_3_2_0	3	2	0	0001101
13	I_16x16_0_0_1	0	0	15	0001110
14	I_16x16_1_0_1	1	0	15	0001111
15	I_16x16_2_0_1	2	0	15	000010000
16	I_16x16_3_0_1	3	0	15	000010001
17	I_16x16_0_1_1	0	1	15	000010010
18	I_16x16_1_1_1	1	1	15	000010011
19	I_16x16_2_1_1	2	1	15	000010100
20	I_16x16_3_1_1	3	1	15	000010101
21	I_16x16_0_2_1	0	2	15	000010110
22	I_16x16_1_2_1	1	2	15	000010111
23	I_16x16_2_2_1	2	2	15	000011000
24	I_16x16_3_2_1	3	2	15	000011001

Slika 2: Tipi "macroblock".

MVD	*code_num*	*codeword*
0	0	1
1	1	010
-1	2	011
2	3	00100
-2	4	00101
3	5	00110
-3	6	00111
4	7	0001000
-4	8	0001001
5	9	0001010
-5	10	0001011
6	11	0001100
-6	12	0001101
7	13	0001110
-7	14	0001111
8	15	000010000
-8	16	000010001
9	17	000010010
-9	18	000010011
…	…	…

Slika 3: MVDs in pripadajoče kodne besede

(iii) Šifriranje koeficientov ostankov

Da zadostimo potrebam po visokih varnostnih standardih, šifriramo koeficiente ostankov v I-okvirjih in P-okvirjih. I-okvirji so najmanj stisnjeni okvirji in se uporabljajo kot referečni okvirji za P-okvirje. P-okvirji vsebujejo samo spremebe iz prejšnjih okvirjov, zato za generiranje dejanskega okvirja sklicujejo na prejšnje okvirje (torej I okvirje in že dekodirane P-okvirje). Koeficienti ostanko so opisanimi s sledečimi lastnostmi [Coeff token, Sign of TrailingOnes, Level, Total zeros, Run before], kjer se sama enkripcija izvede le na nivojih (Level) in predznaku (Sign of TrailingOnes). Predznak je zapisan samo z enim bitom, tako da je enkripcija enostavna, tako da XOR-amo vsak bit posebaj z nekim psevdo-naključnim zaporedjem, ki ga generiramo z nekim enkripcijskim ključem. Podobno velja tudi za kodne besede nivojev, ki jih lahko vidmo na sliki 4, le da se tu omejimo na zadnje bite. Na primer, ko je dolžina korena = 1, so kodne besede, ki pripadajo ravnem "2" in "-2", "010" in "011". Imata torej isto dolžino, kar pomeni da smo zadostili standardu. Pri enkripciji spustimo kodne besede nivojev s korenom dolžine (suffix length) nič, saj v tem primeru ne zadostimo standaru, v primeru da bi XOR izvedli na zadnjem bitu.

B) Vstavljanje podatkov

V šifrirani predstavitvi podatkov vstavljanje podatkov dosežemo z zamenjavo primernih kodnih besed glede na definirane kodna polna nivojev na sliki 5. Skrivanje se izvaja samo na kodnih besedah nivojev in to kodnih besed P-okvirjev. I-okvirji niso primerni, saj se iz njih generirajo P-okvirji, s čemer bi propagirali napako naprej na vse P-okvirje in s tem precej vplivali na kakovost slike. Zamenjava kodnih besed mora zadostiti sledečim pogojem:

(i) Bitna predstavitev podatkov po zamenjavi kodnih besed mora ohraniti sintaksno pravilnost, kar omogoča dešifriranje v nadaljevanju s standardnim dekodirnikom.

(ii) Ker želimo ohraniti število bitov, s katerimi vzorčimo, morajo biti zamenjane kodne besede istih dolžin kot izvirne kodne besede.

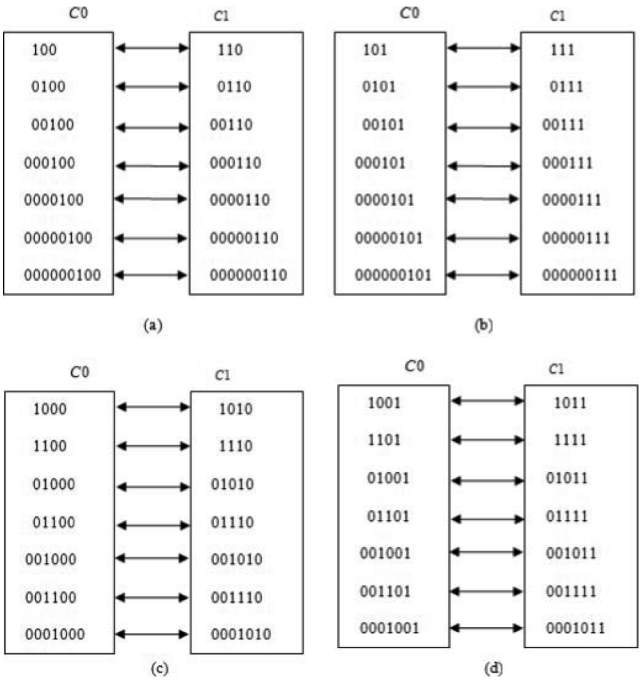
(iii) Skrivanje podatkov vpliva na kvaliteto videa. Stremimo k čim manjšim spremembam v kvaliteti. Da zadostimo prvi predpostavki se uporabijo samo kodne besede nivojev s korenom večjim od 1.

Predpostavimo, da želimo vstaviti binarno predstavljene podatke $B = \{b(i)|i = 1, 2, ..., L, b(i)\epsilon\{0, 1\}\}$.

- Z uporabo algoritma, ki generira psevdo naključno zaporedje števil, močno izboljšamo končno varnost in učinkovitost postopka. Psevdonaključno zaporedje P je generirano z uporabo logistične funkcije map [], kateri določimo neko začetno vrednost oziroma ključ. Če ključa ne poznamo, je skrite podatke veliko težje odkriti.
- Kodne besede različnih nivojev pridobimo z obdelavo šifriranega bitnega zapisa H.264/AVC.

suffixLength	*Level(>0)*	*Codeword*	*Level(<0)*	*Codeword*
0	1	1	-1	01
	2	001	-2	0001
	3	00001	-3	000001
	4	0000001	-4	00000001
1	1	10	-1	11
	2	010	-2	011
	3	0010	-3	0011
	4	00010	-4	00011
	5	000010	-5	000011
	6	0000010	-6	0000011
	7	00000010	-7	00000011
	8	000000010	-8	000000011
2	1	100	-1	101
	2	110	-2	111
	3	0100	-3	0101
	4	0110	-4	0111
	5	00100	-5	00101
	6	00110	-6	00111
	7	000100	-7	000101
	8	000110	-8	000111
	9	0000100	-9	0000101
	10	0000110	-10	0000111
	11	00000100	-11	00000101
	12	00000110	-12	00000111
	13	000000100	-13	000000101
	14	000000110	-14	000000111
3	1	1000	-1	1001
	2	1010	-2	1011
	3	1100	-3	1101
	4	1110	-4	1111
	5	01000	-5	01001
	6	01010	-6	01011
	7	01100	-7	01101
	8	01110	-8	01111
	9	001000	-9	001001
	10	001010	-10	001011
	11	001100	-11	001101
	12	001110	-12	001111
	13	0001000	-13	0001001
	14	0001010	-14	0001011

Slika 4: Nivoji in pripradajoče kodne besede.



Slika 5: CAVLC preslikave kodnih besed na podlagi kodnih polj C0 in C1 (a) suffix Length = 2 & Level > 0. (b) suffix Length = 2 & Level > 0. (c) suffix Length = 3 & Level > 0. (d) suffix Length = 3 & Level < 0.

- Recimo, če je raven = +1 in dolžina korena = 3, potem je pripadajoča kodna beseda "1000". Torej glede na to v katero kodno polje pripada glede na sliko 5 (C0 ali C1) zamenjamo kodno besedo. Torej, če želimo skirti 0 ostane kodna beseda nespremenjena, saj pripada kodnemu polju C0 medtem ko v primeru da želimo skirti 1 zamenjamo kodno besedo glede na definicijo na sliki 5 v kodno besedo "1010" (kodno polje C1).
- Izberemo si naslednjo kodno besedo in pri ponovitvi prejšnjega koraka razberemo, če ostajajo podatkovni biti, ki še niso bili vstavljeni. V nasprotnem primeru je postopek vstavljanja oziroma skrivanja končan.

Če želimo skriti podatkovni niz "1001" in so CAVLC kodne besede nivojev "01 010 00100 00100 0001011 0000100", šifrirno zaporeje pa je "10111", potem se enkripcija in skrivanje izvede kot je prikazano na sliki 6 a). Opazimo, da zaradi pametno definiranih kodnih polj C0 in C1 spremenimo vrednost zgolj za ena, s čemer zagotovimo, da je vpliv na kvaliteto skrivanja podatkov znotraj video vsebin neopazen s prostim očesom.

C) Izvlečenje podatkov

Podatke lahko izvlečemo, če so šifrirani ali ne. Postopek izvčelenja je predstavljen na sliki 6 a) iz šifriranih video vsebin in b) iz nešifriranih vsebin.

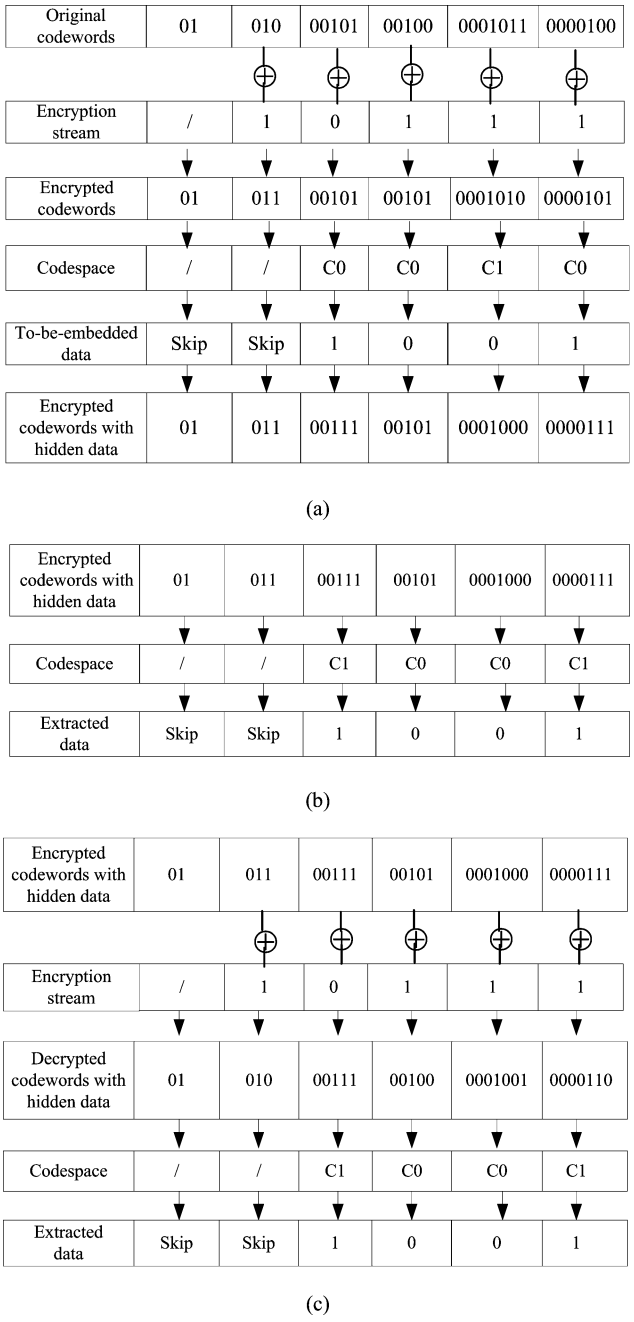
(i) Izvlečenje podatkov v šifrirani obliki

V namen zaščite podatkov, ima skrivni ključ le upravitelj oziroma entita skivanja podatkov (računalnik v oblaku).

- Kodne besede nivojev so sprva prepoznane preko obdelave šifriranega bitnega zaporedja podatkov.
- Če kodna beseda pripada kodnemu območju C0, je izvlečen bit "0", sicer pa "1".
- Glede na skrivni ključ lahko poustvarimo isto psevdo naključno zaporedje števil P, ki smo ga uporabili tekom vstavljanja podatkov. To nam omogoči dešifriranje iskanih podatkov. Ker je celoten postopek v celoti izveden na strani strežnika in v šifrirani obliki, nam to zagotovi, da podatki nikoli ne uidejo.

(ii) Izvlečenje podatkov, če ti niso šifrirani

- Generiramo šifrirne tokove s ključi, ki so bili uporabljeni v postopku šifriranja.
- Prepoznamo kodne besede IPMs, MVDs in koeficientov ostankov tekom obdelave šifriranega bitnega podatkovnega zaporedja.
- Postopek dešifriranja je enak postopku šifriranja, saj je operacija XOR simetrična. Kodne besede lahko ugotovimo z izvajanjem operacije XOR nad generiranimi šifrirnimi podatkovnimi tokovi, kar nam razkrije izvirno besedilo oziroma podatke, ki pa so še vedno šifrirani.
- Z uporabo istega psevdo naključnega zaporedja P lahko podatkovne bite dešifriramo.



Slika 6: Primer vstavljanja/skrivanja podatkov in izvlečenja a) Vstavljanje podatkov b) Izvlečenje podatkov iz šifriranih video vsebin c) Izvlečenje podatkov iz dešifriranih video vsebin.

3. PRIDOBLJENI REZULTATI

Tehnika skrivanja podatkov je bila preizkušena preko uporabe 6 poznanih standardnih video zapisov (Stefan, Table, Tempete, Mobile, Hall, News) v formatu QCIF in pri vzorčenju 30 sličic na sekundo. Med testiranjem so avtorji besedila uporabili prvih 100 sličic vsakega izmed video zapisov. Uporabljeni algoritmi in tehnike skrivanja podatkov vključujejo postopke, ki zagotavljajo tako kriptografsko varnost kot nedvoumno interpretacijo rezultatov. Za šifriranje podatkov in podatkovnih tokov uporabimo poljubno tehniko za generiranje naključnih zaporedji (npr. RC4) in kaotično psevdo-naključno zaporedje generirano z logistično funkcijo map za skrivanje. Ze obe tehniki je bilo večkrat pokazano, da sta odporni na kriptografske napade.

3.1 Kvaliteta video zapisa

Glede na rezultate v tabeli 7 je končna kvaliteta video zapisa skoraj enaka oziroma zelo podobna kvaliteti izvirnega video zapisa. S tem je dosežen en izmed najpomembnejših in hkrati najtežje dosegljivih ciljev skrivanja podatkov. Ugotovimo, da lahko vstavimo sorazmerno veliko količino podatkov znotraj P-okvirjev, pri čemer še vedno ohranimo visoko kakovost video zapisa.

Poleg subjektivnih ocen, lahko ohranitev kvalitete in količino izgube informacije predstavimo z izračuni količin:

- PSNR (*Peak Signal to Noise Ratio*)
- SSIM (*Structural Similarity Index*)
- VQM (*Video Quality Measurement*)

Sequence	*QP*	*Maximum Capacity* (kbits/s)	*PSNR* (dB)		*SSIM*		*VQM*	
			Non-stego	*Stego*	*Non-stego*	*Stego*	*Non-stego*	*Stego*
Stefan	24	17.80	39.60	38.33	0.9833	0.9825	0.7385	0.7855
	28	3.63	35.84	35.50	0.9692	0.9688	1.0334	1.0586
	32	0.57	31.68	31.62	0.9447	0.9446	1.3834	1.3967
Table	24	8.27	38.44	37.91	0.9542	0.9537	0.6896	0.7193
	28	3.45	35.15	34.87	0.9091	0.9087	0.9246	0.9428
	32	0.82	32.31	32.17	0.8451	0.8449	1.1829	1.1970
Tempete	24	7.89	38.68	38.16	0.9855	0.9850	0.8586	0.8913
	28	1.13	34.83	34.74	0.9716	0.9714	1.2056	1.2210
	32	0.14	30.88	30.87	0.9448	0.9448	1.6372	1.6399
Mobile	24	1.81	38.44	38.32	0.9871	0.9871	0.8652	0.8724
	28	0.22	34.51	34.49	0.9741	0.9741	1.2552	1.2575
	32	0.01	30.63	30.63	0.9501	0.9501	1.7154	1.7154
Hall	24	0.60	40.33	40.26	0.9744	0.9743	0.6536	0.6573
	28	0.13	37.92	37.90	0.9658	0.9658	0.7848	0.7867
	32	0.02	34.98	34.98	0.9527	0.9527	0.9542	0.9552
News	24	0.50	40.82	40.75	0.9848	0.9848	0.5847	0.5872
	28	0.11	37.78	37.76	0.9727	0.9727	0.7745	0.7751
	32	0.02	34.57	34.56	0.9531	0.9531	1.0064	1.0065

Slika 7: Tabela z rezultati.

PSNR je najpogosteje uporabljena količina za ocenitev kvalitete video zapisa, čeprav ni popolnoma usklajena z oceno, ki bi jo navadno podal uporabnik. SSIM poda oceno na intervalu [0,1], pri čemer 1 pomeni popolno ujemanje signalov predstavljenih slik. VQM je bližje oceni, ki bi jo subjektivno podal uporabnik. Nižje vrednosti v splošnem pomenijo boljšo kakovost, pri čemer 0 izraža popolno ujemanje.

3.2 Količina podatkov, ki jih lahko skrijemo

Količina podatkov, ki jih lahko skrijemo, se ocenjuje v [kbits/s] [2]. Največje kapacitete video zapisov, ki so šifrirani z uporabo različnih QP vrednosti (določa bitrate - večji je QP manj kvalificiranih kodnih besed imamo), so prestavljeni v tabeli na sliki 7. Sama velikost podatkov, ki jih lahko skrijemo je odvisna od števila kvalificiranih kodnih besed, ki jih lahko zamenjamo. Opazimo torej, da v primeru da imamo video vsebina z veliko gibanja (Stefan, Table) ali z veliko tekstur (Tampete, Mobile) imamo posledično več kodnih besed, ki jih lahko izkoristimo za skrivanje.

V primeru dokaj statičnih video vsebin (Hall, News), je možno skriti manj podatkov. Da bi povečali kapaciteto podatkov, ki jih lahko skrijemo, se lahko poslužimo hibridnega načina zamenjave kodnih besed, kjer za zamenjavo uporabimo tudi kodne besede razlik vektorjev premikanja. Ta način je primeren samo za dokaj statične video vsebine, saj je njegov vpliv na kakovost pri ostalih tipih prevelik.

4. ZAKLJUčEK

Skrivanje podatkov v šifriranih video vsebinah H.264/AVC je pretežno novo področje digitalnega procesiranja signalov oziroma forenzičnih tehnik, ki zaradi vsestranske uporabnosti in nudenja varnosti zanima predvsem ponudnike računalniških omrežij v oblaku. V besedilu smo glede na pridobljena besedila predstavili algoritem za skrivanje podatkov, ki vključuje šifriranje, vključevanje in izvlečenje podatkov, in je preprost za implementacijo. Entiteta oziroma računalnik, ki podatke skriva, to stori brez poznavanja oziroma možnosti interpretacije vsebine in s pomočjo zamenjave kodnih besed. Omenjen postopek podatke oziroma informacije skrije popolnoma. Skrbi, da bi se kdorkoli do podatkov dokopal, ni. Rezultati eksperimentov kažejo, da skrivanje podatkov načeloma ohrani velikosti datotek, kvalitete vsebovanih video zapisov pa skoraj ne spremeni, tj. spremembe niso zaznavne.

Literatura

[1] W. Lu, A. Varna, and M. Wu. Secure video processing: Problems and challenges. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 5856–5859, May 2011.

[2] T. Shanableh. Data hiding in mpeg video files using multivariate regression and flexible macroblock ordering. *Information Forensics and Security, IEEE Transactions on*, 7(2):455–464, April 2012.

[3] T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, and G. Sullivan. Rate-constrained coder control and comparison of video coding standards. *Circuits and Systems for Video Technology, IEEE Transactions on*, 13(7):688–703, July 2003.

Samodejno prepoznavanje kopiranih kriminalnih spletnih strani z metodo gručenja

Rok Kralj
Fakulteta za računalništvo in informatiko
1000 Ljubljana
Slovenia
research@rok-kralj.net

Nejc Novak
Fakulteta za računalništvo in informatiko
1000 Ljubljana
Slovenia
nejc.novak92@gmail.com

Luka Krivec
Fakulteta za računalništvo in informatiko
1000 Ljubljana
Slovenia
luka@krivec.net

POVZETEK

V interesu spletnih kriminalcev je, da pride v stik z njihovo prevaro kar se da veliko število ljudi. S tem v mislih si želijo čim ceneje izdelati veliko skupino spletnih strani, ki so vsaj na videz čim bolj različne. Ker pa je izdelava spletne strani vsakič na novo enostavno predraga, za izdelavo uporabljajo že narejene predloge ali pa delčke preteklih spletnih strani, kar vključuje HTML, CSS in JavaScript kodo, še bolj pogosto pa njihovo zvito sestavljeno in kar se da prepričljivo besedilo.

Uporabili bomo metodo nenadzorovanega strojnega učenja za združevanje podobnih spletnih strani v gruče. Na ta način lahko ugotovimo, katere izvirajo iz iste kriminalne združbe. Kakovost rezultatov bomo tudi ovrednotili, tako da bomo rezultate primerjali z mnenjem človeškega poznavalca te tematike.

1. UVOD

Spletna enciklopedija Wikipedia navaja več kot 100 različnih vrst spletnega kriminala. Osnova delovanja večine takih zlorab pa je predvsem elektronska pošta in pa spletne strani. Za primer si oglejmo primer spletne lekarne, ki ponujajo cenejša zdravila ali pa zdravila, ki jih je sicer mogoče dobiti le na recept. Ko stranka na tako spletno stran vpiše podatke o svoji kreditni kartici, pogosto izgubi celotno stanje na računu, težko pričakovane pošiljke pa jasno, ne dobi.

Večina takih spletnih strani ima zelo nizek odstotek uspešnosti. Članek *Spamalytics: An Empirical Analysis of Spam Marketing Conversion* [8] nam postreže z bolj konkretnimi številkami. Pri pošiljanju neželene elektronske pošte, ki oglašuje spletno lekarno s ponarejenimi zdravili, samo 23,8% sporočil prispe v prejemnikov predal, preostanek pa izločijo filtri za neželeno elektronsko pošto. Spletno lekarno obiščejo samo tri tisočinke odstotka prejemnikov, dejanski nakup pa izvede samo vsak stomilijonti prejemnik. Vendar se tovrstne prevare izvajalcem vseeno izplačajo, saj se izvajajo v res velikem obsegu.

Cilj spletnih prevarantov je torej čim ceneje razširiti svojo prevaro na čim več kontinentov, saj tako pride z njo v stik veliko potencialnih žrtev. Pomembno je, da ustvarijo čim več različnih spletnih portalov, ki so si med seboj kar se da malo podobni, saj bi sicer tvegali, da jih odkrijejo organi pregona, ali pa bi sumničave postale kar žrtve same.

Ker bi bilo grajenje novega spletnega portala vsakič na novo enostavno predrago, spletni kriminalci običajno ohranijo vsaj nek del že obstoječe spletne strani, ki so jo že uporabljali. To je lahko HTML struktura spletne strani ali pa njena oblikovna (CSS) datoteka, še bolj pogosto pa je to kar besedilo. Spletni kriminalci običajno prihajajo iz držav tretjega sveta, za uspešno prevaro pa je nujno, da je uporabljena angleščina v skladu s strogimi pravopisnimi in slovničnimi pravili. Pisanje vsebine vsakič na novo bi bilo tako nesmotrno.

Na sliki 1 je primer štirih spletnih storitev, za katere bi človeški ocenjevalec rekel, da pripadajo isti kriminalni združbi. Čeprav so oblikovno spletne strani zelo različne, pa je videti, da imajo enako poimenovane datoteke, še več, celo besedilo, ki predstavlja storitev ali izdelek, je popolnoma enako.

Cilj naše metode je iz velikega nabora tovrstnih spletnih strani ugotoviti, katere prihajajo iz istega vira, želimo pa si tudi ugotoviti, kateri kriminalni združbi pripada spletna stran, ki se bo pojavila v prihodnosti.

Razvili bomo metodo strojnega učenja, s katero bomo določene spletne strani združili v isto skupino.

2. DVE VRSTI KRIMINALNIH SPLETNIH STRANI

V delu smo raziskovali dve vrsti kriminalnih spletnih strani, spletne Ponzi sheme z visokim donosom ter lažne spletne posredovalnice. Za oba tipa spletnih strani pa imamo na voljo tudi spletne zbirke, ki nam omogočajo, da na lahek način pridemo do velike količine tovrstnih spletnih storitev.

2.1 Spletne Ponzi sheme (HYIP)

Visoko-donosni naložbeni sklad (HYIP) je vrsta Ponzi sheme, ki vlagateljem obljublja nerealistično visoke donose, tudi do 3 odstotke - na dan! Uporaba obrestno-obrestnega računa nam pokaže, da samo enoodstotna obrestna mera, ki se

Figure 1: Primer podobnosti v spletni strukturi in besedilu, ki namiguje, da so vse štiri spletne strani pod nadzorom iste kriminalne združbe.

nabere na koncu vsakega dne, prinese 3778-odstotno letno obrestno mero. Jasno je, da tak sklad ni trajnosten in se slej ko prej podre, saj pologi novih strank ne morejo dolgo vzdrževati obresti, ki se podeljujejo že obstoječim.

S tem namenom obstajajo spletne strani, ki vlagatelje opozorijo tik preden se taka shema poruši, saj lahko tako pravočasno dvignejo svoj denar. Te sledilne spletne strani nam služijo kot zbirka, preko katere lahko zajamemo HTML dokumente skupaj s pripadajočimi datotekami za nadaljnjo analizo.

Kriminalci, ki vodijo take sheme, so zelo vešči *socialnega inženiringa*, torej uporabe socialnih omrežij za pridobitev novih strank, zato je res pomembno, da imamo orodja, s katerimi se lahko proti njim borimo.

2.2 Lažne posredovalnice (fake-escrow)

Mnogo spletnih prodaj, ki poteka med dvema fizičnima osebama, je v svoji osnovi transakcija med dvema entitetama, ki si ne zaupata. Če kupec prvi pošlje svoj denar, obstaja nevarnost, da mu prodajalec izdelka ne bo poslal. Če pa je prodajalec tisti, ki prvi pošlje izdelek, je mogoče, da ga kupec ne bo plačal.

S tem namenom na spletu obstajajo podjetja, ki služijo kot posrednik pri transakciji (*escrow*) in zadržijo denar, dokler prejemnik izdelka ne zatrdi, da je izdelek res prejel. Ko se to zgodi, nakažejo denar prodajalcu.

Ker trenutno za izvajanje spletnega posredovanja ne potrebujemo nobene licence, ne moremo biti prepričani, da je neka spletna posredovalnica zaupanja vredna. Spletni kriminalec, ki ima tako spletno posredovalnico se tako pretvarja, da prodaja neko dobrino, na primer računalnik. Če kupec zahteva posredovanje, mu jasno predlaga, da uporabita kar njegovo storitev. Ko kupec nakaže sredstva, ostane brez računalnika in brez denarja.

V boju proti takim prevaram so se organizirale skupine prostovoljcev, ki gradijo podatkovne zbirke takih lažnih spletnih posredovalnic. Te nam bodo služile kot agregator takih spletnih strani za zajem in kasnejšo analizo.

3. POSTOPEK ODKRIVANJA SKUPIN POVEZANI SPLETNIH STRANI

Iz prej omenjenih zbirk kriminalnih spletnih strani smo dobili veliko količino spletnih URL naslovov. Naša naloga je bila, da razvijemo metodo, s katero bomo ugotovili, kateri pripadajo isti kriminalni združbi.

3.1 Prvi korak: Zajem spletne vsebine

Vsak spletni naslov, ki se pojavi v eni izmed zbirk je potrebno zajeti čim prej, saj sicer tvegamo, da bomo podatke izgubili, če bodo spletno stran umaknili organi pregona ali pa kar avtorji sami.

Pri zajemu je pomembno, da znamo zajeti celotno imeniško

strukturo z vsemi pripadajočimi datotekami. S tem namenom smo uporabili orodje `wget` [1].

3.2 Drugi korak: Analiza in izbor atributov

Ko imamo datoteke shranjene na disku, je naša naloga, da iz njih izbrskamo čim več pomenljivih atributov.

Prvo skupino atributov lahko dobimo iz same datotečne strukture, torej imen HTML, CSS in JavaScript datotek.

Drugo skupino atributov lahko dobimo iz HTML strukture spletne strani. To smo naredili kar se da enostavno, tako da smo za vsako značko prešteli, kolikokrat se ponovi. Atributi so tako ključi oblike `<p>8`, kar bi pomenilo, da dokument vsebuje osem odstavkov.

Tretjo skupino atributov pa nam da tekstovna vsebina spletne strani. V ta namen smo uporabili HTML izrisovalnik, ki vsebine ne prikaže na zaslon (*headless browser*), temveč samo izračuna, kateri deli besedila se dejansko prikažejo na zaslonu. Te dele besedila lahko slovnično razčlenimo z NLP razčlenjevalnikom. To je pomembno, saj nam iste besedne oblike preslika v isti ključ (**primer:** dela, delava, delajo, delaš, delajte $\Longrightarrow$ glagol delati).

3.3 Tretji korak: Izračun matrike podobnosti

V tem koraku analiziramo vse mogoče pare spletnih strani in primerjamo vse zbrane atribute, kot je to opisano v prejšnjem pododseku. Za cenilko uporabimo kar Jacartovo razdaljo, ki je definirana takole:

Če je A množica atributov, ki se pojavijo v prvi strani v paru in množica B množica atributov, ki se pojavijo v drugi strani v paru, je njuna podobnost enaka:

$$1 - J(A, B) = 1 - \frac{|A \cap B|}{|A \cup B|},$$

kjer $|\bullet|$ označuje moč množice.

Denimo, da ima stran A 100 različnih atributov, stran B pa 80, od tega 50 skupnih s stranjo A. Potem je njuna podobnost enaka $1 - J(A, B) = 1 - \frac{50}{130} = \frac{8}{13} \approx 61,5384615\%$

3.4 Četrti korak: Hierarhično gručenje

Hierarhično gručenje je metoda nenadzorovanega strojnega učenja. Nenadzorovano strojno učenje sloni na predpostavki, da nimamo vnaprej podane zbirke učnih primerov, ki bi nam povedala, katere spletne strani spadajo v isto skupino.

Za razliko od navadnega gručenja nam hierarhično gručenje omogoči, da poljubno spreminjamo število kriminalnih združb. Rezultat hierarhičnega gručenja je drevo sosednosti, ali tudi *dendrogram*.

Gručenje lahko izvedemo na podlagi matrike razdalj. Ker pa ni vedno mogoče gručenja narediti tako, da popolnoma ustreza tej matriki, smo se odločili, da za razdaljo med dvema skupinama uporabimo *min-linkage*, kar je v tem primeru zelo pomenljivo. Čeprav imata dve spletni strani povsem različno datotečno in HTML strukturo, imata pa enako vsebino, ju obravnavamo, kot da sta zelo podobni.

4. PREUČEVANJE GRUČENIH KRIMINALNIH SPLETNIH STRANI

Na vsakem od obeh naborov podatkov je bila uporabljena najuspešnejša metoda gručenja. Pri spletnih straneh s Ponzi shemami (HYIP) je bilo 4191 spletnih strani razvrščenih v 864 gruč velikosti dva ali več, ostalih pa je še 530 spletnih strani, ki so razvrščene v gruče velikosti ena. Od 1216 spletnih strani z lažnimi posredovalnicami so bile vse razen sedmih razvrščene v 161 gruč velikosti dva ali več.

4.1 Analiza velikosti gruč

Gruče s spletnimi stranmi z HYIP imajo manjše število primerkov, saj kar 530 gruč (40% vseh gruč) vsebuje samo eno spletno stran. 662 gruč (47%) vsebuje med 2 in 5 spletnih strani, 175 gruč (13%) vsebuje med 6 in 10 strani, 27 (2%) gruč pa vsebuje več kot 10 strani. Ti podatki nakazujejo, da se prevare z HYIP ne izvajajo tako pogosto kot ostale vrste spletnega kriminala.

Pri spletnih straneh z lažnimi posredovalnicami je samo 7 gruč, ki vsebujejo le eno spletno stran. 80 gruč (28%) vključuje med 2 in 5 spletnih strani, 79 (28%) gruč pa je velikosti med 6 in 20. 2 gruči pa vključujeta kar 113 in 109 spletnih strani. To nakazuje, da je pri tem tipu spletnega kriminala kopiranje lažnih spletnih strani dosti bolj pogosto kot pri spletnih straneh z HYIP.

4.2 Analiza načina kopiranja

Kriminalci pri kopiranju spletnih strani najpogosteje uporabljajo dva pristopa kopiranja. Pri prvem pristopu vzporedno ustvarijo več spletnih strani, s čemer si hitro zagotovijo velik doseg potencialnih žrtev. Pri drugem pristopu pa spletne strani ustvarjajo zaporedno oziroma po preteku določenega časa od vzpostavitve prejšnje spletne strani. Gruče z dolgo vztrajnostjo nakazujejo, da lahko kriminalci kopije spletnih strani vzdržujejo zelo dolgo, brez da bi bili kaznovani.

Na sliki 2 je 10 največjih gruč spletnih strani z lažnimi posredovalnicami. Spletne strani v gručah so razvrščene po datumu vzpostavitve. V dveh največjih gručah se spletne strani pogosto pojavljajo po več vzporedno na isti dan, medtem ko se v manjših gručah pojavljajo bolj zaporedno in poredko. Pri vseh desetih gručah se spletne strani pojavljajo v celotnem času opazovanja, kar pomeni, da kriminalci uporabljajo predlogo za kopiranje strani, dokler jih ne odkrijejo in ustavijo.

4.3 Analiza vztrajnosti gruč

Vztrajnost gruče je definirana kot čas med prvo pojavitvijo prve spletne strani v tej gruči in zadnjo pojavitvijo zadnje spletne strani iz te gruče. Za prikaz vztrajnosti gruč je na sliki 3 uporabljena verjetnost preživetja gruče. Funkcija preživetja *S(t)* meri verjetnost, da bo posamezna gruča prisotna več kot t časa. Polna črta na sliki prikazuje verjetnost preživetja, črtkane črte pa ponazarjajo 95% interval zaupanja. Iz levega grafa je možno razbrati, da 75% gruč z lažnimi posredovalnicami preživi vsaj 60 dni. Mediana preživetja je 90 dni. Okoli 25% gruč je preživelo 150 dni,

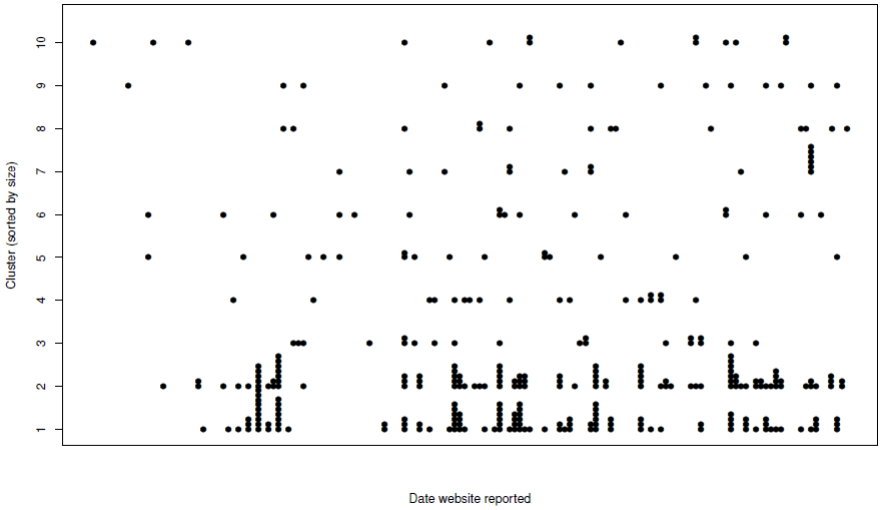


Figure 2: 10 največjih gruč spletnih strani z lažnimi posredovalnicami glede na datum odkritja.

torej celoten čas opazovanja. Za te gruče ni točnega podatka o vztrajnosti.

Čas opazovanja spletnih strani z HYIP je bil daljši, zato so podatki bolj popolni kot pri spletnih straneh z lažnimi posredovalnicami. 20% vseh gruč je preživelo vsaj 500 dni, medtem ko samo 25% gruč ni preživelo več kot 100 dni. Mediana preživetja je okoli 250 dni. Gruče s spletnimi stranmi z HYIP so dosti bolj vztrajne, zaradi tega pa jih je tudi lažje odkriti in kaznovati.

5. AVTOMATSKO PREPOZNAVANJE LAŽNIH SPLETNIH STRANI Z USCAP METODO

Kriminalci ustvarjajo lažne spletne strani z različnimi nameni. Eden bolj pogostih namenov je pridobivanje zaupnih informacij. Najpogosteje so to gesla, kode za dostop do sistemov, bančni podatki itd. Za ta namen ustvarjajo kopije originalnih spletnih strani, na katere želijo privabiti nič hudega sluteče obiskovalce. Takšne spletne strani se imenujejo "phishing" strani.

5.1 SCAP

Določevanje profila avtorja glede na izvorno kodo oziroma SCAP (Source Code Autorship Profile) je metoda, s katero se glede na izvorno kodo programov, določi avtor posameznega programa. Deluje na podlagi n-gramov, ki so lahko na nivoju bajtov, znakov ali besed. Glede na distribucijo n-gramov v vseh avtorjevih delih, se ustvari avtorski profil. Ta vsebuje L najbolj pogostih n-gramov. Nov program se dodeli tistemu avtorju, ki ima največji presek n-gramov med avtorskih profilom in programom. Zaradi enostavnega preverjanja avtorstva novih programov oziroma dokumentov, je ta metoda hitrejša kot predhodne metode, ki se uporabljajo za podobne namene. Natančnost te metode pa je v primerjavi z drugimi enaka ali večja [5].

5.2 USCAP razširitev

Nenadzorovano določevanje profila avtorja glede na izvirno kodo oziroma USCAP (Unsupervised Source Code Autorship Profile) je metoda, ki razširjuje metodo SCAP. Ta namreč deluje na naborih podatkov, pri katerih so primeri označeni z razredi oziroma v tem primeru avtorji. To pri avtomatskem prepoznavanju lažnih spletnih strani ni na voljo, zato je potrebna razširitev USCAP[10]. Sestavljena je iz treh korakov: ustvarjanje matrike podobnosti, gručenja in združevanja.

5.2.1 *Ustvarjanje matrike podobnosti*

V tem koraku se ustvari distribucija n-gramov za vse dokumente (pri SCAP se distribucija ustvari za vse avtorje). Distribucija za posamezen dokument je ustvarjena tako, da se izbere L najbolj pogostih n-gramov v tem dokumentu. Na podlagi raziskave [4] so primerne vrednosti za n od 2 do vključno 9, ter za L vrednosti 10, 50, 100, 200, 500, 1000, 1500, 2000 in 3000, odvisno od dokumentov. Podobnost dveh dokumentov se izračuna isto kot pri SCAP metodi, torej s presekom distribucij obeh dokumentov. Ta vrednost je normalizirana z deljenjem z L. Vrednosti bližje 0 pomenijo, da dva dokumenta zelo verjetno nimata enakega avtorja, medtem ko vrednosti bližje 1 pomenijo, da je dokumenta zelo verjetno spisal isti avtor. Ustvari se več matrik glede na različne parametre n in L.

5.2.2 *Gručenje*

V prejšnjem koraku ustvarjene matrike podobnosti so v tem koraku uporabljene za izdelavo gruč. Za to je uporabljen FMC algoritem (Fuzzy c-means MST Clustering algorithm). Algoritem FMC se uporabi posebej na vsaki izmed matrik iz prejšnjega koraka.

5.2.3 *Združevanje*

V zadnjem koraku se različna gručenja iz prejšnjega koraka združijo v končno gručenje. Posamezna gručenja so si med sabo podobna, prihaja pa do odstopanj in osamelcev. Združevanje poteka tako, da se najprej ustvari matrika podobnosti C. Posamezen vnos C_{ij} je procent gručenj iz prejšnjega koraka v katerih sta dokumenta i in j v isti gruči.

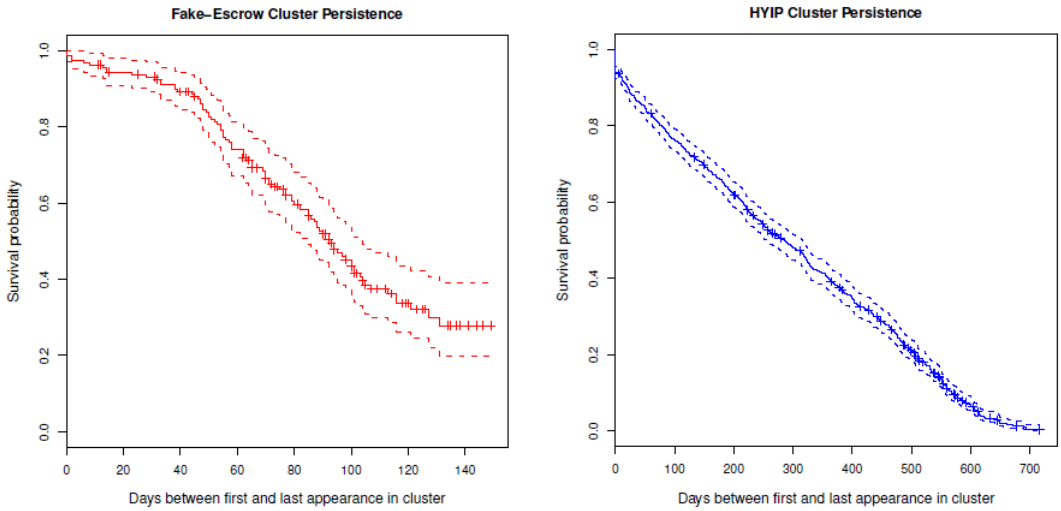


Figure 3: Vztrajnost gruč z lažnimi posredovalnicami (levo) in z HYIP (desno).

Kočno gručenje se nad matriko podobnosti C izvede z algoritmom FMC.

Končno gručenje se na koncu še preveri z metodo NMI (Normalized Mutual Information), ki meri stabilnost gručenja. Visoka vrednost NMI pomeni, da je bilo gručenje v drugem koraku stabilno in je visoko korelirano z gručenjem v tretjem koraku. Nizka vrednost NMI pa nakazuje, da je bilo v drugem koraku pri gručenju preveč variacije in da je slabo korelirano z gručenjem v tretjem koraku.

5.3 Uporaba na resničnih podatkih

V članku [1] je bila metoda USCAP uporabljena na 700 spletnih straneh, ki so ponarejale spletne strani večje Avstralske bančne institucije. Nabor podatkov vsebuje celotno izvorno kodo teh spletnih strani. Vse spletne strani so bile potrjene kot lažne s strani ekspertov.

Na teh podatkih je bilo odkritih 19 gruč. 11 gruč vsebuje 6-9 strani in 7 gruč vsebuje 11-25 strani. Največji gruči vsebujeta 154 in 338 spletnih strani. Matrike podobnosti so definirale goste in dobro ločene gruče z dobrimi NMI vrednostmi. Povprečje NMI je bilo 0.7685, standardni odklon pa 0.0346. To pomeni, da je v povprečju 85.9% odločitev za gručenje enakih v drugem in tretjem koraku, kar nakazuje dobro korelacijo.

6. VREDNOTENJE GLEDE NA RESNIČNE PODATKE

En izmed osnovnih izzivov gručenja logičnih kopij kriminalnih spletnih strani je pomanjkanje resničnih podatkov za vrednotenje natančnosti avtomatiziranih metod. Nekateri raziskovalci so se oprli na oceno ekspertov za ocenjevanje podobnosti, ampak večina pa se je odrekla sistematičnemu vrednotenju zaradi pomanjkanja resničnih podatkov. [9]

Ročno smo gručili 687 od 4191 HYIP spletnih strani in 684 od 1221 spletnih strani z lažnimi posredovalnicami. Izračunali smo prilagojen *Rand index* [11] za vrednotenje kombiniranih metod gručenja. Prav tako smo ovrednotili druge metode gručenja za primerjavo. Rand index je število v mejah od 0 do 1, kjer rezultat 1 pomeni popolno ujemanje med dvema gručama. Tabela spodaj prikazuje prilagojen Rand index za oba podatkovna nabora in za vse kombinacije vhodnih atributov. Prvi trije stolpci prikazujejo Rand index za vsako gručo individualno. Kot primer za storitve z lažnimi posredovalnicami, gručenje na podlagi označb ima Rand index 0.672. Zato je gručenje samo na podlagi stavkov veliko bolj učinkovito kot z datotečno strukturo ali stavki s spletnih strani samih (Rand indexi 0.072 in 0.10). Ko kombiniramo te vhodne atribute, opazimo izboljšanje. Gručenje na podlagi minimalnih razdalj med spletnimi stranmi glede na HTML označbe in stavke ima Rand index 0.9, medtem ko če vzamemo minimum od treh vhodnih atributov to prinese oceno Rand index 0.952. Zato so kombinirane ocene veliko boljše od katerikoli drugih primerjav.

Ker se spletni kriminalci obnašajo zelo različno, ko ustvarjajo logične kopije spletnih strani za različne vrste prevar, se lahko vhodni atributi, ki so najbolj podobni spremenijo. Na primer za HYIP lahko vidimo, da je gručenje po besedah na spletni strani najbolj učinkovito glede na Rand index, medtem ko so pri lažnih posredovalnicah HTML označbe najboljše. Prav tako lahko opazimo, da kombiniranje vhodnih atributov za nekatere prevare ni najbolj učinkovito.

Uporabili so številne različne splošno namenske metode gručenja iz *R Clue* [7] paketa kot ocene za vrednotenje najboljšega minimalnega pristopa.

1. **SE** - Implementira algoritem fiksne točke za pridobitev mehkih najmanjših kvadratov evklidskih soglasnih particij z minimiziranjem z uporabo evklidskih različnosti. [2, 7]

2. **DWH** Uporablja razširitev požrešnega algoritma za implementacijo mehkih najmanjših kvadratov evklidskih soglasnih particij [2, 7]

	Tags	Files	Sent.	T&F	T&S	S&F	Combined
Fake-Escrow Services							
Minimum	0.672	0.072	0.100	0.603	0.900	0.097	**0.952**
Average	0.672	0.072	0.100	0.071	0.077	0.086	0.075
Max	0.672	0.072	0.100	0.075	0.076	0.093	0.080
DISTATIS	0.672	0.072	0.100	0.069	0.076	0.071	0.070
Clue SE	0.672	0.072	0.100	0.223	0.207	0.053	0.128
Clue DWH	0.672	0.072	0.100	0.214	0.307	0.081	0.126
Clue GV3	0.672	0.072	0.100	0.665	0.513	0.086	0.562
Clue soft/symdiff	0.672	0.072	0.100	0.132	0.115	0.087	0.095
High-Yield Investment Programs							
Minimum	0.388	0.245	**0.710**	0.276	0.444	0.255	0.301
Average	0.388	0.245	0.710	0.351	0.516	0.402	0.443
Max	0.388	0.245	0.710	0.314	0.668	0.638	0.623
DISTATIS	0.388	0.245	0.710	0.374	0.565	0.542	0.563
Clue SE	0.388	0.245	0.710	0.187	0.307	0.262	0.245
Clue DWH	0.388	0.245	0.710	0.275	0.464	0.389	0.472
Clue GV3	0.388	0.245	0.710	0.281	0.549	0.415	0.508
Clue soft/symdiff	0.388	0.245	0.710	0.259	0.423	0.340	0.401

Figure 4: Primerjava metrike *Rand index* v odvisnosti od vrste spletne strani in izbora atributov.

3. **GV3** - Uporablja SUMT algoritem, kar je enakovredno iskanju pripadništva matriki m za katero je vsota kvadratnih napak med C(m)=mm' in uteženimi povprečji sočlanov matrike minimalna [6, 7]

4. **soft/symdiff** - Podano največje število razredov uporablja SUMT pristop za minimizacijo z uporabo Manhattanske različnosti sosednih matrik z syymdiff partcioniranjem različnosti v primeru trdih matrik... [3, 7]

Spodnja slika združuje najboljše meritve za različne kombinirane in soglasne metode gručenja.

	Escrow	HYIPs
Minimum	0.952	0.301
Average	0.075	0.443
Max	0.080	0.623
Best Min.	**0.952**	**0.710**
DISTATIS	0.070	0.563
Clue SE	0.128	0.245
Clue DWH	0.126	0.472
Clue GV3	0.562	0.508
Clue soft/symdiff	0.095	0.401
Click trajectories [18]	0.022	0.038

7. DODATEK: POVEZANO DELO

Veliko raziskovalcev uporablja strojno učenje za gručenje kriminalnih spletnih strani. Ward je opazoval datotečno strukturo opazovanih *phishing* strani [12]. Njegova hipoteza je, da z opazovanjem MD5 izvlečkov datotek, ki vplivajo na izkušnjo spletne strani, kot so .jpg in .gif datoteke, prav tako pa tudi .css in .js datoteke, lahko odkrijemo dovolj velike podobnosti, da potrdimo *phishing napad*, čeprav so bile narejene manjše spremembe na glavni HTML strani razlog za različno izračunan MD5 izvleček na tisti strani. Veliko *phisherjev* večkrat uporablja iste HTML, CSS, JavaScript in slikovne datoteke vsakič, ko ustvarja *phising* stran za napad na določeno znamko. Uspešni paketi za *phishing* napad na strani so zapakirane v *phishing kit*.

Njihova metoda najprej pridobi vse datoteke v *phishing* URL lokalnem imeniku. Prav tako, pa tudi vse datoteke v podimenikih. Za to uporabljajo program `wget` [1]. Po prenosu datotek izračunajo MD5 izvleček vsake datoteke s programom `md5deep`. MD5 izvlečki so uporabljeni za merjenje podobnosti do ostalih *phishing* strani. Te MD5 izvlečki se shranjujejo v javno dostopno podatkovno bazo. Dve spletni strani sta statistično primerjani z izračunom števila skupnih MD5 vrednosti v podatkovni bazi. Prav tako upoštevajo povprečno število ujemajočih datotek, skupno število ujemajočih strani, odstotek ujemajočih strani in število strani, ki imajo vsaj eno skupno datoteko. Te statistični podatki so uporabljeni za asociiranje mere podobnosti med različnimi spletni stranmi.

Za testiranje uspešnosti njihove metode so uporabili tri podatkovne nabore. V prvem je 12060 enoličnih MD5 vrednosti, ki so jih označili profesionalcei na tem področju. Podatki vključujejo glavno HTML stran iz kolekcije več kot 46000 potrjenih *phising* strani in 335 znamk, primarno banke, kreditne unije in druga spletna podjetja. Z opazovanje prvega podatkovenega nabora so našli velik procent *phishing napadov* proti določnih znamkam iz izbora majhnega seta URL-jev za to znamko. Veliko MD5 vrednosti je ostalo enoličnih. Drugi podatkovni nabor vsebuje nazadnje zbrane podatke o *phishing* straneh. Vzeli so podatke o 1030 *phising* URL-jih. Te strani vsebujejo 7156 datotek in 2575 enoličnih MD5 vrednosti. Tretji podatkovni nabor pa vsebuje 236 URL-jev. Ti URL-ji so bili zbrani za analizo ene same napadene znamke. Ta set vsebuje 1497 datotek in 658 enoličnih MD5 vrednosti.

V pregledu enoličnih *pshishing* strani, ki nimajo MD5 ujemanj v obstoječi podatkovni bazi so ugotovili, da napadalci uporabljajo tehniko za prikritje podatkov, ki preprečuje enostavno MD5 uspešno ujemanje. Primer tega je spodnja eBay

phishing shema.

```
marcsevigny.dyndns.org
ebayISAPII.dll
index.php@cmd=Validate &54fjcyhaag
nfgvebdxgelob3exzt4rl9or
vpm1343ingormpl4 marcsevigny.dyndns.org
ebayISAPII.dll
index.php@cmd=Validate &819wdcjhaagrtadcitzemrps
pcfheaan9gbe0db62nosnu733hr marcsevigny.dyndns.org
ebayISAPII.dll
index.php@cmd=Validate &8n002m3cshja7n6hxensyedqb
sanndc5d6yc3you072hb8dqsv
```

Jasno je, da tu URL-ji vodijo do iste lokacije, čeprav so zgenerirani MD5 izvlečki različni. V tem primeru strani niso enostavne HTML strani ampak PHP strani. Del PHP programa zgenerira naključni niz v vsebovani izvorni kodi strani, tako da vsaka posamezna spletna stran zgenerira enoličen MD5. Ugotovili so, da od 236 napadenih znamk podatkovnih setov URL-jev, 120 *phishing* spletnih strani vsebuje vsaj eno datoteko, ki ustreza drugim *phishing* spletnim stranem. To pomeni, da 116 spletnih strani ne vsebuje nobenih datotek, ki ustrezajo ostalim spletnim stranem. 29 od 236 URL-jev je bilo nedostopnih, ko so bile datoteke prenešene. Rezultati kažejo, da ima 66 URL-jev 40 ali več datotek, ki se ujemajo z ostalimi spletnimi stranmi. 79 URL-jev ima vsaj 11 ujemanj. Analiza je pokazala, da tri strani vsebujejo 64 datotek, ki so popolnoma enake. Nekatere od teh 64 datotek so prav tako vsebovane v 29 ostalih *phishing* straneh, ki vsebujejo 411 ujemajočih datotek. Teh 411 datotek ustreza 31 različnim spletnim stranem. Rezultati kažejo, da 70 od 236 URL-jev vsebuje HTML datoteke z MD5 vrednostmi, ki ustrezajo tistim shranjenim v javni podatkovni bazi. To pomeni, da je 29.7% *phishing* URL-jev lahko avtomatično označenih kot *phishing napad*. Rezultati kažejo, da podatkovni set vsebuje potrjenih več kot 50 URL-jev.

Ugotovitve iz te raziskave kažejo, da je trenutno mogoče z uporabo MD5 ujemanja avtomatsko označiti okoli 30 % spletnih strani samo z uporabo glavnih HTML MD5 vrednosti. Ko pa upoštevamo še ostale kriterije in zgoščene vrednosti ostalih datotek, pa lahko ugotovimo še več vzorcev.

8. LITERATURA

[1] *Wget*. http://www.gnu.org/software/wget/, 2008.

[2] A. W. E. Dimitriadou and K. Hornik. *A combination scheme for fuzzy clustering*. International Journal of Pattern Recognition and Artificial Intelligence, 2002.

[3] A. V. Fiacco and G. P. McCormick. *Nonlinear programming: sequential unconstrained minimization techniques*. Siam, 1990.

[4] S. G. G. Frantzeskou, E. Stamatatos and C. E. Chaski. Identifying authorship by byte-level n-grams: The source code author profile (scap) method. *Int. Journal of Digital Evidence*, 2007.

[5] S. G. G. Frantzeskou, E. Stamatatos and S. Katsikas. Effective identification of source code authors using byte-level information. In *ICSE '06: Proceedings of the 28th international conference on Software engineering*, page 893–896. ACM, 2006.

[6] A. Gordon and M. Vichi. *Fuzzy partition models for fitting a set of partitions*. Psychometrika, 2001.

[7] K. Hornik. *A clue for cluster ensembles*. 2005.

[8] C. Kanich, C. Kreibich, K. Levchenko, B. Enright, G. M. Voelker, V. Paxson, and S. Savage. Spamalytics: An empirical analysis of spam marketing conversion, 2008.

[9] T. M. N. Leontiadis and N. Christin. *Pick your poison: pricing and inventories at unlicensed online pharmacies In Proceedings of the fourteenth ACM conference on Electronic commerce, pages 621–638*. ACM, 2013.

[10] P. W. R. Layton and R. Dazeley. Automatically determining phishing campaigns using the uscap methodology. *eCrime Researchers Summit (eCrime)*, pages 1–8, october 2010.

[11] W. M. Rand. *Objective criteria for the evaluation of clustering methods*. Journal of the American Statistical Association, 1971.

[12] B. Wardman and G. Warner. *Automating phishing website identification through deep MD5 matching*. In eCrime Researchers Summit, 2008.