

FORENZIČKA ISTRAGA MALICIOZNIH PROGRAMA

Nikola Bojić

Univerzitet Singidunum, Beograd
nikolabojić@live.com

Gojko Grubor, doc. dr.

Univerzitet Singidunum, Beograd
ggrubor@singidunum.ac.rs

Apstrakt: Sa gledišta forenzičke nauke, maliciozni programi predstavljaju poseban problem. Postoji dosta metoda za otkrivanje malicioznih programa koje su danas u upotrebi. Sve ove metode imaju svoje prednosti i mane, a procenti uspešnosti variraju. Ovaj rad se bavi analizom novih metoda koje se zasnivaju na istraživanju podataka. Predviđa se da će upotreba istraživanja podataka, a u skladu sa povećanjem obima digitalne istrage i digitalnog dokaza vremenom rasti. U cilju poboljšanja digitalne forenzičke istrage malicioznih programa, obrađeno je nekoliko naučnih radova koji se bave problemom otkrivanja malicioznih programa pomoću tehnika istraživanja podataka. Ove tehnike su se pokazale kao dosta brže i tačnije od tradicionalnih.

Ključne reči: maliciozni programi, forenzička analiza, istraživanje podataka

1. UVOD

Maliciozni programi predstavljaju realnu pretnju za sve korisnike informacionih tehnologija, bilo da koristimo usluge elektronskog bankarstva, da posedujemo *Facebook* profil, ili samo upotrebljavamo elektronsku poštu. Postojeće metode zaštite često ne pružaju dovoljnu zaštitu korisnicima, a broj štetnih i ilegalnih programa je u porastu svake godine. Dva kompetentna izvora, antivirusne kompanije *McAfee* i *Symantec* se u svojim godišnjim izveštajima za 2008. godinu slažu da je broj malicioznih programa u opticaju tokom 2008. godine iznosio preko milion, od kojih je većina i kreirana tokom te godine [1][2]. Samo broj trojanaca je procenjen na oko 120.000 različitih vrsti. Istovremeno sa porastom broja virusa i crva, raste i njihov uticaj na forenzičku istragu.

U prošlosti je bilo moguće razvrstati maliciozne programe prema funkcionalnosti ili vrsti napada - na viruse, crve, trojance i ostale. Ovi programi su vremenom evoluirali i često sadrže više funkcionalnosti, pa se striktna podela izgubila. Složenost i mogućnosti koje poseduju neke vrste trojanaca, na primer, trojanci za udaljeni rad na računaru (*Remote administration trojan*), prevazilaze tek dosadnu smetnju na korisnikovom računaru. Pomoću ovih programa, udaljeni korisnik je u stanju da preuzme potpunu kontrolu nad zaraženim računarem, a bez znanja njegovog vlasnika. Na ovaj način, otvoreno je novo poglavlje u analizi računarskog kriminala, kada nije samo potrebno dokazati da je napad izvršen sa nekog računara, već i da li je vlasnik tog računara bio svestan napada.

Mnogi korisnici su svesni šta pomoću trojanaca može da se izvede sa njihovog računara, a da oni to, dok rade na istom tom računaru i ne primete. Vremenom, osumnjičeni za neki oblik kompjuterskog kriminala su naučili da kao osnov svoje odbrane iskoriste upravo mogućnost da je neko drugi preuzeo njihov računar. Dosta slučajeva u Velikoj Britaniji i Sjedinjenim američkim državama je ukazalo na efikasnost trojanske odbrane. Dva školska primera su slučajevi Grin i Kafri.

U oktobru 2002. godine, Džulijen Grin (*Julian Greene*) je uhapšen u Devonu, nakon što je policija tokom istrage njegovog računara pronašla primere dečije pornografije, [3]. Internet provajder je predao i log datoteke koji su Grina identifikovali kao osobu odgovornu za preuzimanje spornog materijala. Međutim, digitalni forenzičar koji je radio na odbrani Grina je pronašao tragove postojanja 11 trojanca i virusa na Grinovom kompjuteru. Detaljnom istragom istražitelj je dokazao da neki trojanci poseduju funkcionalnost za preuzimanje spornih slika na računar. Kao što je i slučaj sa većinom trojanaca, Grin je mogao biti potpuno nesvestan njegovog prisustva. U nedostatku dokaza da je Grin svojevolumno preuzeo ilegalni materijal, tužilac je odbacio optužbu.

Iako na prvi pogled sličan, ali dosta različit slučaj se dogodio septembra 2001. godine. Eron Kafri (*Aaron Caffrey*), 19-ogodišnji haker, je optužen da je pokrenuo *Denial of Service* napad na kompjutere luke u Hjustonu, u američkoj državi Teksas, [4]. Veb server luke je bio opterećen, a zapisi Internet provajdera su ukazali da je izvor napada bio Kafrijev računar. Za razliku od Grinovog slučaja, forenzička istraga Kafrijevog računara nije otkrila tragove trojanca, već upravo suprotno. Kafri je posedovao dosta hakerske literature i softverskih alata koji su mogli biti upotrebljeni za ilegalne aktivnosti. Tokom suđenja, Kafri se jednostavno branio činjenicom da je trojanac odgovoran i da je nakon napada obrisan sa njegovog računara. Sem osnovane sumnje, forenzički istražitelji nisu posedovali čvrste dokaze da je Kafri izvršio napad. Porota je u oktobru 2003. godine donela oslobađajuću presudu.

U međuvremenu, pojavila se još jedna ozbiljna pretnja. Tokom decembra 2003. godine, zabeležena je masovna pojava nove vrste kompjuterskog napada - trojanska ucena, [5]. Naime, autori malicioznih programa, uglavnom iz istočne Evrope, pretili su da će pomoću trojanaca podmetnuti ilegalni sadržaj na korisničke računare ukoliko im žrtve ne isplate oko 30-ak američkih dolara. Ova pretnja je dosta realna i dalje aktuelna, jer iskusan autor trojanca može uspešno da ukloni mnoge tragove postojanja malicioznog programa.

2. FORENZIČKA ISTRAGA

Razlika između forenzičke istrage i analize malicioznih programa je u različitom cilju pronalazjenja dokaza. Forenzička istraga treba da odgovori na pitanje da li je trojanac odgovoran za kriminalno delo koje je počinjeno sa osumnjičenog računara. Za razliku od analize malicioznih programa, kada se ispituju osobine programa, potrebno je rekonstruisati događaje koji su prethodili ili nastali nakon kriminalnog dela. Međutim, u sklopu forenzičke istrage se nalazi i analiza spornog programa, u cilju otkrivanja funkcionalnosti programa. Rezultati analize malicioznih programa moraju da budu tačni i ponovljivi, tako da mogu da budu uvršteni u dokazni materijal istrage ili tužbe.

Čak i pri istrazi kada ne postoji osnovana sumnja da je računar zaražen, forenzičar ne sme da isključi mogućnost prisustva malicioznih programa na računaru. Najbolje je da se istraga tragova zlonamernih programa odvija paralelno sa ostatkom istrage. Postoji nekoliko priznatih modela toka procesa za sprovođenje digitalne forenzičke istrage. Kao jedan od potpunijih modela može da se uzme model Ministarstva pravde Sjedinjenih američkih država - *Electronic Crime Scene Investigation Guide*, [6]. Apstraktovani model koji je više prilagođen istrazi malicioznih programa se zasniva na šest celina:

1. Forenzičko ispitivanje i očuvanje kratkotrajnih podataka
2. Ispitivanje memorije
3. Ispitivanje čvrstog diska
4. Statička analiza malicioznog softvera
5. Dinamička analiza malicioznog softvera
6. Kreiranje izveštaja

Ukoliko je moguće, forenzičku istragu treba početi na računaru u radnom stanju. Dokazi koji se prikupe tokom ove faze istrage mogu se pokazati kao krucijalni. Na primer, ukoliko maliciozni softver ima ugrađene neke od anti-forenzičkih tehnika, kao što je prepoznavanje virtuelnog okruženja, istražitelj možda neće biti u stanju da reprodukuje njegovo ponašanje u laboratoriji. Tokom ispitivanja pokrenutog računara treba sakupiti što više informacija o zatečenom stanju računara i mreže. Treba dokumentovati sve pokrenute programe, servise, drajvere i datoteke, otvorene portove i mrežne konekcije. Tek naknadnom, detaljnom analizom, forenzičar može metodom eliminacije da iz istrage isključi benigne procese i programe. Pri ovom koraku, posebnu pažnju treba obratiti upravo na anti-forenzičke tehnike koje su možda primenjene. Proces koji nosi naziv *iexplorer.exe*, iako liči na legitiman *Internet Explorer* proces može zapravo biti trojanac. Na primarnom računaru koji istražujemo može, ali i ne mora biti instaliran neki antivirusni program. Čak i kada antivirusni skener poseduje najsvežije definicije virusa, forenzičar ne sme u potpunosti da se osloni na njegove nalaze.

Kada je napravljena forenzička kopija čvrstog diska, ili kada su izdvojeni samo neki sumnjivi programi sa računara, jedan od prvih koraka koji forenzičar treba da preduzme je ispitivanje digitalnog dokaza u potrazi za malicioznim programima. Istražitelju je na raspolaganju nekoliko metoda i mnoštvo alata za otkrivanje zlonamernih programa. Najkorišćenija metoda kojom se antivirusne kompanije služe danas je ona zasnovana na potpisima poznatih virusa. Podrazumeva se da je forenzička laboratorija uvek snabdevena najsvežijom bazom potpisa. Čak i tada, ovo nije savršen metod za otkrivanje virusa iz brojnih razloga. Sa jedne strane, računar može biti zaražen novijom verzijom nekog trojanca koji još nije identifikovan i/ili čiji unikatni potpis još nije izdvojen. Realna pretnja je i da je virus namenski napisan za određenog korisnika i da antivirusne kompanije neće ni dobiti priliku da ga analiziraju. Sa druge strane, sam antivirusni softver može biti slabijeg kvaliteta, ili njegova podešavanja mogu biti ili fabrički ili slučajno pogrešno postavljena.

Jedan od predloženih alternativnih pristupa se zasniva na kreiranju bele liste benignih datoteka, pri čemu se svi programi koji se ne nalaze na listi smatraju malicioznim, što je upravo suprotno metodi izdvajanja potpisa virusa. Upotrebom nekog od heš algoritama kao što su MD5 ili SHA-1, kreiraju se unikatni potpisi poznatih čistih datoteka. Ove vrednosti, dobijene u laboratorijskim uslovima, se zatim mogu uporediti sa heš vrednostima datoteka koje čine digitalni dokaz i sve nepravilnosti se mogu izdvojiti radi dalje analize. Međutim, mana ove metode je odmah uočljiva kada se istražitelj nađe pred velikom količinom datoteka za koje nisu izdvojeni unikatni potpisi.

Ukratko, ni jedna automatizovana metoda koja je u upotrebi u forenzičkoj laboratoriji za otkrivanje malicioznih programa nije savršena. Videli smo da metoda bazirana na potpisima malicioznih programa ima svoje nedostatke, a do sada ni jedna druga metoda nije dostigla taj nivo tačnosti i brzine. Ukoliko postoji osnovana sumnja da je računar ipak zaražen, dok antivirusni skener ne pronalazi tragove virusa, forenzičar uvek može i treba da izvrši detaljnu statičku i dinamičku analizu svih sumnjivih programa. Primenom istih ovih metoda i antivirusne kompanije izdvajaju unikatne potpise malicioznih programa. Iako ni ovo nisu savršene metode, pomoću njih se sa velikom tačnošću mogu utvrditi zlonamerne datoteke, ali je za njihovu primenu potrebno dosta više vremena i stručnosti. Na statičku i dinamičku

analizu u cilju otkrivanja malicioznih programa treba gledati kao na ručnu, sporu i detaljnu analizu. One nisu primerene za brzo izdvajanje malicioznih programa iz digitalnog dokaza kao što to čini upotreba antivirusnog skenera.

2.1 Statička analiza

Statička analiza, ili analiza koda je dosta korišćena tehnika pri otkrivanju zlonamernih programa. Najbolji način da se razume delovanje nekog programa je putem ispitivanja njegovog koda. Programi se ispituju bez izvršavanja istih. Naravno, izvorni kod većine programa nije lako dostupan. Ovi programi se uglavnom prenose kao binarne datoteke, što znači da se moraju koristiti *debugger*-i i disasembleri. Potrebni su specijalizovano znanje i alati za tumačenje ovog koda. Ovaj proces se naziva i reverzni inženjering.

Windows operativni sistemi, 32-bitni i 64-bitni, koriste *Portable Executable* (PE) format fajla za izvršne datoteke, objekte koda i dinamične biblioteke (DLL). Ovaj format sadrži neophodne informacije da bi operativni sistem mogao da izvršava datoteke. Ovo uključuje reference ka dinamičkim bibliotekama, import i eksport API tabele, resurse i drugo. PE format se koristi za *exe*, *dll*, *obj*, *sys* (drajveri) i druge tipove fajlova. PE fajl se sastoji od niza zaglavlja (*header*) i sekcija koje govore dinamičkom linkeru kako se taj fajl prenosi u radnu memoriju. Samo zaglavlje dosta govori o arhitekturi procesora za koji je program napravljen, sadržaju i karakteristikama fajla, lokaciji tabele stringova i simbola i drugim informacijama. Svaka sekcija sadrži informacije o količini potrebne memorije, kao i dozvoljenim radnjama nad tom sekcijom, kao što su čitanje, pisanje, izvršavanje, itd.

Maliciozni, ali i ostali izvršni fajlovi, često dolaze u kompresovanom obliku. Ovo podrazumeva skladištenje algoritma za kompresiju i odgovarajućeg koda za dekompresiju u jedan fajl. Pri pokretanju kompresovanog programa, kod se prvo otpakuje, a potom i izvrši. Na prvi pogled, forenzičar ne može znati da li je neki program kompresovan, jer se ne primeće nikakva promena u načinu izvršavanja programa. Kompresovane fajlove treba posmatrati kao na varijantu samo-otpakujuće arhive. Programi se načešće kompresuju radi smanjivanja njihove veličine. Sa gledišta autora malicioznog softvera, prednost kompresije je i radi otežavanja reverznog inženjeringa, posebno ukoliko se koristi i neki algoritam za šifrovanje. Kao što se može videti iz istraživanja sprovedenog 2007. godine čak 47,8% svih malicioznih programa koristi pakovanje koda kao metod samozaštite, [7].

Možda najjednostavniji oblik statičke analize je izdvajanje svih stringova unutar izvršne datoteke. Pod stringovima mogu da se podrazumevaju i povezane dinamičke biblioteke i funkcije unutar njih. Iako je autor malicioznog programa u mogućnosti da dosta lako izmeni ili obriše stringove, pretraga za stringovima nam može otkriti: naziv virusa, poruke za korisnika, sakrivene komande i opcije, lozinke, internet i *e-mail* adrese ili već pomenute biblioteke i funkcijske pozive.

Disasembliranje se izdvaja kao posebna, napredna tehnika statičke analize. Disasembler je alat koji konvertuje mašinski jezik, nerazumljive karaktere tipa *0x5589e566*, u nešto razumljiviji asemblerski kod. Asembler je programski jezik niskog nivoa, čija najveća snaga leži u mogućnosti direktnog obraćanja procesoru što rezultuje velikom brzinom rada. Međutim, asemblerski kod nije lako razumljiv ni iskusnijim istraživačima.

2.2 Dinamička analiza

Dinamička analiza se zasniva na proučavanju ponašanja programa pri njegovom izvršenju. Analiza se obavlja pod laboratorijskim uslovima u kontrolisanom okruženju. Uglavnom se koriste virtuelne mašine tako da se svaki proces koji program pokrene ili izmeni

može posmatrati. Beleže se promene u okruženju (sistem, registri, mreža, itd), prati se mrežna komunikacija, promene na disku i ostale. Dobijene informacije se pažljivo dokumentuju, analiziraju i sastavlja se profil ponašanja programa. Prednosti dinamičke, u odnosu na statičku analizu je to što je potreban manji nivo stručnosti istraživača.

Postoji dosta različitih specifičnih programa za očitavanja parametara sistema. Računar na kome se obavlja ispitivanje treba da bude opremljen sa alatima za:

- posmatranje fajl sistema
- proveru integriteta fajlova
- posmatranje procesa
- posmatranje mrežne aktivnosti
- skenerom portova
- posmatranje registra
- debugovanje programa

Ukoliko postoji osnovana sumnja da posmatrani program primenjuje neke od kernel *rootkit* metoda za manipulisanje analizom sistemskih parametara, test računar se mora umrežiti sa drugima i mrežna aktivnost test računara se može posmatrati preko ostalih računara u mreži. Takođe, može se koristiti i pomoćna, na primer, *Linux* distribucija, za otkrivanje promena na hard disku test mašine koje je *rootkit* uspešno sakrio. Forenzički alati su česta meta malicioznih programa, jer što je duže vremena potrebno antivirusnoj kompaniji da izdvoji unikatan potpis, duži je životni vek tog programa.

Primena dinamičke analize se nekada može pokazati kao brža i korisnija u odnosu na statičku analizu. Naravno, svi dobijeni rezultati moraju da budu ponovljivi. Obe vrste analize treba izvršavati zajedno jer samo kompletne informacije koje se prikupe na takav način mogu da se upotrebe kao dokazni materijal. Međutim, obim digitalnog dokaza je u porastu i vremenski nije isplativo da se za svaki sumnjivi program na sistemu vrši ručna statička i dinamička analiza. Potrebno je pronaći način za automatizaciju istrage, a koji bi istovremeno doveo i do nedvosmislene interpretacije dobijenih dokaza.

2.3 Primena istraživanja podataka u digitalnoj forenzici

Digitalna istraga postaje sve složenija i zahteva više vremena, ljudskih i računarskih resursa, kako količina digitalnih podataka sve više raste. Za bolje razumevanje o kojoj količini podataka se radi, kao dobar primer može da posluži istraga sprovedena tokom *Enron* skandala. Tokom istrage, pojavilo se preko 31 terabajt podataka koje je trebalo ispitati. Prema nekim poređenjima, jedan terabajt je ekvivalent oko 250 miliona strana teksta. Potencijalni dokaz je činilo preko 130 kompjutera, hiljade *email*-ova i više od 10 miliona stranica dokumenata, [8]. Količina i složenost podataka koje se pojavljuju u istrazi nadmašuju mogućnosti ručne analize i predviđanja su da će količina podataka samo rasti. Procena je da količina elektronske pošte u organizacijama godišnje raste za 29%, a da se količina podataka udvostručuje na svakih devet meseci, [9].

Postoji dosta primera gde je istraživanje podataka uspešno primenjeno pri forenzičkoj analizi i oni nisu vezani samo za digitalne istrage. Upravo mogućnost da se istraživanje podataka izvrši nad različitim skupovima podataka je često bilo ključno za forenzičke istrage. Mnogi autori su opisali tehnike koje su uspešno primenili pri istragama.

Abraham i de Vel, u svom radu iz 2002. godine su pokazali kako se asocijativne veze mogu iskoristiti da bi se napravio profil ponašanja korisnika računara, [10]. Oni su istraživali *log* fajlove korisnika i napravili su skup pravila koji opisuju uobičajeno korišćenje računara.

Na ovaj način, bili su u mogućnosti da lako uoče sve nepravilnosti i anomalije pri korišćenju računara, koje mogu biti pokazatelj nedozvoljene upotrebe sistema.

Carrier i Spafford, u istraživanju sprovedenom 2005. godine su pokazali način na koji se analiza *outlier*-a može iskoristiti za otkrivanje potencijalnih dokaza u fajlovima i direktorijumima, koji su sakriveni ili se dosta razlikuju od datoteka u svom okruženju. *Outlier*-i su slučajevi čije vrednosti se jako razlikuju od većine vrednosti u datom setu podataka. Da bi se pronašle sakrivene datoteke, karakteristike svakog fajla unutar direktorijuma se porede da bi se pronašli moguće anomalije.

Support vector machine (SVM) su takođe korišćenje u nizu forenzičkih istraga. SVM je algoritam za klasifikaciju koje proizvodi kategorizovane podatke na osnovu fundamentalnih osobina podataka. *Corney et al* su 2002. godine objavili rad u kome su pokazali kako se SVM može iskoristiti za otkrivanje godišta i pola autora *e-maila*, na osnovu korišćene gramatike i izraza, [11].

Istraživanje fotografija i grafičkih fajlova na računaru su jedni od najčešćih aktivnosti tokom forenzičke istrage. SVM se može uspešno iskoristiti za prepoznavanje određenih delova fotografija (*Brown et al*, 2005.). Takođe, SVM se može iskoristiti za automatsko filtriranje slika, bazirano na prepoznatim osobinama datoteke, čime se smanjuje obim digitalnog dokaza. *Mukkamala i Sung*, su 2002. godine pokazali još jedan uspešan primer korišćenja SVM, primenjiv u mrežnoj forenzici za otkrivanje upada na računarske sisteme i mreže, [12].

Značajno istraživanje su 2004. godine sproveli *Carney i Rogers*, u kome su iskoristili diskriminantnu analizu da bi utvrdili da li su zabranjeni sadržaji, kao što je dečija pornografija, svesno preuzeti sa Interneta, ili korisnik nije bio upoznat sa njihovim preuzimanjem, [13]. Optuženi za posedovanje ilegalnog sadržaja se često brane time da je maliciozni softver, kao što je trojanski konj, odgovoran za nedozvoljene radnje preduzete sa njihovog računara, ili za postojanje ilegalnih fajlova na računarima. Autori ovoga rada su uvideli potrebu za standardizovanom i efektnom metodom koja bi takozvanu *trojansku odbranu* mogla da potvrdi ili opovrgne. Metoda se zasniva na rekonstrukciji događaja koji su prethodili ilegalnoj aktivnosti, a slično kao i pri korišćenju DNK dokaza, u obzir se uzima mogući procenat greške.

Takođe, kao i druge forenzičke discipline, digitalna forenzika se zasniva na postavljanju hipoteza zasnovanih na dostupnim dokazima i činjenicama. Iako postoje procedure i alati za pribavljanje digitalnog dokaza, malo radova se bavi tačnosti postavljene hipoteze. Bez osnovanog i naučnog modela, zaključak koji donese forenzičar uvek može biti doveden u pitanje. Bajesove mreže su uspešno korišćene za izradu modela koji meri validnost hipoteze u procentima i na taj način povećava osnovanost i analitički rezultat do koga je dovela digitalna istraga, [14].

2.4 Otkrivanje malicioznih programa pomoću tehnika istraživanja podataka

Mnogi naučnici se bave problemom prepoznavanja zlonamernih programa pomoću metoda istraživanja podataka. U ovom poglavlju biće prikazano nekoliko radova i modela istraživanja podataka koji su razvijeni i pokazuju različite stepene uspešnosti. Većina pokazuje i dosta bolje performanse od tradicionalne metode upoređivanja potpisa. Ovi radovi su uglavnom bazirani na otkrivanju virusa za *Windows* operativni sistem, što je razumljivo kada se uzme u obzir količina virusa napisanih za ovaj sistem. Međutim, primenjene metode su opšte, što znači da bi trebalo da pomognu pri otkrivanju zlonamernih programa na svim platformama. Razlike između radova se pored upotrebe različitih algoritama, svode i na upotrebu različitih ulaznih podataka, tj. različitih osobina programa. Svi pristupi se zasnivaju

na istraživanju osobina koje se prikupljaju tokom statičke analize. Razlog za ovo je što se proces za dobijanje ovih osobina može standardizovati i time ubrzati pri zadržavanju dosta visoke tačnosti modela.

Autori *J. Zico Kolter* i *Marcus A. Maloof* su tokom 2006. godine objavili rezultate istraživanja u kome su pokušali ne samo da utvrde malicioznu prirodu nekog programa već i kategoriju kojoj taj program pripada, [15]. Ovo je dosta zahtevniji posao, ali i dosta korisniji sa gledišta forenzike. Sakupili su ukupno 3,622 programa, od čega su 1, 971 benigni i 1, 651 maliciozni program. Benigne programe su činili i sistemski i aplikativni, a zlonamerne sve kategorije od *keylogger*-a do *backdoor*-ova. Interesantno je da su svi maliciozni programi ispitani pomoću jednog komercijalnog antivirusnog paketa i od ukupne kolekcije od 1, 651 programa, čak 50 nije prepoznato kao maliciozni, iako se već godinama nalaze u javnom vlasništvu, dostupni svima.

Ovo istraživanje se zasniva na izdvajanju sekvenci bitova iz izvršnog koda i njihovu konverziju u n-grame. Pomoću alata *hexdump* svi izvršni fajlovi su konvertovani u ASCII format. Potom su izdvojeni n-grami, tako što su kombinovani sve sekvence od po 4 bajta u jednu reč. Na primer, za sekvencu ff 00 ab 3e 12 b3, odgovarajući n-gram bi bio ff00ab3e, 00ab3e12, ab3e12b3. Ukupno je kreirano 255,904,403 različitih n-grama. N-grami su posmatrani kao reči i korišćeni su algoritmi za klasifikaciju teksta, gde reč ili postoji (T) ili ne (F). Jasno je da su n-grami zapravo binarna reprezentacija stringova ili programskih funkcija i bilo je relativno lako utvrditi prirodu nekih sekvenci. Zanimljiv podatak je da se sekvenca 0000000a nalazi u 75% malicioznih programa, ali autori nisu mogli da utvrde čemu sekvenca odgovara.

Od ukupno 255,904,402 n-grama, odabrano je 500 najrelevantnijih atributa, u ovom slučaju n-grama, računanjem težine svakog atributa pomoću *Information Gain* metode. Računanje težine je često korišćen algoritam za odlučivanje koji atributi su od najveće važnosti, tako da se oni mogu proveriti pri samom korenu u drvetu odlučivanja. Konstruisano je nekoliko klasifikatora, *J48*, *IBk* (koristi *kNN* algoritam), naivni Bajes, *SVM*, drvo odlučivanja, *boosted* naivni Bajes, *booster SVM* i *boosted* drvo odlučivanja. Kako su svi uzorci programa sakupljeni tokom 2003. godine, tokom pisanja rada se pojavilo još malicioznih programa, ukupno 291. Autori su izvršili analizu i ovih novih programa, koji se nisu pojavili ni u skupu za obučavanje ni u test skupovima. Rezultati ove realne provere su dati u tabeli 1.

Metoda	Dozvoljeni procenat lažnih pozitiva					
	0.01		0.05		0.1	
	O	D	O	D	O	D
Boosted J48	0.94	0.86	0.99	0.98	1.00	1.00
SVM	0.82	0.41	0.98	0.90	0.99	0.93
Boosted SVM	0.86	0.56	0.98	0.89	0.99	0.92
iBk, k = 5	0.90	0.67	0.99	0.81	1.00	0.99
Boosted naivni Bajes	0.79	0.55	0.94	0.93	0.98	0.98
J48	0.20	0.34	0.97	0.94	0.98	0.95
Naivni Bajes	0.48	0.28	0.57	0.72	0.81	0.83

Tabela 1. Rezultati realne provere. O predstavljaju očekivane vrednosti, D dobijene

Već je pomenuto da se ovaj rad bavi i utvrđivanjem kategorije kojoj maliciozni programi pripadaju. Metode kompresije i šifrovanja su dosta puta onemogućile utvrđivanje prirode malicioznog programa, ali ne mogu da spreče grubu podelu na maliciozne ili benigne. Takođe, autori slabe performanse određivanja kategorije, pripisuju i nedostatku informacija koje su sakupili o svakom malicioznom programu. Od 1,651 zlonamernog programa, za samo 525 su uspeali da pronađu detaljne informacije o delovanju programa, koje su neophodne radi klasifikacije.

Takođe, treba razumeti da je procenat dozvoljenih lažnih pozitivna od 0.1 potpuno prihvatljiv u forenzičkoj laboratoriji. Lažni pozitivi su zapravo benigni programi koje je model prepoznao kao maliciozne. Mnogo je lakše izvršiti detaljnu analizu na benignom programu i utvrditi njegovu pravu prirodu, nego dopustiti da se zbog praga osetljivosti kroz model provuče nedetektovani maliciozni program.

Drugi rad, autora *Muazzam Siddiqui*, *Morgan C. Wang* i *Joohan Lee*, je predstavio novinu u odabiru ulaznih promenljivih koje se koriste za istraživanje podataka, [16]. Autori su odlučili da je najbolje koristiti sekvence instrukcija promenljive dužine, za razliku od uobičajenog korišćenja stringova ili n-grama. Ideja je da se na osnovu samog toka programa, koji opisuju instrukcije, prepozna zlonamerno ponašanje programa.

Za razliku od korišćenja fiksiranog broja instrukcija, ili n-grama, promenljiva dužina instrukcija omogućava verno skladištenje čitavog programskog bloka. Prednosti su:

- Sekvence instrukcija sadrže kontrolni tok instrukcija programa, što se gubi pri korišćenju n-grama
- Informacije o programu se skladište više na semantičkom nivou, a ne skup pojedinačnih instrukcija
- Ove sekvence se mogu prepoznati u originalnom fajlu, nakon rezultata istraživanja podataka, radi detaljne analize
- Spajanjem više sekvenci, može se povećati efikasnost klasifikacije
- Veliki broj sekvenci koje se pojavljuju i u benignim i u malicioznim programima se mogu zanemariti, kako bi se ubrzao proces modelovanja

```
inc  jb
ins  cmp    in  popa  jz
dec  arpl   xor  xor   jb
```

Slika 1. Izdvojene instrukcije pomoću PHP parsera

Korišćenjem PHP parsera, izdvojeni su samo imena funkcija i operandi su odbačeni, što je i prikazano na slici 1. Svaki red, prikazan na slici 1, predstavlja jednu sekvencu instrukcija. Ukupno je izdvojeno 1,972,920 sekvenci. 47% sekvenci pripada malicioznim programima i 53% benignim. Korišćenjem *Feature Extraction Mechanism*-a izdvojeno je 213,330 jedinstvenih sekvenci sa različitom učestanošću pojavljivanja. Sve sekvence koje se pojavljuju samo jednom, odbačene su, jer ukoliko bi se one koristile, model bi se sveo na prepoznavanje potpisa. Ukupno je odbačeno 94% sekvenci i samo je 23,738 sekvenci pronađeno i u čistim i u zlonamernim programima. Još skoro 97% sekvenci je odbačeno jer se pojavljuju u manje od 10% slučajeva. Ukupan broj sekvenci nakon ovog koraka je bio 679. *Chi-Square* test nezavisnosti je izvršen nad svakom sekvencom, da bi se odredilo da li postoji veza između te sekvence i ciljane promenljive. Promenljive su prikazane u binarnom obliku kao T ukoliko postoje i F ukoliko ne postoje. Za *p* vrednost od 0.01, odbačeno je skoro pola

osobina koje nisu pokazali nikakvu važnu statističku vezu sa ciljanom promenljivom. Broj sekvenci nakon ovoga je bio 268. Kolekcija je podeljena na 70% skupa za obučavanje i 30% test skupa. Korišćeni su algoritmi drveta odlučivanja, *Bagging* i *Random Forest*-a.

Klasifikator	Odnos detekcije	Lažni pozitivni	Ukupna tačnost
<i>Random Forest</i>	95.6%	3.8%	96%
<i>Bagging</i>	94.3%	6.7%	93.8%
<i>Drvo odlučivanja</i>	93.4%	13.4%	90%

Tabela 2. Rezultati eksperimenta nad instrukcijama promenjive dužine

Uporedni rezultati korišćenih algoritama se mogu videti u tabeli 2. U ovom eksperimentu, sekvence su izdvojene samo iz izvršne sekcije PE fajla. U ovoj sekciji se sadrži izvorni kod i srž programa. Autori smatraju da se tačnost od 96% može popraviti korišćenjem i ostalih osobina zapisanih u PE fajlu, kao što su tabela importa, API pozivi, PE zaglavlje i ostali.

Još jedan rad, autora *Matthew G. Schultz et al.* se uglavnom bavi poređenjem metoda istraživanja podataka sa tradicionalnom metodom otkrivanja potpisa, [17]. Da bi se izvršilo ovo poređenje, autori su konstruisali i automatski generator potpisa. Nije postojao način da se koristi neki od komercijalnih programa za ovu namenu, jer oni sadrže potpise svih programa u kolekciji, tako da se ne bi mogao testirati njihov rad sa novim, nepoznatim programima. I ovaj model je obučen samo nad skupom za obučavanje, a proveren nad test skupom. Od svih upotrebljenih algoritama, skener potpisa ima najmanju meru lažnih pozitivna od 0%, ali i najmanju meru detekcije od 33,96% i tačnosti od 49,31%. Osobine koje su posmatrane su informacije o resursima sistema, stringovi i sekvence bajtova. Korišćeni su tri algoritma za klasifikovanje i tri grupe podataka: *RIPPER*, naivni Bajes i Multi-naivni Bajes.

Kolekcija se sastojala od ukupno 4,301 programa, od kojih su 3,301 maliciozni i 1,000 benigni. Kako je rad nešto stariji, programi su se nalazili ili u *Windows* ili u *MS-DOS* formatu. Iz programa koji su se nalazi u PE formatu (ukupno 244), bilo je moguće izdvojiti informacije o resursima (funkcijskim pozivima ili povezanim bibliotekama). Tri tipa osobina koji su mogli biti izdvojeni iz PE programa su:

1. Lista dinamičkih biblioteka, DLL-ova koje program poziva
2. Lista DLL funkcijskih poziva
3. Broj različitih funkcijskih poziva unutar svakog DLL-a

Prva grupa je kreirana na osnovu liste DLL-ova koje program poziva. Napravljen je vektor sa 30 bulovih vrednosti, u zavisnosti da li se ta biblioteka poziva ili ne, u formatu:

$$\neg \text{advapi32} \wedge \text{avicap32} \wedge \dots \wedge \text{winmm} \wedge \neg \text{wsock32}$$

Drugi pristup je izdvajanje korišćenih DLL-ova, ali i imena njihovih funkcija. Vektor se sastojao od 2,229 bulovih vrednosti, u formatu:

$$\text{advapi32.AdjustTokenPrivileges()} \wedge \text{advapi32.GetFileSecurityA()} \wedge \dots \wedge \text{wsock32.recv()}$$

Pri trećem pristupu, meren je broj različitih funkcijih poziva unutar jednog DLL-a. Kako se lista korišćenih DLL-ova sastojala od 30 biblioteka, tako se i ovaj vektor sastoji od 30 celobrojnih vrednosti. Na primer, biblioteka *advapi32* je pozivana dva puta, *avicap32* deset puta, itd.

$$advapi32 = 2 \wedge avicap32 = 10 \wedge \dots \wedge winmm = 8 \wedge wsock32 = 0$$

Funkcijske pozive je moguće izdvojiti samo iz *Windows* datoteka u PE formatu, pa su autori izdvojili i sve stringove, kao i sekvence bajtova unutar programa.

Tip profila	Odnos detekcije	Odnos lažnih pozitivna	Ukupna tačnost
Metod potpisa	33.96 %	0 %	49.31 %
RIPPER			
1. Korišćeni DLL-ovi	57.89 %	9.22 %	83.62 %
2. DLL-ovi sa f. pozivima	71.05 %	7.77 %	89.36 %
3. DLL sa brojem f. poziva	52.63 %	5.34 %	89.07 %
Naivni Bajes			
Izdvojeni stringovi	97.43 %	3.80 %	97.11 %
Multi-naivni Bajes			
Sekvence bajtova	97.76 %	6.01 %	96.88 %

Tabela 3. Rezultati klasifikacije prikazani za svaki algoritam i izdvojene osobine

Tabela 3. prikazuje rezultate klasifikacije nepoznatih programa za tri algoritma kao i za metodu potpisa. Metoda potpisa pokazuje najslabije rezultate, ali ima i najmanji odnos lažnih pozitivna od 0%. Kao što smo već zaključili, u forenzičkoj laboratoriji odnos lažnih pozitivna nije od ključnog značaja. Najbolji odnos detekcije pokazuje Multi-naivni Bajes algoritam sa bajtovima, dok je ukupna tačnost na strani naivnog Bajes algoritma sa upotrebom stringova.

3. ZAKLJUČAK

U trenutku kada paralelno raste i obim digitalne istrage i opasnost od malicioznih programa, jasna je potreba za novim metodama otkrivanja zlonamernih programa. Pažnja mnogih istraživača je usmerena na upotrebu algoritama istraživanja podataka pri detekciji i klasifikaciji, prvenstveno još novih i neotkrivenih, malicioznih programa. Možda je

istraživanje podataka još daleko od svakodnevne upotrebe od strane kućnih korisnika, ali itekako može da pomogne pri obimnoj digitalnoj istrazi. Ovaj rad je pokazao neke inventivne načine upotrebe istraživanja podatak za razne forenzičke poslove, od vrednovanja hipoteza do klasifikacije malicioznih programa.

4. REFERENCE

1. Computer viruses hit one million, <http://news.bbc.co.uk/2/hi/technology/7340315.stm> (Posećeno: 23.02.2010.)
2. Grupa autora, 2008. McAfee Virtual Criminology Report, strana 4.
<http://www.mcafee.com/>
3. Man blames Trojan horse for child pornography, Sophos Anti-Virus reports
http://www.sophos.com/pressoffice/news/articles/2003/08/va_porntrojan.html (Posećeno: 23.02.2010.)
4. Caffrey acquittal a setback for cybercrime prosecutions,
http://www.theregister.co.uk/2003/10/17/caffrey_acquittal_a_setback (Posećeno: 23.02.2010.)
5. Trojans used for online extortion,
<http://news.zdnet.co.uk/security/0,1000000189,39199958,00.htm> (Posećeno: 23.02.2010.)
6. Electronic Crime Scene Investigation: A Guide for First Responders. <http://www.ncjrs.org>
7. <http://www.viruslist.com/analysis?pubid=204791949> (Posećeno: 23.02.2010.)
8. <http://www.fbi.gov/page2/may07/rcfl050707.htm> (Posećeno 23.02.2010.)
9. Beebe, N. i Clark, J. 2005. *Dealing with terabyte data sets in digital investigations. Advances in Digital Forensics*, pp. 3-16. Springer.
10. <http://www2.computer.org/portal/web/csdl/doi/10.1109/ICDM.2002.1183880> (Posećeno 23.02.2010.)
11. Corney et al., 2002. *Language and gender author cohort analysis of e-mail for computer forensics*.
12. Mukkamala et al., 2002. *Modeling intrusion detection systems using linear genetic programming approach*.
13. Megan Carney, Marc Rogers, 2004. *The trojan made me do it: A first step in statistical based computer forensics event reconstruction, Advances in Digital Forensics*
14. Duval, T., Jouga, B. i Roger, L. 2005. *The Mitnick case: how Bayes could have helped*.
15. J. Zico Kolter, Marcus A. Maloof, 2006, *Learning to Detect and Classify Malicious Executables in the Wild*
16. Muazzam Siddiqui, Morgan C. Wang, Joohan Lee, *Detecting Internet Worms Using Data Mining Techniques*

17. *Matthew G. Schultz, Eleazar Eskin, Erez Zadok, Salvatore J. Stolfo, Data Mining Methods for Detection of New Malicious Executables*

Abstract: From the standpoint of forensic science, malicious programs represent a special problem. There are number of methods of malicious software detection in use today. All of them have their advantages and flaws, and detection rates vary. This paper analyze new methods for malicious software detection based on data mining techniques. It is anticipated that the use of data mining, in accordance with the increasing volume of digital investigation and digital evidence, will only grow over time. In order to improve digital forensics investigation, several scientific papers that suggest new methods for software profiling are analyzed. These methods are recognized as faster and their performance is better than traditional.