# Mozgalo: Detekcija pakiranih datoteka

Erik Banek Iva Manojlović Ivan Lazarić Daniel Paleka

***Abstract*—Opisujemo kako pomoću strojnog učenja detektirati pakirane datoteke. Pristup podacima za treniranje i testiranje je omogućila tvrtka Reversing Labs. Temeljni dio rješavanja problema bio je dobro odabrati koje su bitne značajke dobivene statičkom analizom binarnih datoteka i naučiti što se sve mijenja prilikom pakiranja. Problem je bio zadan kao jedan od nagradnih zadataka studentskog natjecanja Mozgalo te se ovaj dokument uglavnom bazira na našem rješenju, s kojim smo osvojili 1. mjesto.**

## I. Uvod

Pakiranje je metoda izmjene izvršnih datoteka bez mijenjanja njihove izvorne funkcionalnosti [1], s ciljem da se smanji veličina originalne izvršne datoteke ili da se obfuscira maliciozni izvršni kod, često na način da se računalo zaštiti od reverznog inženjeringa. Pakiranje podrazumijeva izmjenu sadržaja datoteke te dodavanje instrukcija koje će prilikom izvršavanja taj sadržaj obnoviti. Za sakrivanje većine malwarea najčešće se koristi kombinacija različitih packera pa je zbog toga vrlo korisno imati metodu koja će automatski detektirati pakirane datoteke. Iako packer ne mora nužno služiti za sakrivanje malwarea, sustav koji bi detektirao je li datoteka pakirana bi sigurno bio koristan i u detekciji malwarea. Packeri mogu mijenjati izvršnu datoteku na nekoliko načina:

- kompresijom podataka
- enkripcijom podataka
- obfuskacijom
- dodavanjem detekcije izvršavanja unutar debuggera ili virtualnog računala
- modificiranjem raznih dijelova formata izvršne datoteke

U području računalne sigurnosti česti su packeri za Windows-PortableExecutable, tj. PE datoteke.

Cilj ovog projekta je napraviti model strojnog učenja koji će što točnije detektirati pakirane PE datoteke. Pretpostavka projekta je da se pomoću informacija dobivenih statičkom analizom PE binanih datoteka i samih PE datoteka mogu izvući podaci pomoću kojih ćemo nekim algoritmom strojnog učenja moći vrlo uspješno klasificirati pakirane datoteke. Dobar sustav prepoznavanja mogao bi nam omogućiti [3]:

- ranu detekciju nikad viđenog malwarea
- prepoznavanje malware kampanja
- brže prepoznavanje malwarea
- skeniranje i proučavanje virusa osobama s manje domenskog znanja

## II. Opis problema

Skup podataka koji smo koristili u ovom projektu napravila je tvrtka ReversingLabs za potrebe natjecanja Mozgalo te se sastoji od originalne datoteke i TitaniumCore izvještaja za tu datoteku. Uzeli su neki određen skup poznatih packera i njima zapakirali neke binarne datoteke, i na kraju za svaku datoteku napravili i TitaniumCore izvještaj. TitaniumCore izvještaji [2] su napravljeni statičkom analizom PE datoteka, uzimajući u obzir njihov format. Ti izvještaji su vrlo iscrpni, i glavni cilj pro- jekta je odabirom značajki iz izvještaja napraviti najbolje predviđanje je li neka datoteka pakirana. Iako je posao izvlačenja informacija iz same binarne podatke uvelike napravljen u izvještajima, pokušali smo i sami konstruirati neke značajke direktno iz binarnih datoteka. Skup podataka podijeljen je na skup za treniranje i skup za testiranje. Skup za testiranje nije bio dostupan tijekom natjecanja. Skup za treniranje sastoji se od 43784 primjerka,od toga 21962 nepakiranih datoteka, a broj pakiranih datoteka je po (ovisno o kategoriji korištenog packera) grupama 1956 , 1926 , 14604 i 3336. Cilj projekta je samo binarna klasifikacija, tj. odrediti je li datoteka pakirana ili nije, ali promatranjem preciznosti i osjetljivosti za svaku od grupa pakera dobivali smo dodatne informacije o tome na što je model koji konstruiramo osjetljiv.

Skup za testiranje sastoji se 6928 datoteka, a distribuirani su u omjeru 3 : 1 : 1 : 1, nepakirani/raspakirani : pakirani12 : pakirani3 : pakirani4 Omjer će biti 3 : 1 : 1 : 1 , nepakirani/raspakirani : pakirani12 : pakirani3 : pakirani4. Otprilike 20% pakiranih primjera je od dosad neviđenih packera.

## III. Opis metode i pristupa za rješavanje problema

Ovaj problem probat ćemo riješiti izvlačenjem značajki iz TitaniumCore izvještaja koji su u .json formatu u kombinaciji sa značajkama izvučenim iz PE datoteka. Razmišljali smo o tome što se sve mijenja prilikom pakiranja. Za očekivati je da će prilikom enkripcije ili kompresije porasti entropija. Sumnjivo je ako neke sekcije zovu vrlo mali broj api-ja, pogotovo ako su to api-ji koji mogu pozivati druge api-je pa prije izvršavanja nemamo dobar uvid u to što će sve bit pozvano. Korisnim se pokazalo gledati koja dopuštenja imaju pojedine sekcije jer bi one s puno ovlasti mogle sadržavati maliciozni kod. Na početku smo se vodili takvim razmišljanjima, što nam je dalo dobre rezultate. Neke od značajki iz TitaniumCore izvještaja koje smo izvukli su:

- entropy
- broj api-ja koje sekcije pozivaju
- ukupna entropija sekcija
- maksimalna entropija sekcija
- broj sekcija s entropijom većom od 7
- veličina filea

- veličina koda
- veličina stoga
- veličina rezervnog stoga
- veličina inicijaliziranih podataka
- veličina neinicijaliziranih podataka
- verzija glavnog linkera
- adresa na kojoj počinju podaci
- verzija saporednog linkera
- broj izvršivih sekcija
- broj sekcija po kojima se može pisat
- broj sekcija iz kojih se može čitati
- broj sekcija koje imaju inicijalizirane podatke
- broj sekcija koje imaju neinicijalizirane podatke
- broj sekcija iz kojih se može i čitati i pisati
- broj sekcija koje imaju neinicijalizirane podatke i koje se mogu izvršavati
- broj poziva funkcija koje pokazuju na adrese

Značajke dobivene direktno iz binarnih datoteka daju više informacija o samoj distribuciji bajtova unutar filea. Izračunali smo relativne frekvencije svih bajtova, među njima smo uzeli najčešćih 20 vrijednosti i na tome računali očekivanje, medijan, standardnu devijaciju, koeficijent asimetrije, gornji i donji kvartil i interkvartil. Dalje, za svaku datoteku smo 100 puta uzeli nasumičan uzastopni podniz od 500 bajtova, izračunali entropiju za njega, za tih 100 vrijednosti smo opet računali iste statistike. Uz to smo računali i entropije prvih 50,100 i 150 bajtova.

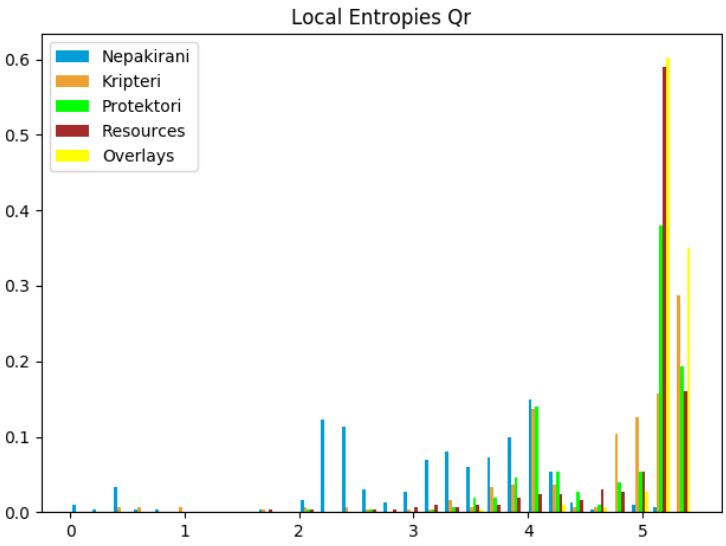


Fig. 1. Gornji kvartil entropija.

Pri evaluaciji modela promatrali smo i osjetljivost na pojedinim klasama packera iako je cilj bio samo binarna klasifikacija. Kod overlay packera smo imali najmanju osjetljivost

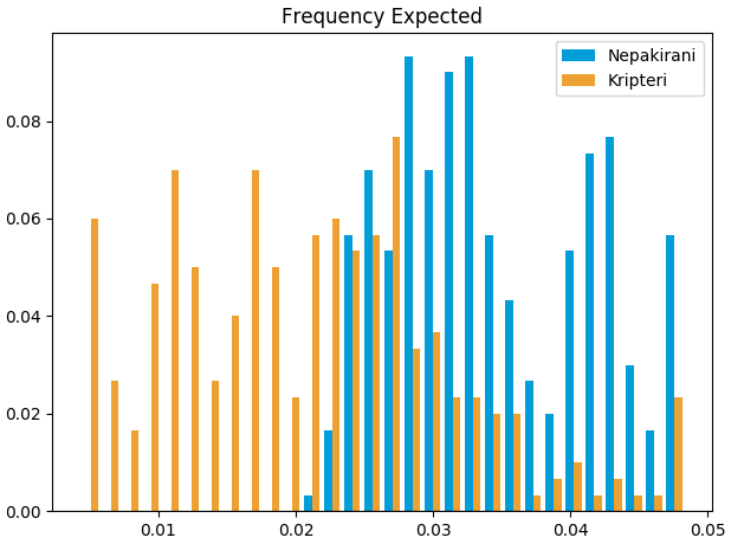


Fig. 2. Očekivana entropija.

te smo se koncentrirali na poboljšanje tog rezultata.
Odabrali smo jednako velik podskup nepakiranih i overlay pakiranih datoteka. Promatrali smo zastavice u tim izvještajima te ukoliko je postojala velika razlika u učestalosti pojavljivanja zastavice u ova dva tipa podataka, tu zastavicu smo dalje promatrali kao potencijalnu značajku. Nakon odabranih zastavica koje su se činile korisne, odredili smo matricu korelacije zastavica te smo neke od njih odbacili jer su bile visoko korelirane.

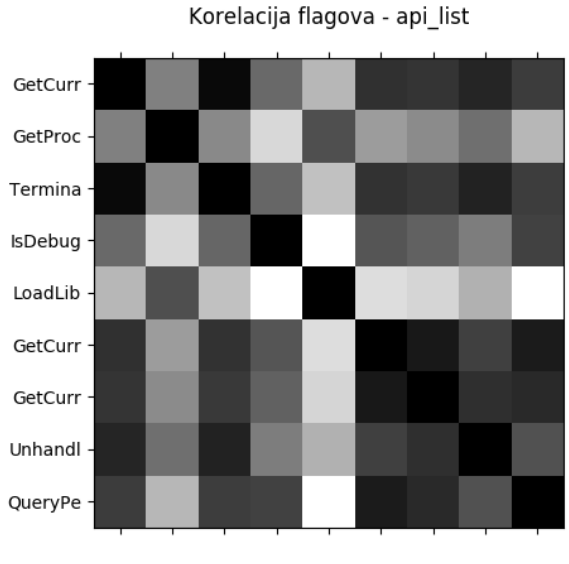


Fig. 3. Korelacijska matrica zastavica u listi apija.

Glavni modeli koje smo trenirali su stabla odlučivanja i

. Za provjeru točnosti modela i usporedbu različitih

hiperparametara i značajki koristili smo 5-struku unakrsnu validaciju. Za potrebe natjecanja, odlučili smo se za Random Forest Classifier kao model koji ćemo koristiti. Razlozi zbog kojih smo odabrali baš to što je to jedna od ansambl metoda strojnog učenja te se u pravilu ne prenauči. Također, više smo se orijentirali na istraživanje pakiranih i nepakiranih datoteka te imamo različite vrste značajki: kategoričke i numeričke, diskretne i kontinuirane varijable u različitim skalama. Random forest dobro radi sa svim vrstama varijabl te nam je to olakšalo pripremu podataka i omogućilo d steknemo što bolje domensko znanje. Grid-searchem sm pretraživali prostor hiperparametara:

- broj stabala
- maksimalni broj značajki koji se gleda u jednom stablu
- maksimalna dubina stabla

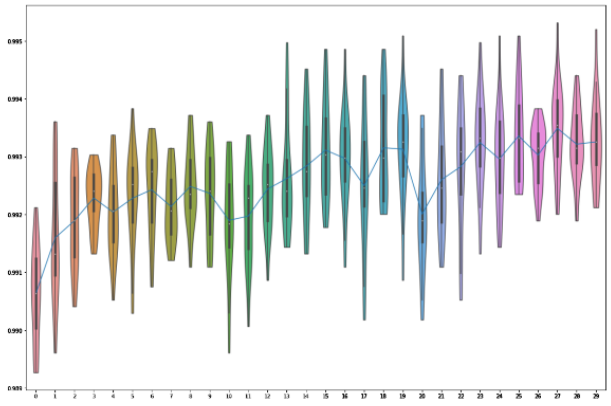

Fig. 4. Odabir broja stabala.

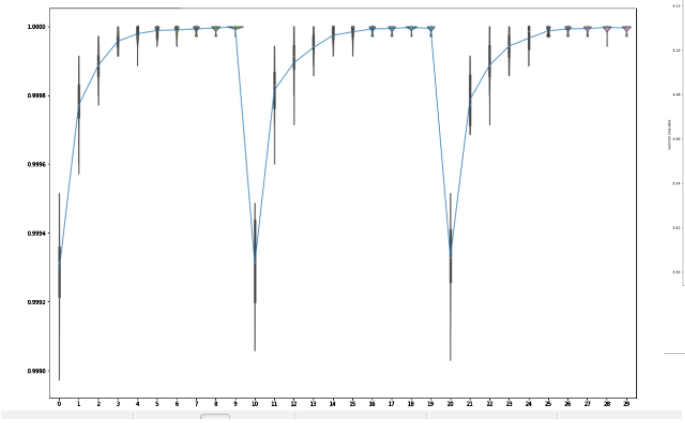

Fig. 5. Odabir broja stabala.

Odlučili smo se za 40 stabala, 17 maksimalnih broj značajki i 30 kao maksimalna dubina stabla. Za usporedbu smo još i isprobali AdaBoost na stablima odluke. Izvukli smo ukupno 83 značajke. Kako Random Forest Classifier već u sebi ima uključeno računanje važnosti značajki, to smo iskoristili za odabir onih koje ćemo koristiti u našem modelu. Odlučili smo se odbaciti 23 značajke koje su se pokazale gotovo nevažne. Uklanjali smo manje važne značajke rekurzivnom eliminacijom značajki te promatrali kad nam počinje padati točnost. Značajke nismo uklanjali jednu po jednu, već u grupama od 5.

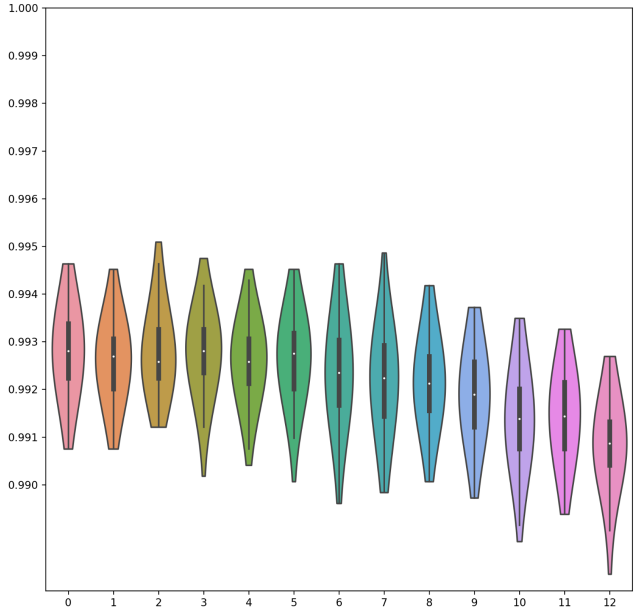


Fig. 6. Rezulati unakrsne 5-struke validacije

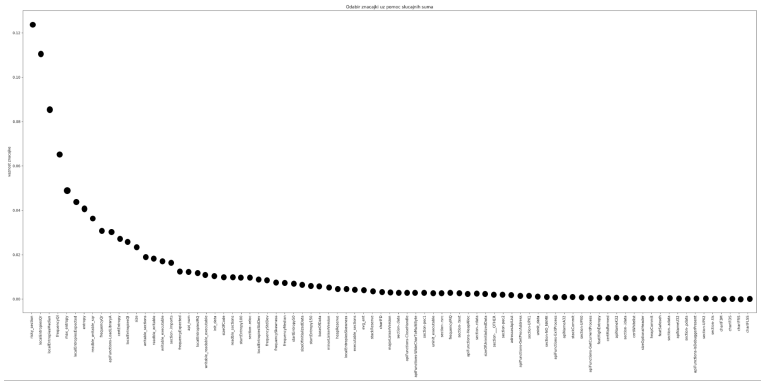

Fig. 7. Važnost značajki.

## IV. Osvrt na druge pristupe

## V. Prikaz rezultata

Za provjeru točnosti modela i usporedbu različitih hiperparametara i značajki koristili smo 5-struku unakrsnu validaciju.

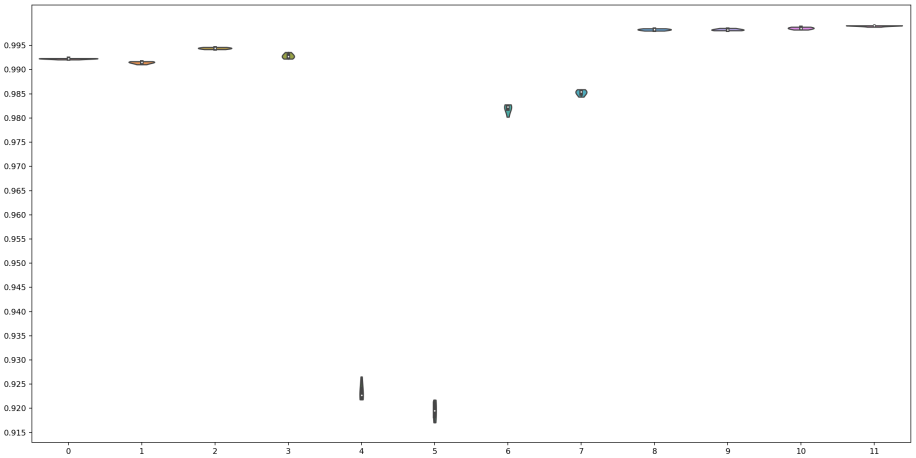


Fig. 8. Rezultati sa i bez korištenja značajki iz binarnih datoteka, redom nepakirani sa/bez, kripteri sa/bez itd.

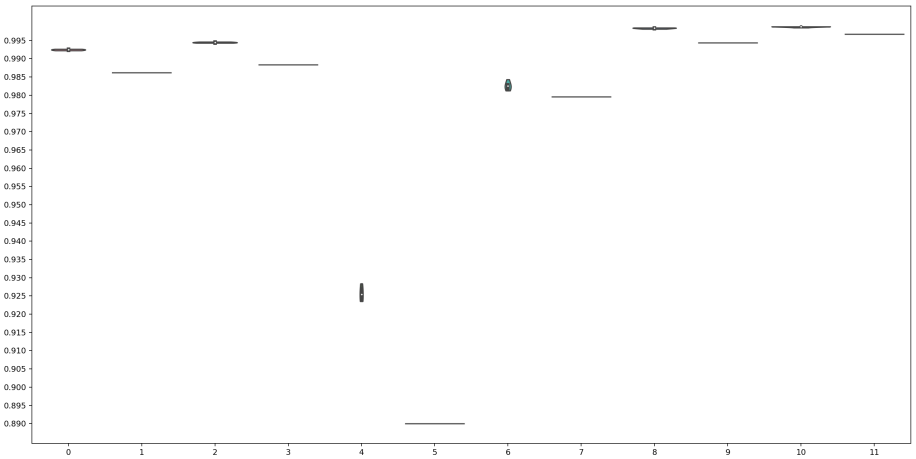


Fig. 9. Osjetljivost po klasama, nepakirani rf/adaboost itd.

| Osjetljivost po klasama | | | | |
|---|---|---|---|---|
| | nepakirani | overlay i kripter | kompresor | protektor |
| osjetljivost trening (unakrsna krosvali-dacija) | 0.9943 | 0.9562 | 0.9984 | 0.9989 |
| rezultati osjetljivosti test | 0.9824 | 0.9312 | 0.9930 | 0.9819 |

## VI. Osvrt na druge pristupe

Ovaj problem bio je u sklopu natjecanja Mozgalo. Skup podataka koji smo dobili je mnogo manji nego što je to u praksi tvrtke ReversingLabs koja je zadala zadatak i zbog toga su rješenja finalista bila dosta slična. Dosta drugačiji pristup od ovdje opisanog bio je više pristup reverznog inženjera, nego nekog tko se bavi strojnim učenjem. Naime, pretpostavka koju je tim imao je da su sve nepakirane datoteke međusobno slične te ukoliko nađu podatke koji ih povezuju, mogu zaključiti i kad je datoteka pakirana. Naš pristup se od ostalih razlikovao po tome što smo osim značajki iz izvještaja koristili i značajke dobivene iz binarnih datoteka. Većina timova koristila je Random Forest Classifier, a naš model je imamo nešto više značajki od ostalih(njih 60) te puno manje stabala.

## VII. Mogući budući nastavak istraživanja

Mogućnost napredovanja postoji u analizi binarnih datoteka. Neke od mogućnosti su promatranje binarnih datoteka te analiza bajtova ili entropija uzastopnih podskupova bajtova kao vremenski niz, npr. LongShortTermMemory mrežom. Jedan od zanimljivijih pristupa koji smo našli u [4] bio je korištenje konvolucijskih neuronskih mreža za detekciju pakiranih datoteka. Naime, ideja je organizirati bajtove u datoteci kao sliku te onda pomoću konvolucijskih filtera ukloniti eventualne manje nepravilnosti u datoteci te na taj način lakše uočiti je li datoteka pakirana ili nije. Pretpostavljamo da bi se bolji rezultati dobili kombinacijom našeg i ovdje navedenih modela.

## Literatura

## References


[1] P. F. F. Guo and T. Chiue, "A study of the packer problem and its solutions," 2008.

[2] ReversingLabs, "Portable executable format, titaniumcore report and packers," 2018. [Online]. Available: https://www.estudent.hr/wp-content/uploads/2018/03/mozgalo_radionica.pdf.

[3] ——, "Mozgalo: Detekcija pakiranih datoteka," 2018. [Online]. Available: https://www.estudent.hr/wp-content/uploads/2018/03/ReversingLabs-zadatak.pdf.

[4] J. Saxe and B. Konstantin, "Deep neural network based malware detection using two dimensional binary," 2015.

[5] W. Tzu-Yen and C. Wu, "Detection of packed executables using support vector machines," 2011.

[6] R. Lyda and J. Hamrock, "Using entropy analysis to find encrypted and packed malware," 2007.