SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
ELEKTROTEHNIČKI FAKULTET

Sveučilišni studij

OTKRIVANJE DOS NAPADA POMOĆU STROJNOG UČENJA

Završni rad

Josip Šokčević

OSIJEK, 2013

SADRŽAJ

1. UVOD

DoS napad (engl. Denial of Service attack) je vrsta napada na računalne sustave kod kojeg dolazi do opterećenje pojedinog elementa unutar mreže te je krajnji rezultat uskraćivanje usluge. Napadi se najčešće izvode DDoS (engl. Distributed DoS), to jest s više klijenata, što omogućavaju virusi i računalni crvi. Prvi napadi dogodili su se već kod samog začetka Interneta slanjem mnogih SYN paketa s krivom IP adresom klijenta [1].

Kod otkrivanja i blokiranja napada potrebno je više razina sigurnosti te ne postoji jedinstveno rješenje s kojim će poslužitelj biti zaštićen. Raniji pokušaji otkrivanja DoS i DDoS napada otkrivali su napade i anomalije u sustavu na četvrtoj razini mrežnog OSI modela [2] [3] [4]. Ovaj rad usmjeren je otkrivanju napada na sedmoj, zadnjoj razini OSI modela te prepuštanje vatrozidu i drugim algoritmima otkrivanje na nižim razinama. Otkrivanje na sedmoj razini ima prednost transparentnosti podataka, no ujedno traži i najviše resursa za obradu podataka. Stoga, važno je imati i zaštitu na nižim razinama.

U drugom poglavlju, DoS napad, opisan je sam napad, što ga karakterizira te koje metode napada postoje. U trećem poglavlju, strojno učenje, opisuje se što je strojno učenje, koje vrste algoritama postoje te koji su algoritmi korišteni u ovom završnom radu. Četvrto poglavlje govori o zaštiti od DoS napada te podešavanju vatrozida tako da se poslužitelj može obraniti od osnovnih i jednostavnih metoda napada. Peto poglavlje govori o otkrivanju DoS napada korištenjem strojnog učenja pomoću Naive Bayes i One-Class SVM algoritma. Također je demonstriran napad te su rezultati napada predočeni grafički i tablicom.

1.1 Zadatak završnog rada

Zadatak završnog rada jeste otkriti napad uskraćivanja usluge na poslužitelja koristeći strojno učenje. Potrebno je proučiti TCP/IP, točnije HTTP protokol te šablonu DoS napada – učestalost zahtjeva, značajke napada i drugo. Zatim je potrebno analizirati takve informacije te pronaći svojstva s kojima bi računalo moglo otkriti sam DoS napad. Nakon što su svojstva pronađena treba ih provjeriti testiranjem, to jest simulacijom DoS napada i rekonstrukcijom regularnih zahtjeva korisnika.

2. DOS NAPAD

DoS je vrsta napada na računalne sustave pri kojem dolazi do opterećenja poslužitelja, preusmjernika ili čak primatelja te je krajni cilj napada uskraćivanje usluge drugim korisnicima [5]. Razlozi DoS napada mogu biti razni, no cilj je ili privremeno ili trajno onesposobiti, to jest isključit poslužitelja s Internet mreže. Najraširenij način DOS napada je slanjem mnogostrukih zahtjeva prema poslužitelju preko više klijenata kako bi rezultat napada bio što bolji. Takav se napad zove DDoS (engl. Distributed Denial Of Service).

Napadači u pravilu ciljaju na stranice velikih korporacija (banke, kreditne kuće, osiguravajuće kuće), pa čak i glavne DNS (engl. Domain Name System) poslužitelje. Napadi mogu biti i političke naravi te iskaz neslaganja. Primjer je napad grupe Anonymous 2010. godine. Richard Stallman, kreator GNU te aktivist, komentirao je da se DDoS napad može smatrati virtualnim prosvjedom [6].

Prema definiciji US-CERT [7] simptomi DoS napada su:

- usporen pristup poslužitelju
- nedostupnost određenih dijelova servisa
- nemogućnost pristupanju drugim Internet stranicama
- povećan broj neželjene pošte
- prekinuta žičana ili bežična Internet mreža

2.1 Metode napada

Kao što je opisano u prethodnom poglavlju, cilj napada je uskraćivanje usluge drugim, legitimnim korisnicima. Postoje dvije glavne vrste DoS napada – DoS napad pri kojem se servis ruši te DoS napad pri kojem servis postaje zagušen. Ispod su navedene četiri glavne metode napada.

ICMP (engl. Internet Control Message Protocol) poplave

Za ICMP poplave (engl. ICMP flood) može se reći da su prvi oblici DoS napada na računalne sustave. Danas su rijetkost i većina se sustava može obraniti od njih. U teoriji, napad se izvodi tako da se pošalje veliki broj ICMP paketa koristeći emitiranje (engl. broadcast). ICMP paket sadrži izvornu IP adresu (engl. source IP address) od žrtve. Stoga sva računala koja prime takav

paket, odgovorit će žrtvinoj IP adresi umjesto napadaču. Uz kombinaciju s velikim paketima, ishod može biti zagušenje mreže ili zagušenje servera.

SYN poplava

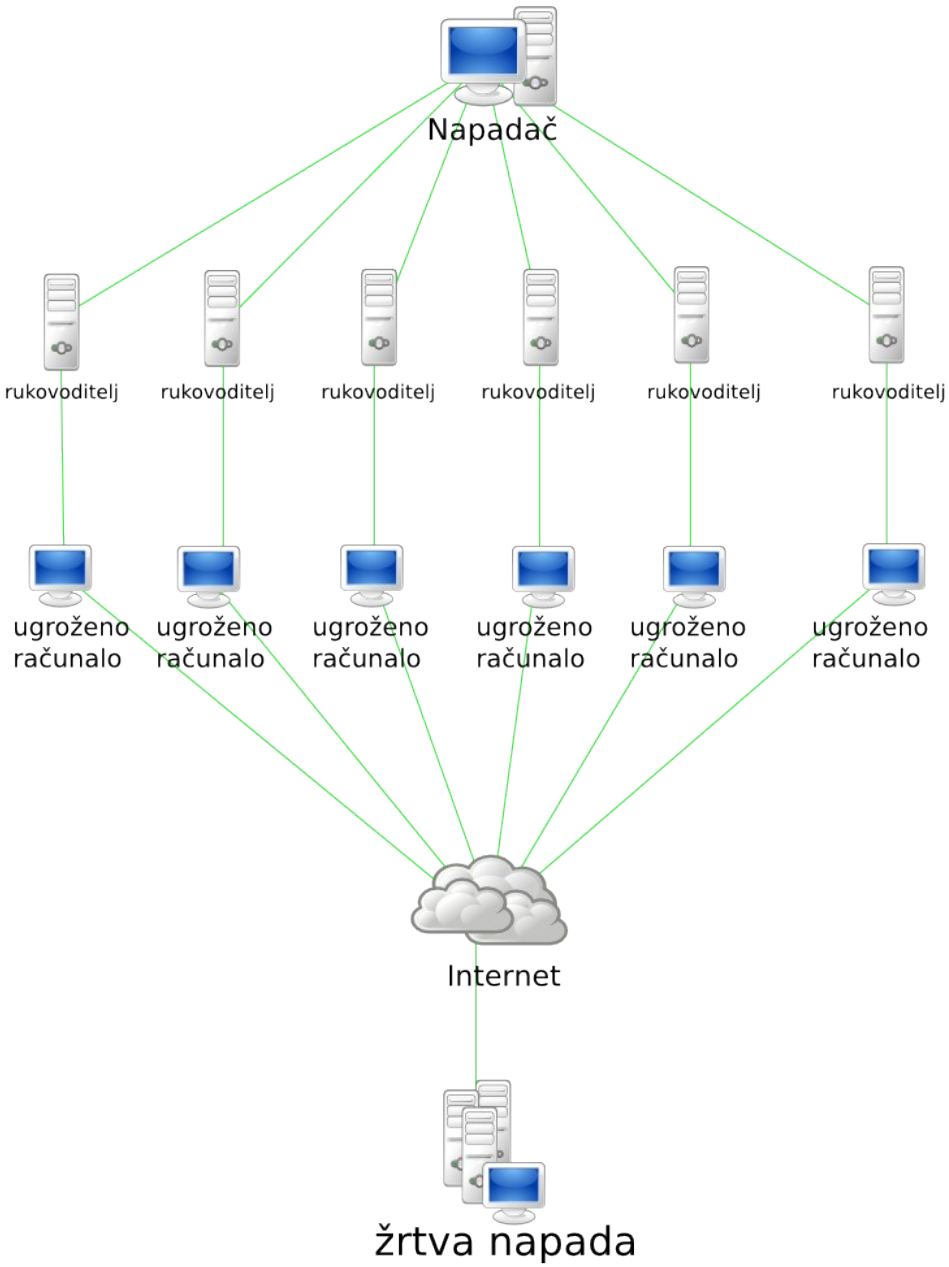
SYN poplava je tip DoS napada pri kojem se šalje serija SYN poruka bez ACK poruke. SYN poruka je sinkronizacijska mrežna poruka koja se šalje pri prvom TCP/IP zahtjevu. Nakon što primatelj dobije SYN poruku, odgovara sa SYN-ACK porukom te očekuje od pošiljatelja ACK poruku. Kako napadač ne šalje ACK nazad, server drži kanal otvoren određeno vrijeme (definirano u aplikaciji). Ukoliko se red (engl. queue) popuni, novi, legitimni zahtjevi se ne mogu ispuniti.

Aplikacijski DoS napadi

Aplikacijski DoS napadi trenutno su najčešće vrste napada na sustav. Za razliku od prethodno navedenih napada koji se izvode na četvrtom sloju OSI mrežnog modela, ova se vrsta napada izvodi na zadnjem, sedmom sloju. Kako bi se ostvario glavni cilj, onesposobljavanje poslužitelja, koristi se loš dizajn aplikacije koja je dostupna preko Interneta. Poslužitelji se najčešće onesposobljavaju tako da se popune radna memorija, popuni tvrdi disk ili da se preoptereti procesor.

2.2 Ditribuirani DoS napadi

U današnje je vrijeme sve više DDoS napada koji mogu napraviti mnogo više štete nego pojedinačni DoS napad. Napad se razlikuje od DoS napada po broju računala koji sudjeluju u napadu. Također, prema slici 2.1, napadač se ne može izravno otkriti jer ne radi izravne zahtjeve prema žrtvi već preko posrednika.



Sl. 2.1. *Shema DDoS napadač - wikipedia.org*

3. STROJNO UČENJE

Strojno je učenje dio umjetne inteligencije (engl. Artificial Intelligence) koji se bavi izgradnjom i proučavanjem sustava koji mogu učiti iz podataka. Stoga se može reći da je strojno učenje dio računarne znanosti koja daje mogućnost računalima da uče i donose zaključke bez da ih eksplicitno definira programer. Također je važno napomenuti da strojno učenje nije egzaktna znanost.

Strojno učenje može se podijeliti na tipove algoritama te na same algoritme. Tipovi algoritama definiraju se po rezultatu ishoda algoritma te po tipu ulaznih varijabli unutar treniranja.

3.1 Tipovi algoritama

Nadzirani algoritam (engl. supervised algoritm) je algoritam koji je učen s označenim primjerima, to jest zna se točno koji je rezultat ulaza. Ovakav tip algoritma pokušava generalizirati ulaz te izlaz algoritama. Takav generaliziran ulaz i izlaz korišten je za dobivanje rezultata kod do tada nepoznatih ulaznih varijabli.

Nenadzirani algoritam (engl. unsupervised algoritm) je onaj algoritam koji je učen s neoznačenim primjerima, to jest nije poznat rezultat primjera. Cilj algoritma je otkrivanje strukture podataka te ne generalizira rezultat ulaza i izlaza, kao što je slučaj kod nadziranog učenja.

Učenje pojačavanjem (engl. reinforcement learning) udružuje nadzorni i nenadzirani algoritam, to jest koristi označene (izrazito malo) i neoznačene primjere za treniranje algoritma. Svaki rezultat algoritma može biti ocijenjen te se takva ocjena vraća nazad u algoritam kako bi se mogla usavršiti.

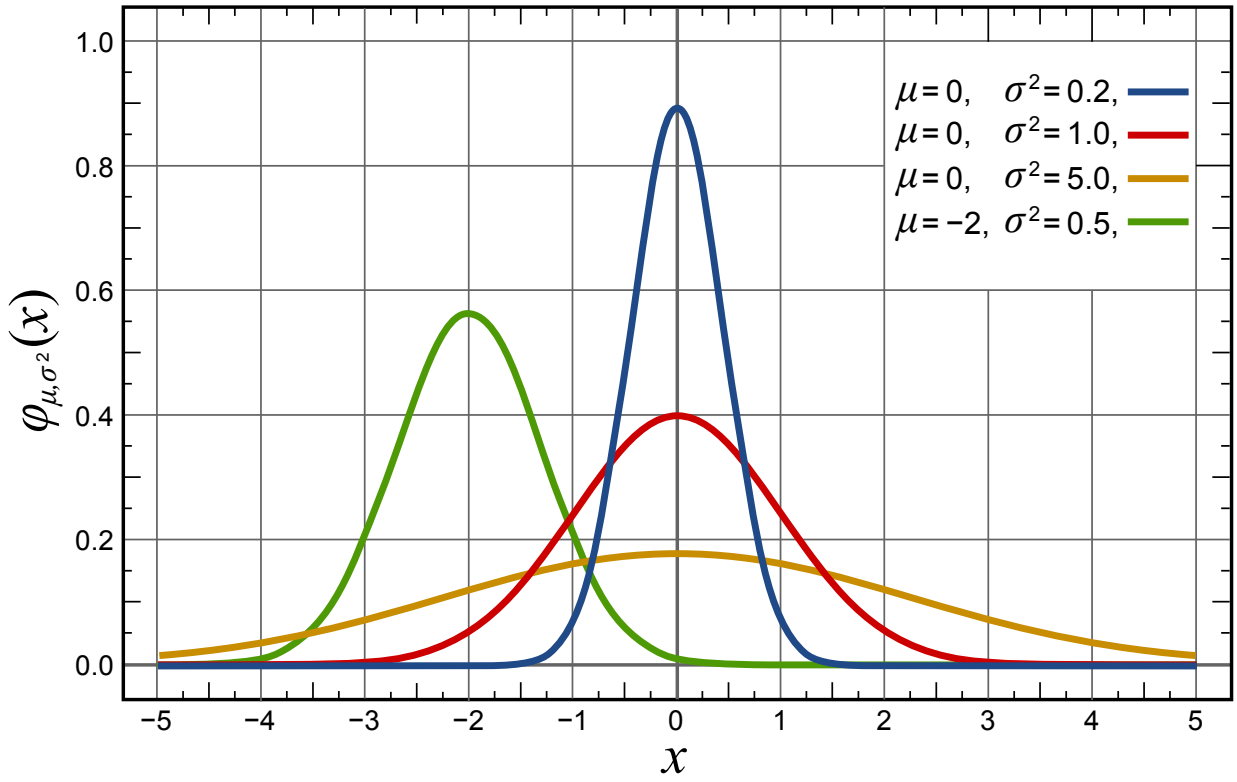
3.2 Algoritmi

Naive Bayes algoritam korišten u ovom radu koristi normalnu raspodjelu. Normalna raspodjela ili Gaussovu krivulja jedna je od najkorištenijih raspodjela kod otkrivanja anomalija unutar sustava [8]. Funkcija vjerojatnosti distribucije dana je formulom 3-1 te grafički prema slici 3.1.

$$P(x;\mu,\sigma^2)=\frac{1}{\sigma\sqrt{2\pi}}e^{\left(-\frac{x-\mu^2}{2\sigma^2}\right)} \tag{3-1}$$

Gdje je:

- x – ulazna varijabla
- μ – očekivana vrijednost
- σ – standardna devijacija



Sl. 3.1. *Normalna raspodjela – wikipedia.org*

Kako klasifikator Gauassian Naive Bayes smatra sve ulaze nezavisne o drugim ulazima, formulu je moguće napisati za svaki ulaz kao prema jednadžbi 3-2.

$$P(x_i; y) = \frac{1}{\sigma_y \sqrt{2\pi}} e^{\left(-\frac{x_i - \mu_y^2}{2\sigma_y^2}\right)} \tag{3-2}$$

Gdje je

- P(x ; y) – vjerojatnost pojedinog ulaza
- x_i – pojedina vrijednost ulaza

- μ_y – očekivana vrijednost ulaza
- σ_y – standardna devijacija ulaza

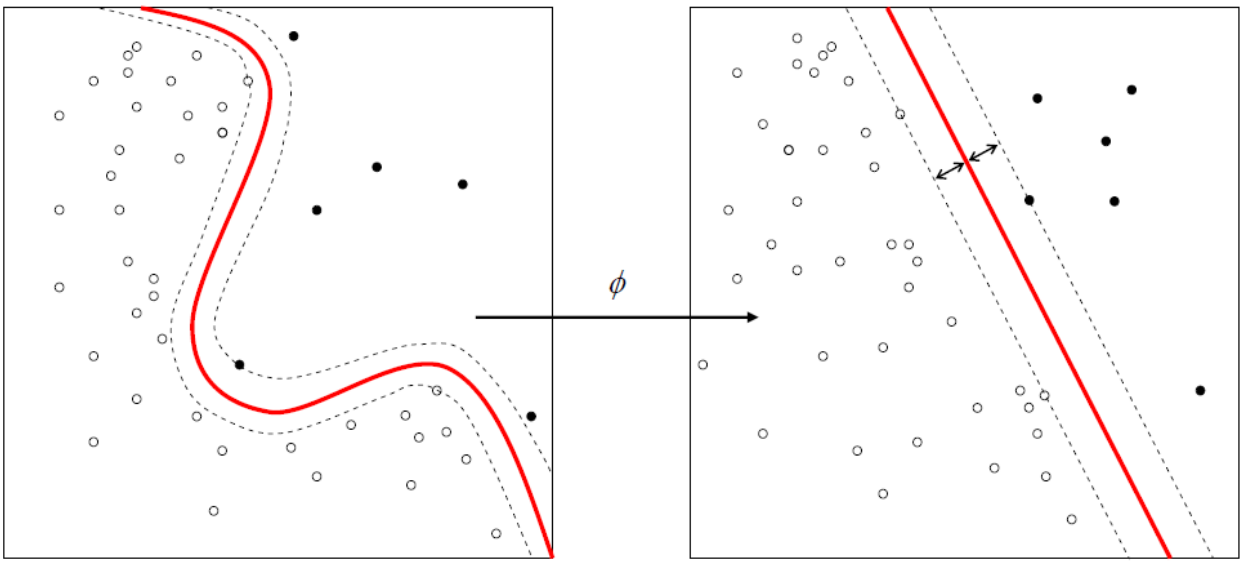
Formula vjerojatnosti cijelog modela dana je jednadžbom 3-3.

$$P(x; y) = \prod_{i=1}^{n} P(x_i; y) \tag{3-3}$$

Gdje je

- P(x; y) – vjerojatnost obzirom na sve ulaze
- $P(x_i; y)$ – vjerojatnost pojedinog ulaza

Algoritam One-Class SVM (engl. Support Vector Machine) tip je nenadziranog algoritma koji grupira podatke u dvije skupine – unutar i van limita, to jest je li nova vrijednost jednaka setu za treniranje ili odstupa od okvira. SVM koristi kernel koji omogućuje nelinearnu klasifikaciju (Slika 3.2). U ovom radu koristi se RBF kernel (engl . Gaussian radial basis function kernel).



Sl. 3.2. *RBF kernel – linearizacija – wikipedia.org*

4. ZAŠTITA

Prije samog otkrivanja DoS napada potrebno je na aplikacijskoj razini napraviti preliminarnu zaštitu kojom se mogu blokirati jednostavni i primitivni napadi. Preliminarna zaštita omogućit će bržu i jeftiniju zaštitu nego na aplikacijskoj zaštita jer koristi manje resursa.

4.1 Zaštita od DoS napada

Bilo koje računalo koje je dostupno na Internetu mora biti zaštićeno od DoS napada. Najčešći način zaštite je korištenjem vatrozida (engl. firewall) te se njim može zaustaviti većina napada na transportnom sloju, kao što je SYN poplava ili ICMP poplava.

Usmjernici (engl. switch) mogu imati ograničenje učestalosti paketa (engl. rate limit) i kontrolu pristupa (engl. ACL – Access Control List). Prvi ovakvoj vrsti zaštite napadač je ograničen po broju veza prema serveru.

Najinteligentniji sustav otkrivanja DoS napada je na aplikacijskom sloju te pri tom dolazi do detaljne analize svih paketa te klasificiranju istih u određene grupe. Aplikacijski sloj može biti pisan u bilo kojem programsku jeziku te može koristiti bilo koji algoritam. Ovakvi inteligentni sustavi najčešće se vežu za vatrozid nakon što otkriju napad.

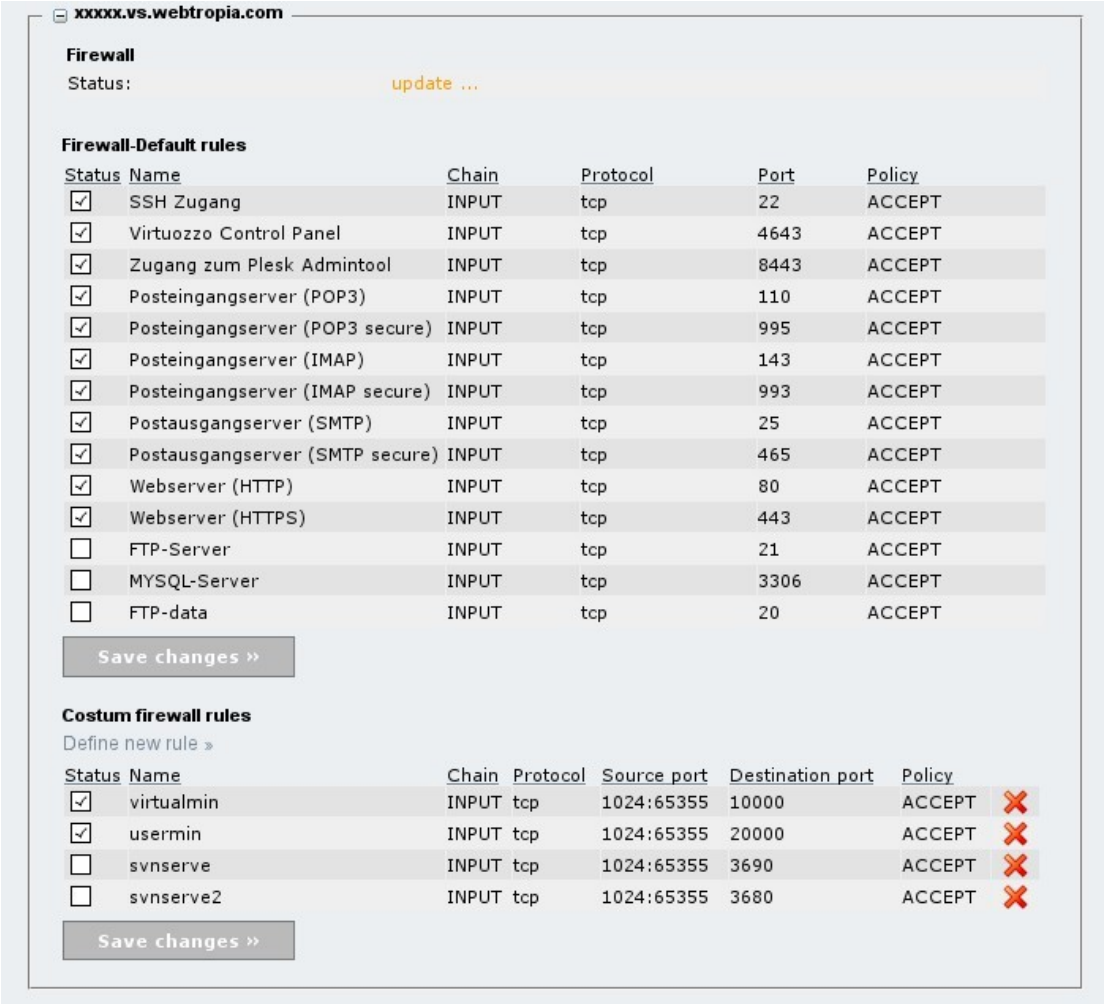
4.2 Vatrozid

Vatrozid pruža osnovnu vrstu zaštite protiv DoS napada. Stoga je prvi korak podesiti vatrozid tako da propušta samo određene portove (engl. port). Većina današnjih pružatelja najma servera (engl. hosting) ima fizički vatrozid kojeg korisnici mogu podešavati preko internetskog preglednika.

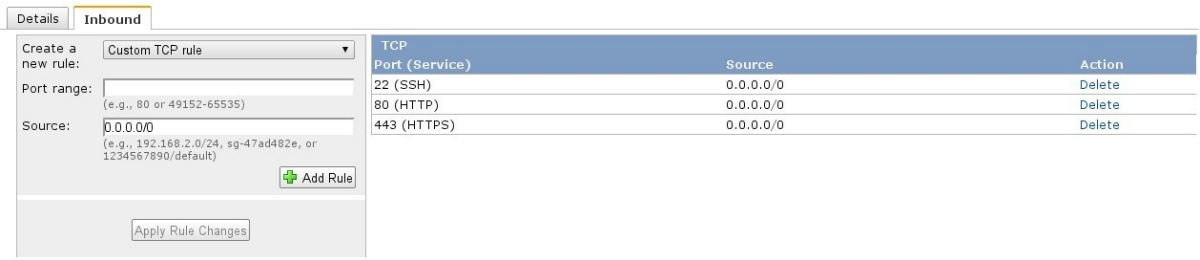
Vatrozid je potrebno podesiti tako da se omogući pristup samo nužnim dijelovima. Na primjer, ukoliko se radi o web stranici s bazom podataka te pri tom web aplikacija komunicira preko porta, a baza podataka na portu 3306, potrebno je propustiti samo port 80 na vatrozidu. Web aplikacija će moći komunicirati s bazom podataka jer se nalazi na istom računalu, stoga nije potrebno ići preko vatrozida. Ovakvim podešavanjem vatrozida napadač neće imati mogućnost komunicirati s bazom podataka te se smanjuje mogućnost DoS.

Isto tako, ako se na serveru nalazi FTP ili SSH server, poželjno je i isključiti takve portove te omogućiti ih samo za željenu IP adresu. Nažalost, to može biti problematično jer se IP adrese mijenjaju te je svaki put potrebno promijeniti postavke vatrozida.

Primjeri podešavanja vatrozida preko internetskog preglednika dani su na slikama 4.1 i 4.2.



Sl. 4.1. *Primjer vatrozida upravljan preko internetskog preglednika*



Sl. 4.2. *Primjer vatrozida na oblaku AWS*

Osim fizičkog vatrozida, moguće je koristiti unix vatrozid iptables koji može biti na zasebnom poslužitelju ili 'pak na istom gdje se nalazi internetska aplikacija. Vatrozid iptables može se, te je i poželjno, podesiti preko komandne linije.

Primjer dobre konfiguracije vatrozida dan je u kôdu 4.1. Vatrozid se podešava tako da odbija sve

dolazne zahtjeve te zahtjeve za prosljeđivanje. Zatim, omogućava se port 80 (HTTP) te port 443 (HTTPS) tako da korisnici web aplikacije mogu komunicirati s poslužiteljem. Port 22 (SSH) omogućen je samo unutar lokalne mreže, 192.168.1.0/24. Port 22 dopušten je unutar intervala jer nije potrebno, niti je poželjno, da svi imaju pristup poslužitelju. Ukoliko je potrebno, moguće je dodati zasebnu IP adresu unutar postojeće list dozvoljenih adresa.

```
#!/bin/bash
# obrisi sve postojece postavke
iptables --flush
# ukloni sve zahtjeve
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT DROP
# dopusti svima port 80 (web)
iptables -A INPUT -i eth0 -p tcp --dport 80 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o eth0 -p tcp --sport 80 -m state --state ESTABLISHED -j ACCEPT
# dopusti port 22 samo IP unutar intervala 192.168.1.1 - 192.168.1.254
iptables -A INPUT -i eth0 -p tcp -s 192.168.1.0/24 --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o eth0 -p tcp --sport 22 -m state --state ESTABLISHED -j ACCEPT
```

Kod 4.1. *Podešavanje iptables vatrozida*

Provjera ispravnosti vatrozida može se testirati na više načina, a jedan je od njih korištenjem nmap alata. Primjer nmap skeniranja dan je unutar kôda 4.2.

```
nmap -p 80,443,3306 IP_ADRESA

Starting Nmap 6.40 ( http://nmap.org ) at 2013-07-17 10:58 CEST
Nmap scan report for IP_ADRESA
Host is up (0.054s latency).
PORT     STATE    SERVICE
22/tcp   open     ssh
80/tcp   open     http
443/tcp  open     https
3306/tcp filtered mysql

Nmap done: 1 IP address (1 host up) scanned in 1.50 seconds
```

Kod 4.2. *nmap skeniranje*

Prilikom dodavanja pravila unutar iptables vatrozida, potrebno je pripaziti ukoliko se korisnik spaja preko udaljene lokacije. Naime, ukoliko se ta IP adresa ne nalazi u popisu dozvoljenih, korisnik može izgubiti pravo pristupa te će biti potrebna fizička intervencija na samom poslužitelju kako bi korisnik mogao dobiti prava opet.

5. OTKRIVANJE DOS NAPADA POMOĆU STROJNOG UČENJA

Otkrivanje DoS napada, ukoliko je vatrozid dobro podešen, najefikasnije je na aplikacijskom sloju. Unutar tog sloja mogu se izvući korisni podatci koji će biti temeljni ulaz unutar algoritma strojnog učenja. Iako se svi podatci unutar aplikacijskog sloja mogu mijenjati, pa čak i unutar četvrtog, transportnog sloja OSI modela, računalo treba otkriti taku vrstu napada.

Efektivno otkrivanje DoS napada je u traženju anomalija, to jest zahtjeva koji odudaraju od uobičajenih zahtjeva korisnika. Na primjer, napadač može češće izvoditi POST metodu, može imati drugačiji HTTP agent. Ukoliko se radi o DDoS napadu, može se izdvajati neuobičajeni broj zahtjeva sa specifične geografske lokacije.

5.1 Odabiranje varijabli za daljnju obradu

Kako bi bilo koji algoritam strojnog učenja mogao raditi, potrebno je odabrati ulazne varijable u sustav. Prilikom odabira ulaza važno je pronaći ključne ulaze – premalo ulaza rezultira nepreciznim rezultatom, dok previše ulaza troši više računalnih resursa. Stoga je potrebno pomnom izabrati ulaze kako bi bili unikatni, dovoljni za dobre rezultate te optimalni glede korištenja resursa.

Korištenje aplikacijskog sloja omogućava korištenje više varijabilnih ulaza unutar algoritma strojnog učenja te samim time preciznije i točnije klasificiranje mrežnih zahtjeva. Poželjno je iskoristiti podatke koji već postoje, stoga će se podatci iščitavati iz log datoteke (apache format).

Ulazne varijable za daljnju obradu koje se koriste unutar ovog rada jesu:

1. IP adresa
2. Otisak računala (engl. fingerprint)
3. Geografska lokacija korisnika
4. URL putanja (engl. path)
5. Metoda HTTP zahtjeva
6. Referent (engl. referrer)

Šest varijabli dovoljno je da se može opisati korisnika i njegovo ponašanje te pronaći anomalije, a da opet imamo dovoljno brzu analizu.

Svako računalo ima otisak koji ostavlja prilikom HTTP zahtjeva. Otisak se može dobiti

korištenjem HTTP zaglavlja kao što su HTTP korisnički agent (engl. User-Agent) i preferirani jezik, IP adresa s otiskom računala omogućit će da otkrijemo koliko često određena računala pristupaju poslužitelju te koliko računala ima pod zajedničkom IP adresom. Geografskom lokacijom moguće je pronaći anomalije u posjetima određene regije. Na primjer, ukoliko je web poslužitelj namijenjen hrvatskom tržištu te u jednom trenutku dolazi ogroman broj zahtjeva iz Kine, moguće je zaključiti da se radi o napadu. Naravno, potrebno je napraviti detaljnu analizu podskupine, no takva je analiza daleko učinkovitija nego nad cijelim setom podataka.

Korištenjem ulaza kao što su URL, metode te referenti, moguće je napraviti tipičan profil korisnika. Ukoliko korisnik radi previše zahtjeva s metodom POST, postoji velika mogućnost da se radi o DoS napadu ili čak neželjenom sadržaju (jer POST metoda znači kreiranje sadržaja).

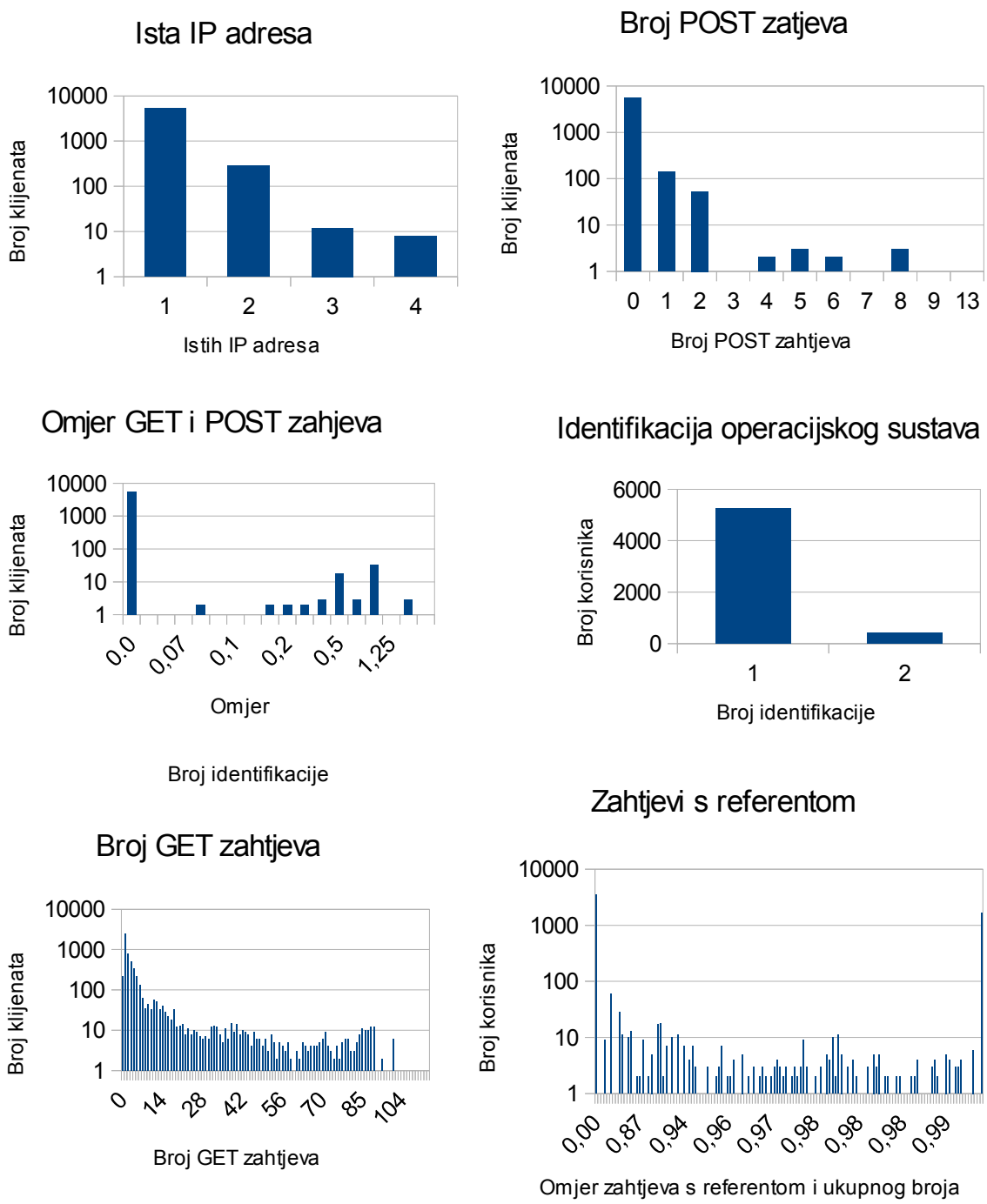
5.2 Grupiranje

Kako bi se mogao efikasno opisati svaki korisnik, potrebno je grupirati zahtjeve. Grupirati zahtjeve možemo koristeći otisak računala te IP adresu, grupiranje po geografskoj lokaciji. Važno je napomenuti da grupiranje po prosječnom trajanju zahtjeva također može doprinijeti otkrivanju napada, no takav podatak ne postoji unutar log datoteke te je stoga izostavljen.

Grupiranje po otisku računala i IP adresi izvedeno je spajanjem (engl. aggregate) podataka u podesivom intervalu. Kao rezultat spajanja dobiveni su vektori ulaznih varijabli:

- Broj istih IP adresa s drugačijim otiskom računala
- Broj GET zahtjeva
- Broj POST zahtjeva
- Broj ostalih zahtjeva (HEAD, PUT, DELETE)
- Identifikacija operacijskog sustava (nikakva, tip, verzija)
- Identifikacija internetskog preglednika (nikakva, tip, verzija)
- Omjer GET zahtjeva
- Geografska dužina i širina

Rezultat grupiranja po otisku računala i IP adresi za dani set za treniranje (5681 podatak) dan je slikom 5.1 te tablicom 5.1.



Sl 5.1. *Rezultat grupiranja podataka*

Naziv	Minimalna vrijednost	Maksimalna vrijednost	Srednja vrijednost	Standardna devijacija
TSIP (ista IP adresa)	1	4	1.058	0.26
POST (broj POST zahtjeva)	0	13	0.06	0.423
GET (broj GET zahtjeva)	0	150	7.141	16.357
OTHER (broj ostalih zahtjeva)	0	7	0.02	0.193
OS (broj identifikacije)	1	2	1.569	0.495
BROWSER (broj identifikacije)	1	2	1.074	0.262
RATIOPG (omjer POST I GET)	0	5	0.01	0.117

Tablica 5.1. *Rezultat grupiranja po otisku i IP*

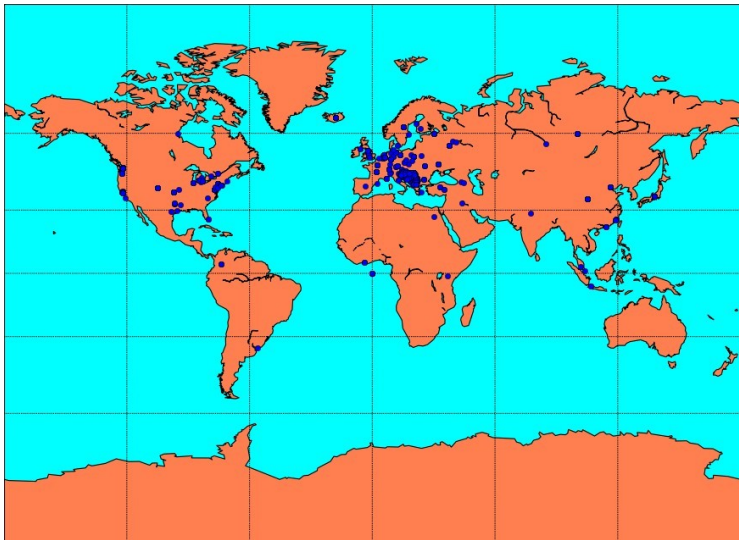
Grupiranje po geografskoj lokaciji koristi K-Means klaster (engl. K-means cluster). Cilj K-means je kreiranje proizvoljnog broja centroida te grupiranje podataka oko istih. Kao ulaz u k-means klaster koristit će se podatci grupirani po IP adresi i otisku računala.

Kako bi se dobio podatak o geografskoj lokaciji potrebno je koristiti bazu podataka relacije IP adrese te geografske lokacije. Za bazu je izabrana GeoLite City od firme MaxMind [9]. Rezultat grupiranja je sličan kao i u prethodnom slučaju:

- Broj različitih IP adresa
- Broj različitih računalih otiska
- Broj GET zahtjeva
- Broj POST zahtjeva
- Broj ostalih zahtjeva (HEAD, PUT, DELETE)
- Broj zahtjeva s identifikacijom operacijskog sustava

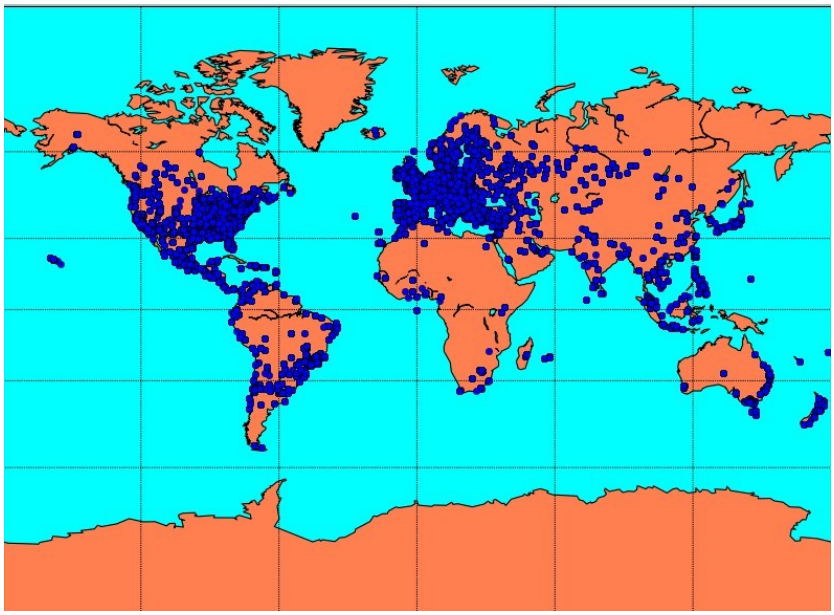
- Broj zahtjeva s identifikacijom internetskog preglednika
- Omjer GET i POST zahtjeva

Rezultati posjetitelja jednog regionalnog portala dani su slikom 5.2. Iz slike se može zaključiti da je većina posjetitelja upravo iz Hrvatske i okolnih zemalja te da jako mali broj posjetitelja dolazi izvan Europe.



Sl. 5.2. *Prikaz posjetitelja regionalnog portala*

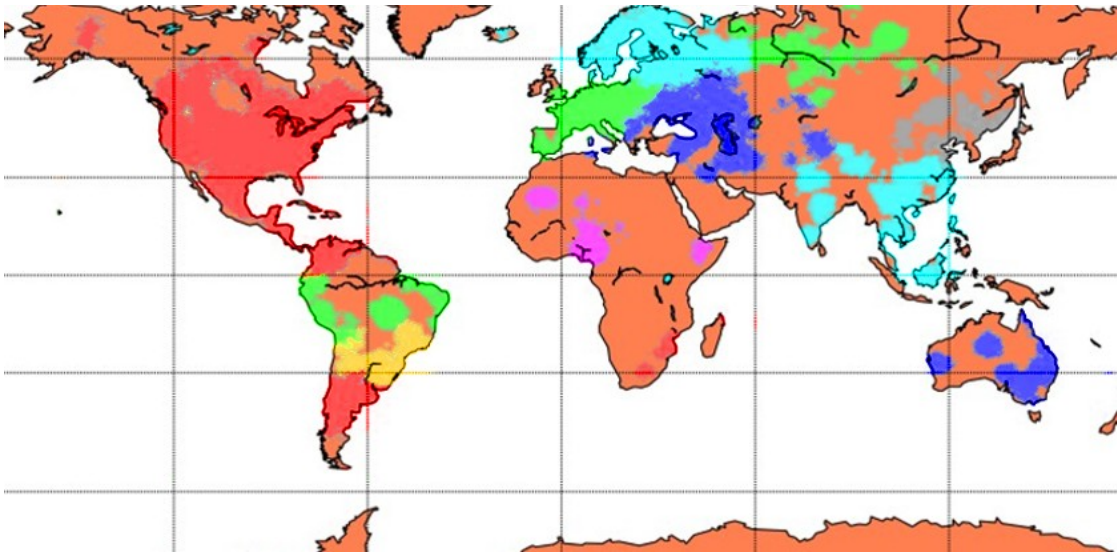
Rezultati jednog internacionalnog portala slikom 5.3 te se iz nje može zaključiti da je distribucija posjetitelja uvelike raspršena. Najviše posjetitelja dolazi iz Europe te Sjedinjenih Američkih Država, ali je i znatan broj onih iz južne Azije, Južne Amerike te Australije.



Sl. 5.3. *Prikaz posjetitelja internacionalnog portala*

U prikazu regionalnog portala može se vidjeti da je većina posjeta iz područja Hrvatske i zemalja bivše Jugoslavije. Ukoliko dođe do velikog broja posjetitelja iz Azije te istovremeno šalju veći broj zahtjeva, može se zaključiti da se radi o napadu.

Napad se može otkrivati i pri internacionalnim servisima. Kao što se može vidjeti na slici 5.3, najviše posjeta dolazi iz Europe i Sjedinjenih Američkih Država. Primjer grupiranja je dan na slici 5.4 gdje su posjetitelji podijeljeni u 14 grupa.



Sl 5.4. *Grupiranje posjetitelja pomoću x-means (Weka), varijanta K-Means*

Ukoliko pojedina grupa počne kreirati veći broj zahtjeva te su poslužitelji pod opterećenjem, moguće je isključiti cijelu regiju ili detaljnu analizu napraviti nad manjom grupom, što je brže, efikasnije i jeftinije nego nad cijelom grupom. Također, minimalan i maksimalan broj grupa može se regulirati unutar samog K-Means algoritam. Nažalost, zbog tehničkih nemogućnosti, otkrivanje DoS napada pomoću geografske lokacije nije moguće testirati.

5.3 Izlazne varijable

Kod otkrivanja DoS napada postoje dvije izrazite izlazne varijable: zahtjev je napad ili nije. Međutim, kako bi se uklonili eventualni lažni alarmi (engl. false positive), dodaje se i treća izlazna varijabla koja opisuje stanje nesigurnosti algoritma. Razlog zašto nije loše dodati ovakvu treću varijablu jeste u tome što ipak želimo legitimnom korisnicima omogućiti uslugu. Također, takvu informaciju možemo iskoristiti kasnije za treniranje mreže kako bi se takav slučaj ne bi ponovio (učenje pojačavanjem). Tu je naravno poželjno pripaziti te napraviti dobru zaštitu kako napadač ne bi dozvolio svoje napade.

5.4 Obrada Apache Log datoteke

Kada posjetitelj posjeti HTTP poslužitelja, HTTP poslužitelj u pravilu zapisuje informaciju o posjetitelju. Najčešće korišten način zapisa jeste Apache Log format. Iako se format zove po Apache HTTP poslužitelju, ustaljen je kao standard. Mnogi brojni HTTP poslužitelji zapisuju ili imaju mogućnost zapisivanja formata u tom obliku kao što su nginx i Varnish.

Apache Log zapis sastoji se od IP adrese posjetitelja, korisnika (ukoliko je korištena HTTP autorizacija), datum i vrijeme posjeta, zahtjev, status rezultata, veličina odgovora, referent, HTTP agent. Pojedina polja odvojena su razmakom.

```
173.236.154.XX - - [18/Aug/2013:06:18:26 +0200] "POST /wp-login.php
HTTP/1.1" 200 1917 "http://domain.com/wp-login.php" "Mozilla/5.0
(Windows NT 6.1; WOW64; rv:18.0) Gecko/20100101 Firefox/18.0"
```

Zahtjev unutar zapisa jeste prva linija zahtjeva prema HTTP poslužitelju te se sastoji od metode zahtjeva, putanje te protokola. Metoda zahtjeva može biti GET, POST, HEAD te manje korištene DELETE, PUSH, PUT. Putanja može, ali ne mora sadržavati nastavak (engl .extension).

Status zahtjeva odnosi se na status HTTP odgovora na korisničkov zahtjev. Najčešći statusi jesu 200 (ispravan), 403 (neautorizirano), 404 (nije pronađeno), 500 (interna greška – najčešće greška u aplikaciji) [10].

Aplikacija koja obrađuje zapise zove se *parse_log.py*. Rezultat obrade jeste vektor sa šest elemenata: otisak računala, IP adresa, metoda zahtjeva, agent, status rezultata te referent.

Svakih određeni vremenski period, koji je definiran prilikom pokretanja skripte parametrom *-f*, a koji ima početnu vrijednost 10 sekundi, pokreće se grupiranje rezultata. Prikupljeni se rezultati šalju na obradu algoritmu za grupiranje podataka te se zatim rezultat obrade šalje klasi koja će ispisati rezultate (npr. za ručnu provjeru), spremiti u datoteku (npr. za naknadno učenje algoritma) ili koristiti za analizu DoS napada. Ukoliko se radi o analizi DoS napada, poželjno je dodati parametar -t pri pokretanju *parse_log.py* aplikacije kako bi se nove vrijednosti stalno isčitavale iz log datoteke (engl. log tail).

Grupiranje, kao što je napisano u prethodnom paragrafu, izvodi se svaki određeni vremenski period. Sve prikupljene informacije grupiraju se po otisku računala. Klasa koja je zadužena za grupiranje jeste *DataAggregator* te se nalazi u datoteci *aggregation.py*. Rezultat grupiranja nalazi se unutar metode *results* te je njen rezultat matrica. Redak matrice predstavlja korisnika s jednim otiskom, a stupci su redom: broj istih IP adresa s drugim otiskom, broj POST zahtjeva,

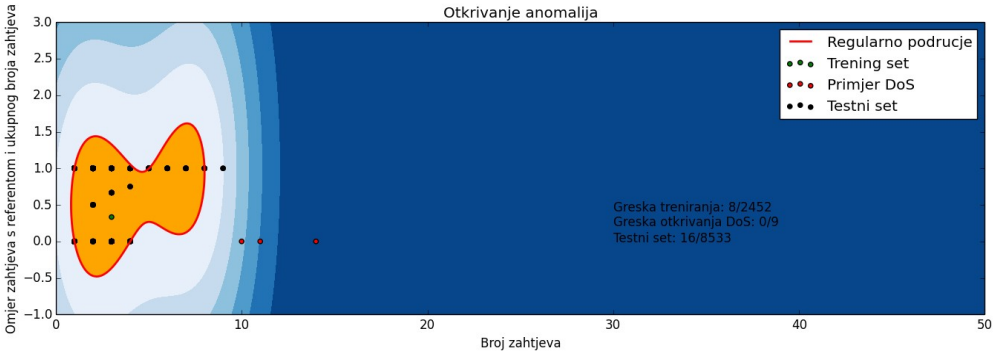
broj GET zahtjeva, broj ostalih zahtjeva, identifikacija preglednika, identifikacija operacijskog sustava, omjer POST i GET, omjer zahtjeva s referentom i ukupnog broja zahtjeva, geografska dužina, geografska širina. Za dobivanje geografske dužine korištena je GeoIP biblioteka od firme MaxMind, a za operacijski sustav i verziju te Internet preglednik i verziju user_agents bibiloteka [11].

Nakon grupacije rezultat se može poslati na jednu od klasa koje nasljeđuju klasu *LogFlusher*: *NetworkFlusher*, *FileFlusher* i *StdoutFlusher*. *NetworkFlusher* unijet će podatke u algoritam za otkrivanje napada, FileFlusher spremit će rezultat u TSV datoteku (engl. tab separated values), dok će *StdoutFlusher* rezultate vraćati u standardni izlaz (tipa konzola) te se takvi podatci mogu preusmjeriti na aplikaciju ili u određenu datoteku.

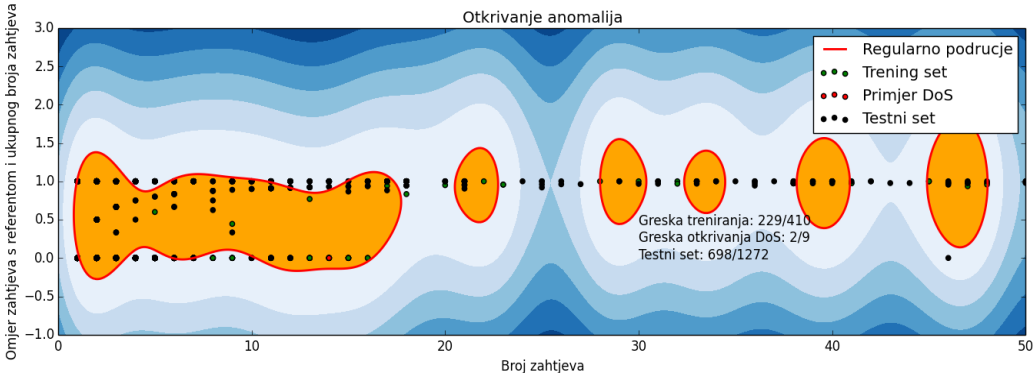
5.5 Podešavanje i testiranje algoritma

Za uspješno podešavanje algoritma strojnog učenja potrebno je imati klasificiran set podataka (eng. labeled data), testni set napada te testni set normalnih posjetitelja. Prilikom testiranja potrebno je paziti na efikasnost algoritma – broj detekcija napada te broj krivo označenih napada. U ovom radu korištena su dva algoritma: SVM te Naive Bayes na dva različita tipa portala: internacionalni RESTful API te regionalni portal. Regionalni portal također služi statičke datoteke (slike, css, javascript), a internacionalni isključivo JSON rezultat. Napad je simuliran slanjem velikog broja HTTP paketa korištenjem alata Apache ab te python skripte.

Rezultat SVM, na slici 5.5 za internactionalni portal te 5.6 za regionalni, pokazuju da je SVM učinkovit kod internetskih servisa koji imaju relativno mali broj zahtjeva. Na X-osi je broj zahtjeva u intervalu od 10 sekundi, a na Y-osi je omjer broja zahtjeva s referentom te ukupnog broja zahtjeva (interval od 0 do 1). Dok SVM algoritam kod internacionalnog portala izvrsno radi (pogrješka manja od 0.01% uz otkrivanje DoS napada, uz testni set, od 100%), SVM kod regionalnog portala nije učinkovit (pogrješka od 55% te učinkovitost otkrivanja 80%). Razlog je velik broj zahtjeva zbog učitavanja statičnih dijelova portala. SVM nije u mogućnosti napraviti dobru aproksimaciju između normalnih zahtjeva te malicioznih.



Sl. 5.5. *Internacionalni portal, SVM*



Sl. 5.6. *Regionalni portal, SVM*

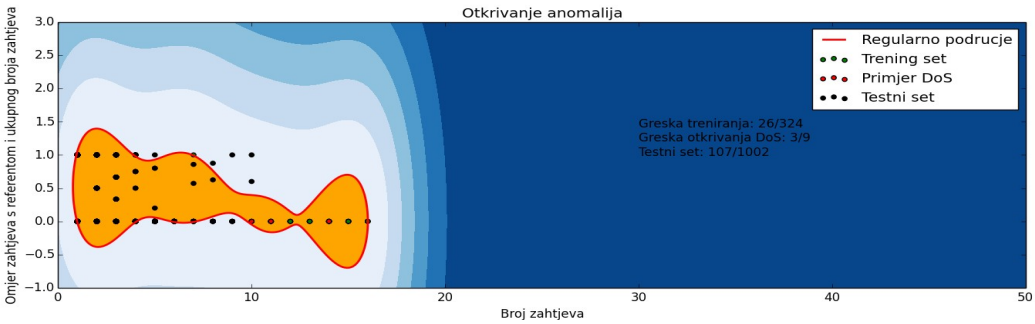
Ukupni rezultati Naive Bayes i SVM algoritam dani su tablicom 5.2. Kao što se može zaključiti i iz prethodnih slika, otkrivanje DoS napada nije uspješno kod regionalnog portala. Točnije, velik broj zahtjeva regularnih korisnika ocijenjen je kao napad – false positive – greška preko 55%.

S druge strane, internacionalni portal ima odlične rezultate – pogrješka od 0% te *false positive* 0.003% kod SVM algoritma te 3.83% kod Naive Bayes.

SET PODATAKA ALGORITAM	Pogrješka trening seta	Pogrješka terniung seta [%]	Pogrješka DoS seta	Pogrješka DoS seta [%]	Pogrješka testnog seta	Pogrješka testni seta [%]
SVM regionalni	229/410	55.85	2/9	22.22	698/1272	54.87
SVM internacionalni	8/2452	0.003	0/9	0.00	16/8533	0.002
Naive Bayes reg.	287/410	70.00	0/9	0.00	771/1272	60.61
Naive Bayes inter.	94/2452	3.83	0/9	0.00	239/8533	2.80

Tablica 5.2. *Pogrješke otkrivanja napada i "false positive"*

Filtriranjem statičkih datoteka (css, javascript te slike – gif, png i jpg) iz regionalnog portala dobiva se sasvim drugačiji rezultat SVM i Naive Bayes algoritma. Rezultat SVM dan je grafički na slici 5.7.



Sl. 5.7. *Regionalni portal bez statičkih datoteka; SVM*

Može se zaključiti da je algoritam višestruko precizniji i točniji, iako i dalje ima veće odstupanje nego kod internacionalnog portala. Isto tako, važno je odvojiti statičke datoteke od dinamičkih datoteka kako bi se otkrio DoS napad. Osim mogućnosti otkrivanje napada, smanjuje se i opterećenje poslužitelja, pojednostavljuje se njegova uloga te se omogućava lakše skaliranje aplikacije.

Tablicom 5.3 dani su rezultati nakon izdvajanja statičkih datoteka iz jednadžbe. Pogrješka je

smanjena s 55.85% na 8.02% za trening set SVM algoritma te s 54.87% na 10.68% za testni set. Pogrješka za otkrivanje DoS napada povećala se s 22.22% na 33.33%. Naive Bayes ima mnogo bolji rezultat nego SVM, pogrješka se smanjila s 70% na 14.81% za trening set te s 60.61% na 0.90% za testni set dok pogrješka otkrivanja DoS napada ostala nepromijenjena – 0%.

SET PODATAKA ALGORITAM	Pogrješka trening seta	Pogrješka terniung seta [%]	Pogrješka DoS seta	Pogrješka DoS seta [%]	Pogrješka testnog seta	Pogrješka testni seta [%]
SVM regionalni	26/324	8.02	3/9	33.33	107/1002	10.68
Naive Bayes regionalni	48/324	14.81	0/9	0.00	9/1002	0.90

Tablica 5.3. *Pogrješka otkrivanja napada i false positive, bez statičnih datoteka; regionalni portal*

6. ZAKLJUČAK

Otkrivanje DoS napada definitivno je zanimljivo područje te izazov za mnoge velike korporacije koje temelje posao upravo preko interneta. Iako postoje mnoga programatska rješenja, takvi programi rješavaju određeni problem te nisu u mogućnosti učiti, to jest programer mora intervenirati kako bi se nadogradila zaštita. Isto tako, takva se rješenja mogu primijeniti samo na određenu skupinu internetskih servisa. Strojno učenje ima mogućnost prilagođavanja okolini te se jedno rješenje može primijeniti na više različitih poslužitelja.

U ovom radu rekonstrukcijom korisničkih zahtjeva i simulacijom DoS napada te dodanim reguliranjem parametara, kao što je filtriranje statičkih datoteka, uspješnost otkrivanja DoS napada je 100% dok je postotak *false positive* 2 do 3%. Na krajnjem korisniku je da izabere najbolji algoritam uz najoptimalnije parametre. Iako je strojno učenje vrlo učinkovito kod otkrivanja DoS napada s vrlo malom pogrješkom *false positive* kao što smo mogli i vidjeti u poglavlju otkrivanja DoS napada, važno je napomenuti da zahtjeva više resursa nego programatska rješenja te rješenja na nižim razinama OSI mrežnog modela. Osim toga, napadi se otkrivaju tek nakon što se klijent posluži te je sustav ranjiv u ranoj fazi otkrivanja napada.

Daljnja poboljšanja ovog softverskog rješenja su dodavanje većeg broja algoritama strojnog učenja (šira primjena), podrška za druge log datoteke te, najvažnije, podrška za treću klasifikacijsku grupu. Treća klasifikacijska grupa bila bi skupina korisnika koji bi imali mogućnost potvrditi jesu li pravi korisnici. Time bi se omogućila povratna veza te bi računalo učilo na greškama, a korisnici bi mogli normalno pristupiti aplikaciji.

LITERATURA

[1] Denial of service attack – predavanje https://www.cs.columbia.edu/~smb/classes/f06/l22.pdf; listopad 2006

[2] Denning, Dorothy, "An Intrusion Detection Model," Proceedings of the Seventh IEEE Symposium on Security and Privacy, May 1986

[3] Application of Machine Learning Algorithms to KDD Intrusion Detection Dataset within Misuse Detection Context http://neuro.bstu.by/ai/To-dom/My_research/Papers-0/For-research/D-mining/Anomaly-D/KDD-cup-99/mlmta03.pdf; 2003

[4] Gerhard Muunz, Sa Li, Georg Carle, "Traffic Anomaly Detection Using K-Means Clustering"; srpanj 2009.

[5] Denial-of-service attack; https://en.wikipedia.org/wiki/Denial-of-service_attack; kolovoz 2013.

[6] The Anonymous WikiLeaks protests are a mass demo against control, http://www.theguardian.com/commentisfree/2010/dec/17/anonymous-wikileaks-protest-amazon-mastercard; 17. studeni 2010.

[7] Internet Denial-of-Service Considerations, https://tools.ietf.org/html/rfc4732; studeni 2006.

[8] Guassian distribution; https://en.wikipedia.org/wiki/Gaussian_distribution; kolovoz 2013.

[9] MaxMind GeoIP database; http://dev.maxmind.com/geoip/legacy/geolite/, kolovoz 2013

[10] HTTP status, http://httpstatus.es/; kolovoz 2013

[11] User-agents bibilioteka https://pypi.python.org/pypi/user-agents/0.1; kolovoz 2013

SAŽETAK

U radu je opisana definicija DoS napad, vrste DoS napada, opisano je strojno učenje da je dan prikaz algoritama strojnog učenja korištenih u ovom radu. Objašnjen je postupak pronalaženja varijabli za daljnju obradu te detaljan opis svake, postupak grupiranja korisničkih zahtjeva po otisku računala i geografskoj lokaciji, klasificiranje klijenata u dvije skupine (napadač, posjetitelj). Objašnjeno je i softversko rješenje od obrade log datoteke do algoritma strojnog učenja. U prilogu su priložene upute o pokretanju programa iz komadne linije.

Ključne riječi: Strojno učenje, SVM, Naive Bayes, DoS, DDoS

ABSTRACT

This paper describes what DoS attack is, what types of DoS exist, describes what machine learning is and what algorithms were used. Moreover, the paper describes a process of finding the right properties for further processing, a process of client requests' aggregation using its fingerprint and geographical location and classification of a client in two buckets (valid client and attacker). The paper describes a software solution for detection of DoS attacks using machine learning staring from log processing to machine learning algorithm. Appendix has instructions how to execute the software solution from command line.

Keywords: Machine Learning, SVM, Naive Bayes, DoS, DDoS

ŽIVOTOPIS

Josip Šokčević rođen je 8. rujna 1990. godine u Vinkovcima te tamo završio osnovnu i srednju tehničku školu, smjer mehatronika. Tijekom osnovnoškolskog i srednjoškolskog obrazovanja nastupao je na brojnim natjecanjima iz matematike i informatike, ali i sporta. Višestruki je prvak županije u matematici i informatici te ima brojne pehare s teniskih natjecanja.

2005. godine radi mnoge web projekte s ciljem promocije otvorenog koda. 2006. godine osvaja nagradu Vidi Web Top 100 u kategoriji "Mediji i novosti" za stranicu croblogeri.com (danas blogeri.hr). 2007. s kolegom pokreće jednu od najpopularnijih portala otvorenog koda linuxzasve.com te također osvaja nagradu Vidija 2012. godine. Josip se 2009. godine upisuje na Elektrotehnički fakultet u Osijeku, smjer računarstvo kako bi usavršio svoje znanje u računarnoj znanosti. Dvije godine poslije, Josip sudjeluje u projektu ETFOS mobi s izradom Android aplikacije te dobiva rektorovu nagradu. Iste godine, 2011., dobiva i posao u The Backplane, Inc. kao softverski inženjer. Nakon godinu dana nastavlja školovanje na Elektrotehničkom fakultetu.

JOSIP ŠOKČEVIĆ

PRILOG

Instalacija potrebnih biblioteka na Unix sustavu

Program je pisan u programskom jeziku Python, verzija 2.7. Instalacija na Ubuntu i Debian distribuciji moguća je preko upraviteljem paketa apt-get te su potrebne root ovlasti.

$ apt-get install python2.7 pip2 # instalacija python2 i pip

$ pip install pyyaml ua-parser user-agents pygeoip scikit-learn # (instaliranje biblioteka)

Potrebno je skinuti i prevesti u strojni kod (engl. compile) bibliteku basemap http://sourceforge.net/projects/matplotlib/files/matplotlib-toolkits/basemap-1.0.6/

Nakon toga je program spreman za korištenje.

Dokumentacija

ddos_tester_naive_bayes.py – testiranje algoritma pomoću seta za treniranje testnog seta

Korištenje:

ddos_tester_naive_bayes.py [-h] [-f flush time] [-s skip_static] good_file bad_file log

Gdje je:

- -h oznaka za ispis pomoći
- -f interval agregacije (početna vrijednost 10 sekundi)
- -s korištenjem parametra statične datoteke (jpg, gif, png, css, js) neće ulaziti u agregaciju
- good_file je putanja do apache log datoteke kod koje nije došlo do napada
- bad_file je putanja do apache log datoteke simuliranog napada (data/ab.log se može koristiti)
- log je putanja do apache log datoteke koja će se koristiti za testiranje ispravnosti mreže

Poželjno je da količina good_file bude 30 do 50% veličine log datoteke. To se može postići korištenjem unix programa *split*.

ddos_tester_svm.py – testiranje algoritma pomoću seta za treniranje testnog seta

Korištenje:

ddos_tester_svm.py [-h] [-f flush time] [-s skip_static] good_file log

Gdje je:

- -h oznaka za ispis pomoći
- -f interval agregacije (početna vrijednost 10 sekundi)
- -s korištenjem parametra statične datoteke (jpg, gif, png, css, js) neće ulaziti u agregaciju
- good_file je putanja do apache log datoteke kod koje nije došlo do napada
- log je putanja do apache log datoteke koja će se koristiti za testiranje ispravnosti mreže

Poželjno je da količina good_file bude 30 do 50% veličine log datoteke. To se može postići korištenjem unix programa *split*.

ddos_tester_svm_visualize.py – grafički prikaz omjera broja posjeta i omjer referenata

Korištenje:

ddos_tester_svm_visualize.py [-h] [-f flush time] [-s skip_static] good_file log

Gdje je:

- -h oznaka za ispis pomoći
- -f interval agregacije (početna vrijednost 10 sekundi)
- -s korištenjem parametra statične datoteke (jpg, gif, png, css, js) neće ulaziti u agregaciju
- good_file je putanja do apache log datoteke kod koje nije došlo do napada
- log je putanja do apache log datoteke koja će se koristiti za testiranje ispravnosti mreže

Poželjno je da količina good_file bude 30 do 50% veličine log datoteke. To se može postići korištenjem unix programa *split*.

plot_world.py – grafički prikaz geografske distribucije korisnika

Korištenje:

ddos_tester_svm_visualize.py input

Gdje je:

- input – TSV datoteka grupiranih podataka

parse_log.py – obrada log datoteke te grupiranje po otisku računala. Izlaz je TSV

Korištenje:

parse_log.py [-h] [-f flush time] [-o output file] Input file

Gdje je:

- -h oznaka za ispis pomoći
- -f interval agregacije (početna vrijednost 10 sekundi)
- -o izlazna datoteka (ukoliko je nedefinirano, izlaz je standardni izlaz)
- Input file – apache log datoteka

ddos_detection.py – alat za aktivno otkrivanje napada

Korištenje:

ddos_detection.py [-h] [-f flush time] good_file bad_file log

Gdje je:

- -h oznaka za ispis pomoći
- -f interval agregacije (početna vrijednost 10 sekundi)
- -s korištenjem parametra statične datoteke (jpg, gif, png, css, js) neće ulaziti u agregaciju
- good_file je putanja do apache log datoteke kod koje nije došlo do napada
- bad_file je putanja do apache log datoteke simuliranog napada (data/ab.log se može koristiti)
- log je putanja do apache log datoteke koja će se koristiti za testiranje ispravnosti mreže