

ZAVOD ZA ELEKTRONIKU, MIKROELEKTRONIKU, RAČUNALNE I INTELIGENTNE SUSTAVE
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA
SVEUČILIŠTE U ZAGREBU

SUSTAVI ZA OTKRIVANJE MREŽNIH NAPADA

Igor Požgaj
SEMINARSKI RAD

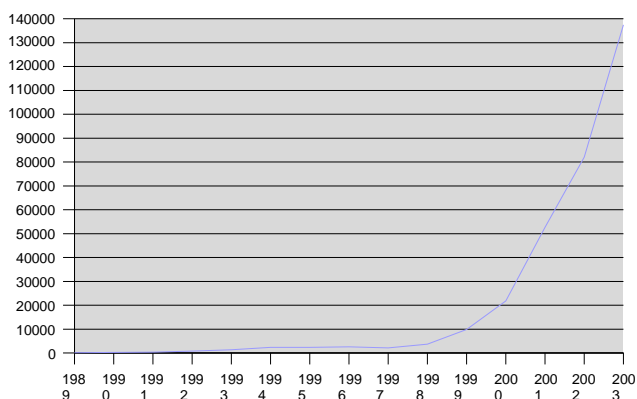
Zagreb, 2006.

Sadržaj

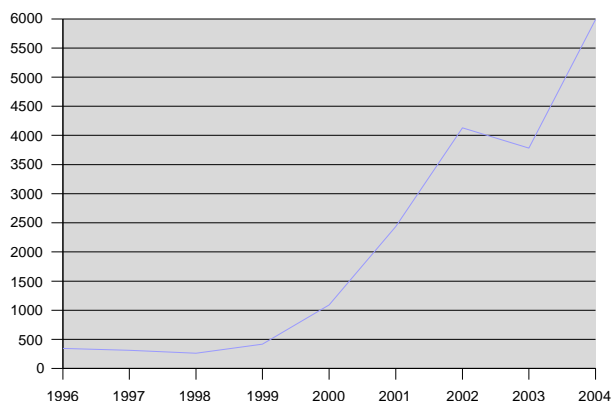
1. Uvod.....	1
2. Sustavi za otkrivanje mrežnih napada.....	3
2.1. Definicija i funkcionalnost.....	3
2.2. Vrste IDS-a.....	4
2.3. Položaj u različitim mrežnim topologijama.....	5
2.3.1. Smještaj IDS-a.....	5
2.3.2. Rješavanje problema sa preklopnima.....	6
2.3.3. Mrežna priključnica.....	7
2.4. Arhitektura.....	8
2.4.1. Uobičajena arhitektura IDS-a.....	8
2.4.2. Dekoder paketa.....	9
2.4.3. Pretprocesor.....	9
2.4.4. Sustav detekcije.....	10
2.4.5. Sustav za vođenje dnevnika i upozoravanje.....	11
2.4.6. Izlazni moduli.....	11
2.5. Problemi i nedostaci IDS-ova.....	12
2.6. Zaštita IDS-a.....	13
2.7. Standardizacija.....	13
3. Praktični rad.....	15
3.1. Snort IDS.....	15
3.2. Ispitno okruženje i ispitivanje napada.....	16
3.3. Rezultati ispitivanja.....	17
3.3.1. Skeniranje pristupnih vrata.....	17
3.3.2. Napadi shell kôdovima.....	18
3.3.3. Zaobilaženje otkrivanja.....	20
4. Zaključak.....	21
5. Literatura.....	22

1. Uvod

Računalna sigurnost je posljednjih nekoliko godina jedan od najčešće spominjanih pojmova u računalnoj industriji. Novi propusti i metode napada na informacijske sustave otkrivaju se gotovo svakodnevno. Prema podacima preuzetih od CERT-a (eng. *Computer Emergency Response Team*), broj prijavljenih sigurnosnih incidenata u razdoblju od petnaest godina porastao je gotovo 1000 puta (Slika 1.1), a praktički se udvostručuje svake sljedeće godine. Razlozi za ovakav drastičan porast broja sigurnosnih incidenata su mnogobrojni. Posljednjih godina pristup Internetu je sve jednostavniji i jeftiniji, a razvojem tehnologije veze postaju sve brže pa je sve teže analizirati sav promet koji prolazi takvim visoko propusnim mrežama. Nadalje, tržištem trenutno dominira vrlo mali broj operacijskih sustava pa pronalaženjem propusta napadač automatski dobiva veliki broj potencijalnih žrtava na kojima može iskoristiti pronađeni propust. Brzi razvoj tehnologije često izbacuje na tržište nekompletna i neprovjerena rješenja koja na kraju rezultiraju velikim brojem sigurnosnih propusta. Takav primjer se može vidjeti na slučaju WEP (eng. *Wired Equivalent Privacy*) protokola u bežičnim mrežama gdje se u samo nekoliko mjeseci nakon objavljivanja protokola pojavio niz alata za iskorištavanje njegovih nedostataka. Uza sve to, popularizacijom Interneta informacije o novim propustima se trivijalno i vrlo brzo šire među velikim brojem ljudi, a dobavljanje alata za napade na razne informacijske sustave se svodi na jednostavno upisivanje ključnih riječi u tražilicu te preuzimanje gotovih alata s jedne od mnogobrojnih *hackerskih* stranica. Zbog toga je znanje potrebno za uspješno provođenje napada na neki sustav sve nevažniji faktor pa sve veći udio u populaciji napadača čine takozvani *script kiddies*, napadači koji koriste već gotove alate bez detaljnog znanja o alatu koji koriste ili propustu koji iskorištavaju.



Slika 1.1: Prijavljeni sigurnosni incidenti [CERT]



Slika 1.2: Pronađeni sigurnosni propusti [CERT]

Struktura napada se tijekom vremena drastično promijenila. Prema podacima iz CERT-a, prije desetak godina većinu prijavljenih incidenata su sačinjavale prijave vezane uz štetno djelovanje virusa, pogađanje korisničkih lozinki raznim metodama pretraživanja (eng. *brute force*), te iskorištavanje dobro poznatih propusta u sustavima. Porastom kompleksnosti informacijskih sustava drastično

se povećava i broj sigurnosnih propusta (Slika 1.2). Danas većinu prijava CERT-u sačinjavaju razni tipovi mrežnih napada u rasponu od jednostavnog prikrivenog skeniranja pristupnih vrata (eng. *stealth port scanning*) pa sve do DDoS (eng. *Distributed Denial of Service*) napada i lažiranja paketa (eng. *packet spoofing*). U vrijeme kada je gotovo svako računalo priključeno na Internet i zbog svih prethodno navedenih razloga, mrežna sigurnost i razvoj sigurnosnih rješenja za obranu od mrežnih napada čine jedno od najbrže razvijajućih područja računalne industrije. Posebnu ulogu u tome ima i razvoj alata za obranu ranjivih mrežnih usluga.

Paralelno s razvojem raznih metoda napada na informacijske sustave razvijali su se i obrambeni mehanizmi i tehnike. Jedna od prvih, a danas vjerojatno i najčešće korištenih metoda obrane je korištenje mrežne sigurnosne stijene (eng. *firewall*). Sigurnosna stijena je u svojoj osnovi softver ili hardver koji djeluje kao filtar paketa i služi za osnovnu kontrolu prometa između raznih područja povjerenja (eng. *zones of trust*) čime je moguće u potpunosti zabraniti ili selektivno propuštati promet prema određenim grupama računala. Tipična takva područja koja se koriste su *lokalna mreža* (eng. *local zone*) s najvećim stupnjem povjerenja, demilitarizirano područje (eng. *demilitarized zone*) s manjim stupnjem povjerenja te *Internet* s najmanjim ili nikakvim stupnjem povjerenja. Moguće su i implementacije s puno više područja povjerenja. Prve implementacije sigurnosne stijene su bile statične i nisu pratile stanje veze (eng. *stateless packet filter*) pa su mogle blokirati promet samo u ovisnosti o mrežnim adresama i pristupnim vratima. Danas je takav pristup uglavnom manje zastupljen i koriste se sigurnosne stijene koje mogu pratiti i stanje veze (eng. *statefull packet filter*). Praćenje stanja veze omogućuje izradu puno složenijih pravila jer se svakoj vezi pridružuje i jedno od mogućih stanja. Moguća stanja su npr. *nova* (eng. *new*), *uspostavljena* (eng. *established*), *povezana* (eng. *related*) i slično. Ova tehnika omogućuje bolje funkcioniranje sigurnosne stijene s problematičnim servisima poput protokola za prijenos podataka FTP (eng. *File Transfer Protocol*). Sve veća potreba za ovakvim rješenjima pojavila se i nakon uvođenja novijih mrežnih tehnologija poput CIDR (eng. *Classless Inter Domain Routing*) i NAT (eng. *Network Address Translation*).

Glavni nedostatak zaštite sigurnosnom stijenom je što većina takvih rješenja radi na mrežnom (L3) i transportnom (L4) sloju čime je moguće propuštanje ili zabranjivanje prometa samo u ovisnosti o podacima dostupnim na tim slojevima. Moderne metode zaštite zahtijevaju i analizu prometa na slojevima višim od mrežnog, posebice na aplikacijskom (L7) sloju. Primjerice, paket koji stiže na poslužitelj i sadrži štetni programski kôd koji će napadaču omogućiti pristup ljusci operacijskog sustava (eng. *shell code*), sigurnosna stijena neće moći odbaciti jer su na mrežnom i transportnom sloju dostupne samo informacije o mrežnim adresama i pristupnim vratima pa sigurnosna stijena ne može znati ništa o sadržaju paketa (eng. *payload*). Očito je da je za puno veću kontrolu prometa koji prolazi mrežom potrebno rješenje koje će između ostalog moći analizirati i sadržaj samoga paketa te na temelju tih informacija odlučiti da li će paket biti propušten ili odbačen.

Još jedno od često korištenih sigurnosnih rješenja su i *honeypotovi*. Honeypot je računalo koje se nalazi u istoj lokalnoj mreži na kojoj se nalazi i stvarni

poslužitelj kojeg treba zaštititi, a čija je osnovna namjena zavarati napadača tako da povjeruje da radi sa stvarnim poslužiteljem. Cilj ove tehnike je mogućnost pravovremenog dobijanja informacije o napadu te dobijanje informacija o tehnikama koje napadač koristi da bi ostvario taj napad. Honeypot bi po adresi trebao biti blizak stvarnom poslužitelju, obično ima pokrenute usluge koje napadači najčešće napadaju (*telnet, ssh, ftp, httpd, nfs, samba...*), a sigurnosni mehanizmi na njemu su uobičajeno puno slabiji od onih na stvarnom poslužitelju. Često se i pristupna vrata navedenih mrežnih usluga sa stvarnog poslužitelja preusmjeravaju na pristupna vrata honeypota. Samo računalo koje služi kao honeypot mora biti dobro izolirano od stvarnog poslužitelja da napadač ne dobije mogućnost napada na ostala računala u mreži nakon što kompromitira računalo s honeypotom.

Danas se, zahvaljujući vrlo brzom razvoju virtualizacijskih tehnika i alata (*vmware, xen, qemu, Microsoft Virtual Machine...*) za honeypotove najčešće ne koriste stvarni već virtualni poslužitelji. Prednost takvog pristupa je puno manji trošak postavljanja pojedinih honeypotova, ali i mogućnost postavljanja većeg broja honeypotova za istu cijenu čime se povećava vjerojatnost da će napadač pri napadanju mreže naići baš na honeypot, a ne na stvarni poslužitelj. Tvrtke koje se bave računalnom sigurnošću često pružaju već gotove programske proizvode koji obuhvaćaju honeypot, sigurnosnu stijenu, sustav za otkrivanje napada i druge korisne alate na *live CD-u* čime postavljanje takvih rješenja postaje izuzetno jednostavno. Jedno od takvih popularnih rješenja je *Honeynet*, slobodno dostupni skup programa za zaštitu temeljen na principima otvorenog programskog kôda. Honeynet obuhvaća sve prethodno navedene metode zaštite na jednom live Linux CD-u.

2. Sustavi za otkrivanje mrežnih napada

2.1. Definicija i funkcionalnost

Sustavi za otkrivanje napada (eng. *Intrusion Detection Systems*) su jedna od novijih tehnologija za podizanje ukupne razine sigurnosti sustava. Njihov rad se zasniva na prikupljanju informacija s čitavog niza mrežnih i računalnih izvora te analiziranju tih informacija s ciljem otkrivanja eventualnih nedozvoljenih aktivnosti i zlouporabe sustava na kojem se nalaze. Sustavi za otkrivanje napada (u daljnjem tekstu koristi se u literaturi uobičajena kratica *IDS*) prate i analiziranju mrežni promet i različite dijelove operacijskog sustava, međutim ne poduzimaju nikakve akcije koje bi spriječile štetni i neželjeni promet. Ukoliko se takav promet uoči, najčešće se samo zabilježi u datoteke dnevnika te se obavještavaju nadležne osobe putem različitih mehanizama obavještavanja. U literaturi se često razlikuju sustavi koji otkrivaju napade od onih koji otkrivaju pogrešno korištenje sustava. Uobičajeno se napadom smatra štetna aktivnost koja na sustav djeluje s lokacije izvan sustava, a iste takve aktivnosti koje djeluju unutar samog sustava se smatraju pogrešnim korištenjem. Na primjer, zaposlenik prijavljen za rad na jednom od računala u lokalnoj mreži koji više puta za redom unese krivo korisničko ime kod autentifikacije na neku mrežnu uslugu u korporacijskom intranetu krivo koristi sustav, dok se isti postupak ali s računala izvan tvrtkine lokalne mreže smatra napadom. U većini stvarnih implementacija ovakvih programa najčešće se ne pravi razlika između napada i pogrešnog korištenja sustava već se oboje tretiraju jednako.

Rad *IDS*-a zasniva se na nadziranju rada operacijskog sustava te analiziranju sadržaja mrežnih paketa. Obje vrste podataka sadrže informacije o aktivnostima korisnika pa je na temelju analize njihova sadržaja moguće identificirati neovlaštene aktivnosti i napade. Pri analiziranju mrežnog prometa *IDS* ni na koji način ne smije usporavati mrežni promet – njegov rad mora biti u potpunosti nevidljiv korisnicima mreže. Pošto analiza svog prometa na visoko propusnim mrežama može biti prilično zahtjevan posao, primjenjuju se posebne metode (npr. mrežne priključnice) da bi se izbjegle takve vrste problema. Analizu mrežnog prometa moguće je obavljati na računalu posebno namijenjenom samo svrsi prepoznavanja i praćenja napada na određeni poslužitelj, aplikaciju ili grupu računala u mreži. Iako je moguće da se *IDS* nalazi na istom računalu na kojem je instaliran i poslužitelj koji se želi zaštititi, najčešće se ipak nalazi na zasebnom računalu namijenjenom posebno toj svrsi. U literaturi se ovakvo računalo koje se koristi isključivo u svrhu rada *IDS*-a naziva *senzor*.

Za prepoznavanje štetnih aktivnosti na računalnom sustavu ili mreži, *IDS* koristi dvije uobičajene metode: *prepoznavanje potpisima* i *prepoznavanje anomalijama*. Kod prepoznavanja potpisima (eng. *signatures detecting, pattern matching*) sadržaj dnevnčkih zapisa ili mrežnih paketa šalje se u posebni pretraživački program gdje se uspoređuju s unaprijed poznatim i definiranim potpisima napada. Potpis napada je skup podataka pomoću kojih se ispravno korištenje sistemskih resursa može razlikovati od napada ili nedozvoljenog načina

2. Sustavi za otkrivanje mrežnih napada

korištenja, S druge strane, sustavi koji koriste prepoznavanje anomalijama rade na principu prikupljanja i statističke obrade podataka o uobičajenom radu sustava. Statistička obrada podataka služi da bi sustav za otkrivanje napada odredio granice koje definiraju dozvoljeno korištenje sustava. Kada se primijeti znatnije odstupanje od tako sakupljenih i statistički obrađenih podataka, IDS će smatrati da je u tijeku napad na sustav. Sustav za otkrivanje koji koristi prepoznavanje anomalijama u pravilu koristi više sistemskih resursa, ali će nakon početnog razdoblja “učenja” moći prepoznati i nove vrste napada, dok je sustav koji koristi prepoznavanje potpisima toliko učinkovit koliko je ažurna njegova baza podataka s potpisima napada. Usporedba pojedinih važnijih karakteristika sustava koji su zasnovani na prepoznavanju potpisima te onih koji koriste prepoznavanje anomalijama prikazana je u sljedećoj tablici:

Prepoznavanje potpisima	Prepoznavanje anomalijama
Mali broj lažnih sigurnosnih upozorenja	Mogući veći broj lažnih sigurnosnih upozorenja
Proizvođač IDS mora održavati i izdavati potpise poznatih vrsta napada	Prilagodljivi sustav, može otkriti čak i nepoznate vrste napada
Brza obrada napada	Zahtjeva više računalnih resursa
Nije potrebno “učenje” sustava da prepoznaje napade	Potrebno je “učiti” sustav da prepoznaje napade, može biti problem u dinamičnim sredinama
Teško održavanje pravila	Manje pravila i lakše održavanje

2.2. Vrste IDS- a

IDS-ovi se prema svojoj namjeni dijele u dvije skupine:

- *računalno bazirani IDS*
- *mrežni IDS*

Računalno bazirani IDS (eng. *Host Based Intrusion Detection System - HIDS*) je sustav namijenjen analiziranju mrežnog prometa koji je usmjeren prema samom računalu na kojem je postavljen IDS senzor, te analiziranju datoteka, procesa i drugih važnijih dijelova operacijskog sustava. Vrste napada koje može otkrivati ovakav sustav su vrlo raznolike. Rad HIDS-a se zasniva na prikupljanju potpisa ili sažetka (eng. *hash*) te praćenju dozvola na kritičnim sistemskim datotekama, nadziranju rada s datotečnim sustavom, nadziranju procesa, memorije itd. Važan dio HIDS-a je baza podataka koja sadrži informacije o ranjivim programima i uslugama na temelju kojih sustav može procijeniti rizičnost pojedinih pokrenutih aplikacija te dati ukupnu ocjenu o sigurnosti sustava. Primjeri često korištenih HIDS programa su kompletne aplikacije, npr. *Nessus* i *samhain*, ali i manji alati poput *chkrootkit* i *tripwire*.

Mrežni IDS (eng. *Network Intrusion Detection System – NIDS*) je sustav namijenjen prikupljanju i analiziranju mrežnog prometa koji nije nužno generiran ili usmjeren prema računalu na kojem je postavljen IDS, to jest prema IDS senzoru. Cilj NIDS-a je nadzirati rad ranjivih mrežnih usluga i aplikacija, pratiti rad korisnika na mreži te paziti na dobro poznate propuste u mrežnim protokolima. NIDS u svojoj osnovi djeluje vrlo slično kao i aplikacijska (L7) sigurnosna stijena

ili aplikacijski prilaz (eng. *application level gateway*). Za razliku od sigurnosne stijene, NIDS ne može utjecati na stanje veze pa se njegove aktivnosti svode na prepoznavanje napada i upozoravanje. Nadalje, način rada i funkcija NIDS-a se bitno razlikuje od L7 sigurnosne stijene koja je primarno namijenjena selektivnom propuštanju mrežnog prometa koji generiraju određeni programi. Glavna tema ovog seminara su mrežni IDS pa će se pod pojmom “IDS”, ukoliko nije drukčije navedeno, u nastavku seminara podrazumijevati mrežni IDS (NIDS).

Mrežni IDS mogu otkriti veliki broj vrsta napada preko mreže zahvaljujući već unaprijed poznatim potpisima napada. Glavne vrste takvih potpisa vezane su uz nadziranje karakterističnih ciljeva u mrežnim paketima. Najčešće se nadziru sadržaj paketa, zaglavlje paketa te pristupna vrata. Primjer niza znakova u sadržaju paketa koji se može tražiti je niz:

```
cat "+ +" > /.rhosts
```

koji ukoliko se izvrši na Unix računalu može učiniti sustav vrlo izloženim daljnjim napadima. Nizovi znakova koji se također često traže su “cgi-bin”, “ifs”, “aglimpse” itd. Druga vrsta potpisa traži sumnjive pakete koji stižu na računalo i cilj im je pristupiti karakterističnim pristupnim vratima (IMAP, SunRPC, Telnet, FTP...), a na lokalnom računalu nisu pokrenute mrežne usluge na tim pristupnim vratima. Posljednja vrsta potpisa zadužena je za otkrivanje opasnih i nelogičnih kombinacija u zaglavlju mrežnih paketa. Najpoznatiji takav primjer je program *Winnuke*, koji je slao pakete na NetBIOS pristupna vrata s uključenim opcijama URG (eng. *urgent pointer*) i OOB (eng. *out of band*). Zbog propusta u operacijskom sustavu *Microsoft Windows*, takvi paketi su uzrokovali trenutno rušenje sustava. Sličan primjer malicioznih paketa su i paketi koji u zaglavlju imaju uključenu i SYN i FIN opciju čime napadač pokušava uspostaviti i zatvoriti vezu u isto vrijeme. Najpoznatiji programi koji mogu otkriti sve prethodno navedene vrste napada su *Snort*, *Axent*, *Cisco Secure IDS*, *ISS*, *Shadow* itd.

2.3. Položaj u različitim mrežnim topologijama

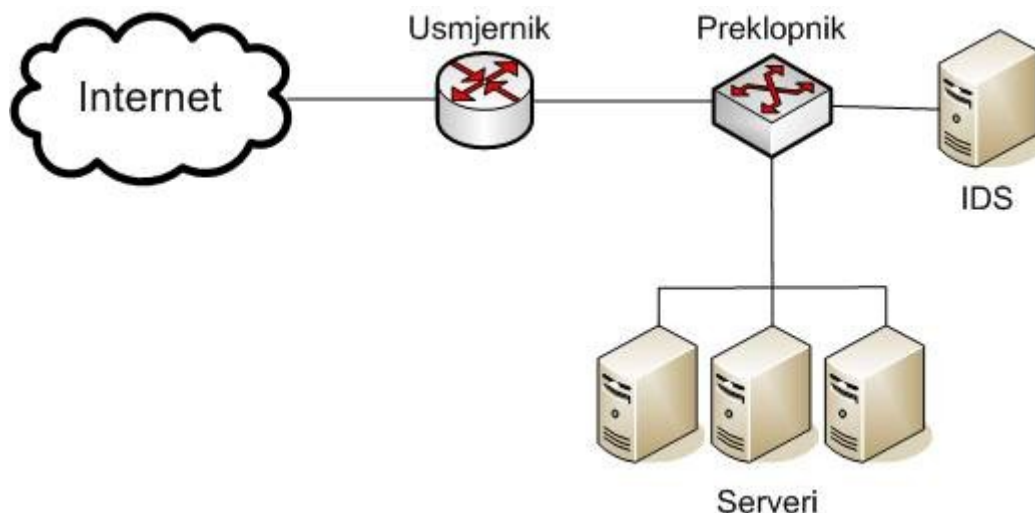
2.3.1. Smještaj IDS-a

Ovisno o mrežnoj topologiji na koju se postavlja, IDS može biti smješten na jedno ili više mjesta u mreži. Smještaj također ovisi o prirodi napada koji želimo otkrivati: napade s lokalne mreže (*unutrašnje napade*), napade izvan lokalne mreže (*vanjske napade*) ili oboje. Na primjer, ako želimo otkrivati samo vanjske napade, a mreža se sastoji od samo jednog usmjernika (eng. *router*) preko kojega lokalna mreža s nekoliko segmenata ima pristup Internetu, najbolje je IDS postaviti odmah iza usmjernika jer će sav promet koji dolazi s Interneta u lokalnu mrežu morati proći kroz usmjernik pa samim time i kroz IDS senzor. Ako mreža ima više izlaza na Internet, tada se IDS mora nalaziti na svakoj izlaznoj točki iz mreže. Ako želimo otkrivati i unutrašnje napade, tada je IDS senzore potrebno postaviti na svaki mrežni segment kako bi senzori imali pristup svom prometu. U većini slučajeva nema potrebe za nadziranjem svakog pojedinačnog segmenta nego se nadziru samo pojedini povjerljivi dijelovi mreže, primjerice segment mreže na kojem se nalazi sigurnosna stijena ili aplikacijski poslužitelj. U pravilu se IDS senzor postavlja iza svakog usmjernika i sigurnosne zaštitne stijene u

lokalnoj mreži. Mjesta koja se obično izbjegavaju su redundantni dijelovi mreže poput paralelnih mrežnih veza, nakupina računala (eng. *clusters*), višestrukih preklopnika, sustava s *failover* rješenjima protiv ispada pojedinih mrežnih kartica itd.

Posebnu pažnju kod postavljanja IDS senzora i dimenzioniranja mreže treba obratiti na način na koji će se obrađivati prikupljeni podaci. Ovi problemi posebice dolaze do izražaja u složenijim konfiguracijama IDS-a gdje je potrebno postaviti više IDS čvorova i senzora koji međusobno komuniciraju. U takvim slučajevima najčešće je potrebno odabrati rješenje koje će biti kompromis između potrebne procesorske snage te količine prometa koja će se izmjenjivati između pojedinih IDS čvorova. Podaci iz pojedinih senzora mogu se obrađivati *distribuirano* ili *centralizirano*. Kod distribuiranog pristupa podaci se obrađuju u onim sensorima u kojima su i prikupljeni, dok se kod centraliziranog pristupa podaci sakupljeni u sensorima šalju u središnji senzor ili *analizator* gdje će biti detaljnije analizirani. Prednost centraliziranog pristupa projektiranju IDS-a je lakše upravljanje cjelokupnim sustavom, međutim takvo rješenje znatno povećava količinu mrežnog prometa koji se razmjenjuje između pojedinih senzora te zahtjeva znatno veću procesorsku moć centralnog senzora. Distribuirani je pristup znatno teže održavati, ali se zato smanjuje količina mrežnog prometa između pojedinih senzora te je potrebna puno manja procesorska snaga za obradu ovako prikupljenih podataka. Dodatna činjenica koja ide u prilog distribuiranom pristupu u oblikovanju IDS je bolja otpornost na ispadu u sustavu. Pošto su pojedini čvorovi u distribuiranom sustavu potpuno autonomni, ukoliko jedan od distribuiranih čvorova prestane s radom, ostatak sustava može i dalje ispravno raditi. Centralizirani sustav nije tako fleksibilan i otporan je samo na ispadu senzorskih računala, a ukoliko se pokvari centralni čvor tada će prestati ispravno funkcionirati i ostatak sustava. U praksi se najčešće javlja kombinacija oba prethodno navedena pristupa pri čemu distribuirani senzori prikupljaju podatke i na njima obavljaju neke osnovne akcije (npr. pretprocesiranje mrežnih paketa) te ih zatim prosljeđuju jednom ili više centralnih senzora koji će ih detaljnije analizirati. Svim čvorovima u takvim sustavima može se upravljati s jednog mjesta putem posebne upravljačke konzole.

Tipični smještaj IDS-a u jednostavnoj lokalnoj mreži prikazuje Slika 2.1. U slučaju da mreža mora sadržavati i demilitarizirano područje, također je potrebno smjestiti senzor i u taj segment mreže, iako bi u pravilu postavke za generiranje sigurnosnih upozorenja u tom području trebale biti puno blaže od onih u privatnim dijelovima mreže.



Slika 2.1: Tipičan smještaj IDS-a

2.3.2. Rješavanje problema sa preklopticima

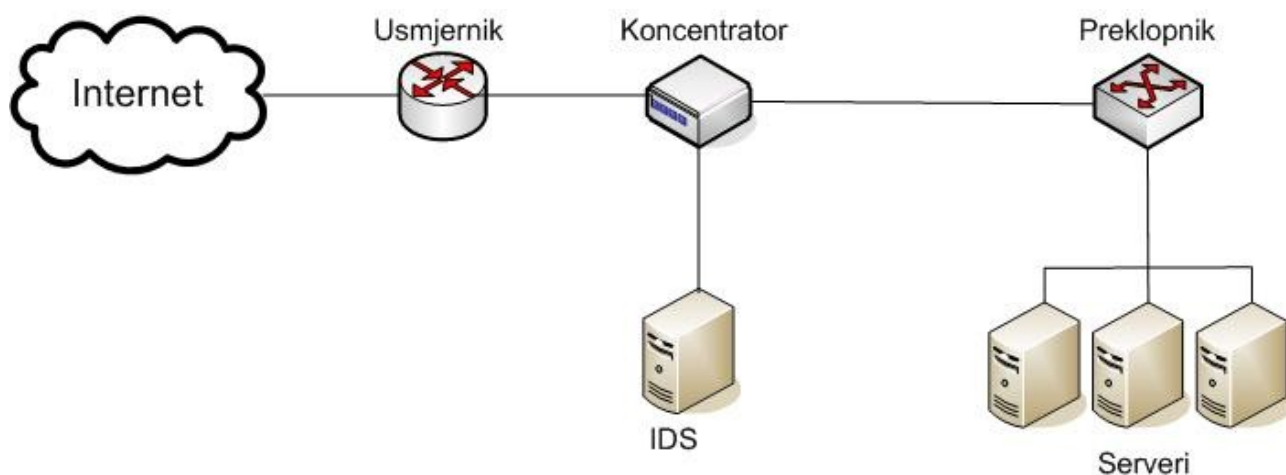
Poseban problem u smještaju IDS-a i pojedinih senzora predstavlja mrežna oprema koja razdvaja mrežu na više područja vidljivosti mrežnih adresa. Primjer takve mrežne opreme je preklopnik (eng. *switch*). Problem se može pojaviti u slučajevima poput onoga koji prikazuje Slika 2.1. Mrežni preklopnik će pakete koje dobije na jedan od izlaza proslijediti na točno onaj izlaz koji je priključen na onaj segment mreže na kojemu se nalazi ciljna adresa (mrežni preklopnik radi sa adresama na podatkovnom sloju, tj. s MAC adresama). Ovo predstavlja ozbiljno ograničenje u radu IDS-a jer je u tom slučaju potrebno postavljati senzore na svaki od izlaza preklopnika što bitno uvećava troškove i vrijeme potrebno za postavljanje, ali i održavanje takvog rješenja.

Većina novijih preklopnika (npr. tvrtke Cisco) su tako dizajnirani i mogu raditi u takvome načinu rada da se TX/RX (*transmit* i *receive*) linije s jednog izlaza mogu zrcaliti na drugi izlaz preklopnika. Takav izlaz na koji se zrcali se obično naziva SPAN (eng. *Switch Port Analyser*) ili zrcalni izlaz (eng. *mirrored port*). Ukoliko se senzor priključi na takav izlaz, a preklopnik je podešen tako da zrcali izlaz koji je priključen prema usmjerniku na SPAN izlaz, sav promet koji prolazi od usmjernika k poslužiteljima kroz preklopnik dolaziti će i do IDS senzora. Na taj način se izbjegava potreba za postavljanjem dodatnih senzora u svaki pojedini segment mreže na koji je spojen preklopnik. Korištenje SPAN izlaza je vrlo jednostavno i ne zahtjeva nikakvo dodatno mijenjanje postojeće mrežne topologije. Priključivanjem IDS senzora na SPAN izlaz preklopnika nije potrebno mijenjati konfiguracije ostalih mrežnih komponentni poput tablica prosljeđivanja u usmjernicima ili pravila u sigurnosnim stijenama. Međutim korištenje SPAN izlaza ima i niz nedostataka. Na jednom preklopniku se može nalaziti samo jedan SPAN izlaz. Ukoliko je potrebno nadzirati više od jednog izlaza preklopnika tada se na SPAN izlaz umjesto samo jednog mora zrcaliti promet sa čitavog raspona izlaza. Ovakvo zrcaljenje prometa sa više izlaza nije poželjno jer može vrlo brzo zakrčiti SPAN izlaz i smanjiti performanse preklopnika, pogotovo ako se nadzire promet u dvosmjernoj vezi (eng. *full duplex link*). Bez dodatnih promjena u podešenju IDS-a, ovakav način rada omogućava napadaču napad direktno na IDS, osim ako je zaštićen radom u prikrivenom načinu rada kako je opisano u

2. Sustavi za otkrivanje mrežnih napada

poglavlju o zaštiti IDS-a. Dodatni je nedostatak ovakvog rješenja i činjenica da će preklopnik odbacivati pakete za koje izračunato zaštitno CRC polje ne odgovara onome u paketu. IDS inače analizira i ovakve pakete jer napadači koriste razne tehnike u kojima namjerno generiraju pakete sa krivim CRC poljima kako bi zaobišli mehanizme zaštite.

Drugo rješenje problema s preklopnima je korištenje dodatne mrežne opreme poput koncentratora (eng. *hub*). Koncentrator se koristi u paričnim mrežama i radi na takav način da će sve pakete koje dobije na jedan od svojih izlaza proslijediti na sve ostale izlaze, tj služi kao obnavljač signala sa više izlaza (eng. *multiport repeater*). Ovakav način rada koncentratora može se iskoristiti za slanje kopije prometa prema IDS-u pri čemu je potrebno napraviti takvu topologiju mreže u kojoj će se koncentrator nalaziti između usmjernika i preklopnika, a IDS se priključi na dodatni izlaz koncentratora kao što prikazuje slika Slika 2.2.



Slika 2.2: Smještaj IDS-a u mrežama sa preklopnima

U ovom slučaju sav promet koji ide od usmjernika prema preklopniku biti će prosljeđen i preklopniku i IDS senzoru. Potrebno je i napomenuti da je ovakva topologija mreže pogodna samo za otkrivanje vanjskih napada, jer IDS može vidjeti samo promet koji dolazi sa Interneta. Zbog načina rada preklopnika IDS neće vidjeti promet koji međusobno razmjenjuju računala koja se nalaze iza preklopnika. Korištenje koncentratora za rješavanje problema sa preklopnima ima niz prednosti. Koncentratori s četiri izlaza koji se koriste u ovakvim konfiguracijama su relativno jeftini uređaji i vrlo se lako postavljaju u već postojeću mrežnu topologiju. Postavljanjem koncentratora također nije potrebno mijenjati postavke usmjernika i sigurnosnih stijena. Međutim korištenje koncentratora ima i svoje nedostatke. Ako je veza između usmjernika i preklopnika dvosmjerna, zbog učestalih kolizija paketa propusnost veze može postati bitno smanjena. Kolizije će naročito doći do izražaja ako se uz usmjernik, preklopnik i IDS senzor na četvrti izlaz koncentratora spoji nadzorna konzola IDS-a što će bitno utjecati na tok podataka između usmjernika i koncentratora.

2.3.3. Mrežna priključnica

Za rješavanje prethodno navedenih problema kod rada IDS-a s preklopnicama danas se najčešće koriste posebni mrežni uređaji namijenjeni prvenstveno toj zadaći. Mrežna priključnica (eng. *passive network tap*) je uređaj namijenjen kopiranju prometa u Ethernet, 802.11, FDDI i ATM mrežama. Funkcionalnost uređaja odvija se na fizičkom i podatkovnom sloju, a dodatna je pogodnost što može raditi i na žičanim i na optičkim medijima. Priključnica se obično koristi za nadziranje i analiziranje prometa na najkritičnijim i najzahtjevnijim segmentima mreže iz razloga što kvarovi i gubitak napajanja na uređaju ni na koji način neće utjecati na povezanost i performanse mreže. Ponegdje se u literaturi izraz “tap“ tumači kao kratica od engleskog izraza *Test Access Port*.

Mrežne priključnice su uređaji s četiri izlaza, uobičajeno označene sa *A*, *B*, *A tap* i *B tap*. Izlazi *A* i *B* su podatkovni izlazi i priključuju se izravno na segment mreže koji je potrebno nadzirati, a *tap* izlazi se priključuju na IDS. *Tap* izlazi služe za zrcaljenje podatkovnih izlaza, tako da *tap* izlaz *A* zrcali podatkovni izlaz *A*, a *tap* izlaz *B* zrcali podatkovni izlaz *B*. Mrežna priključnica se u mrežu postavlja na isto mjesto na koje se postavlja i koncentrator.

Prednosti korištenja mrežnih priključnica nad koncentratorom i SPAN izlazom preklopnika su mnogobrojne. Priključnica je tolerantna na ispade u napajanju jer su podatkovni izlazi unutar priključnice spojeni žicom pa ne ovise o izvoru napajanja. Priključnica ne utječe na protočnost podataka između usmjernika i preklopnika, ne zahtjeva promjene u podešenjima ostalih mrežnih uređaja poput sigurnosnih stijena i usmjernika, ne zahtjeva promjenu mreže u cjelini te ne odbacuje pakete sa lošim CRC poljem. Konačno, priključnica ne dozvoljava izravno povezivanje napadača na IDS jer je njezino djelovanje na mreži potpuno transparentno pa pridonosi i cjelokupnoj sigurnosti sustava. Nedostatak korištenja priključnica je njihova relativno visoka cijena te činjenica da se bez dodatnih modifikacija ne mogu se koristiti za nadzor prometa u oba smjera. U slučaju nadzora prometa u oba smjera potrebno je primijeniti udruživanje više kanala u jedan (eng. *channel bonding*) na IDS senzorima koji se priključuju na takve uređaje. Nadalje, IDS koji je priključen na priključnicu mora biti podešen za rad u prikrivenom načinu rada.

2.4. Arhitektura

2.4.1. Uobičajena arhitektura IDS-a

IDS je logički podijeljen na nekoliko zasebnih komponenti. Pojedine komponente sustava međusobno surađuju s zajedničkim ciljem efikasnog otkrivanja pojedinih vrsta napada te generiranja sigurnosnih upozorenja i njihovog spremanja u datoteke dnevnika (eng. *log files*). Komunikacija između pojedinih komponenti ostvarena je razmjenom poruka. Primarna namjena razdvajanja cjelokupnog sustava na više međusobno ovisnih komponenti je jednostavnost razvoja, bolja podjela zadataka unutar sustava te mogućnost da se koriste moduli iz različitih IDS sustava u jednoj heterogenoj cjelini. Pojedine komponente se mogu smještati i na zasebnim računalima čime se ostvaruje mogućnost izrade sustava koji će maksimalno iskoristiti paralelni rad (npr. *clustering* rješenja) i dati najbolje

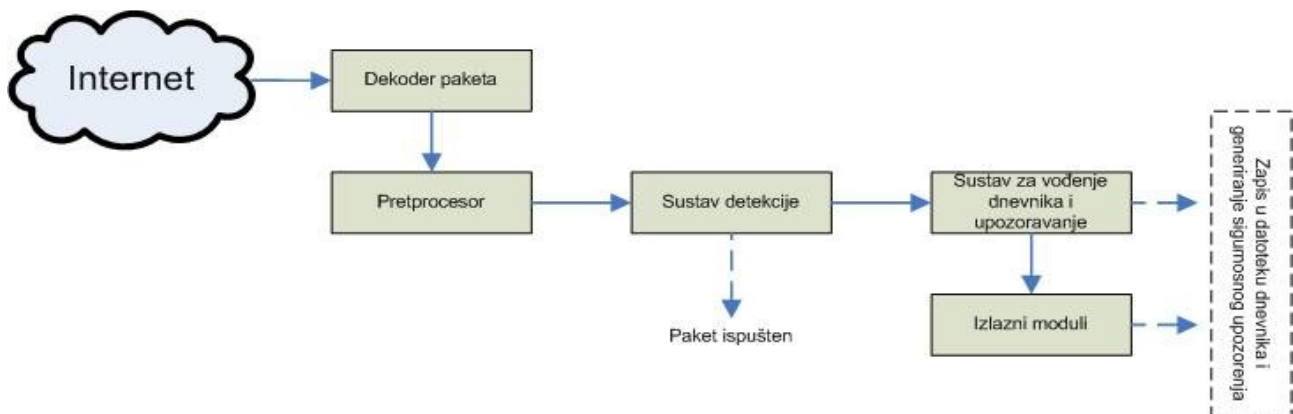
performanse. Nažalost, arhitektura IDS rješenja u stvarnosti dosta varira od sustava do sustava, bilo zbog nedovoljne standardiziranosti i nedostatka standarda ili proizvođačeve namjene da onemogući korištenje pojedinih komponenti i od konkurentskih proizvoda.

Tipične komponente od kojih se sastoji IDS su sljedeće:

- dekodler paketa
- pretprocesor
- sustav detekcije
- sustav za vođenje dnevnika i upozoravanje
- izlazni moduli

Ponegdje se kao zasebni dio IDS-a izdvaja i konzola za nadziranje rada sustava. Konzola za nadziranje sustava služi za praćenje rada i upravljanje pojedinim dijelovima sustava te IDS sensorima. U većini stvarnih implementacija IDS-a konzola za nadziranje sustava nije ništa drugo nego posebno izdvojeni senzor ili svaki senzor ima mogućnost upravljanja ostalim sensorima. Moguća su i grupiranja pojedinih funkcija u jedinstveni modul, primjerice dekodler paketa i pretprocesor mogu biti dio modula za pripremanje mrežnih paketa.

Međusobnu ovisnost pojedinih modula prikazuje Slika 2.3. Tok paketa koji ulazi u IDS prikazan je punom strijelicom. Akcije koje IDS može poduzeti na paketu prikazane su isprekidanom crticom. Na putu od dekodera paketa do izlaznih modula paket može biti ili ispušten ili će se za njega generirati odgovarajuće sigurnosno upozorenje i zapis u datoteci dnevnika.



Slika 2.3: Arhitektura tipičnog IDS sustava

2.4.2. Dekoder paketa

Svi paketi koji dolaze s mreže prvo ulaze u dekodler paketa. Zadaća ovog dijela IDS-a je pripremiti pakete s različitih mrežnih sučelja prije daljnjeg prosljeđivanja pretprocesoru. Dekoder paketa omogućuje da se ostatak sustava koristi na jednak način bez obzira na vrstu sučelja koja se koristi na ulazu u IDS, na primjer Ethernet, SLIP ili PPP. Pošto u IDS dolaze sirovi paketi (eng. *raw packets*) koji će biti različiti ovisno o vrsti sučelja s kojeg dolaze, nužno je ostalim

dijelovima sustava osigurati unificirani format dekodiranog paketa. Bez ovakvog pristupa u projektiranju IDS-a, sustav bi bio vrlo ograničen jer bi se sve komponente kojima dekodirane pakete morali dizajnirati za neku specifičnu vrstu paketa. Umjesto toga, nakon dekodiranja sadržaja dalje se prosljeđuje i koristi vlastiti format paketa.

2.4.3. Pretprocessor

Pretprocessor je vrlo važna komponenta IDS-a koja može preurediti i mijenjati nizove znakova u paketu prije nego što ih sustav detekcije počne detaljnije analizirati. Neke vrste IDS-a su tako dizajnirane da čak i pretprocessor može generirati sigurnosna upozorenja nakon što otkrije pogreške u pojedinim dijelovima paketa. Pogreške koje na paketima može otkriti pretprocessor su vrlo jednostavne i u osnovi se svode na elementarnu provjeru zaglavlja u primljenim podacima. IDS može imati i više od jednog pretprocessora. U takvom modularnom slučaju svaki od pretprocessora obavlja uže specijalizirani dio posla koji inače obavlja cijeli pretprocessor. U različitim kombinacijama distribuiranih i centraliziranih IDS arhitektura obično svaki od senzora ima svoj vlastiti pretprocessor. Prednost IDS-a koji sadrže više pretprocessora je puno lakši razvoj i nadogradnja sustava te smanjenje zahtjeva nad centralnim senzorom u slučaju da je arhitektura IDS-a centralizirana.

Uloga pretprocessora može se najbolje opisati na jednostavnom primjeru otkrivanja napada na hipotetsku ranjivu mrežnu aplikaciju, primjerice na web poslužitelj. Možemo pretpostaviti da će svaki bolji sustav za otkrivanje napada sadržavati pravilo koje će generirati sigurnosno upozorenje ako se tijekom nadgledanja HTTP sjednice i jednom od paketa otkrije pokušaj pristupa važnoj sistemskoj datoteci poput administratorske datoteke *scripts/iisadmin*. Ukoliko sustav detekcije doslovce uspoređuje staze do datoteka s onima navedenima u bazi potpisa, napadač može zaobići otkrivanje napada na nekoliko vrlo jednostavnih načina. Najjednostavniji takvi načini su korištenje zamjenskih ili *escape* znakova te korištenje oznaka “.” za pristup tekućem kazalu i oznake “..” za pristup roditeljskom kazalu:

- `scripts ./iisadmin`
- `scripts / examples ../iisadmin`
- `scripts \iisadmin`
- `scripts /.\iisadmin`

Stvar se može i dodatno zakomplicirati uvođenjem Unicode heksadecimalnih kodova u URI (eng. *Uniform Resource Identifier*). Web poslužitelj će uvijek moći otkriti da se radi o stazi do datoteke *scripts/iisadmin*, međutim ukoliko IDS tumači staze doslovno onda neće moći otkriti ovakve vrste napada. Uloga pretprocessora je upravo rješavanje ovakve vrste problema.

Pretprocessor se također koristi za defragmentaciju paketa. Kada se mrežom prenosi veliki blok podataka paket se uobičajeno razdvaja na više dijelova – fragmenata. Na primjer, najveća veličina bloka podataka koji se može prenositi u Ethernet mrežama je 1500 okteta. Iznos ove veličine se kontrolira pomoću MTU

(eng. *Maximum Transfer Unit*) vrijednosti mrežnog sučelja. Ukoliko se mrežom želi prenositi blok podataka veći od MTU, podaci prvo moraju biti razdvojeni na fragmente veličine manje ili jednake MTU. Nakon primanja svih fragmenata primalac podataka može ponovno sastaviti izvorni blok podataka i očitati izvornu poruku. Fragmentacija paketa je jedan od uobičajenih načina kako napadači pokušavaju zaobići otkrivanje napada i to ne samo kod IDS-a. Prije nego što IDS počne provjeravati da li primljeni paket zadovoljava koji od uvjeta iz baze potpisa, prvo ga mora ponovno sastaviti iz svih njegovih primljenih fragmenata. U slučaju prethodnog primjera s pristupom važnim sistemskim datotekama, jedan dio URI-a bi se mogao nalaziti u jednom, a ostatak u drugom fragmentu. Kada bi se baza potpisa počela pretraživati za svaki fragment zasebno, sustav detekcije ne bi mogao otkriti napad.

Preprocesor može riješiti sve prethodno navedene probleme. U većini danas dostupnih IDS-ova preprocesor je iznimno važna komponenta i obavlja funkcije defragmentacije paketa, dekodiranja URI-a te praćenja TCP veza (eng. *association*) bez kojih IDS ne bi mogao korektno raditi.

2.4.4. Sustav detekcije

Sustav detekcije je najvažniji dio IDS-a. Njegova uloga je provjeravanje da li primljeni paketi zadovoljavaju neki od uvjeta iz baze potpisa i pravila tj. da li je sadržaj paketa u dozvoljenim granicama rada sustava u slučaju da se radi o sustavu koji prepoznaje napade prepoznavanjem anomalija. Baza potpisa i pravila organizirana je u interne strukture podataka u obliku povezanih lista (eng. *chains*). Pravila su u listama grupirana tako da ako se pronađe određeni uzorak u ulaznim podacima onda se aktivira čitavi niz zadovoljenih pravila. Time se izbjegava nepotrebno ponovno ispitivanje za isti uvjet, ali se i omogućava izrada složenijih nizova pravila. Ako paket zadovoljava bilo koje od pravila u bazi potpisa, poduzet će se odgovarajuća akcija koja može biti generiranje sigurnosnog upozorenja, stvaranje zapisa u datoteci dnevnika ili oboje. Ako paket nije zadovoljio niti jedno od pravila, sustav će ga smatrati ispravnim korištenjem sustava te će ga ispustiti.

Sustav detekcije je vremenski najkritičnija i najosjetljivija komponenta IDS-a. Ovisno o tome koliko je brzo računalo na kojem se nalazi te o broju pravila i potpisa, vremenski odziv na različite vrste paketa može jako varirati. Ovo može biti naročito važno ukoliko se od sustava zahtjeva da radi u stvarnome vremenu (eng. *real time*) gdje vrijeme prepoznavanja nikad ne smije prekoračiti unaprijed zadanu granicu. Problem posebno dolazi do izražaja kada sustav radi na mreži s iznimno velikim prometom. U takvim uvjetima sustav neće moći raditi u stvarnome vremenu, a moguća su i situacije gdje će neki paketi biti propušteni a da uopće nisu bili pretraživani za tragove napada. U najgorem slučaju može se dogoditi i da se nestručno postavljen IDS u potpunosti zaguši i da prekine komunikaciju na pojedinim segmentima mreže.

Detekcija se može obavljati na raznim dijelovima paketa:

- zaglavlja na mrežnom sloju (IP, IPX, ICMP, itd.)
- zaglavlja na transportnom sloju (TCP, UDP, itd.)

- zaglavlja na aplikacijskom sloju (DNS, FTP, SNMP, SMTP, NNTP itd.)
- sadržaj paketa (eng. *payload*)

Sustav detekcije može raditi na nekoliko načina. U prvom se načinu za paket koji zadovoljava jedan od uvjeta odmah generira sigurnosno upozorenje i zapis u datoteci dnevnika, a paket se ispušta. Posljedica ovakvog načina rada je činjenica da će se generirati samo jedno sigurnosno upozorenje za paket iako možda zadovoljava uvjete više pravila. Ovakva tehnika će doprinijeti rasterećenju sustava detekcije zato jer se neće morati pretraživati ostatak potpisa, međutim predstavlja ozbiljno ograničenje cjelokupnog sustava ukoliko bi paket zadovoljavao više pravila a prvo pronađeno je najmanje važnosti. Djelomično rješenje ovog problema je i prethodno navedeno grupiranje pravila i liste, međutim to samo umanjuje problem jer će još uvijek postojati grupe međusobno isključivih pravila koja mogu biti aktivirana istim paketom. Drugo rješenje uzima u obzir i ovakve probleme te svakome pravilu pridružuje i odgovarajuću težinu u ovisnosti o opasnosti koju paket može uzrokovati. Nakon što se pronađe prvo odgovarajuće pravilo za paket, detekcija se nastavlja dalje sve dok nisu pretražena sva pravila u bazi potpisa. Nakon što se pretraži cijela baza i sve liste pravila, paket može zadovoljavati i više pravila. Ovisno o tome kako je sustav podešen, kao konačni rezultat pretrage se mogu prikazati sva zadovoljena pravila ili samo ona najvišeg prioriteta.

2.4.5. Sustav za vođenje dnevnika i upozoravanje

Ovisno o tome što sustav detekcije pronađe unutar paketa, sustav za vođenje dnevnika i upozoravanje može generirati sigurnosno upozorenje te odgovarajući zapis u datoteci dnevnika. Datoteka dnevnika sadrži podatke primljene od sustava detekcije i obično se sastoji od sljedećih podataka o aktiviranom pravilu te paketu koji ga je aktivirao:

- vrijeme kada je pravilo aktivirano
- izvorišna i odredišna adresa paketa
- identifikacijski broj i opis pravila
- sadržaj i zaglavlje paketa

Većina IDS sustava pruža dodatne opcije za kontrolu ovog dijela sustava čime se omogućuje filtriranje određenih vrste upozorenja. Mogućnost filtriranja kod generiranja upozorenja je važna jer će u većini slučajeva IDS generirati veliki broj lažnih upozorenja iste vrste te bez razloga puniti datoteke dnevnika i generirati dodatni mrežni promet. Filtriranjem jednostavno možemo ograničiti broj zapisivanja određene vrste pravila u jedinici vremena ili ga u potpunosti zabraniti.

2.4.6. Izlazni moduli

Izlazni moduli su zaduženi za formatiranje podataka primljenih od sustava za vođenje dnevnika i upozoravanje. Ovaj dio IDS-a određuje format datoteke zapisa te metodu dojave sigurnosnog upozorenja. Izlazni moduli su vrlo važna komponenta IDS-a jer omogućavaju rad IDS-a na heterogenim okruženjima. Na primjer, sustav detekcije i ostale kritične komponente se mogu nalaziti na Linux

operacijskom sustavu, dok se zahvaljujući izlaznom modulu koji omogućava stvaranje zapisa pomoću SMB poruka dnevnicima mogu čuvati na računalima koja koriste drukčiji operacijski sustav poput Microsoft Windowsa. Dodatni razlog za različite vrste izlaznih formata dnevničkih zapisa je činjenica da se u većini slučajeva datoteke zapisa ne čitaju direktno nego se dalje analiziraju s različitim alatima. Neki od uobičajenih i najčešćih formata datoteka zapisa te načina dojavljivanja su sljedeći:

- tekstualne datoteke
- XML (eng. *Extensible Markup Language*) datoteke
- slanje SNMP upozorenja (eng. *SNMP trap*)
- korištenje usluga operacijskog sustava (npr. *syslog*)
- korištenje sustava za upravljanje relacijskim bazama podataka
- mijenjanje podešenja sigurnosnih stijena i usmjernika
- slanje SMB (eng. *Service Message Block*) poruka
- slanje upozorenja elektroničkom poštom

2.5. Problemi i nedostaci IDS-ova

Uz brojne prednosti, postoje i određeni nedostaci IDS-ova. Pregledavanjem sadržaja mrežnih paketa te datoteka sa dnevničkim zapisima često se dozvoljeni promet krivo okarakterizira kao nedozvoljeni, odnosno nedozvoljeni kao dozvoljeni. Do ove pojave dolazi uslijed previše općenitih definicija potpisa pojedinih vrsta napada. IDS-ovi uglavnom ne mogu zaustaviti ili usporiti aktivne mrežne napade. Zbog mogućih dojava o lažnim napadima na sustav, nije niti je poželjno da IDS-ovi prekidaju uspostavljene veze jer bi to rezultiralo velikim brojem neopravdanih prekida u radu. Posljedica činjenice da IDS-ovi ne mogu prekinuti napade je generiranje onoliko sigurnosnih upozorenja koliko ima dnevničkih zapisa o samim napadima. Pošto se ponekad i ispravni promet neopravdano prepoznaje kao napad, količina upozorenja za systemske administratore se time dodatno povećava.

Sljedeći problem je vremenski raskorak između napada i otkrivanja napada. Iz razloga što IDS mora analizirati veliki broj zapisa te moguće potrebe za naknadnom provjerom dojava o napadu od strane systemskog administratora, IDS-ovi pružaju spor odgovor na napade na računalne i mrežne resurse. Još jedan od nedostataka IDS-ova je činjenica da oni ne mogu otkriti nove vrste napada, nego mogu ustanoviti samo već postojeće. Kako ovakvi sustavi uglavnom uspoređuju sadržaje dnevničkih zapisa i mrežnih paketa s definiranim potpisima za napade, novi napadi nisu sadržani u bazi potpisa pa ih IDS niti ne prepoznaje. Otkrivanje nepoznatih vrsta napada moguća je samo ukoliko je IDS zasnovan ili kombiniran sa metodom prepoznavanja statističkih anomalija.

Na kraju, u zadnje vrijeme je prisutna sve veća tendencija kriptiranja prometa koji prolazi kroz mrežu. Sve više mrežnih usluga razmjenjuje poruke preko sigurnih kanala i virtualnih privatnih mreža (VPN). IDS-ovi imaju velikih problema u nadziranju veza zaštićenih IPsec rješenjima ili ostvarenih putem npr. SSH

tuneliranja. Problem postaje sve veći, a pogotovo dolazi do izražaja kada se uzme u obzir da će prelaskom s IPv4 na IPv6 gotovo sav promet postati kriptiran.

Zbog spomenutih nedostataka očito je da je nužno razinu sigurnosti pomaknuti s otkrivanja zlonamjernih aktivnosti na njihovo sprečavanje, pa je tako došlo do razvoja sustava za sprečavanje mrežnih napada (eng. *Intrusion Prevention System* – *IPS*). IPS-ovi omogućavaju preventivno djelovanje glede računalnih napada i izbjegavanje nastajanja štete koja bi bila prouzročena tim napadima.

2.6. Zaštita IDS- a

IDS je poput svake ostale komponente na mreži podložan raznim vrstama napada. Uobičajene su dvije mjere zaštite IDS-a: rad na prikrivenom sučelju i rad na sučelju bez adrese.

Sučelja na kojima radi IDS obično su podešena i za primanje i za slanje podataka, te rade u takvome načinu rada da primaju i pakete koji nisu namijenjeni direktno njima (eng. *promiscuous mode*). Prikriveno sučelje (eng. *stealth interface*) je mrežno sučelje na kojem IDS senzor može samo primati podatke, ali ih ne može slati nazad. Takvo sučelje se može ostvariti na dva načina. Prvi način je korištenje mrežne priključnice koja sama po sebi ne dozvoljava vraćanje prometa, a drugi je korištenje posebne vrste UTP kabela (eng. *sniffing cable*). Ova vrsta kabela se može napraviti od običnog UTP kabela na nekoliko načina. Najjednostavnije je na jednom kraju kabela kratko spojiti linije 1 i 2, a linije 3 i 6 se spoje s istim tim linijama na drugom kraju kabela. Umjesto da se linije 1 i 2 kratko spoje, između njih se može umetnuti kondenzator koji će služiti kao visoko propusni filter. Iznos kondenzatora se može izračunati izrazom $f = 1 / 2 RC$ i iznosi 150 pF za 10 megabitni te 15 pF za 100 megabitni Ethernet. Postoje i naprednije izvedbe prikrivenog sučelja poput UTP Y-kabela koji je detaljnije opisan na web adresi <http://www.snort.org/docs/faq/1Q05/node31.htm>.

Druga metoda zaštite IDS-a je rad na sučelju bez dodijeljene IP adrese. Ako IDS senzor nema dodijeljenu IP adresu tada se na njega ne može pristupiti izvana. Postavljanje ovakvog sučelja je vrlo jednostavno i svodi se na podizanje sučelja bez adrese i onemogućavanje protokola poput DHCP-a. Senzor obično ima dva sučelja od kojih je jedno, ono prema Internetu, bez IP adrese ili ostvareno putem prikrivenog sučelja, dok je drugo prema lokalnoj mreži ili konzoli za upravljanje, sa dodijeljenom IP adresom.

Još jedna popularna metoda zaštite su i transparentni IDS prenosnici (eng. *bridges*). Takvi sustavi sadrže dva mrežna sučelja i mogu se postaviti na bilo koje mjesto u lokalnoj mreži što ne zahtjeva dodatne promijene u mrežnoj topologiji ili već postojećim konfiguracijama usmjernika. Transparentni IDS prenosnici kombiniraju korištenje Ethernet prenosnika s IDS-om i sigurnosnom stijenom što ima niz prednosti nad npr. implementacijom sigurnosne stijene na usmjerniku. Pošto se u takvim konfiguracijama sigurnosna stijena i IDS nalaze na prenosniku, a ne na usmjerniku, to ih čini nevidljivima izvan mreže i stoga ih štiti od napada. Svaki paket prolazi kroz prenosnik sve dok IDS ne pronađe potpis napada, a u tom slučaju daju se instrukcije sigurnosnoj stijeni koja zatim blokira sve pakete računala s kojeg dolazi napad.

2.7. Standardizacija

Jedan od prvih pokušaja da se standardiziraju protokoli i aplikacijska programska sučelja (eng. *Application Programming Interface* – API) koja se koriste u IDS sustavima rezultirao je stvaranjem zajedničkog IDS okvira znanog kao CIDF (eng. *Common Intrusion Detection Framework*). Glavni cilj CIDF-a je pojednostavniti dijeljenje informacija i resursa između različitih IDS sustava te omogućiti da se njihove pojedine komponente mogu ponovno iskoristiti u drugim sustavima. Sam CIDF definiran je u četiri Internet nacrtu (eng. *draft*):

- “*The Common Intrusion Detection Framework Architecture*”
- “*Communication in the Common Intrusion Detection Framework*”
- “*A Common Intrusion Specification Language (CISL)*”
- “*CIDF APIs: Their Care and Feeding*”

Prvi nacrt je najvažniji i opisuje temeljne pojmove u CIDF terminologiji te daje detaljni opis CIDF arhitekture. Ostali dokumenti definiraju protokole, format poruka i programska sučelja putem kojih pojedine komponente iz CIDF arhitekture IDS-a mogu međusobno komunicirati na siguran i efikasan način te obavljati međusobne provjere autentičnosti i raspoloživosti. CIDF opisuje IDS kao sustav koji sačinjavaju četiri diskretne komponente. Komponente su sljedeće:

- generatori događaja (eng. *event generators, E – box*)
- analizatori događaja (eng. *event analyzers, A – box*)
- baze podataka o događajima (eng. *event databases, D – box*)
- jedinice za odgovaranje na događaje (eng. *response units, R – box*)

Sve četiri komponente međusobno komuniciraju razmjenom poruka (eng. *message passing*) standardiziranog formata i korištenjem jezika CISL. S vremenom je CISL koji je prilično kompliciran zamijenjen jednostavnijim oblikom poruka za razmjenu podataka poznatim pod imenom IDMEF (eng. *Intrusion Detection Message Exchange Format*). Uloga generatora događaja je prikupljati podatke s mreže i slati poruke o događajima ostalim komponentama sustava. Analizatori događaja su komponente na koje se uobičajeno misli kada se govori o IDS-ovima. Njihova uloga je primiti poruke od ostalih komponenti, analizirati ih te vratiti poruke koje predstavljaju sažetak analize ulaznih poruka. Baza podataka o događajima i jedinice za odgovaranje na događaje imaju sličnu namjenu kao i baza potpisa i izlazni moduli u uobičajenim IDS arhitekturama.

Neke od ideja koje su proizašle iz CIDF projekta potakle su nastanak IETF radne grupe (eng. *Internet Engineering Task Force Working Group*) koja je dobila naziv *Intrusion Detection Working Group*. Glavni cilj ove radne grupe je predstaviti ideje CIDF projekta široj javnosti.

Svi pokušaji standardizacije nastali u okviru IETF-ovih radnih grupa i CIDF projekta datiraju još iz 1998. godine i nisu rezultirali nekim pretjeranim uspjehom niti su ikad zaživjeli u nekom od popularnijih dostupnih IDS-ova, iako se neke ideje iz tih nacrtu i danas koriste u promijenjenom obliku. Umjesto ovakvih pravnih i službenih standarda u većini današnjih IDS-ova prisutni su

neformalni (*de facto*) standardi proizašli iz nekih od najpopularnijih IDS programa.

3. Praktični rad

3.1. Snort IDS

Snort je danas najpopularniji sustav za otkrivanje i prevenciju mrežnih napada i *de facto* industrijski standard. Program koristi vlastiti *engine* koji koristi jezik temeljen na događajima (eng. *rule driven language*). Otkrivanje napada se obavlja kombiniranjem tehnika prepoznavanja potpisima i prepoznavanja anomalija. Snort se razvija prema principima otvorenog programskog kôda i slobodno je dostupan s web adrese <http://www.snort.org>.

Snort može raditi u nekoliko načina rada:

- sniffer
- packet logger
- NIDS
- inline

U *sniffer* načinu rada (`snort -v`) program čita mrežne pakete i prikazuje ih na odabranoj konzoli (konzoli u Unix kontekstu – *tty* ili *pty*), slično poput programa *tcpdump*. Ispis je moguć s raznim opcijama poput ispisa cijelog paketa, ispisa zaglavlja, ispis samo određenih vrsta paketa, selekciju protokola itd. *Packet logger* način rada (`snort -l`) u osnovi djeluje vrlo slično, ali pakete ne ispisuje na konzolu nego u datoteku dnevnika. Zapisivanje u datoteke dnevnika je dostupno u dva osnovna formata: tekstualnom i binarnom. *NIDS* način rada je najsloženiji, ali i najmoćniji način rada u kojem program provodi punu analizu primljenih paketa i u ovisnosti o njihovom sadržaju provodi nekoliko dodatnih akcija. U novijim verzijama dostupan je i *inline* način rada u kojem program ne čita pakete preko standardne biblioteke *libpcap* (eng. *packet capture library*) nego preko *netfilter* i *iptables* infrastrukture. *Netfilter* je dio Linux jezgre koji služi za primanje paketa i provođenje raznih operacija na paketima, dok je *iptables* korisnički program koji služi za upravljanje *netfilter* pravilima.

Snort dnevnički zapisi, ukoliko se zapisuju kao obične tekstualne datoteke, dijele jedinstveni način zapisivanja podataka. Na primjer:

```
[**] [122:19:0] (portscan) UDP Portsweep [**]  
09/08-06:34:55.414399 161.53.65.175 -> 85.169.202.17  
PROTO255 TTL:0 TOS:0x80 ID:2677 IpLen:20 DgmLen:167 DF
```

Prva linija je informacijska i nalazi se unutar delimitera `[**]` radi lakšeg pretraživanja datoteka programima poput alata *grep* i automatiziranja analize zapisa pomoću raznih skriptnih programskih jezika. Informacijska linija sadrži tri broja i kratak opis aktiviranog pravila. Brojevi označavaju identifikator generatora, identifikator potpisa te identifikator revizije potpisa. Identifikator generatora (eng. *generator ID*, *GID*) govori koja je komponenta Snorta generirala sigurnosno upozorenje i obično označava grupu međusobno sličnih pravila.

Identifikator potpisa (eng. *signature ID, SID*) je jedinstveni broj pridružen svakom potpisu napada unutar neke grupe potpisa koju identificira *GID*. Broj revizije govori koja je trenutna dostupna verzija pravila. Početni broj revizije pravila je nula, a svakim nadograđivanjem definicije pravila broj se inkrementira. Nakon informacija o aktiviranom pravilu slijede informacije o vremenu i datumu aktivacije, određenoj i izvornoj mrežnoj adresi i pristupnim vratima te pojedine informacije iz zaglavlja mrežnih paketa vezane uz samu prirodu napada za koju se zapisuje dnevnički zapis. Za pisanje vlastitih pravila i potpisa koristi se posebni programski jezik koji je detaljno opisan u službenoj programskoj dokumentaciji.

Prava snaga Snorta leži u činjenici da može raditi i kao samostalni *daemon* i kao dio kompletnog NIDS sustava. Ako radi kao samostalni *daemon*, podaci o napadima se zapisuju u običnu tekstualnu ili binarnu datoteku i mogu se kasnije pogledati tekst uređivačem ili u slučaju binarnih datoteka uz pomoć specijalnih programa poput *barnyard*. Takva jednostavna instalacija omogućuje i slanje SNMP upozorenja na SNMP upravitelje kao i SMB poruka na računala koje rade s Microsoft Windows operacijskim sustavom. Međutim, ako su uz instalaciju Snorta dostupni i dodatni alati, moguće su i naprednije metode rada poput zapisivanja podataka o napadima u relacijske baze podataka te pregled i analiza tih podataka putem web sučelja. Popis dodatnih programa koji se mogu koristiti sa Snortom je iz dana u dan sve veći, a uobičajeno kompletno Snort okruženje obično obuhvaća sljedeće programe i programske biblioteke:

- MySQL – sustav za upravljanje relacijskim bazama podataka koji služi za zapis podataka o napadima. Umjesto MySQL-a može se koristiti bilo koji drugi sustav za upravljanje relacijskim bazama podataka koji ima odgovarajuće ODBC (eng. *Open Database Connectivity*) upravljačke programe, npr. PostgreSQL ili Oracle
- Apache – web poslužitelj putem kojeg će biti dostupna web aplikacija za pregled i analizu podataka o napadima
- PHP – programski jezik, koristi se kao sučelje za komunikaciju između Apachea i MySQL baze podataka te kao jezik za izgradnju web aplikacije za upravljanje sustavom i analizu podataka
- ACID – PHP paket namijenjen analizi Snort dnevničkih zapisa
- GD – grafička biblioteka, koristi je ACID za stvaranje grafova
- PHPLOT – PHP modul za predstavljanje podataka u obliku grafikona
- ADODB – služi ACID-u za povezivanje na MySQL bazu
- oinkmaster – program za automatizirano obnavljanje baze potpisa napada

3.2. Ispitno okruženje i ispitivanje napada

Za potrebe ispitivanja mogućnosti otkrivanja različitih vrsta napada u okviru ovog seminarskog rada, instalirano je jednostavno Snort okruženje. Ispitna platforma sastoji se od sljedećih komponenti:

- zavodsko računalo (`taurus.zemris.fer.hr`) koje služi primarno kao pristupna točka za lokalnu bežičnu mrežu
- operacijski sustav Debian GNU/Linux 3.1 (Sarge)
- procesor Pentium III, 735 MHz, 376MB fizičke memorije i 430MB swapa
- tri 100 Mbit Ethernet sučelja, Snort sluša samo na jednom sučelju (izlaz na Internet)
- filtriranje paketa obavlja se skupom iptables pravila
- Snort verzije 2.3.2.12 u NIDS načinu rada, instaliran iz standardnog Debian repozitorija paketa, bez ikakvih posebnih dodataka za analizu dnevnih zapisa ili upravljanja pravilima te podešenim načinom zapisivanja sigurnosnih upozorenja u obične tekstualne datoteke

Ispitivanje je zamišljeno tako da se pokuša niz napada na ispitno računalo, te ispita razne mogućnosti otkrivanja postupno sve sofisticiranijih napada. Ispitivanja su radi jednostavnosti i preglednosti podijeljena u nekoliko logičkih cjelina - scenarija, a svaki scenarij se sastoji od više testova. Testovi unutar jednog ispitnog scenarija su vrlo slični, a svaki sljedeći test unutar istog scenarija koristi sve bolje i efikasnije tehnike (npr. prikriivanje ili fragmentiranje paketa). Kao izvor ideje za većinu vrsta napada u opisanim scenarijima poslužili su razni *security portali* poput Securityfocus ili CERT-a, ali i razni *hackerske* stranice poput npr Milw0rma. Za ispitivanje su se koristili standardni alati koji se mogu pronaći na većini računala s instaliranim Linux operacijskim sustavom, poput alata nmap, hping2, netcat itd.

Ispitni scenariji i testovi su sljedeći:

1. Skeniranje pristupnih vrata
 - TCP connect() skeniranje
 - SYN prikriveno (*stealth*) skeniranje
 - FIN skeniranje
2. Napadi shell kôdovima
 - `exec() + /bin/sh` shell kôd
 - iptables -F shell kôd
3. Zaobilaženje otkrivanja
 - napadi fragmentiranim paketima

3.3. Rezultati ispitivanja

3.3.1. Skeniranje pristupnih vrata

Cilj skeniranja pristupnih vrata je ustanoviti da li je na određenim pristupnim vratima aktivna neka usluga, tj. ustanoviti da li su pristupna vrata "otvorena". Skeniranje pristupnih vrata prvi korak u kompromitiranju određene usluge, a

nakon njega obično slijedi utvrđivanje verzije usluge koja je pokrenuta na tim pristupnim vratima te na kraju iskorištavanje poznatih propusta te verzije usluge.

TCP connect() skeniranje je najjednostavnija vrsta skeniranja koja koristi kompletnu uspostavu veze između inicijatora i poslužitelja (eng. *three way handshake*). Inicijator pokreće uspostavu veze slanjem TCP paketa s uključenom opcijom SYN. Na to poslužitelj potvrđuje primljeni zahtjev tako da vrati paket s uključenim opcijama SYN i ACK, a veza je potpuno uspostavljena nakon što i inicijator potvrdi primljeni SYN/ACK paket s novim paketom koji ima uključenu ACK opciju. Za skeniranje ovim postupkom poslužiti ćemo se alatom *nmap*:

```
$ nmap -P0 -sT taurus.zemris.fer.hr
Starting Nmap 4.11 ( http://www.insecure.org/nmap/ ) at 2006-09-15
21:12 CEST
Interesting ports on Taurus.zemris.fer.hr (161.53.65.175):
Not shown: 1678 filtered ports
PORT      STATE SERVICE
22/tcp    open  ssh
Nmap finished: 1 IP address (1 host up) scanned in 350.411 seconds
```

Opcija *-sT* govori *nmap*-u da koristi TCP connect() skeniranje, a opciju *-P0* koristimo zato da se ne provjerava da li je određeno računalo raspoloživo slanjem ICMP poruka. Određeno računalo ignorira ICMP poruke jer je tako navedeno u skupu iptables pravila. Ukoliko bi pokušali skenirati bez *-P0* alat bi zaključio da je računalo nedostupno i prekinuo daljnje ispitivanje. Ovakva vrsta skeniranja traje relativno dugo jer *nmap* ne radi vlastite (*raw*) pakete nego koristi programsko sučelje operacijskog sustava, te je iznimno lako za otkriti čak i ako se ne koristi IDS pošto će sistemski dnevnički zapisi biti puni poruka o pokušaju spajanja na sva poznatija pristupna vrata.

Nakon pregleda Snortovih dnevničkih zapisa ustanovljeno je da je sustav detekcije uspješno prepoznao ovu vrstu skeniranja:

```
[**] [122:1:0] (portscan) TCP Portscan [**]
09/15-21:39:10.671166 83.131.95.179 -> 161.53.65.175
PROTO255 TTL:0 TOS:0x80 ID:34487 IpLen:20 DgmLen:164 DF
```

TCP connect() skeniranje je neučinkovito i vrlo jednostavno za primijetiti jer se sekvencijalno pokušava uspostaviti veza s velikim brojem pristupnih vrata. Stoga se češće koristi SYN prikriveno skeniranje. SYN skeniranje se razlikuje od prethodno opisane metode po tome što se neće uspostaviti veza između inicijatora i poslužitelja. Čim inicijator primi SYN/ACK paket kao potvrdu na poslani SYN paket, može biti siguran da su pristupna vrata otvorena. Takvu vrstu skeniranja je nešto teže otkriti. Za potrebu ovog testa također je korišten *nmap*, no sada je umjesto *-sT* korištena opcija *-sS*.

3. Praktični rad

```
# nmap -P0 -sS taurus.zemris.fer.hr
Starting Nmap 4.11 ( http://www.insecure.org/nmap/ ) at 2006-09-15
21:40 CEST
Interesting ports on Taurus.zemris.fer.hr (161.53.65.175):
Not shown: 1011 filtered ports, 666 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh

Nmap finished: 1 IP address (1 host up) scanned in 43.370 seconds
```

Zapis u datoteci dnevnika je isti kao i u prethodnom slučaju (Snort ne pravi razliku između SYN i connect() skeniranja) te otkriva da je Snort uspješno prepoznao i ovu vrstu skeniranja. Kao treću vrstu skeniranja primijenit ćemo FIN skeniranje. FIN skeniranje šalje paket s uključenom FIN opcijom što bi prema RFC-u 793 koji opisuje TCP protokol trebalo rezultirati vraćanjem paketa s uključenom RST opcijom ako su pristupna vrata otvorena. Ako su pristupna vrata zatvorena neće se vraćati nikakav paket. Za obavljanje FIN skeniranja ponovno se koristi nmap, ali ovaj puta sa -sF opcijom. Kao rezultat, pojavljuje se novi zapis u datoteci dnevnika koji potvrđuje da Snort prepoznaje i ovu vrstu skeniranja:

```
[**] [1:621:7] SCAN FIN [**]
[Classification: Attempted Information Leak] [Priority: 2]
09/15-21:57:28.640256 83.131.95.179:36087 -> 161.53.65.175:1425
TCP TTL:28 TOS:0x80 ID:17017 IpLen:20 DgmLen:40
*****F Seq: 0x18AADB04 Ack: 0x0 Win: 0x800 TcpLen: 20
[Xref => http://www.whitehats.com/info/IDS27]
```

Dodatni zanimljiv detalj u ovom isječku iz Snort datoteke dnevnika je zadnja linija koja daje referencu o detaljima za pojedinu vrstu napada.

3.2.1. 3.3.2. Napadi shell kôdovima

Shell kôd je maliciozan programski kod, obično pisan u assembleru, koji iskorištava propuste u ranjivim mrežnim uslugama. Shell kôd radi na sljedećem principu: ranjive usluge su prevedene iz izvornog kôda pisanog u jeziku C koji koristi funkcije koje ne provjeravaju granice spremnika u koje upisuju podatke poput strcpy() ili sprintf(). Zbog načina na koji C programi oblikuju okvir stoga, ukoliko takve funkcije pokušaju pisati više podataka nego što stane u međuspremnik to će rezultirati prebrisanim sadržajem na stogu smještenih registara EBP (pokazivač okvira stoga) i EIP (programsko brojilo). Prebrisavanjem sadržaja spremljenog registra EIP napadač može preusmjeriti tok programa na proizvoljnu adresu, što će mu omogućiti da izvrši shell kôd i ostvari pristup ljusci operacijskog sustava na napadnutom računalu, ugasi sigurnosnu stijenu, otvori neka pristupna vrata itd.

Svi shell kôdovi korišteni za potrebe ove grupe testova otkrivanja preuzeti su sa stranice <http://www.milw0rm.com/shellcode/linux/x86>. Za slanje shell kôda korišten je program netcat, a kao odredište SSH pristupna vrata (tcp/22). Potrebno je napomenuti da ovakvo jednostavno slanje neće rezultirati uspješnim napadom pošto je potrebno ispred samog shell kôda umetnuti odgovarajući broj NOP instrukcija i offset adresu da bi početna adresa kôda došla točno u EIP registar na stogu.

Prvi primjer je slanje najjednostavnijeg oblika shell kôda koji pokreće ljusku /bin/sh na napadnutim pristupnim vratima:

```
$ echo
"\xeb\x1f\x5e\x89\x76\x08\x31\xc0\x88\x46\x07\x89\x46\x0c\xb0\x0b\x89\x
f3\x8d\x4e\x08\x8d\x56\x0c\xcd\x80\x31\xdb\x89\xd8\x40\xcd\x80\xe8\xdc
\xff\xff\xff/bin/sh" | nc taurus.zemris.fer.hr 22
```

Snort je ponovno uspješno prepoznao napad:

```
[**] [1:1324:6] EXPLOIT ssh CRC32 overflow /bin/sh [**]
[Classification: Executable code was detected] [Priority: 1]
09/15-22:21:55.247962 83.131.95.179:60771 -> 161.53.65.175:22
TCP TTL:51 TOS:0x80 ID:47090 IpLen:20 DgmLen:60 DF
***AP*** Seq: 0xDCBE7E12 Ack: 0xD9B00576 Win: 0x5AC TcpLen: 32
TCP Options (3) => NOP NOP TS: 4573681 716067694
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=2001-0572][Xref
=> http://cve.mitre.org/cgi-bin/cvename.cgi?name=2001-0144][Xref =>
http://www.securityfocus.com/bid/2347]
```

Međutim, ukoliko se na odredišna pristupna vrata pošalje samo niz znakova “/bin/sh”, Snort će također prijaviti da je u tijeku napad. Očito je da Snort ima definirano pravilo koje će se aktivirati uvijek kada se pojavi karakteristični niz znakova na ssh pristupna vrata. Ukoliko se pokuša isto s nekim pristupnim vratima na kojima inače sluša neka usluga koja komunicira preko nekriptiranog kanala, npr FTP, sustav neće prijaviti da je u tijeku napad:

```
$ echo "/bin/sh" | nc taurus.zemris.fer.hr 21
```

Ovo je očekivano ponašanje pošto je promet na ssh pristupnim vratima inače kriptiran, a pokušali smo slati čisti ASCII tekst s karakterističnim nizom znakova koji se nalazi unutar shell kôdova. Još jedan primjer shell koda je i sljedeći koji po aktiviranju spušta zaštitu sigurnosnom stijenom tako da izvrši naredbu iptables -F:

```
$ echo
"\x31\xd2\x52\x66\x68\x2d\x46\x89\xe6\x52\x68\x62\x6c\x65\x73\x68\x69\x
70\x74\x61\x89\xe7\x68\x62\x69\x6e\x2f\x68\x2f\x2f\x2f\x73\x89\xe3\x5
2\x56\x57\x89\xe1\x31\xc0\xb0\x0b\xcd\x80" | nc taurus.zemris.fer.hr
22
```

Ovoga puta Snort nije uspio detektirati napad. Ovo je posljedica činjenice da se u bazi potpisa ne nalazi uzorak ovakvog napada. Ovaj primjer također pokazuje i nedostatak otkrivanja prepoznavanjem potpisa jer je IDS toliko učinkovit koliko mu je ažurna i velika baza potpisa. Za što bolju učinkovitost prepoznavanja napada preporuča se što češće i redovito obnavljati bazu potpisa koristeći neki od alata za automatsku sinkronizaciju pravila poput *oinkmastera*.

3.3.3. Zaobilazanje otkrivanja

Jedna od najčešćih metoda kojom napadači zaobilaze otkrivanje napada je korištenje fragmentiranja paketa. Za ovu potrebu ponovno je provedeno

3. Praktični rad

skeniranje pristupnih vrata pomoću nmapa, ali ovaj puta koristeći XMAS skeniranje ie korištenjem fragmentiranja paketa. Ideja korištenja fragmentiranja za zaobilaznje zaštite je sljedeća: pošto su TCP paketi enkapsulirani unutar IP paketa, ako se koristi fragmentiranje tada će se TCP zaglavlje nalaziti razdvojeno u više IP paketa. Nmap može razdvojiti pakete u fragmente duljine samo 8 okteta. XMAS skeniranje koje se koristi u ovom primjeru slično je FIN skeniranju samo što šalje pakete s uključenim opcijama PUSH, URG i FIN. Skenirano računalo bi trebalo odgovoriti paketom s uključenom RST opcijom ako su pristupna vrata otvorena, a ikakav odgovor bi trebao izostati u slučaju da su pristupna vrata zatvorena,

```
# nmap -P0 -sX -f taurus.zemris.fer.hr
```

I ovaj napad je uspješno otkriven, unatoč korištenju tehnike fragmentiranja:

```
[**] [1:1228:7] SCAN nmap XMAS [**]
[Classification: Attempted Information Leak] [Priority: 2]
09/15-22:43:45.045016 83.131.95.179:48209 -> 161.53.65.175:4662
TCP TTL:25 TOS:0x80 ID:58364 IpLen:20 DgmLen:40
**U*P**F Seq: 0xE64823BC Ack: 0x0 Win: 0xC00 TcpLen:20 UrgPtr:0x0
[Xref => http://www.whitehats.com/info/IDS30
```

Kao dodatni primjer možemo navesti i slanje fragmentiranih neispravnih paketa poput onih koji imaju uključenu i FIN i SYN opciju. U tu svrhu poslužiti će nam alat hping koji može oblikovati proizvoljne pakete:

```
# hping2 -f -S -F -p 22 taurus.zemris.fer.hr
```

Snort je ponovno prepoznao napad i klasificirao ga kao SYN FIN skeniranje. Ovakve vrste napada obično se koriste za jednostavnu detekciju sustava za prevenciju mrežnih napada i prisutnosti sigurnosne zaštitne stijene koja obično filtrira ovakve vrste neispravnih paketa.

IDS nikad ne može prepoznati sve vrste napada, a napadači različitim tehnikama mogu zaobići otkrivanje. Za primjer može služiti prikriveno skeniranje korištenjem hping- a. Prikriveno skeniranje radi na principu nadgledanja identifikacijskih brojeva unutar IP paketa. Primjerice želimo otkriti da li su na računalu faramir otvorena pristupna vrata tcp/22. Za prikrivanje (eng. *spoofing*) ćemo koristiti računalo taurus. Početak ove tehnike je slanje paketa na računalo za prikrivanje:

```
# hping -p 22 -A -r taurus.zemris.fer.hr
len=40 ip=161.53.65.246 ttl=51 DF id=46917 sport=22 flags=R seq=0
win=0 rtt=60.8 ms
len=40 ip=161.53.65.246 ttl=51 DF id=+1 sport=22 flags=R seq=1 win=0
rtt=60.1 ms
```

Ostavimo li da se naredba i dalje izvršava vidimo da se ID povećava uvijek za 1 (važno je računalo preko kojeg se sakrivamo nema puno mrežnog prometa da bi lakše pratili identifikacijske brojeve). Ukoliko u drugom terminalu izvršimo:

```
# hping -a taurus.zemris.fer.hr -S -p 22 -i u10000 faramir
```

i ostavimo da se izvršava nekoliko sekundi, ukoliko su skenirana vrata otvorena ID će porasti za točno toliki broj koliko smo poslali paketa s drugom naredbom. IDS je nemoćan u otkrivanju pravog izvora ovakvih skeniranja jer će otkriti samo adresu preko koje smo preusmjerili skeniranje,

4. Zaključak

Sustavi za otkrivanje mrežnih napada su još uvijek tehnologija koja se izrazito brzo razvija i poboljšava. S vremenom su postali nezamjenjivi dio cjelokupnog sigurnosnog sustava mnogih tvrtki i toliko ih je uobičajeno naći koliko je uobičajeno da se koristi i sigurnosna stijena. Međutim, brzim razvojem tehnologije NIDS-ovi se također moraju prilagođavati sve većim zahtjevima tržišta i moraju odolijevati sve sofisticiranijim tehnikama napada. U samo dvadesetak godina koliko postoje evoluirali su od najprimitivnijih metoda nadziranja do današnjih nevidljivih distribuiranih sustava za rad u stvarnom vremenu koji mogu nadzirati i mreže s najvećim prometom.

Danas je gotovo nemoguće govoriti isključivo o NIDS-u da se ne spomene u kontekstu složenijih sustava poput sustava za prevenciju napada (IPS). Kao što je prije navedeno, otkrivanje napada je samo po sebi prilično beskorisno, pa je za očekivati da će se industrija u budućnosti u potpunosti orijentirati razvoju sustava za prevenciju napada i razvoju integriranih sustava koji će objedinjavati sva popularnija rješenja unutar jednog programskog paketa. Također, danas sve je manja razlika između računalno i mrežno temeljenih sustava za detekciju, tako da su sve češća integrirana rješenja koja objedinjuju i ove dvije tehnike. Postoje čak i ekstremna rješenja kod kojih se neki koncepti sustava za otkrivanje napada koriste i u ojačavanju jezgre operacijskog sustava. Primjer takvoga rješenja je projekt *LIDS*.

Pred sustavima za otkrivanje mrežnih napada je nekoliko velikih izazova. Jedan od najvećih će zasigurno biti riješiti problem nadziranja kriptiranih kanala koji se sve češće koriste, a problem će kulminirati potpunim prelaskom na IPv6. Kao što smo pokazali, postoji i niz dobro poznatih tehnika kojima je moguće zaobići otkrivanje od strane NIDS-a pa ostaje i mnogo mjesta za poboljšavanje metoda otkrivanja. Jedan od čestih prigovora na implementaciju NIDS rješenja unutar lokalne mreže ide račun na ukupne cijene postavljanja i održavanja (eng. *total cost of ownership, TCO*). Čak i u slučajevima kada se koriste Open Source rješenja, a ne komercijalne inače IDS-a, postoje relativno veliki dodatni troškovi koji posebice rastu ukoliko je potrebno napraviti robusno rješenje u danas već standardnim mrežama koje se temelje na radu s preklopnim. U takvim slučajevima je potrebno intenzivno koristiti mrežne priključnice koje su još uvijek vrlo skupi uređaji. Nadalje, problem predstavlja i sve veći rast propusnosti mreža.

Bez sumnje, sadašnja NIDS rješenja će još dugo vremena biti nezamjenjivi dio mreže na sigurnosnom planu, a razvijaju se i sve bolje verzije programa. Rastući broj sigurnosnih problema ide samo u korist ovoj tvrdnji. Posljednji trendovi u razvoju NIDS-a su sve veća upotreba matematičkih koncepata te uvođenje umjetne inteligencije (eng. *artificial intelligence, AI*) u proces otkrivanja. Neki proizvođači su u njihove produkte (npr. ISS) počeli ugrađivati tehnologije kojima će sustav moći modificirati sam sebe na temelju prethodno dobivenih informacija o napadima, a sustav neće više biti pasivan tj. neće više samo slušati na nekom mrežnom sučelju nego će provoditi aktivnu analizu mreže poput traženja propusta i skeniranja pojedinih segmenata mreže. Primjer ovakvih modernih

rješenja su i Cisco *self – aware* ili *self – defending* mreže koje u sebi imaju integrirane i IDS i IPS sustave. Takve mreže pružaju integrirano sigurnosno rješenje na svim aspektima zaštite i sposobne su se samostalno prilagoditi novim prijetnjama. Budućnost svakako donosi još puno posla za poboljšavanje NIDS-ova koji su nedvojbeno već i sada tehnologija koju mora imati implementiranu svatko tko drži do sigurnosti vlastitih podataka.

5. Literatura

1. R. Rehman, *Intrusion Detection Systems with Snort*, Prentice Hall PTR, Upper Saddle River, New Jersey, 2003.
2. J. McGibney, *Intrusion Detection Systems & Honeypots*, Waterford Institute of Technology, Ireland, Prezentacija sa konferencije INET/IGC, Barcelona, 2004
3. B. Laing, *Implementing a Network Based Intrusion Detection System*, Sovereign House, Reading, 2000.
4. CARNet CERT i LS&S, *Sustavi za sprečavanje neovlaštenih aktivnosti*, CCERT-PUBDOC-2006-01-145, Zagreb, 2006.
5. Sans Institute, Intrusion Detection FAQ, URL: <http://www.sans.org/resources/idfaq/> (28/06/2006)
6. Carnegie Mellon University, CERT/CC Statistics 1998 – 2006, URL: <http://www.cert.org/stats> (26/06/2006)
7. IETF, RFC 793, Transmission Control Protocol, URL: <http://rfc.net/rfc793.html>