SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1729

SIGURNOSNI SUSTAVI ZA OTKRIVANJE NAPADA

Edi Modrić

Zagreb, lipanj 2008.

Ovaj diplomski rad posvećujem svojim roditeljima.
Hvala vam na svemu što ste mi pružili tijekom mog studija.

Zahvaljujem svima koji su mi pomogli pri izradi ovog rada svojim savjetima, preporukama i ostalim ne tako beznačajnim sitnicama, a posebno mom mentoru doc. dr. sc. Marinu Golubu.

There's a lady who's sure all that glitters is gold
And she's buying a stairway to heaven
And when she gets there, she knows, if the stores are all closed
With a word she can get what she came for
Led Zeppelin - Stairway To Heaven (1971.)

Sažetak

Sigurnosni sustavi za otkrivanje napada

Ovaj diplomski rad opisuje sigurnosne sustave za otkrivanje napada, način njihovog rada, strategije implementacije te njihove prednosti i nedostatke. Dodatno su opisana dva postojeća sustava za otkrivanje napada: jedan sklopovski i komercijalan te drugi besplatan i otvorenog izvornog teksta. Pojašnjena je njihova instalacija, konfiguracija i korištenje. Za praktični dio ovog rada implementiran je jednostavan sustav za otkrivanje napada u programskom jeziku *C#*. Ostvareni sustav je ispitan te su prikazani rezultati ispitivanja.

Abstract

Intrusion detection systems

This diploma thesis describes intrusion detection systems, the way they work, implementation strategies and their advantages and disadvantages. In addition, two existing intrusion detection systems are described: first one, commercially available hardware device and second one, freely available and open source software package. As a practical part of this thesis, a simple intrusion detection system was implemented in *C#* programming language. The implemented system was tested and test results are displayed.

Sadržaj

1. Uvod

Kako je u današnjem svijetu apsolutni imperativ kvalitetna zaštita vlastitog poslovanja svake organizacije, na tržištu postoji mnogo različitih mehanizama za obranu od napada, bilo komercijalnih ili slobodno dostupnih, bilo sklopovskih ili programskih. Osobito je važna kvalitetna edukacija prije ikakve odluke o implementaciji tih mehanizama.

Sustavi za otkrivanje napada su relativno nova tehnologija u cjelokupnoj povijesti zaštite računalnih sustava. Svrha ovog diplomskog rada je da kroz pregled teorijskih informacija o ovim sustavima te praktičnih prednosti i nedostataka postojećih sustava pokuša educirati buduće korisnike o svim aspektima njihova rada. Ovaj rad bi trebao olakšati odluku o izboru sustava za otkrivanje napada kao i odluku treba li korisnicima uopće ovakva vrsta zaštite njihovih računalnih sustava.

Rad je sastavljen od osam poglavlja. Tri poglavlja su ovo uvodno razmatranje, zaključak rada i popis korištene literature. Nakon uvodnog poglavlja, slijedi poglavlje koje za početak daje pregled povijesti razvoja računalnih sustava po desetljećima, počevši od pedesetih godina dvadesetog stoljeća. Nakon toga je opisan pojam sigurnosti računalnih sustava te je dan pregled osnovnih vrsta napada i načina obrane od napada. Spomenuti su sustavi za otkrivanje napada kao kratki uvod u glavnu temu ovog rada.

Treće poglavlje predstavlja glavnu temu razmatranja ovog diplomskog rada. Objašnjen je pojam otkrivanja napada te je dan kratak povijesni pregled sustava za otkrivanje napada, počevši od njihovih začetaka pa sve do danas. Dalje u poglavlju su opisane vrste sustava za otkrivanje napada te je prikazana osnovna arhitektura mrežnih sustava za otkrivanje napada. Na kraju poglavlja su opisane strategije implementacije sustava za otkrivanje napada te je dan pregled koraka koje je potrebno obaviti prije implementacije sustava, s obzirom na postojeća tehnološka rješenja i postojeće sigurnosne politike organizacije u kojoj se sustav implementira. Opisani su i najvažniji nedostaci sustava za otkrivanje napada.

U četvrtom poglavlju je opisan slobodno dostupni programski sustav za otkrivanje napada *Snort*. Nakon općenitog opisa *Snorta*, opisana su pravila kojima programski paket *Snort* obavlja otkrivanje napada te su pojašnjene instalacija i konfiguracija *Snorta* na operacijskom sustavu *Windows*. Na kraju je dan kratki pregled prednosti i nedostataka programskog paketa *Snort*.

U petom poglavlju je pobliže pojašnjena serija komercijalnih sklopovskih i programskih sustava za sprječavanje napada *Proventia M*, koji među ostalim funkcionalnostima imaju i modul za otkrivanje napada. Prvo su opisane razlike između pojedinih uređaja iz serije, a zatim su opisane mogućnosti konfiguracije tih uređaja korištenjem konfiguracijskog sučelja ugrađenog u uređaje. Na kraju je naveden popis glavnih prednosti i nedostataka ovih uređaja.

Zadnje poglavlje opisuje implementaciju i korištenje jednostavnog sustava za otkrivanje napada koji predstavlja praktični dio ovog diplomskog rada. Opisane su glavne tehničke karakteristike i preduvjeti za pokretanje i korištenje sustava. Osim toga, prikazane su

tehnologije pomoću kojih je sustav bio ispitan te proces ispitivanja. Za ispitivanje je iskorišteno šest postojećih ranjivosti u operacijskom sustavu *Windows* i nekoliko aplikacija. Na kraju poglavlja su prikazani rezultati ispitivanja te je dan pregled nedostataka implementiranog sustava kao i moguća poboljšanja u daljnjem razvoju.

2. Sigurnost računalnih sustava

2.1. Povijesni pregled računalnih sustava

Povijest razvoja digitalnih računalnih sustava se često prikazuje u generacijama. Pod pojmom generacija se smatraju epohe vremena odijeljene velikim događajima u računarskoj znanosti kao što su otkrivanje revolucionarnih tehnologija (npr. otkriće tranzistora koji je zamijenio katodnu cijev) i veliki napredak u razvoju programskih jezika (npr. razvoj viših programskih jezika nauštrb asemblerskog programiranja).

Kompletna povijest računanja uz pomoć uređaja seže na početak 17. stoljeća kad su matematičari poput Blaisea Pascala i Gottfrieda Leibnitza koristili mehanička računala koja su bila sposobna izvoditi četiri osnovne matematičke operacije: zbrajanje, oduzimanje, množenje i dijeljenje.

Prva električna računala su se pojavila tek 300 godina poslije, početkom 20. stoljeća (prva generacija računalnih sustava), a prava povijest električnih (što kasnije uključuje i digitalna) računala počinje izumom ENIAC-a (*eng. Electronic Numerical Integrator And Computer*) koje je bilo prvo električno računalo koje se moglo programirati te par godina mlađeg EDVAC-a (*eng. Electronic Discrete Variable Automatic Computer*) koje je bilo prvo binarno računalo (ENIAC je bio zasnovan na dekadskom sustavu) i prvo računalo sagrađeno po Von Neumannovoj arhitekturi računala. Razdoblje nastanka ENIAC-a i EDVAC-a se smatra drugom generacijom računalnih sustava (50-e godine 20. stoljeća).

Treća generacija računalnih sustava (60-e godine 20. stoljeća) počinje izumom i početkom široke upotrebe tranzistora kao osnove razvoja računala. Ova generacija povijesti računalnih sustava je također poznata po razvoju prva tri programska jezika visoke razine: ALGOL-a (*eng. ALGOrithmic Language*), FORTRAN-a (*eng. FORmula TRANslator*) i COBOL-a (*eng. COmmon Business Oriented Language*).

Četvrta generacija (70-e godine 20. stoljeća) razvoja računalnih sustava je označena razvojem prvih LSI (*eng. Large Scale Integration*) i VLSI (*eng. Very Large Scale Integration*) čipova kao i razvojem logičkog i funkcionalnog programiranja. Osim toga, ova generacija je poznata i po razvoju operacijskog sustava *Unix* i programskog jezika *C*.

Peta generacija računalnih sustava (80-e godine 20. stoljeća) je svojevrsni preokret u kompletnom svijetu informacijskih tehnologija. Naime, pojavom i masovnim širenjem IBM-ovog osobnog računala počela je nova era kućnog računarstva koja je dovela do toga da je današnji dizajn i koncept osobnog računala direktni nasljednik IBM-ovog PC-a (*eng. Personal Computer*). Ove godine su poznate i po razvoju i masovnom širenju WAN (*eng. Wide Area Network*) i LAN (*eng. Local Area Network*) mreža računala što je svojevrsni početak razvoja Interneta kakvog danas poznajemo i označava prelazak sa računanja upotrebom centralnog računala (*eng. mainframe*) na korištenje računala za jednostavne i puno manje zahtjevne poslove.

Od početka 90-tih godina 20. stoljeća pa sve do danas je bilo više novih i revolucionarnih stvari nego u cijeloj povijesti razvoja računalnih sustava i to u svim granama računarskih znanosti: od sklopovske opreme, preko programskih rješenja do i dan danas korištenih koncepata i tehnologija. Razvoj sve bržih i bržih računala i mreža sa sve većim i većim brzinama prijenosa podataka postavlja pitanje kakva budućnost očekuje računarsku znanost jer sadašnja situacija odaje dojam da je dosad otkriveno sve što se imalo za otkriti.

2.2. Zaštita računalnih sustava

Od pojave prvih virusa u osamdesetim godinama 20. stoljeća kao i od pojave prvih neželjenih elektroničkih poruka krajem 70-tih godina 20. stoljeća, korisnicima računalnih sustava se počela razvijati svijest o potrebi sve bolje i bolje zaštite njihovih računala i računalnih sustava. Kako je kroz povijest čovječanstva uvijek bilo pojedinaca i grupa ljudi kojima je jedina preokupacija bila nanošenje zla nevinim osobama, tako ni računalni sustavi nisu bili (niti će ubuduće biti) pošteđeni zlonamjernih osoba.

Zbog toga se razvila posebna grana računarske znanosti koja se bavi proučavanjem sigurnosti računalnih sustava. Ciljevi ove grane računarske znanosti su široki, no postoje četiri osnovna cilja istraživanja.

Glavni ciljevi istraživanja u ovoj grani računarske znanosti su:

- ispitivanje sigurnosnih rizika u računarstvu
- razmatranje raspoloživih zaštitnih mjera i kontrola
- podizanje svijesti o informacijskoj sigurnosti
- identifikacija područja u kojima se zahtjeva više rada na postizanju bolje sigurnosti

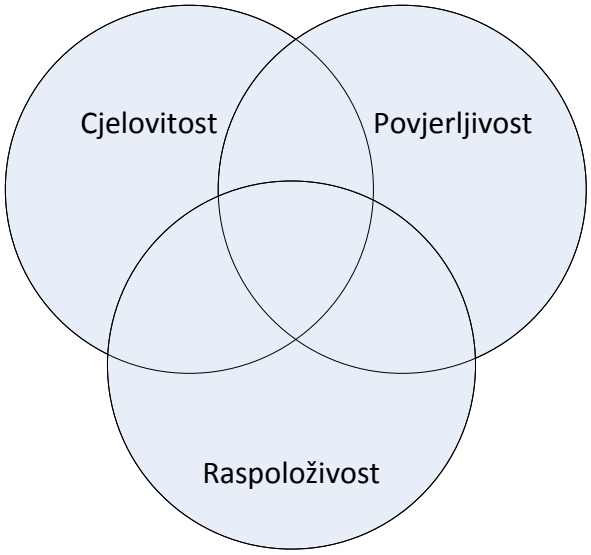
Glavne tehnike zaštite računalnih sustava su:

- provjera autentičnosti korisnika
- kontrola pristupa
- nadzorni zapisi
- neporecivost komunikacije
- povjerljivost informacija
- cjelovitost informacija
- raspoloživost informacija i računalnog sustava

Zadnje tri stavke su se kroz cijelu povijest računalne sigurnosti pokazale kao tri osnovne stavke u razvoju sigurnih računalnih sustava. Slika 2.1 prikazuje njihov međusobni odnos. Kao što se vidi na toj slici, povjerljivost, cjelovitost i raspoloživost su isprepletene i dvije komponente ne mogu funkcionirati bez treće.

Potreba za visokom sigurnošću računalnih sustava je potakla razvoj operacijskih sustava koji su dizajnirani s namjerom da budu sigurni, koristeći pri tom sve principe proizašle iz istraživanja o sigurnosti računalnih sustava. Najpoznatiji takav operacijski sustav je *CapROS* (*eng. Capability-based Reliable Operating System*) koji je zasnovan na principu mogućnosti i otvorenog je izvornog teksta. Svakom objektu u modelu ovog operacijskog

sustava je pridružena mogućnost zajedno s pravima pristupa koja govori ima li neki objekt mogućnost pristupa nekom drugom objektu.



Slika 2.1. Odnos cjelovitosti, povjerljivosti i raspoloživosti u računalnoj sigurnosti

2.3. Najčešće vrste napada na računalnu sigurnost

2.3.1. Preljev spremnika

Preljev spremnika (*eng. buffer overflow*) predstavlja programsku grešku koja rezultira prekidom rada programa na način da program u nekom trenutku u polje određene duljine želi zapisati podatak koji je veći od veličine polja. Uslijed toga dolazi do prepisivanja memorijskih lokacija koje nisu namijenjene za smještanje podataka koji se zapisuju u polje. Takve memorijske lokacije mogu biti rezervirane za neki drugi podatak unutar istog programa ili za potpuno drugi program kojem je operacijski sustav dodijelio tu memorijsku lokaciju.

Iako preljev spremnika na prvi pogled djeluje kao pogreška koja nije pogodna za iskorištavanje, to ipak nije tako. Naime, velika većina napada izvedenih pomoću preljeva spremnika funkcionira iskorištavajući dobro poznati princip rada poziva funkcija sa parametrima unutar programa. Pri pozivu funkcija unutar bilo kojeg programa koristi se podatkovna struktura nazvana stog. Stog je organiziran na LIFO ili "zadnji unutra, prvi vani" principu (*eng. LIFO - Last In, First Out*). Takva organizacija stoga omogućuje programu da pri pozivu funkcije na njega pohranjuje parametre koji se prenose funkciji, da pohranjuje povratnu adresu na kojoj program nastavlja izvršavanje nakon završetka funkcije i da pohranjuje lokalno definirane varijable funkcije. Svakom stogu je pridijeljena i kazaljka na vrh stoga koja se dinamički mijenja kako se pozivaju funkcije unutar programa.

Prilikom poziva funkcije, na stog se prvo pohranjuju parametri koji se predaju funkciji, a zatim se na stog pohranjuje sadržaj instrukcijskog registra u procesoru koji služi kao kazaljka na instrukciju koja se izvodi nakon trenutne (*eng. IR - Instruction Register*). Ta se kazaljka na stog pohranjuje da program, nakon što se izvođenje preusmjeri na funkciju i

nakon što ona obavi svoj posao, može nastaviti na mjestu gdje je izvođenje prekinuto pozivom funkcije. Nakon što se na stog pohrani vrijednost instrukcijskog registra i nakon što funkcija započne s izvođenjem, ona na stog pohranjuje svoje lokalne varijable. To je trenutak u kojem napadač može iskoristiti prije opisanu grešku. Naime, ako se u neku od lokalnih varijabli koja je polje određene duljine, pokuša zapisati podatak koji je veći od veličine tog polja, zbog organizacije stoga može doći do prepisivanja sadržaja instrukcijskog registra. Pažljivim biranjem podatka koji će se zapisati na mjesto na kojem je pohranjen sadržaj instrukcijskog registra napadač može kontrolirati koju će instrukciju program izvesti nakon što završi izvođenje funkcije. Napad se izvodi tako da se sadržaj instrukcijskog registra prepiše sa adresom memorijske lokacije na koju je napadač prethodno pohranio nekakav zlonamjerni program koji se dalje izvršava neprimjetno za korisnika.

2.3.2. Uskraćivanje posluživanja

Napad uskraćivanjem posluživanja (*eng. DoS - Denial of Service*) je vrsta napada na računalni sustav koja se provodi s namjerom da se legitimnim korisnicima računalnog sustava privremeno ili trajno onemogući pristup resursima tog sustava. Najčešći pristup provođenju DoS napada je slanje velike količine zahtjeva računalnom sustavu u vrlo kratkom vremenskom roku. Posljedica toga je da računalni sustav ne može u nekom normalnom vremenskom okviru odgovoriti na one zahtjeve koje šalju legitimni korisnici, tako da sustav gledajući sa strane legitimnog korisnika postaje neupotrebljiv.

Do neupotrebljivosti resursa računalnog sustava koji je napadnut DoS napadom najčešće dolazi zbog dvije stvari:

- zagušenja strojeva koji su dio napadnutog računalnog sustava, tako da sustav jednostavno prestaje obrađivati zahtjeve koji mu stalno pristižu ili
- prekida komunikacije između korisnika i računalnih sustava.

DoS napadi se provode na velik broj načina, no najpoznatiji i najkorišteniji napadi su sljedeći:

- *smurf* napad - oslanja se na neispravno podešene mreže računala koje dozvoljavaju slanje paketa na sva računala u mreži putem mrežne adrese za razašiljanje paketa (*eng. broadcast address*). Napadači u ovom slučaju lažiraju IP adresu izvorišta paketa da odgovara meti napada, tako da sva računala koja dobiju pakete preko adrese za razašiljanje odgovaraju meti napada koja postaje zatrpana.
- *teardrop* napad - oslanja se na grešku u funkciji za sastavljanje paketa u raznim operacijskim sustavima tako da šalje pakete sa prevelikim podatkovnim dijelom ili sa podatkovnim dijelovima koji se preklapaju u fragmentima nekog paketa.
- raspodijeljeni napad (*eng. DDoS - Distributed Denial of Service*) - napad koji koristi više računala za napad na istu metu. Ta računala su većinom zaražena virusima, trojancima, crvima i ostalim zloćudnim softverom koji služi za koordinaciju tih računala prilikom napada.
- raspodijeljeni reflektirani napad (*eng. DRDoS - Distributed Reflected Denial of Service*) - ova vrsta napada se izvodi tako da se na više mrežnih odredišta na Internetu šalju zahtjevi sa lažiranom izvorišnom adresom koja odgovara adresi mete napada, tako da svi odgovori stižu na metu.

2.3.3. Zlonamjerni programi

Pojam zlonamjernog programa označava bilo koju vrstu programa kojem je namjera uzrokovati štetu, kako računalnom sustavu tako i njegovim korisnicima, bila ona materijalna, podatkovna ili novčana (npr. krađom podataka o kreditnim karticama i bankovnim računima). Postoji puno vrsta zlonamjernih programa, no većinom se mogu svrstati u pet kategorija: viruse, crve, trojance, *adware* i *spyware*.

Virusi su zlonamjerni programi koji koriste druge legitimne programe za svoje širenje na način da na izvršni program legitimne aplikacije dodaju zlonamjerni izvršni program (*eng. payload*). Prilikom pokretanja legitimnog programa koji je zaražen virusom prvo se izvršava zlonamjerni program tako da korisnik nije svjestan toga. Nakon što zlonamjerni program obavi svoju zadaću, kontrola izvršavanja se vraća legitimnom programu koji normalno nastavlja svoj rad. Upravo zbog činjenice da virusi, nakon što obave svoju zadaću, vraćaju kontrolu legitimnom programu, korisnici najčešće nisu svjesni da su izvršne datoteke na računalu zaražene virusom. Viruse najčešće šire korisnici prenoseći ih na disketama, CD i DVD medijima ili USB memorijama.

Crvi su zlonamjerni programi koji za razliku od virusa imaju mogućnost samostalnog širenja nakon što su inicijalno pokrenuti od strane napadača, dakle nije im potrebna pomoć korisnika. Crvi za širenje najčešće iskorištavaju jako poznate i neispravljene greške u operacijskim sustavima ili drugim programima. Još jedna učestala metoda širenja je koristeći programe za čitanje elektroničke pošte i to tako da sami sebe pošalju svim kontaktima u adresaru na zaraženom računalu. Priroda crva čini ga puno destruktivnijim od bilo kojeg virusa jer je crv u stanju zaraziti velik broj računala u vrlo malom vremenskom roku. Jednom zaražena računala mogu biti iskorištena od strane napadača na više načina. Najčešće se ta računala koriste za napade protiv nekih drugih računala pretvarajući ih u sudionike raspodijeljenog napada uskraćivanjem usluge.

Trojanci su zlonamjerni programi koji se predstavljaju kao legitimni programi (male igre ili jednostavni programi za specifičnu namjenu su dobar primjer takvih programa), a u pozadini izvršavaju zlonamjerni program koji najčešće otvara pozadinska vrata (*eng. backdoor*) u računalnom sustavu, dopuštajući napadaču neometan pristup resursima sustava.

Adware je vrsta programa koji je najčešće besplatan za korištenje zbog činjenice da se unutar programa prikazuju oglasi kojima se proizvođači programa financiraju. Prikazivanje oglasa unutar programa nije sporno, no proizvođači softvera kao sustave za prikazivanje oglasa koriste druge programe koji su napisani tako da ciljano prikazuju oglase koristeći podatke o navikama korisnika koje prikupljaju sa računala na koja su instalirana. Time je narušena privatnost korisnika računalnih sustava.

Spyware je vrsta zlonamjernog programa kojem je jedina namjera prikupljanje statistika o korištenju programa i radnih navika korisnika koji se potom šalju proizvođačima *spywarea* koji ih iskorištavaju u marketinške svrhe. *Spyware* programi se koriste raznim tehnikama kojima prikupljaju podatke kao što su praćenje teksta koji korisnik tipka po tipkovnici ili pohranjivanje sadržaja ekrana te na taj način mogu prikupiti vrlo povjerljive informacije

kao što su lozinke, brojevi kreditnih kartica, podaci zaštićeni intelektualnim vlasništvom itd.

Od pojave prvog virusa sredinom osamdesetih godina dvadesetog stoljeća pa sve do danas, napadači stalno pronalaze nove, inventivnije načine na koje će širiti svoje zlonamjerne programe. To je dovelo do situacije da se danas zlonamjerni programi šire bez ikakve intervencije i kontrole korisnika, na primjer koristeći mogućnosti modernih pretraživača Interneta da izvršavaju program napisan u skriptnim programskim jezicima poput *JavaScripta*. Kako skriptni programski jezici imaju velik broj mogućnosti normalnih programskih jezika, napadači ih mogu iskoristiti na način da naprave stranicu čijim se otvaranjem automatski pokreće program koji na korisnikovo računalo instalira i pokreće nekakav zlonamjerni program.

2.3.4. Zlonamjerni korisnici

Mogućnost kompromitiranja računalnih sustava direktno od strane korisnika, bili oni legitimni korisnici resursa računalnog sustava ili osobe koje pokušavaju dobiti neovlašteni pristup resursima računalnog sustava, također ne treba zanemariti prilikom razmatranja sigurnosti sustava. Općenito je poznato da je računalni sustav siguran onoliko koliko je sigurna njegova najslabija karika. Zbog te činjenice potrebno je posvetiti veliku pozornost korisnicima i njihovim pravima pristupa te kvalitetno educirati korisnike računalnih sustava da znaju prepoznati najčešće vrste prevara i da uspješno odolijevaju istima.

U situaciji kad su svi pokušaji zlonamjernih korisnika da kompromitiraju računalni sustav blokirani dobro podešenim sigurnosnim rješenjima, oni često pokušavaju dobiti pristup računalnim resursima koristeći tehnike poput socijalnog inženjeringa.

Socijalni inženjering je skup tehnika koje zlonamjerni korisnici koriste pri pokušajima manipulacije legitimnih korisnika računalnog sustava da im odaju informacije ili da naprave niz akcija koje će omogućiti zlonamjernim korisnicima neometan pristup računalnom sustavu. Dva su najčešća pristupa socijalnom inženjeringu: *pretexting* i *phishing*.

Pretexting je tehnika socijalnog inženjeringa u kojoj napadač komunicira s korisnikom računalnog sustava u svrhu dobivanja povjerljivih informacija od korisnika koji bi mogli poslužiti kao osnova za neovlašteni pristup resursima računalnog sustava. Napad se provodi tako da prije komunikacije s korisnikom napadač pripremi uvjerljivi scenarij pomoću kojeg će uvjeriti korisnika da napadaču preda podatke koje zatraži.

Phishing je tehnika kojom se najčešće korištenjem nepoželjne pošte (*eng. spam*) korisnike pokuša natjerati da ostave povjerljive podatke o bankovnim računima, brojevima kreditnih kartica itd. Te nepoželjne poruke najčešće izgledaju kao da su poslane od strane legitimnih tvrtki (banke i njihovi sustavi za upravljanje računima preko Interneta, usluge za plaćanje i kupovinu preko Interneta...) i u kojima se traži od korisnika da na posebno kreiranoj internetskoj stranici ostave svoje podatke (najčešće korisnička imena i lozinke za navedene usluge). Te internetske stranice su doslovne preslike postojećih i legitimnih stranica, tako da korisnik nema razloga posumnjati u prevaru. Najčešći primjer ovakve

vrste prevare je neželjena pošta koja od korisnika traži da se prijavi na najpoznatiju uslugu za plaćanje preko Interneta *PayPal*, i da potvrdi osobne podatke, inače će mu račun na usluzi biti uskoro deaktiviran. Korisnici koji ne sumnjaju u legitimnost primljene elektroničke pošte će se pokušati prijaviti na posebno kreiranoj stranici koja će pohraniti korisničko ime i lozinku u svoju bazu podataka i preusmjeriti korisnika na stvarnu stranicu usluge *PayPal* tako da korisnik ništa ne posumnja.

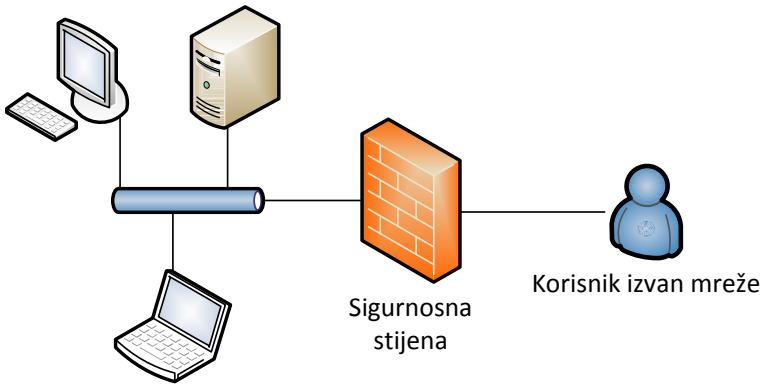
Koliko je socijalni inženjering postao ozbiljna prijetnja sigurnosti računalnih sustava pokazuje i zakon donesen u Sjedinjenim američkim državama u siječnju 2007. godine. Po tom zakonu se socijalni inženjering smatra kaznenim djelom i čijim se kršenjem može dobiti kazna do 250 000 američkih dolara ili do 10 godina zatvora za fizičke osobe, dok je za pravne osobe kazna do 500 000 američkih dolara.

Uz socijalni inženjering tj. napade zlonamjernih korisnika van sustava, ne treba zanemariti ni vrlo stvarnu mogućnost pojave napada unutar računalnog sustava. Naime, vrlo je lako moguće da zaposlenik u nekoj tvrtci (i samim time korisnik računalnog sustava te tvrtke) obavlja radnje koje imaju za cilj omogućavanje neometanog pristupa resursima sustava osobama van sustava, krađu podataka zaštićenih intelektualnim vlasništvom, dobivanje pristupa podacima za koje nema ovlaštenje itd. Ovakve napade je u praksi nemoguće zaustaviti dok korisnici ne budu uhvaćeni. Zato je potrebno stalno obavljati striktne interne kontrole zaposlenika i korisnika računalnog sustava.

2.4. Tehnike obrane od napada

2.4.1. Sigurnosne stijene

Sigurnosna stijena (*eng. firewall*) je uređaj ili aplikacija koja upravlja pravima pristupa resursima računalnog sustava. Resurs računalnog sustava se kod spominjanja pojma sigurnosne stijene može odnositi ili na kompletnu mrežu računala ili na jedno računalo unutar mreže. U slučaju kad se pod resursom računalnog sustava misli na mrežu računala, obično je sigurnosna stijena uređaj koji je smješten negdje u mrežnoj topologiji i nadgleda svaki pokušaj pristupa mreži. Slika 2.2 prikazuje takvu konfiguraciju mreže.



Slika 2.2. Mrežna topologija sa sigurnosnom stijenom

U slučaju kad se kao resurs računalnog sustava promatra jedno računalo, onda je obično sigurnosna stijena aplikacija koja je instalirana na to računalo i nadgleda svaki pokušaj udaljenog pristupa računalu.

Bez obzira na razlike u implementaciji i svrsi ove dvije vrste sigurnosnih stijena, zajednički im je način rada. Obje vrste rade na način da se prilikom njihove konfiguracije definiraju pravila pristupa računalnom resursu. Ta pravila se mogu definirati koristeći mnoge parametre. Najčešće se koriste parametri kao što su protokol (TCP, UDP, IP), stanje TCP veze, smjer prometa (odlazni ili dolazni), IP adrese lokalnog i udaljenog računala, mrežne pristupne točke (*eng. ports*) na lokalnom i udaljenom računalu, vremenske odrednice itd. Nakon što su pravila ispravno konfigurirana, definiraju se akcije koje će se obaviti kada se desi pokušaj pristupa na računalni resurs, a ima odgovarajuće parametre zadane pravilima sigurnosne stijene. Te akcije mogu biti dopuštanje pristupa, blokiranje pristupa, obavijest administratorima o događaju itd.

Osim provjere mrežnog prometa i reagiranja na pojavu pokušaja spajanja na računalni resurs, kvalitetnije sklopovske sigurnosne stijene nude i dodatne razine zaštite. Naime, nadziranje mrežnog prometa je moćno i može spriječiti većinu neautoriziranih pristupa, no činjenica da se izvorišne IP adrese lako mogu lažirati pokazuje potrebu za postojanjem dodatnih razina sigurnosti. Jedan primjer dodatne razine zaštite resursa računalnog sustava je provjera autentičnosti (*eng. authentification*) i autorizacija (*eng. authorisation*) korisnika. Provjera autentičnosti i autorizacija se mogu provoditi na više načina: od jednostavnog zahtjeva za upisivanjem korisničkog imena i lozinke do korištenja sustava enkripcije zasnovanih na javnim ključevima i certifikatima.

Važno je napomenuti da iako sigurnosne stijene izgledaju kao jako moćni sustavi za nadzor i zaštitu računalnih resursa od zlonamjernih korisnika, one u svakom slučaju ne bi trebale biti jedina metoda zaštite računalnog sustava. Metode napada kao što je socijalni inženjering mogu efikasno i vrlo lako dovesti do zaobilaženja sigurnosnih stijena i kompromitiranja računalnog sustava.

2.4.2. Antivirusna rješenja

Antivirusni programi su programi koji identificiraju, neutraliziraju i uklanjaju zlonamjerne programe. Iako su se svi moderni antivirusni programi sposobni boriti protiv većine vrsta zloćudnih programa, ime su zadržali iz povijesnih razloga, kada su se pod pojam zloćudnih programa svrstavali jedino virusi.

Antivirusni programi najčešće koriste dvije tehnike identifikacije, neutralizacije i uklanjanja zloćudnih programa:

- provjera datoteka na računalu i traženje specifičnih nizova okteta koji identificiraju zloćudne programe. Svi antivirusni programi imaju bazu podataka sa velikim brojem odgovarajućih nizova okteta s kojima vrše usporedbu, te
- heuristički pristup koji služi za identifikaciju novih i antivirusnim programima nepoznatih, kao i varijanti postojećih zloćudnih programa.

Dok je način funkcioniranja prve tehnike praktički jasan sam po sebi, potrebno je pobliže opisati način na koji radi heuristički pristup otkrivanju zloćudnih programa.

Heuristički pristup otkrivanju zloćudnih programa se najčešće implementira na dva načina:

- Korištenjem virtualnog stroja u sklopu antivirusnog programa. Antivirusni program unutar tog virtualnog stroja pokreće datoteke koje pregledava heurističkim pristupom, izolirajući ih od ostatka računalnog sustava. Nakon pokretanja datoteke, antivirusni program prati aktivnosti koje pokrenuti program radi. Najčešće se prate uobičajene aktivnosti koje izvode zloćudni programi kao što su mijenjanje drugih datoteka, samostalno širenje, pokušaji skrivanja nekih datoteka itd.
- Prevođenjem datoteka koje se pregledavaju iz binarnog oblika natrag u izvorni tekst (*eng. decompilation*) i analizom dobivenog izvornog teksta. Izvorni tekstovi tih programa se uspoređuju sa izvornim tekstovima postojećih zloćudnih programa.

Iako je heuristički pristup otkrivanju zloćudnih programa svakako dobrodošla funkcija antivirusnih programa, oni se ne mogu i ne smiju isključivo oslanjati na rezultate heurističkog pristupa. Naime, uspješnost heurističkog pristupa je 40 do 50% zbog činjenice da se svakodnevno pojavljuju zloćudni programi koji mogu sadržavati funkcije koje niti jedan zloćudni program nije imao do sad i u takvim situacijama je heuristički pristup nemoćan.

Nakon što antivirusni program detektira neki zloćudni program, najčešće postoje tri akcije koje se mogu obaviti:

- pokušavanje čišćenja zloćudnog programa sa računala,
- prebacivanje zloćudnog programa u karantenu koju antivirusni program na siguran način izolira od ostatka računala ili
- brisanje zaražene datoteke.

Kako antivirusni programi za svoj rad trebaju bazu podataka o postojećim zloćudnim programima, jako je važno da se ta baza podataka konstantno nadopunjava informacijama o novim virusima. Proizvođači antivirusnih programa obično na svojim internetskim stranicama nude dnevne ili tjedne nadopune baze podataka. Uz to, antivirusni programi obično imaju mogućnost samostalnog skidanja tih nadopuna preko Interneta. Uzevši u obzir sve dosad napisano, jasno je da su i antivirusna rješenja, kao i sigurnosne stijene, neizostavan dio alata za zaštitu računalnih sustava. No, kao i kod sigurnosnih stijena vrijedi napomena da se korisnici ne bi trebali oslanjati isključivo na antivirusne sustave kao jedino rješenje za sigurnost računalnih sustava.

2.4.3. Sustavi za otkrivanje napada

Sustavi za otkrivanje paketa su sklopovski uređaji ili programski sustavi koji imaju mogućnost otkrivanja napada u stvarnom vremenu. To rade na način da svaki paket koji prolazi mrežom koju sustav za otkrivanje napada nadzire, analizira u potrazi za određenim uzorcima nizova znakova ili okteta koji bi mogli biti indikator neželjenog ponašanja u

mrežnoj infrastrukturi. Traženje uzoraka se obavlja pretraživanjem unaprijed definirane baze podataka koja sadrži detalje o već poznatim napadima pomoću kojih je moguće identificirati napad koji je u tijeku.

2.4.4. Sustavi za sprječavanje napada

Sustavi za sprječavanje napada su tehnološki napredniji sustavi za otkrivanje napada. Od sustava za otkrivanje napada se razlikuju po tome što uz otkrivanje mogu i aktivno blokirati napade. Blokiranje napada se većinom provodi na dva načina:

- mijenjanjem sadržaja paketa koji je uzrokovao otkrivanje napada. Primjer takvog mijenjanja sadržaja može biti izmjena zlonamjernog programa koji je dio paketa da ne radi ništa ili
- ubacivanjem RST (*eng. reset*) paketa unutar TCP veze kojom se provodi napad, čime se nasilno prekida veza i zaustavlja napad u tijeku.

3. Sustavi za otkrivanje napada

3.1. Otkrivanje napada

Pojam otkrivanja napada podrazumijeva otkrivanje bilo kakvih akcija koje imaju za cilj narušiti cjelovitost, povjerljivost i raspoloživost računalnih sustava. Otkrivanje napada se može provoditi na dva načina, ručno i automatski.

Ručno otkrivanje napada se svodi na čitanje dnevnika događaja i traženje sumnjivih zapisa koji se u sigurnoj radnoj okolini računalnog sustava ne bi trebali pojavljivati. Takav način otkrivanja napada je često zamoran i kontraproduktivan jer zahtijeva da osoba (najčešće sistemski administrator) čita veliku količinu podataka. Takav način otkrivanja napada je podložan greškama zbog ljudskog faktora. Također, ovaj način otkrivanja napada jako rijetko daje dovoljno vremena za reakciju sprečavanja napada jer postoji određena vremenska zadrška od trenutka od kad se dogodi napad do trenutka kad sistemski administrator čita zapise u dnevniku. Dok ta vremenska zadrška prođe, napad će vrlo vjerojatno već biti uspješno izvršen.

Zbog tih činjenica, početkom 80-tih godina 20. stoljeća se krenulo s razvojem automatskih sustava za otkrivanje napada. Prvi nacrti i modeli sustava za otkrivanje napada su razvijeni od strane vlade Sjedinjenih američkih država i to za nadziranje njenih računalnih sustava. Prvi funkcionalni sustav za otkrivanje napada je razvijen za nadzor i analizu informacija o provjeri autentičnosti korisnika mreže ARPANET (*eng. Advanced Research Projects Agency Network*). Razvio ga je *Stanford* istraživački institut (*eng. SRI - Stanford Research Institute*) 1984. godine. 1988. godine je na Kalifornijskom sveučilištu razvijen prvi sustav za otkrivanje napada koji je pratio zapise u dnevniku događaja na temelju predefiniranih uzoraka. Taj sustav je preteča današnjih sustava za otkrivanje napada.

Automatski sustavi za otkrivanje napada mogu biti izvedeni kao programski paketi ili kao sklopovski uređaji. Jedan programski sustav za otkrivanje napada je pobliže opisan u poglavlju 4, dok je jedan sklopovski uređaj za otkrivanje napada opisan u poglavlju 5 ovog rada.

3.2. Vrste sustava za otkrivanje napada

Sustavi za otkrivanje napada se općenito mogu podijeliti na tri vrste: mrežno zasnovane sustave za otkrivanje napada (*eng. NIDS - Network Intrusion Detection System*), računalno zasnovane (*eng. HIDS - Host Intrusion Detection System*) i mješovite sustave za otkrivanje napada koji su kombinacija mrežnih i računalnih sustava. Mrežno zasnovani i računalno zasnovani sustavi za otkrivanje napada su pobliže opisani u ovom potpoglavlju.

3.2.1. Mrežno zasnovani sustavi za otkrivanje napada

Kad se govori o sustavima za otkrivanje napada, većinom se misli na upravo mrežno zasnovane sustave. Mrežno zasnovani sustavi za otkrivanje napada funkcioniraju na način da prate i analiziraju mrežne pakete koji prolaze mrežnim sučeljem. To mrežno sučelje najčešće radi u promiskuitetnom načinu rada koji dozvoljava hvatanje apsolutno svih

paketa koji se pojave na mrežnom sučelju, a ne samo onih koji kao odredište imaju IP adresu tog sučelja. Mrežno zasnovani sustavi za otkrivanje napada su najčešće uređaji koji su smješteni negdje u mrežnoj topologiji (slično kao i sigurnosne stijene) i nadziru sav promet na mreži. Analiza prometa i reagiranje na pojavu sumnjivog prometa se vrše koristeći tri metodologije: traženjem uzoraka u uhvaćenim paketima, praćenjem učestalosti pojave određenih paketa i traženjem određenih anomalija u mrežnom prometu koje mogu biti indikacija da je napad u tijeku. Velika prednost ovog tipa sustava za otkrivanje napada nad računalno zasnovanim sustavima je ta što se otkrivanje provodi u stvarnom vremenu. Time je napadaču praktički onemogućeno brisanje tragova svog napada, za razliku od računalno zasnovanog sustava za otkrivanje napada gdje je napadaču relativno lako zamesti tragove izvršenog napada.

3.2.2. Računalno zasnovani sustavi za otkrivanje napada

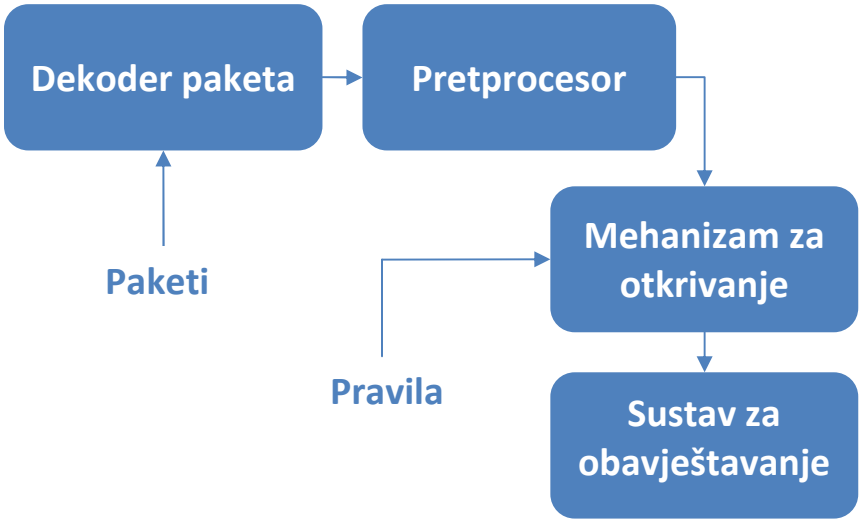
Za razliku od mrežno zasnovanih sustava za otkrivanje napada, ovaj tip sustava je najčešće aplikacija koja se izvršava na jednom računalu i prati zapise u dnevnicima događaja tražeći neuobičajene uzorke. Dnevnici događaja koje ovaj tip sustava za otkrivanje napada prati se razlikuju od operacijskih sustava do operacijskih sustava. Na operacijskom sustavu *Windows* su to najčešće tri dnevnika: aplikativni, sistemski i sigurnosni dnevnici događaja. Na operacijskom sustavu *Linux* se najčešće radi o *syslog* sistemskom dnevniku događaja. Iako se svi ti dnevnici događaja mogu ručno pregledavati, računalno zasnovani sustavi su pogodni zbog svog automatizma i brzine.

Ovaj tip sustava za otkrivanje napada koristi iste metodologije kao i mrežno zasnovani sustavi. No, iako koristi iste pristupe u otkrivanju napada, nepogodni su za ozbiljniju primjenu u velikim mrežama. Naime, kao problem se nameće samo ime ovog tipa sustava, računalno zasnovani. Kao što je već rečeno, ova vrsta sustava za otkrivanje napada su aplikacije koje se izvršavaju na jednom računalu, pa bi za efektivno praćenje svih aktivnosti na velikoj mreži računala bilo potrebno instalirati i izvršavati ovakve sustave na svim računalima u toj mreži. Takvo nešto je praktički nemoguće jer bi bilo potrebno mnogo ljudskih resursa za kvalitetno održavanje svih instanci ovih sustava. Nadalje, ovi sustavi su dosta podložni za napade jer napadači imaju mogućnost, jednom kad dobiju pristup računalnom sustavu, zamesti tragove svojih napada brišući zapise u dnevnicima događaja. Kako većina ovih sustava ne čita dnevnike događaja konstantno već u određenim vremenskim intervalima, napadači mogu biti vrlo uspješni u zaobilaženju ovog tipa sustava za otkrivanje napada.

Bez obzira na spomenute nedostatke, računalno zasnovani sustavi za otkrivanje napada su u nekim aspektima rada bolji od mrežno zasnovanih sustava. Ovaj tip sustava ima znatno veću otpornost na pogreške pri otkrivanju napada. Naime, ovaj tip sustava je čitajući dnevnike događaja u mogućnosti saznati je li određeni napad bio uspješan ili ne, što je kod mrežno zasnovanih sustava gotovo nemoguće. Također, ovaj tip sustava ne mora nužno pratiti mrežne aktivnosti računala na kojem se nalazi. On je sposoban pratiti i druge aktivnosti na računalu, poput neautoriziranog pokušaja pristupa povjerljivim podacima ili pokretanja određenih programa itd.

3.3. Arhitektura mrežno zasnovanih sustava za otkrivanje napada

Iako na tržištu postoje različiti sustavi za otkrivanje napada, različitih funkcionalnosti i od različitih proizvođača, u suštini se rad svakog sustava može svesti na jedno. Rad svih sustava za otkrivanje napada se svodi na povezivanje četiri osnovna modula u jednu cjelinu. Ti moduli su dekoder paketa, pretprocesor, mehanizam za otkrivanje napada te sustav za obavještavanje i vođenje dnevnika. Slika 3.1 prikazuje model sustava za otkrivanje napada sastavljen od navedenih modula.



Slika 3.1. Model sustava za otkrivanje napada

3.3.1. Dekoder paketa

Kao što se vidi na prethodnoj slici, dekoder paketa je prvi modul kroz koji paket prolazi pri dolasku do sustava za otkrivanje napada. Kako postoji više vrsta mrežnih sučelja i svako od njih ima svoj specifičan format paketa koji se šalje i prima kroz to mrežno sučelje, dekoder paketa ima za zadaću da sve vrste tih paketa pretvara u jedinstveni format paketa koji je najčešće specifičan sustavu za otkrivanje napada. Kad ne bi bilo dekodera paketa, sustav za otkrivanje napada bi bio vrlo ograničen u svojoj funkcionalnosti jer bi mogao analizirati samo određenu vrstu mrežnog prometa. Ovako, uz implementiran dekoder paketa, sustav za otkrivanje napada je sposoban analizirati bilo kakvu vrstu mrežnog prometa čitajući pakete u dekodiranom obliku.

3.3.2. Pretprocesor

Pretprocesor je komponenta sustava za otkrivanje napada koja ima sposobnost preuređivanja nizova okteta u paketu (što uključuje pretvorbu, izbacivanje i dodavanje okteta) s ciljem što jednostavnije i lakše kasnije analize paketa. Dizajn pretprocesora u većini sustava za otkrivanje napada je modularan što olakšava dodavanje novih funkcionalnosti u sustav, sukladno pojavi novih standarda i protokola.

Tri su glavne zadaće pretprocesora:

- Sastavljanje fragmentiranih paketa - kako je kroz mrežno sučelje dozvoljeno slanje paketa do neke određene veličine, (standardizirana veličina je 1500 okteta), a nekad postoji potreba za slanjem više podataka u jednom paketu, nužno je paket koji prelazi zadanu veličinu rastaviti na manje dijelove. Takvi dijelovi paketa su i sami paketi, no sustavi za otkrivanje napada se ne mogu pouzdati da će pretragom jednog po jednog paketa iz skupine koji čine originalni paket pronaći određeni zapis pomoću kojeg će prepoznati izvršavanje napada. Zbog toga je potrebno prije analize pakete koji su dio nekog većeg paketa sastaviti u jedan, opet u jedinstvenom formatu koji je specifičan za svaki sustav posebno.
- Dekodiranje uniformnih identifikatora resursa - kako je za pristup datotekama na bilo kojem serveru moguće koristiti više načina pristupa, tj. više načina zapisa putanje do resursa, moguće je da napadači pokušaj pristupa datotekama maskiraju korištenjem alternativnih načina zapisa putanje do te datoteke. Jedan često korišten primjer je zapis putanje do adrese neke internetske usluge koja se želi napasti pomoću UTF-8 (*eng. Unicode Transformation Format*) zapisa znakova. Sustavi za otkrivanje napada koji svoj rad zasnivaju na otkrivanju ovakve vrste napada samo pomoću apsolutnih putanja do datoteka nisu u mogućnosti otkriti napade koji su maskirani na ovaj način. Zbog toga je potrebno putanje u uniformnom identifikatoru resursa moći obraditi i pretvoriti u format koji sustav za otkrivanje napada prepoznaje čime se omogućava otkrivanje napada.
- Praćenje stanja TCP veze - kako je uspješnost otkrivanja nekih napada ovisna o tijeku komunikacije napadača i sustava koji se napada, vrlo važna funkcionalnost koju sustav za otkrivanje napada mora podržati je i praćenje stanja TCP veze. Ova funkcionalnost je podržana upravo kroz pretprocesor.

3.3.3. Mehanizam za otkrivanje napada

Mehanizam za otkrivanje napada je središnji modul sustava za otkrivanje napada. O ovom mehanizmu ovisi brzina i efikasan rad cijelog sustava. Što se brzine rada tiče, potrebno je da je sustav za otkrivanje pokrenut na dovoljno brzom računalu koje će biti u stanju pratiti cjelokupan promet na mreži. Zbog sporog računala ili zbog prevelikog prometa na mreži postoji rizik da dođe do zagušenja i da neki od paketa ne prođe kroz ovaj mehanizam te tako postoji mogućnost da se ne otkriju svi napadi. Pod efikasnim radom se prvenstveno misli na što manju pojavu lažnih uzbuna i, naravno, što veći postotak otkrivenih napada. To se može postići korištenjem dobro podešenog skupa pravila po kojima se obavlja otkrivanje.

Prilikom otkrivanja napada, pravila su obično grupirana u logičke cjeline i pravila u jednoj logičkoj cjelini se odnose na jednu vrstu napada. Time je jednostavno omogućeno isključivanje ili uključivanje otkrivanja pojedinih vrsta napada. Najosnovniji način otkrivanja napada se provodi tako da se svaki paket uspoređuje sa svim pravilima iz svih logičkih cjelina. U trenutku kad se za neki paket pronađe odgovarajuće pravilo, daljnji proces otkrivanja napada se prekida i sustav generira obavijest o pojavi sumnjivog paketa. U nekim slučajevima je potrebno i nastaviti proces otkrivanja napada nakon pronalaska odgovarajućeg pravila jer se vrlo lako može desiti da jedan paket aktivira više pravila, a neka od tih pravila koja su kasnije aktivirana su više bitna za buduće efikasno sprječavanje napada u tijeku od pravila koja su ranije aktivirana. Gledajući na taj način na proces

otkrivanja napada, vrlo lako se objašnjava činjenica zašto može doći do zagušenja sustava za otkrivanje napada. U slučaju kad postoji velik broj paketa i velik broj pravila, računalo na kojem je smješten sustav mora biti vrlo brzo da se u stvarnom vremenu uspiju obraditi svi paketi. Kako ipak u nekim slučajevima nije moguće uskladiti količinu prometa na mreži za zahtjevima za brzim računalom, vrše se razne optimizacije baze podataka koja sadrži pravila. Najčešća optimizacija je sortiranje pravila po prioritetu. Pravilima se unaprijed zadaju prioriteti koji govore o težini napada u tijeku. Veći prioritet znači veću težinu napada. Nakon što su svim pravilima pridodani odgovarajući prioriteti, sortiranje se provodi tako da se paket prvo uspoređuje s pravilima najvišeg prioriteta pa je tako zagarantirano da će prvo pravilo koje je aktivirano uvijek imati viši prioritet od svih ostalih pravila koja bi naknadno mogla biti aktivirana. Zbog toga je sigurno prekinuti daljnje otkrivanje napada.

3.3.4. Sustav za obavještavanje i vođenje dnevnika

Nakon što sustav za otkrivanje napada otkrije neki napad, njegova zadaća je pravovremena reakcija u vidu obavještavanja o otkrivenom napadu. Obavještavanje se uglavnom provodi zapisivanjem određenih podataka o otkrivenom napadu u razne dnevnike događaja koje onda administratori mrežne infrastrukture u kojoj je smješten sustav prate te na temelju zapisa pronađenih u njima mogu pravovremeno reagirati i spriječiti napad.

Informacije koje se zapisuju u dnevnik događaja su većinom zajedničke svim sustavima za otkrivanje napada a uključuju sljedeće:

- izvorišnu i odredišnu IP adresu i pristupne točke
- vrijeme kad je napad otkriven
- identifikacijski broj pravila pomoću kojeg je napad otkriven
- reference na izvore na Internetu koji pobliže opisuju otkriveni napad
- zaglavlje i sadržaj paketa koji su doveli do prepoznavanja napada
- oznaka komponente koja je otkrila napad (npr. pojedini modul pretprocesora)
- prioritet pravila

3.4. Implementacija sustava za otkrivanje napada

Danas postoji nebrojeno mnogo organizacija kojima je imperativ kvalitetno zaštiti svoje poslovanje. Kako svaka organizacija ima mrežnu infrastrukturu koja podupire njeno poslovanje, postoji nebrojeno mnogo mrežnih topologija koje odgovaraju specifičnim zahtjevima pojedine organizacije. Upravo zbog te činjenice, uz dodatak da danas na tržištu postoji mnogo različitih proizvođača i vrsta sustava za otkrivanje napada, postoji mnogo mogućih načina implementacije zaštite mreže infrastrukture pa zadaća kvalitetne zaštite poslovanja neke organizacije nije nimalo laka. U nastavku će biti prikazane neke općenite smjernice i preporuke koje bi se trebale uzeti u razmatranje prilikom odabira sustava za otkrivanje napada, u ovisnosti o potrebama i tehnološkim mogućnostima i ograničenjima.

3.4.1. Razmatranje postojećih tehnoloških rješenja i sigurnosnih politika

Kod razmatranja postojećih tehnoloških rješenja, prvo i osnovno na što bi trebalo obratiti pažnju je postojeća mrežna infrastruktura. Potrebno je detaljno popisati sve aspekte funkcioniranja mrežne infrastrukture. To uključuje crtanje mrežnih dijagrama koji bi trebali sadržavati kompletan prikaz mreže (broj i smještaj radnih stanica, operacijski sustavi na tim radnim stanicama, broj i vrsta mrežnih uređaja koji se koriste u infrastrukturi, konfiguracija mreže itd.).

Iduće na što bi trebalo obratiti pozornost su postojeći sigurnosni sustavi unutar organizacije. To uključuje specifikaciju i popisivanje svih sigurnosnih stijena, servera za provjeru autentičnosti korisnika, sustava za enkripciju, antivirusnih rješenja, virtualnih privatnih mreža i bilo kakvih drugih sigurnosnih mehanizama.

Dalje je potrebno definirati koji su ciljevi u implementaciji sustava za otkrivanje napada. To se najlakše postiže popisivanjem najvjerojatnijih vrsta napada koji mogu zadesiti mrežnu infrastrukturu organizacije. Time se daje pregled tehnoloških rješenja koja su potrebna za obranu od takvih napada. Primjerice, tvrtke koje pružaju usluge držanja internetskih stranica na svojim poslužiteljima (*eng. web hosting*) bi najčešće bile podložne napadima koji za cilj imaju prestanak posluživanja određenih internetskih stranica. Kad se zna da su baš ti napadi najčešći i najvjerojatniji, poduzimaju se mjere nabavke opreme kojoj je zadaća (među ostalima) i sprečavanje takvih vrsta napada.

Nakon što su definirana postojeća tehnološka i sigurnosna rješenja, potrebno je definirati i postojeću sigurnosnu politiku organizacije. Upravo razmatranje sigurnosnih politika najviše pridonosi kvalitetnoj zaštiti upotrebom sustava za otkrivanje napada. Naime, sigurnosne politike definiraju što je u organizaciji dopušteno, a što nije, pa se prema tome sustav za otkrivanje napada profilira ispravnom konfiguracijom.

Također je potrebno popisati sve korisnike računalnog sustava i njihove uloge i prava pristupa u smislu tri standardna sigurnosna zahtjeva (povjerljivost, cjelovitost, raspoloživost). Kad je to na raspolaganju, idući logičan korak bi bio popisivanje akcija koje se provode nad korisnicima koji krše neka od pravila organizacije. To uključuje apsolutno sve moguće akcije i kaznene mjere, od najjednostavnijih pa do najtežih, zakonom i pravnim sustavom definiranih.

Navedeni koraci bi trebali dati dovoljno informacija koje su potrebne za kompletan pregled svih tehnoloških i sigurnosnih aspekata organizacije u cilju implementacije sustava za otkrivanje napada. No, pravi problem predstavlja pronalazak odgovarajućeg sustava za otkrivanje napada koji dovoljno dobro odgovara svim zahtjevima. Gotovo je nemoguće pronaći sustav za otkrivanje napada koji odgovara apsolutno svim zahtjevima organizacije koja ga implementira, tako da se neminovno moraju raditi određeni kompromisi. Čak i ako se pronađe sustav za otkrivanje napada koji naizgled po svim parametrima rada odgovara zahtjevima neke organizacije, potrebno je provesti dodatne mjere koje za cilj imaju i stvarno utvrđivanje sposobnosti sustava za otkrivanje napada. To uključuje kontaktiranje proizvođača sustava za otkrivanje napada i zahtijevanje da se

odabrani uređaj ispita u mrežnoj infrastrukturi organizacije. Nadalje, potrebno je saznati mogućnosti sustava što se tiče nadogradnje i implementacije novih mogućnosti u skladu sa rastom organizacije, mijenjanjem mrežne infrastrukture, sigurnosnih parametara itd.

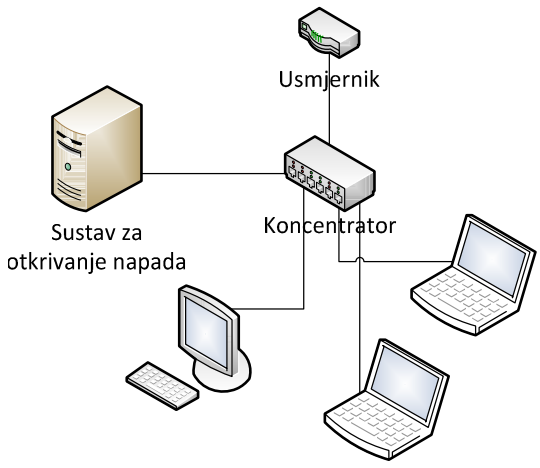
Posebno bitna stavka jednom kad se sustav za otkrivanje napada odabere i implementira, je kvalitetna podrška od strane proizvođača. To uključuje garanciju u slučaju bilo kakvog kvara, nadogradnju programske podrške (*eng. firmware*) te baze podataka o postojećim napadima i načinima obrane od njih te cijenu istih.

3.4.2. Postavljanje sustava za otkrivanje napada

Postoji više načina ispravne implementacije i smještanja sustava za otkrivanje napada u topologiju lokalne mreže. Način na koji će se sustav smjestiti unutar mreže ovisi o konfiguraciji mreže te o namjeni sustava za otkrivanje napada.

Postoje četiri osnovne kombinacije mrežne topologije i vrste sustava za otkrivanje napada koje se mogu realizirati. Te kombinacije ovise o tome je li mreža zasnovana na preklopnicima ili na koncentratorima, te je li sustav za otkrivanje napada mrežno ili računalno zasnovan. Kako su većina današnjih mreža računala zasnovane na preklopnicima te kako se implementacija računalno zasnovanih sustava za otkrivanje napada ne razlikuje od implementacije bilo kojeg drugog programskog paketa, kao jedini pravi problem za postojeće mrežne konfiguracije se nameće implementacija mrežno zasnovanih sustava za otkrivanje napada u mrežama zasnovanima na preklopnicima. Problem nastaje zbog toga što je za ispravno funkcioniranje sustava za otkrivanje napada potrebno sav promet na mreži dovesti do sustava, što u mrežama zasnovanima na preklopnicima nekad nije jednostavna zadaća.

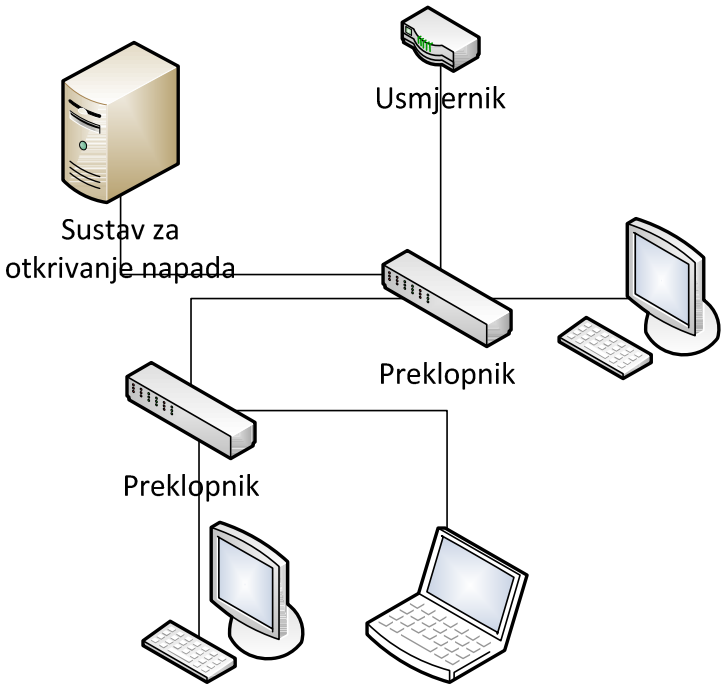
U mrežama zasnovanima na koncentratorima je vrlo jednostavno implementirati mrežni sustav za otkrivanje napada. Naime, koncentratori po dizajnu sav promet koji doputuje na jedan od njegovih mrežnih priključaka šalju na sve ostale mrežne priključke. Ta činjenica dopušta jednostavnu implementaciju sustava za otkrivanje napada jer je dovoljno sustav spojiti na jedan od mrežnih priključaka na koncentratoru i automatski kroz njega prolazi sav promet generiran na mreži. Slika 3.2 prikazuje takvu mrežnu topologiju.



Slika 3.2. Smještanje sustava za otkrivanje napada u mrežu zasnovanu na koncentratoru

U mrežama zasnovanima na preklopnicima postoje tri osnovna načina na koji se može implementirati sustav za otkrivanje napada. To su takozvani *tap*, *hub* i *span* načini spajanja sustava.

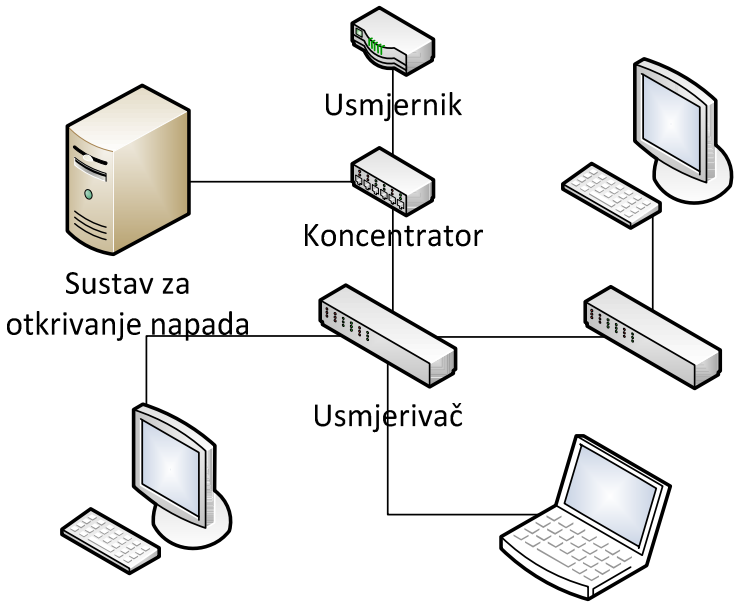
Span način spajanja sustava za otkrivanje napada koristi SPAN (*eng. Switch Port Analyzer*) mrežni priključak na preklopnicima koji služi za prisluškivanje prometa na mreži. SPAN mrežni priključak je obični mrežni priključak, no konfiguriran je tako da se na njega šalje sav promet koji prolazi nekim drugim priključkom. Ovakva implementacija sustava za otkrivanje napada je vrlo jednostavna jer ne zahtijeva nikakve dodatne sklopove za implementaciju sustava niti zahtjeva promjene u infrastrukturi ili konfiguraciji mreže. Međutim, jednostavnost instalacije donosi i nedostatke zbog kojih ovaj način implementacije nije previše raširen. Naime, na jednom preklopniku je moguće imati samo jedan SPAN mrežni priključak, što znači da u slučaju potrebe za nadzorom više mrežnih priključaka na preklopniku svi ti priključci moraju biti povezani s istim SPAN priključkom. U mrežama u kojima postoji puno prometa to znači vrlo vjerojatno smanjenje efikasnosti rada SPAN priključka zbog preopterećenja. Opterećenju može dodatno pridonijeti i sustav za otkrivanje napada budući da je jedini način da sustav šalje upozorenja o otkrivenim napadima korištenje SPAN priključka. Također, pri ovakvom načinu implementacije sustav za otkrivanje napada je podložan napadima. Slika 3.3 prikazuje izgled konfiguracije mreže u *span* načinu spajanja sustava za otkrivanje napada.



Slika 3.3. *Span* način spajanja sustava za otkrivanje napada

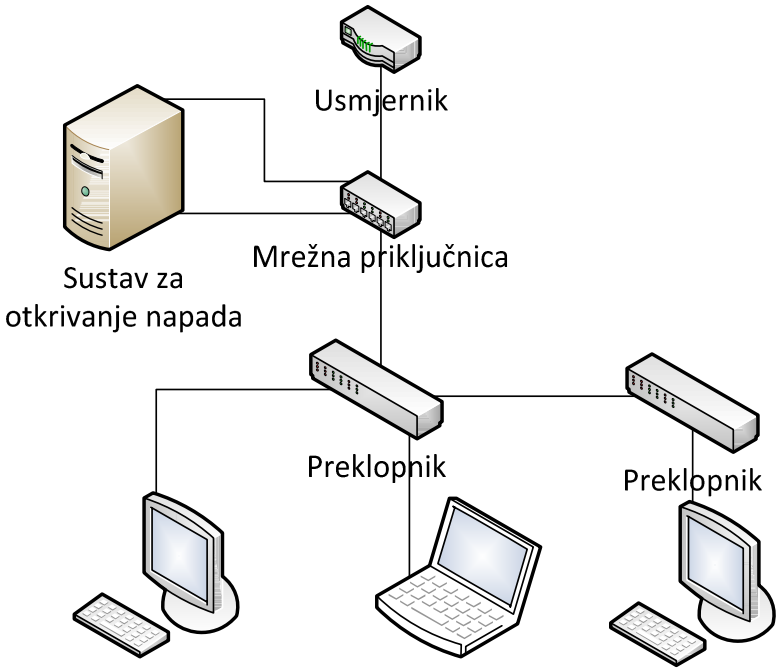
U *hub* načinu spajanja sustava za otkrivanje napada se najčešće između točaka gdje se želi analizirati mrežni promet smješta koncentrator na kojeg se spaja sustav za otkrivanje napada. Zbog načina rada koncentratora nije preporučljivo koristiti ovaj način spajanja jer je mogućnost za nastanak problema pri radu koncentratora vrlo velika. Naime, vrlo lako

se može desiti da se koncentrator preoptereti uslijed velike količine prometa koji mora prolaziti kroz njega. Slika 3.4 prikazuje konfiguraciju mreže u ovom načinu spajanja.



Slika 3.4. *Hub* način spajanja sustava za otkrivanje napada

Većinu nedostataka prethodno opisanih načina spajanja rješava *tap* način spajanja. Ovaj način spajanja koristi mrežnu priključnicu (*eng. tap*) koja efektivno ima jednak način rada kao i koncentrator, no uz drugačiju izvedbu. Mrežne priključnice su uređaji koji imaju četiri mrežna priključka. Dva priključka (najčešće označena kao priključci A i B) služe za spajanje mrežnih segmenata između kojih se želi prisluškivati mrežni promet. Specifičnost ta dva priključka je ta što su unutar mrežne priključnice direktno povezani žicom te su zbog toga mreže s tim uređajima otporne na ispade uslijed mogućih kvarova ili prekida napajanja priključnica. Druga dva mrežna priključka na ovim uređajima (najčešće označena kao *Tap* A i *Tap* B) služe za zrcaljenje prometa koji prolazi priključcima A i B. Tu je važno napomenuti da se na mrežni priključak *Tap* A zrcali promet koji ulazi u priključak A, dok se na mrežni priključak *Tap* B zrcali promet koji ulazi u priključak B. Time je efektivno ostvareno zrcaljenje ukupnog prometa koji prolazi priključnicom (što odgovara načinu rada koncentratora). Priključci *Tap* A i *Tap* B se spajaju na sustav za otkrivanje napada koji je onda u stanju pratiti sav promet na segmentu mreže koji se nadzire. Slika 3.5 prikazuje tipičnu konfiguraciju mreže u kojoj je sustav za otkrivanje napada spojen korištenjem priključnice.



Slika 3.5. *Tap* način spajanja sustava za otkrivanje napada

3.5. Nedostaci sustava za otkrivanje napada

Jedan od glavnih nedostataka mrežnih sustava za otkrivanje napada je taj što se napadi koji su u tijeku ne mogu aktivno spriječiti. Za to je potrebna intervencija mrežnih administratora koji imaju pristup dnevniku događaja i koji će čitati zapise u njima i reagirati na pojavu sumnjivih paketa. To predstavlja problem zbog još jednog nedostatka. Naime, kako je efikasno otkrivanje napada ovisno o dobro podešenim pravilima, u slučaju kad su pravila previše općenita, javlja se situacija da sustavi za otkrivanje napada mogu prijavljivati velik broj lažnih uzbuna. To dovodi do punjenja dnevnika događaja velikim brojem nepotrebnih i neupotrebljivih zapisa, što dodatno opterećuje mrežne administratore koji moraju biti sposobni prepoznati lažne uzbune i ignorirati ih.

Nadalje, budući da se u današnje vrijeme sve više koriste protokoli za kriptiranu komunikaciju, u takvim situacijama su sustavi za otkrivanje napada praktički neupotrebljivi jer nisu u stanju dekriptirati mrežni promet.

Što se tiče računalno zasnovanih sustava za otkrivanje napada, njihov glavni nedostatak je nemogućnost otkrivanja napada u stvarnom vremenu. To neminovno dovodi do određenog vremenskog razmaka između pojave napada i njegovog otkrivanja. Zbog toga se može desiti da je u trenutku otkrivanja napada on već uspješno izvršen. Računalno zasnovani sustavi za otkrivanje napada su također podložni napadima od strane zlonamjernih osoba. Kako su računalno zasnovani sustavi aplikacije koje su pokrenute na nekom od računala koji je aktivni dio mreže, moguće je da napadač dobije pristup tom računalu i jednostavno onemogući sustav, čime osigurava nesmetano provođenje daljnjih napada u mreži.

Svi sustavi za otkrivanje napada, bez obzira bili oni mrežno ili računalno zasnovani, mogu otkrivati samo one napade za koje postoji određeno pravilo u njihovoj bazi pravila. Sustavi nisu u mogućnosti otkrivati nove vrste napada sve dok se ne formiraju pravila koja prepoznaju tu vrstu napada. Nove vrste napada se pojavljuju svakodnevno što pokazuje važnost redovitog osvježavanja baze pravila.

Zbog svega navedenog, vrlo je bitno da se zaštita mreže računala ne oslanja samo na sustave za otkrivanje napada. Njih je potrebno kombinirati sa sigurnosnim stijenama i sustavima za sprječavanje napada čime se razina sigurnosti mreže računala podiže na visoku razinu u odnosu na postojanje samo sustava za otkrivanje napada.

4. Slobodno raspoloživi programski sustav za otkrivanje napada

4.1. Općenito o programskom paketu *Snort*

Snort je besplatni sustav za otkrivanje i sprječavanje napada. Autor *Snorta* je Martin Roesch. Prva verzija *Snorta* je ugledala svjetlo dana 1998. godine pod GPL (*eng. General Public Licence*) licencom otvorenog izvornog teksta i napisana je za *Linux* operacijski sustav. Od tada, *Snort* je doživio mnogo verzija (u trenutku pisanja rada se nalazi u verziji 2.8.0.1) i izdanja za većinu dostupnih operacijskih sustava, tako da je potencijalna baza korisnika *Snorta* vrlo velika. Do 2001. godine *Snort* je razvijan pod vodstvom njegovog autora. Od 2001. godine se *Snort* nalazi pod jurisdikcijom tvrtke *Sourcefire Inc.* koja je osnovana te iste godine od strane autora te je tako, uz zadržavanje postojeće licence, dobio puno više na popularnosti zbog formiranja posebnog tima koji se bavi njegovim razvojem što je pridonijelo podizanju kvalitete sustava.

Snort ima četiri osnovna načina rada: *sniffer*, *packet logger*, *network intrusion detection* i *inline* način rada.

Sniffer način rada je najjednostavniji od svih i služi samo za hvatanje paketa sa mrežnog sučelja i prikazivanje istih na ekranu. U svojim početcima, *Snort* je bio sustav kojem je ovo bio jedini način rada.

Packet logger način rada je sličan *sniffer* načinu rada i služi da se uhvaćeni paketi ne prikazuju na ekranu, već da se njihov sadržaj zapisuje u datoteku na disku.

Network intrusion detection način rada je najsloženiji i ima najviše opcija za podešavanje. Služi za analizu uhvaćenih paketa pomoću vrlo moćnog sustava pravila. Nakon što neki paket bude prepoznat pomoću nekog od pravila, *Snort* ima mogućnosti obaviti više vrsta akcija, kao što su obavještavanje administratora, zapisivanje paketa na disk itd.

Inline način rada ne koristi *libpcap* biblioteku funkcija za hvatanje paketa već pakete preuzima iz programskog sustava *iptables* i ima mogućnosti upravljati tim sustavom da odbacuje ili modificira pakete koje prepozna korištenjem svojih pravila. Time se efektivno postiže funkcionalnost sustava za sprječavanje napada.

4.2. Pravila

Sustav pravila u *Snortu* sadrži niz mogućnosti za analizu mrežnih paketa, od onih najjednostavnijih kao što su IP adrese i pristupne točke, do pretraživanja korisnog sadržaja mrežnih paketa regularnim izrazima. No, bez obzira na obilje mogućnosti, sintaksa pravila je vrlo jednostavna i lako čitljiva.

Snort pravila su logički podijeljena u dva dijela: zaglavlje i tijelo pravila. Zaglavlje je dio pravila u kojima se definiraju najjednostavniji filtri za pretraživanje paketa. Ti filtri su mrežni protokol, IP adrese izvorišta i odredišta i pripadajuće mrežne maske te pristupne

točke izvorišta i odredišta. Uz te filtre, zaglavlje pravila definira i smjer prometa u kojem će se obavljati analiza uhvaćenih paketa kao i akciju koju će *Snort* obaviti kad prepozna paket temeljem nekog od pravila. U tijelu paketa se definiraju svi ostali filtri za analizu paketa. Tijelo paketa nije obavezno stavljati u pravilo. Izgled jednog jednostavnog pravila je prikazano ispod.

```
alert tcp any any -> 192.168.1.0/24 111 (content:"|00 01 86 a5|"; msg:"mountd
access";)
```

Slika 4.1. Izgled jednostavnog *Snort* pravila

Tijelo pravila je sve što se nalazi unutar zagrada, dok je zaglavlje paketa sve ostalo. Da bi paket odgovarao nekom od pravila, njemu moraju odgovarati svi filtri navedeni u pravilu.

4.2.1. Zaglavlje pravila

Prva opcija u zaglavlju pravila je jedna od mogućih akcija koje će *Snort* obaviti kad naiđe na odgovarajući paket.

Iduća opcija u zaglavlju pravila određuje mrežni protokol. Za sada su podržani TCP, UDP, ICMP i IP protokoli.

IP adrese se u pravilu definiraju pomoću adrese mreže i adrese maske i to CIDR (*eng. Classless Inter-Domain Routing*) blokom, npr. `192.168.1.0/24`, gdje prvi dio bloka prije znaka `"/"` predstavlja adresu mreže, a drugi dio predstavlja mrežnu masku. Ispred takve definicije IP adrese može stati i negacija koja se označava uskličnikom. Dodatno, IP adrese se mogu definirati i listom CIDR blokova koja se također može negirati. Elementi te liste su odvojeni zarezom, a lista je omeđena uglatim zagradama kao npr. `![192.168.1.0/24,192.168.2.0/24]`.

Pristupne točke se definiraju ili eksplicitnim navođenjem broja pristupne točke ili skupom na način da se koristi dvotočka koja odvaja granice skupa. Tako se korištenjem skupa pristupne točke mogu definirati na tri načina: bez donje granice, npr. `:1024`, bez gornje granice, npr. `20000:` ili sa objema granicama, npr. `1000:2000`. Umjesto brojčane definicije IP adresa i pristupnih točki se može koristiti i ključna riječ any, koja označava da se radi o bilo kojoj IP adresi i pristupnoj točki. U pravilu uvijek postoje dva para mrežnih adresa i pristupnih točaka, za izvorište prometa i za odredište prometa.

Zadnja opcija koja se definira u zaglavlju pravila je smjer prometa koji će se analizirati. Smjer prometa definiran pomoću znakova `"->"` označava da IP adrese i pristupne točke lijevo od oznake smjera označavaju izvorište prometa, a IP adrese i pristupne točke desno od oznake smjera označavaju odredište. Smjer prometa definiran pomoću znakova `"<>"` označava da IP adrese i pristupne točke i sa lijeve i sa desne strane oznake mogu označavati i izvorište i odredište prometa.

Tablica 4.1 daje pregled svih mogućih akcija u *Snort* pravilu.

Tablica 4.1. Popis mogućih akcija u *Snort* pravilu

Akcija	Pojašnjenje
alert	obavještava o dolasku paketa i zapisuje ga na disk
log	zapisuje paket na disk
pass	zanemaruje paket
activate	obavještava o dolasku paketa i aktivira dinamičko pravilo
dynamic	ignorira pakete dok pravilo nije aktivirano drugim pravilom
drop	odbacuje paket i zapisuje ga na disk
reject	odbacuje paket, zapisuje ga na disk te prekida TCP vezu ili šalje ICMP unreachable port paket ako se radi o UDP vezi
sdrop	odbacuje paket

4.2.2. Tijelo pravila

Tijelo pravila sadržava tri vrste podataka. Prva vrsta su općeniti podaci koji pobliže opisuju pravilo, druga vrsta su filtri koji služe za otkrivanje paketa pomoću podataka iz zaglavlja paketa, dok su treća vrsta filtri koji služe za otkrivanje paketa pomoću korisnih podataka u paketu.

Općeniti podaci se definiraju u formatu `"ključna_riječ:vrijednost"`. Slijedi popis svih mogućih ključnih riječi i pojašnjenja općenitih podataka koji se mogu definirati u pravilu:

- `msg` - poruka koja se ispisuje prilikom dojave o pojavi paketa koji je otkriven pravilom
- `reference` - ova ključna riječ služi za definiranje referenci na izvore na Internetu koji pobliže opisuju napad koji je otkriven pravilom. Format vrijednosti ove ključne riječi je `"izvor,identifikator"` gdje izvor jedinstveno označava o kojem izvoru na Internetu je riječ, dok je identifikator niz znakova po kojima se mogu pronaći informacije o otkrivenom napadu u navedenom izvoru
- `gid` - ova ključna riječ opisuje koji dio *Snorta* je odgovoran za obavljanje akcije definirane u pravilu
- `sid` - vrijednost ove ključne riječi je broj koji jedinstveno identificira *Snort* pravilo
- `rev` - vrijednost ove ključne riječi je broj koji označava trenutnu reviziju pravila
- `classtype` - ova ključna riječ definira vrstu napada koji je otkriven. U *Snortu* je definirano oko 40 vrsta napada
- `priority` - ova ključna riječ označava koliki je prioritet obavijesti o otkrivenom napadu
- `metadata` - dodatni podaci koje autori pravila mogu dodati u pravilo. Ti podaci se definiraju u formatu `"ključna_riječ vrijednost"`

Filtri za otkrivanje paketa pomoću podataka iz zaglavlja se koriste za provjeru vrijednosti kao što su redni broj fragmenta paketa, vrijeme života paketa, tip usluge, jedinstveni

identifikacijski broj paketa, razne IP opcije, bitove za fragmentiranje i rezervirane bitove u paketu, veličinu korisnih podataka u paketu, TCP zastavice (FIN, SYN, RST, PSH, ACK, URG), sekvencijski broj, broj potvrde, veličinu prozora TCP paketa te druge manje korištene vrijednosti.

Filtri za otkrivanje pomoću korisnih podataka u paketu su najsloženiji. Ovakvih filtara u pravilu može biti više i sve vrijednosti definiranih filtara moraju biti prisutne u paketu da bi paket bio detektiran pomoću *Snorta*. Najvažnija ključna riječ kod ovih filtara je `content` koja definira sadržaj koji će se tražiti u paketu. Taj sadržaj može biti definiran pomoću niza tekstualnih znakova, niza okteta ili kombinacije to dvoje. Ostale ključne riječi kod ove vrste filtara su modifikatori koji pobliže određuju kako će *Snort* pretraživati definirane nizove znakova ili okteta u paketu. Svi modifikatori se odnose na prethodno definiranu ključnu riječ `content` i imaju format `"ključna_riječ:vrijednost"`. Slijedi popis najčešće korištenih modifikatora:

- `nocase` - postojanje ove ključne riječi govori *Snortu* da ne obraća pozornost na velika i mala slova u nizu tekstualnih znakova
- `rawbytes` - postojanje ove ključne riječi govori *Snortu* da zanemari rezultate pretprocesora i da promatra pakete kakvi su došli na mrežno sučelje
- `depth` - ova ključna riječ govori *Snortu* do koje dubine u korisnom sadržaju paketa će tražiti definirani niz tekstualnih znakova i okteta
- `offset` - ova ključna riječ označava od koje pozicije u korisnom sadržaju paketa će *Snort* tražiti definirani niz tekstualnih znakova i okteta
- `distance` - svrha ove ključne riječi je ista kao i kod ključne riječi `depth`, no razlika je u tome što se odnosi na relativnu poziciju unutar paketa koja odgovara poziciji zadnjeg prepoznatog niza tekstualnih znakova i okteta, dok se `depth` odnosi na apsolutni odmak od početka korisnog sadržaja paketa
- `within` - ima odnos sa ključnom riječi `offset` kakav ključna riječ `distance` ima sa riječi `depth`
- `http_client_body` - ograničava pretragu na normalizirano tijelo HTTP zahtijeva
- `http_uri` - ograničava pretragu na normalizirani sadržaj u URI-ju HTTP zahtijeva
- `pcre` - služi za definiciju regularnih izraza pomoću kojih se obavlja pretraživanje korisnog sadržaja paketa

4.3. Instalacija i konfiguracija programskog paketa *Snort*

Kao što je već rečeno, programski paket *Snort* je dostupan za mnoštvo operacijskih sustava i platformi. Instalacijske procedure na svakom od operacijskih sustava se razlikuju, no u osnovi su vrlo slične. Na operacijskom sustavu *Windows* instalacija se svodi na pokretanje instalacijske datoteke i odgovaranje na par pitanja koja instalacijska procedura postavlja pred korisnika. Prvo od takvih pitanja je odabir načina zapisivanja podataka o događajima unutar *Snorta* (*eng. event log*). Nude se četiri mogućnosti: zapisivanje u tekstualnu datoteku, zapisivanje u *MySQL* bazu podataka, zapisivanje u *Oracle* bazu podataka i zapisivanje u *SQL Server* bazu podataka. Sljedeće pitanje koje se postavlja pred korisnika je odabir komponenti koje korisnik želi instalirati. Na odabir su ponuđeni osnovni moduli programskog paketa *Snort*, dinamički moduli koji se integriraju u pretprocesor *Snorta*, dokumentacija i sheme odabrane baze podataka. Zadnje pitanje za korisnika je odabir instalacijskog direktorija. Za ispravan rad programskog paketa *Snort* na

operacijskom sustavu *Windows* potrebna je *WinPcap* biblioteka funkcija za hvatanje mrežnih paketa. Nešto više o toj biblioteci funkcija je napisano u poglavlju 6.1.2 ovog rada.

Prije pokretanja programskog paketa *Snort*, potrebno ga je ispravno konfigurirati da bi radio u skladu za zahtjevima okoline u kojoj se pokreće. Kako programski paket *Snort* temelji svoj rad na skupu pravila prema kojima obavlja prepoznavanje uhvaćenih paketa, potrebno je prije ikakve konfiguracije sa internetske stranice programskog paketa *Snort* preuzeti datoteku koja sadrži najsvježiji popis pravila. Ta pravila se trebaju smjestiti u `rules` direktorij unutar instalacijskog direktorija *Snorta*. Pravila su podijeljena u više datoteka od kojih svaka predstavlja jedan skup tih pravila sa sličnim značajkama. Takav način distribucije pravila omogućava da se na jednostavan način u konfiguracijskoj datoteci programskog paketa *Snort* definira koji skup pravila se želi koristiti, a koji ne. Datoteka koja je dostupna na internetskoj stranici programskog paketa *Snort* sadži 50-ak datoteka za isto toliko skupova pravila. Neki primjeri tih skupova pravila su pravila koja služe za nadzor HTTP prometa, POP3 prometa, SMTP prometa, za otkrivanje DoS napada, za otkrivanje napada koji iskorištavaju ranjivosti u aplikacijama, za otkrivanje napada koji se provode SQL injekcijom itd.

Nakon preuzimanja i smještanja pravila u instalacijski direktorij programskog paketa *Snort*, potrebno je obaviti i konfiguraciju *Snorta*, koja je ključna za ispravan rad programskog paketa. Konfiguracija se provodi ručnim uređivanjem niza datoteka koje su smještene u `etc` direktoriju unutar instalacijskog direktorija programskog paketa. Glavna konfiguracija se provodi uređivanjem datoteke `snort.conf` u nekoliko koraka. Kroz uređivanje te datoteke se mogu podesiti svi aspekti rada programskog paketa *Snort*.

Prvi korak pri uređivanju `snort.conf` datoteke je definiranje varijabli koje služe kao zamjena za liste IP adresa i pristupnih točaka koje u određenim mrežnim okolinama mogu biti prevelike i nepraktične za korištenje u pravilima. Za definiranje varijabli se koriste dvije ključne riječi: `var` i `portvar`. Izgled linija koje definiraju jednostavnu listu IP adresa i pristupnih točaka je prikazan ispod.

```
var MY_NETWORK [192.168.1.0/24,192.168.2.0/24]
portvar MY_PORTS 2000:3000
```

Slika 4.2. Primjer definiranja varijabli prilikom konfiguracije *Snorta*

Nakon što su se varijable definirale na ovakav način, svaka pojava definiranih lista IP adresa i pristupnih točaka se može zamijeniti odgovarajućim varijablama na način kako je prikazano ispod.

```
alert tcp $MY_NETWORK $MY_PORTS -> 192.168.1.0/24 111 (content:"|00 01 86 a5|";
msg:"mountd access";)
```

Slika 4.3. Primjer pravila koje koristi korisnički definirane varijable

Idući korak pri uređivanju `snort.conf` datoteke je podešavanje globalnih postavki programskog paketa *Snort*. Globalne postavke se podešavaju korištenjem ključne riječi

`config` nakon koje slijedi naziv postavke koja se podešava, nakon čega može ili ne mora slijediti lista parametara potrebnih za odabranu postavku.

Daljnji korak je podešavanje pretprocesorskih i izlaznih modula. Kako je podešavanje za svaki od modula ovisno o mogućnostima modula, podešavanje tih modula može biti jako zahtjevan posao i upravo o ispravnosti konfiguracije ovih modula ovisi ispravan rad programskog paketa *Snort*. Podešavanje ovih modula se obavlja korištenjem ključnih riječi, na isti način kao i za podešavanje globalnih opcija programskog paketa *Snort*. Jedina razlika je što se umjesto ključne riječi `config` za podešavanje pretprocesorskih modula koristi ključna riječ `preprocessor`, dok se za podešavanje izlaznih modula koristi ključna riječ `output`.

Zadnji korak pri uređivanju `snort.conf` datoteke je podešavanje skupa pravila koji će se koristiti za prepoznavanje paketa. To podešavanje se obavlja upotrebom ključne riječi `include`, nakon koje slijedi putanja do datoteke koja sadrži potreban skup pravila. Višestrukim navođenjem ove ključne riječi i putanja do datoteka sa skupom pravila se može jednostavno urediti lista pravila koji će se koristiti pri radu programskog paketa *Snort*.

4.4. Pokretanje programskog paketa *Snort*

Kao što je već rečeno, programski paket *Snort* ima četiri osnovna načina rada. U daljnjem tekstu će biti opisano na koji se način *Snort* pokreće da radi u nekom od ta četiri osnovna načina rada. Programski paket *Snort* je ostvaren kao konzolna aplikacija pa se pokretanje *Snorta* obavlja navođenjem imena izvršne datoteke nakon koje slijedi lista parametara.

Sniffer način rada je najjednostavniji pa je tako i pokretanje *Snorta* u ovom načinu rada jako jednostavno. Tablica 4.2 prikazuje parametre i njihovo značenje pomoću kojih se programski paket *Snort* pokreće u ovom načinu rada.

Tablica 4.2. Parametri za pokretanje *Snorta* u *sniffer* načinu rada

Parametri	Pojašnjenje
-v	ispisivanje zaglavlja paketa na ekran
-d	ispisivanje sadržaja paketa na ekran
-e	ispisivanje zaglavlja paketa sa podatkovnog sloja mrežnog protokola

Packet logger način rada se odabire navođenjem putanje do postojećeg direktorija na tvrdom disku računala, u koji će se pohranjivati podaci o uhvaćenim paketima. U tom direktoriju *Snort* stvara strukturu direktorija prema IP adresama odredišta ili izvorišta paketa za lakše pretraživanje kada se nakupi velika količina podataka. Parametri za pokretanje *Snorta* u ovom načinu rada se koriste u suradnji s parametrima koji služe za pokretanje u *sniffer* načinu rada. Tablica 4.3 prikazuje moguće parametre za pokretanje *Snorta* u ovom načinu rada.

Tablica 4.3. Parametri za pokretanje *Snorta* u *packet logger* načinu rada

Parametri	Pojašnjenje
-l putanja	definicija putanje do direktorija za pohranjivanje uhvaćenih paketa
-h ip_adresa	definiranje ranga IP adresa lokalne mreže
-b	pohranjivanje uhvaćenih paketa u binarnom obliku

IP adresa lokalne mreže se definira u slučaju ako korisnik želi hvatati pakete iz samo jednog smjera. Ako se, na primjer, definira da je IP adresa lokalne mreže `192.168.1.0/24`, onda će se na tvrdi disk zapisivati samo dolazni paketi, to jest samo oni paketi koji u svom zaglavlju kao odredište imaju IP adresu iz zadanog ranga IP adresa.

Za pokretanje *Snorta* u *network intrusion detection* načinu rada je dovoljno uz već navedene parametre dodati i parametar koji definira naziv konfiguracijske datoteke. Tada će *Snort* koristiti pravila koja su definirana u toj konfiguracijskoj datoteci za prepoznavanje uhvaćenih paketa. Primjer pokretanja programskog paketa *Snort* u ovom načinu rada je prikazan ispod.

```
./snort -d -h 192.168.1.0/24 -l ./log -c snort.conf
```

Slika 4.4. Pokretanje *Snorta* u *network intrusion detection* načinu rada

U ovom načinu rada je moguće definirati i parametre koji kontroliraju način na koji će programski paket *Snort* obavještavati korisnika o otkrivenim napadima. Tablica 4.4 daje pregled načina obavještavanja korisnika i pripadajuća objašnjenja.

Tablica 4.4. Parametri za definiranje obavještavanja u *Snortu*

Parametri	Pojašnjenje
-A fast	jednostavan format obavijesti, prikazuju se vrijeme, poruka i izvorišne/odredišne IP adrese i pristupne točke
-A full	potpuni format obavijesti, dodaje ispis zaglavlja paketa
-A unsock	šalje obavijesti na mrežni *socket* što omogućuje drugim aplikacijama da dobivaju obavijesti
-A none	isključuje obavještavanje
-A console	koristi *fast* format obavijesti, no šalje ih na ekran umjesto u datoteku
-A cmg	dodaje potpuni ispis paketa podacima iz *full* načina obavještavanja
-s	zapisuje obavijesti u sistemski zapisnik

Pokretanje u *inline* načinu rada je nešto složenije od pokretanja u ostalim načinima rada. Naime, za ovaj način rada se programski paket *Snort* povezuje sa programskim paketom *iptables* te preko njega dohvaća pakete, umjesto korištenjem *WinPcap* biblioteke funkcija.

Da bi se programski paketi *Snort* i *iptables* uspješno povezali, oba paketa moraju biti prevedena u izvršni oblik korištenjem posebnih parametara koji omogućavaju povezivanje ta dva programska paketa. Za ispravan rad, prvo je potrebno pokrenuti *iptables* na način prikazan ispod.

```
iptables -A OUTPUT -p tcp --dport 80 -j QUEUE
```

Slika 4.5. Način pokretanja programskog paketa *iptables* za rad sa *Snortom*

Nakon toga je potrebno pokrenuti programski paket *Snort* sa odgovarajućim parametrima koji su do sad opisani, uz dodatak parametra `-Q` koji omogućava dohvat paketa iz *iptablesa*.

4.5. Prednosti i nedostaci programskog paketa *Snort*

Jedna od glavnih prednosti programskog paketa *Snort* je činjenica da je *Snort* besplatan i uz to otvorenog izvornog teksta. Uz veliku bazu potencijalnih korisnika, to efektivno znači da svaka osoba koju to zanima može dobiti kompletan uvid u način rada programskog paketa *Snort*. To je korisno zbog više razloga. Praktički svatko može raditi na unaprjeđivanju *Snorta* ispravljajući uočene pogreške i implementirajući nove funkcionalnosti. To znači da će se jako teško desiti da ovako velik projekt iz bilo kojeg razloga propadne i iznenada mu se prekine daljnji razvoj. Daljnja prednost programskog paketa *Snort* je njegov vrlo moćan sustav pravila i odlična podrška za proširivost dodatnim modulima bez kojih otkrivanje napada ne bi funkcioniralo.

No, taj moćan sustav pravila i mogućnosti proširivosti se u određenim situacijama mogu pokazati i kao jedan od temeljnih nedostataka ovog programskog paketa. Naime, potrebno je dobro poznavati način rada mrežnih protokola da bi se pisala efikasna i univerzalna pravila i da bi se programski paket *Snort* mogao konfigurirati za ispravan i efikasan rad. Svaki od modula dodatno komplicira konfiguraciju *Snorta* svojim mogućnostima. Napisano je mnogo knjiga koje se isključivo bave programskim paketom *Snort*, što dokazuje činjenicu da rad sa *Snortom* nije trivijalan.

5. Komercijalni sustav za otkrivanje napada

5.1. Sustav za sprječavanje napada *Proventia M*

Proventia M je komercijalni sklopovski i programski sustav za sprječavanje napada. Razvijen je od strane tvrtke IBM (*eng. International Business Machines*), tj. njezinog odjela koji se bavi sigurnošću u informatičkim tehnologijama, ISS Inc. (*eng. Internet Security Systems Incorporated*). Uređaji iz serije *Proventia M* su samo jedan u nizu proizvoda iz vrlo široke ponude uređaja ove tvrtke. Serija *Proventia M* modela je sastavljena od robusnih i cijenom pristupačnih uređaja koji uz minimalnu i vrlo jednostavnu konfiguraciju efikasno zaustavljaju mnoge vrste napada na sigurnost računalne mreže kombiniranjem antivirusa, sigurnosne stijene, sustava za obranu od neželjene pošte, sustava za filtriranje sadržaja na Internetu te sustava za otkrivanje i sprječavanje napada.

Serija *Proventia M* je sastavljena od tri modela uređaja: *M10*, *M30* i *M50*. Sva tri modela imaju operacijski sustav sa identičnim karakteristikama, tako da funkcionalnost obrane od napada u niti jednom od ova tri modela nije umanjena. Operacijski sustav je zasnovan na *Linux* jezgri. Ono u čemu se modeli *Proventia M* serije uređaja razlikuju su sklopovske specifikacije. Tablica 5.1 prikazuje osnovne sklopovske razlike uređaja *Proventia M10*, *M30* i *M50*.

Tablica 5.1. Osnovne razlike *Proventia M* modela uređaja

Karakteristike	Proventia M10	Proventia M30	Proventia M50
Broj mrežnih sučelja	3 (do 100 Mbps)	3 (do 100 Mbps)	3 (do 1000 Mbps)
Dodatno napajanje	Ne	Ne	Da
Dodatni tvrdi disk	Ne	Ne	Da
Težina (kg)	1.2	5.5	27.2
Dimenzije (ŠxVxD) (cm)	24.9 x 3.8 x 17.5	42.8 x 36 x 4.4	7.7 x 43 x 64.8
Broj podržanih korisnika	100	500	2500
Propusnost sigurnosne stijene	100 Mbps	200 Mbps	1600 Mbps

Iz priložene tablice se da zaključiti da je model *Proventia M10* namijenjen obrani manjih mrežnih infrastruktura, dok su *Proventia M30* i *M50* namijenjeni obrani srednje velikih i velikih mrežnih infrastruktura, respektivno.

Proventia M serija uređaja je specifično namijenjena sprječavanju napada. Iako su u ovom diplomskom radu opisani sustavi za otkrivanje napada, upravo zbog činjenice da sprječavanje napada ne bi bilo efikasno bez kvalitetnog sustava za otkrivanje napada, u daljnjim potpoglavljima je ova serija uređaja opisana detaljnije.

5.2. Mogućnosti sustava *Proventia M*

5.2.1. Inicijalna konfiguracija

Instalacija i konfiguracija uređaja *Proventia M* se sastoji od nekoliko koraka. Prvo korak u instalaciji je obavljanje inicijalne konfiguracije koja je specifična za mrežu u koju se uređaj

instalira. Inicijalna konfiguracija se provodi korištenjem pomoćnika koji na pregledan i jednostavan način dopušta unos potrebnih parametara i to korištenjem preglednika interneta sa računala spojenog u mrežu sa uređajem preko ukriženog (*eng. crossover*) mrežnog kabela.

Prilikom inicijalne konfiguracije treba odrediti način rada uređaja. *Proventia M* ima dva osnovna načina rada: transparentni i rad kao usmjernik. U transparentnom načinu rada uređaj nije aktivni element mreže računala već samo prosljeđuje pakete na odgovarajuće sučelje na osnovi MAC (*eng. Media Access Control*) adresa koje čita iz paketa. Samim time je i nevidljiv za napadača[1]. Za funkcioniranje u transparentnom načinu rada nije potrebno mijenjati mrežnu infrastrukturu i njenu konfiguraciju (IP adrese, rute, mrežne maske itd.), no za ispravan rad se zahtijeva postojanje usmjernika. U ovom načinu rada uređaj ne podržava prevođenje mrežnih adresa (*eng. NAT - Network Address Translation*).

Ako uređaj funkcionira u načinu rada usmjernika, ponaša se kao aktivni element mreže i nudi sve funkcionalnosti koje nudi i usmjernik, što uključuje podršku za definiciju ruta i prevođenje mrežnih adresa. Da bi uređaj ispravno radio u ovom načinu rada, moraju mu se podesiti rute za svaki segment mreže koji je spojen na njega. Usmjernicima u mreži se također moraju podesiti rute da bi znali kako usmjeravati promet prema svakom od mrežnih segmenata spojenih na uređaj. Potrebno je naglasiti da sprječavanje napada nije ograničeno upotrebom bilo kojeg od ova dva načina rada uređaja.

Nakon što je odabran način rada uređaja, ostaje konfigurirati par dodatnih opcija koje ovise o odabranom načinu rada. Za funkcioniranje u načinu rada usmjernika je potrebno postaviti ime uređaja[2], IP adrese i mrežne maske svih mrežnih sučelja na uređaju, IP adrese DNS (*eng. Domain Name System*) servera te lozinke za pristup uređaju. U transparentnom načinu rada je uz ime uređaja, IP adresa DNS servera i lozinki za pristup uređaju (postavke su identične postavkama u načinu rada usmjernika) potrebno postaviti i IP adresu preko koje se obavlja konfiguracija uređaja.

Ostatak postavki u inicijalnoj konfiguraciji se odnosi na postavljanje prava pristupa uređaju, tj. s kojih se računala može pristupiti konfiguracijskom sučelju uređaja. Također je potrebno postaviti i automatsko preuzimanje novih verzija pravila za otkrivanje napada, baze podataka za antivirusni modul, te sistemskih aplikacija koje upravljaju radom uređaja. Dalje, potrebno je definirati koje usluge će biti pokrenute na uređaju. Te usluge olakšavaju rad sa uređajem, a nudi se izbor sljedećih usluga:

- DHCP (*eng. Dynamic Host Configuration Protocol*) - ova usluga olakšava konfiguraciju postavki mrežnih sučelja automatski pridružujući IP adrese svakom od sučelja.
- SMTP (*eng. Simple Mail Transfer Protocol*) - omogućava obavještavanje administratora o događajima putem elektroničke pošte.

[1] Uređaj je nevidljiv za napadača jer u transparentnom načinu rada mrežnim sučeljima uređaja nisu pridružene IP adrese pa ta mrežna sučelja imaju sposobnost samo prosljeđivati pakete na ostala sučelja.
[2] Potrebno je unijeti FQDN (*eng. Fully Qualified Domain Name*) oblik imena uređaja, npr. abyss.zemris.fer.hr

- SNMP (*eng. Simple Network Management Protocol*) - prati statistike o radu mrežnih uređaja spojenih na sustav u svrhu obavještavanja o svim nepravilnostima u radu.
- SSH (*eng. Secure Shell*) - omogućava udaljeni pristup uređaju.
- HTTP (*eng. Hyper Text Transfer Protocol*) - omogućava preuzimanje nadogradnji operacijskog sustava uređaja direktno iz konfiguracijskog sučelja.

Konačno, potrebno je konfigurirati i sustav obavještavanja o svim događajima vezanim uz uređaj.

5.2.2. Mogućnosti sigurnosne stijene

Podešavanje sigurnosne stijene u *Proventia M* seriji uređaja je ovisno o načinu rada uređaja. U načinu rada usmjernika, sigurnosna stijena omogućava definiranje pristupnih politika koje opisuju kakav promet će se propustiti kroz stijenu, a kakav ne. Tako se može definirati niz specifičnih pristupnih politika u ovisnosti o smjeru prometa (promet koji odlazi iz lokalne mreže ili promet koji dolazi u lokalnu mrežu), specifičnom računalu koje šalje i prima promet, pristupnim točkama na strani primatelja ili pošiljatelja te mrežnom protokolu. Tako je na primjer moguće definirati pristupne politike koje dopuštaju pristup serveru organizacije na kojem su smještene internetske stranice samo pomoću HTTP i HTTPS protokola ili je moguće definirati pristupne politike koje potpuno zabranjuju bilo kakvu vrstu prometa koristeći FTP protokol.

U transparentnom načinu rada se filtriranje mrežnog prometa obavlja na podatkovnom sloju OSI mrežnog modela zbog već spomenute činjenice da u ovom načinu rada mrežna sučelja nemaju dodijeljenu IP adresu. U ovom načinu rada su moguće dvije vrste postavki. Prva je filtriranje po MAC adresama mrežnih sučelja smještenih unutar lokalne mreže. Ovo filtriranje se obavlja tako da se definira lista MAC adresa kojima je dopuštena komunikacija u lokalnoj mreži. Svim paketima u kojima su prisutne MAC adrese koje nisu definirane u toj listi je zabranjen prolazak kroz sigurnosnu stijenu. Druga vrsta postavki je filtriranje po protokolu podatkovnog sloja OSI mrežnog modela. U transparentnom načinu rada je tvornički definirano nekoliko pristupnih politika koje su potrebne da bi se moglo pristupiti uređaju za potrebe konfiguracije uređaja.

Sigurnosna stijena ima mogućnost izvještavanja o svom radu. Izvještavanje se može obavljati putem elektroničke pošte ili SNMP protokola. Izvještavanje se obavlja za nekolicinu sumnjivih radnji na lokalnoj mreži prikazanih u sljedećoj listi:

- prilikom pojave poplave syn paketima[3]
- prilikom pojave pinga smrti[4]
- prilikom pojave lažirane IP adrese u paketu
- prilikom pojave loše formiranog paketa
- prilikom pojave greške u radu sigurnosne stijene

[3] syn poplava je generiranje zahtjeva za TCP vezom u mjeri većoj od one u kojoj odredišno računalo više nije u stanju odgovoriti na sve zahtjeve u normalnim vremenskim okvirima pa prestaje s radom

[4] ping smrti je slanje ping paketa koji su veći od veličine međuspremnika za privremeni smještaj paketa (tipična veličina spremnika je 65535 okteta), pri čemu se dešava preljev spremnika pa računalo prestaje s radom

- prilikom pojave prometa koji je blokiran u skladu s pristupnim politikama
- prilikom pojave prometa koji je propušten u skladu s pristupnim politikama
- kad sigurnosna stijena nije u stanju pronaći pravilo koje bi opisalo paket
- prilikom mijenjanja postavki sigurnosne stijene od strane korisnika
- kad sigurnosna stijena generira statistike o svom radu
- kad sigurnosna stijena otkrije DNS upite i odgovarajuće odgovore

Od ostalih mogućnosti sigurnosne stijene u *Proventia M* seriji uređaja je važno spomenuti i prevođenje mrežnih adresa i pristupnih točaka koje omogućuju sakrivanje lokalnih IP adresa od vanjskog svijeta. Jedan od tipičnih scenarija korištenja prevođenja mrežnih adresa je sakrivanje lokalne IP adrese računala na kojem su smještene internetske stranice, istovremeno dopuštajući neometan pristup internetskim stranicama.

5.2.3. Mogućnosti filtera elektroničke pošte i sadržaja sa Interneta

Filter elektroničke pošte je modul koji koristi više tehnika za filtriranje elektroničke pošte s ciljem smanjivanja broja neželjenih elektroničkih poruka. Te tehnike su analiza sadržaja elektroničke pošte, uspoređivanje sa bazom podataka o postojećim neželjenim porukama te crna i bijela lista:

- analiza sadržaja elektroničke pošte funkcionira na način da uređaj ima bazu podataka u kojoj su pohranjeni podaci o ključnim riječima, slikama i linkovima koji su povezani sa neželjenom poštom. Uređaj skenira svaku elektroničku poruku te se poruka, u slučaju da sadrži dovoljan broj prepoznatih elemenata, označava kao neželjena.
- crna lista je popis adresa elektroničke pošte koji se definira od strane administratora, a predstavljaju adrese koje su poznate kao izvori neželjene pošte
- bijela lista je popis adresa elektroničke pošte koji se također definira od strane administratora, no u ovom slučaju popis predstavlja adrese koje su legitiman izvor elektroničkih poruka.

Filter elektroničke pošte također ima mogućnost obavještavanja o otkrivenim neželjenim porukama (putem elektroničke pošte ili SNMP protokola). Sljedeće akcije generiraju obavijesti:

- označavanje poruke kao neželjene
- skeniranje poruke
- generiranje statistika o radu filtera

Filter sadržaja s interneta radi na sličnom principu kao i filter elektroničke pošte. Također postoji više tehnika kojima se filter služi prilikom otkrivanja nepoćudnog sadržaja. Prva tehnika je analiza sadržaja internetskih stranica koja po funkcionalnosti odgovara analizi sadržaja elektroničke pošte. Dalje, moguća je definicija liste IP adresa koje su poznate kao izvori nepoćudnog sadržaja. Kad korisnik zatraži neki sadržaj koji je smješten ja jednoj od tih IP adresa, taj zahtjev će biti blokiran. Također je moguća definicija liste IP adresa koje su legitimne pa kad korisnik zatraži neki sadržaj koji je smješten na jednoj od takvih IP adresa, taj zahtjev će biti propušten kroz filter. Na posljetku, moguće je definirati i listu lokalnih IP adresa koje imaju neograničen pristup Internetu.

Filter internetskog sadržaja isto koristi sistem obavještavanja i to za sljedeće akcije:

- prilikom blokiranja zahtjeva za sadržajem na Internetu
- prilikom korisničkog zahtjeva za ručno blokiranim sadržajem (koristeći listu IP adresa koje su poznate kao izvori nepoćudnog sadržaja)
- prilikom generiranja statistika o radu filtera

5.2.4. Mogućnosti antivirusnog modula

Antivirusni modul koristi dvije tehnike za otkrivanje virusa i ostalih zloćudnih programa. Prva tehnika je prepoznavanje koristeći bazu podataka koja sadrži takozvane potpise koji opisuju tipične karakteristike svakog od postojećih zloćudnih programa. Uređaj obavlja skeniranje elektroničke pošte, prijenosa datoteka i internetskog prometa. Kad uređaj prepozna neki zloćudni program, smješta ga u karantenu. Skeniranje kriptiranih datoteka, datoteka zaštićenih lozinkom te bilo kojih tipova datoteka koji su korisnički definirani u listi iznimki se ne obavlja.

Druga tehnika je pretraživanje prometa tražeći karakteristične uzorke ponašanja zloćudnih programa, npr. mogućnost samostalnog širenja. Ova tehnika omogućava uređaju otkrivanje zloćudnih programa za koje još ne postoji odgovarajući potpis u za to namijenjenoj bazi podataka.

Obavještavanje u ovom modulu se svodi na dvije stvari: obavještavanje o otkrivenim zloćudnim programima te smještanju istih u karantenu, te obavještavanje prilikom generiranja statistika o radu antivirusnog modula.

5.2.5. Mogućnosti sustava za otkrivanje i sprječavanje napada

Modul za otkrivanje i sprječavanje napada je centralni modul serije uređaja *Proventia M*. Ovaj modul ima mogućnost u stvarnom vremenu otkrivati i blokirati bilo kakvu vrstu napada na sigurnost računalne mreže, uključujući i akcije koje mogu poslužiti otkrivanju informacija o strukturi mreže kao što je skeniranje pristupnih točaka.

Osnovno podešavanje ovog modula se svodi na jednostavno uključivanje i isključivanje rada modula, kao i dodatno uključivanje i isključivanje sprječavanja napada. Tako je moguće ovaj modul podesiti da samo otkriva napade, a po potrebi da ih sprječava.

Kako je ovaj modul (uz sigurnosnu stijenu) najvažniji od svih funkcionalnosti koje nudi serija uređaja *Proventia M*, logično je da nudi jednak broj mogućnosti zapisivanja obavijesti u dnevnik događaja kao i sigurnosna stijena. U sljedećoj listi su nabrojani svi događaji koji izazivaju zapisivanje obavijesti u dnevnik:

- blokiranje napada
- otkrivanje napada
- otkrivanje procesa skeniranja mrežne infrastrukture (npr. *ping* paketima)
- generiranje statistika o radu modula
- kad je karantensko pravilo dodano, obrisano, isteklo ili je pronađen paket koji odgovara pravilu
- otkrivanje neispravno definiranog paketa

- otkrivanje neispravno definiranog protokola
- blokiranje TCP veze

Kao i za sve ostale module, i ovdje vrijedi da se obavještavanje može provoditi pomoću elektroničke pošte ili SNMP protokola.

Zadnja stvar koju je moguće podešavati je filtriranje obavijesti koji se zapisuju u dnevnik događaja. Kako proces otkrivanja i sprječavanja napada može generirati velik broj obavijesti od kojih su neke vrlo bitne, a neke manje ili nimalo bitne, potrebno je moći podešavati koje se vrste obavijesti prikazuju u dnevniku događaja, a koje ne. Filtriranje ovih obavijesti se obavlja po IP adresama napadača ili mete napada, protokolu te pristupnim točkama napadača ili mete napada.

5.3. Prednosti i nedostaci sustava *Proventia M*

Jedna od najvećih prednosti *Proventia M* serije uređaja je bogatstvo ponuđenih opcija i modula. Već spomenuti moduli za antivirus, sigurnosnu stijenu i filtriranje elektroničke pošte i sadržaja s Interneta se jako dobro nadopunjuju sa osnovnom funkcionalnošću ovih uređaja, otkrivanjem i sprječavanjem napada. Svi moduli se redovito i automatski nadopunjuju novim pravilima za otkrivanje svih vrsta napada što je također jedna od velikih prednosti jer su bez redovitih nadogradnji ovi uređaji praktički neupotrebljivi.

Još jedna prednost *Proventia M* serije uređaja je njihov kompaktan dizajn, što omogućava praktično smještanje na svako mjesto, što olakšava integraciju u postojeće mrežne infrastrukture jer nije potrebno brinuti o provođenju mrežnih kabela od uređaja do mjesta njihove integracije u mrežnoj infrastrukturi. Također, konfiguracijsko sučelje je implementirano u programskom jeziku *Java* što omogućava da se uređaji iz *Proventia M* serije mogu konfigurirati koristeći bilo koji Internet preglednik koji ima podršku za izvođenje aplikacija napisanih u programskom jeziku *Java*.

Jedan od najvećih nedostataka *Proventia M* serije uređaja je model naplate korištenja uređaja. Iako je početna cijena uređaja relativno niska u odnosu na druge slične sustave za sprječavanje napada, nakon godine dana besplatnog korištenja automatskih nadogradnji pravila za otkrivanje i sprječavanje napada, potrebno je godišnje nadoplaćivati licencu za preuzimanje novih pravila. Ovaj model dodatno povećava cijenu uređaja pa je potrebno unaprijed razmišljati i o ovim troškovima prilikom procesa izbora sustava za otkrivanje i sprječavanje napada. Također, za korištenje antivirusnog modula i modula za filtriranje elektroničke pošte i sadržaja s interneta je potrebno kupovati godišnju licencu jer ti moduli koriste patentirane tehnologije drugih tvrtki.

6. Ostvarenje jednostavnog sustava za otkrivanje napada

6.1. Način i tehnologija implementacije

6.1.1. Programski jezik

Za ostvarenje jednostavnog sustava za otkrivanje napada su odabrani programski jezik *C#* i *.NET Framework* verzije 3.5. Sustav je napisan koristeći programski paket za razvoj aplikacija (*eng. IDE - Integrated Development Environment*) *Visual Studio 2008 Professional* tvrtke *Microsoft*. Na početku rada na ostvarenju sustava se kao prvi mogući problem pokazala odabrana tehnologija implementacije. Naime, kako prevodioci za programski jezik *C#* ne prevode izvorni tekst direktno u strojni, već koriste takozvani IL (*eng. Intermediate Language*) koji se onda pri pokretanju i korištenju aplikacije prevodi u strojni izvršni program (*eng. JIT - Just In Time compiling*), *C#* nije pogodan za razvoj aplikacija koje moraju proračune raditi u stvarnom vremenu. Kako je hvatanje mrežnih paketa i njihova analiza jako zahtjevna procedura, posebno na jako prometnim mrežama, vrlo lako se moglo desiti da ovo ostvarenje na kraju ne bi funkcioniralo kako treba. Međutim, kako je mreža računala u kojoj je ostvarenje ispitano malena (svega tri računala), ostvarenje sustava za otkrivanje napada je funkcioniralo više nego zadovoljavajuće.

6.1.2. *WinPcap*

Za hvatanje paketa je zaslužan programski paket *WinPcap* koji je *Windows* implementacija *libpcap* API-ja (*eng. Application Programming Interface*) za hvatanje paketa, koji je originalno napisan za operacijski sustav *Unix*.

libcap, a time i *WinPcap* su industrijski priznat alati za hvatanje paketa na podatkovnom sloju OSI mrežnog modela. Upravo zbog činjenice da funkcioniraju na podatkovnom sloju OSI mrežnog modela, dozvoljavaju aplikacijama da hvataju (i usput šalju) pakete bez obzira na aplikacijski protokol. *WinPcap* se sastoji od pogonskog programa (*eng. driver*) preko kojeg se *Windows* operacijski sustav proširuje tako da je dopušten pristup nižim slojevima OSI mrežnog modela. Osim pogonskog programa, *WinPcap* sadrži i biblioteku funkcija (*eng. library*) koje služe za pristupanje tim slojevima. Ovo ostvarenje sustava za otkrivanje napada koristi upravo tu biblioteku funkcija za hvatanje paketa.

WinPcap (i njegovu originalnu varijantu *libpcap*) zbog svoje brzine, optimiranih pogonskih programa i biblioteke funkcija, lakoće korištenja, pouzdanosti i činjenice da je besplatan koristi mnogo programskih paketa kao osnovu za svoj rad, bili oni komercijalni ili otvorenog izvornog teksta. Najpoznatiji primjeri su *Wireshark* (sustav za hvatanje i analizu mrežnog prometa) i već spomenuti *Snort*.

Instalacija *WinPcap* programskog paketa je vrlo jednostavna i svodi se na preuzimanje izvršne datoteke koja sadrži instalaciju paketa i njezino pokretanje. Instalacijska procedura sama instalira pogonski program koji se odmah nakon instalacije i eventualnog ponovnog pokretanja računala može koristiti u aplikacijama.

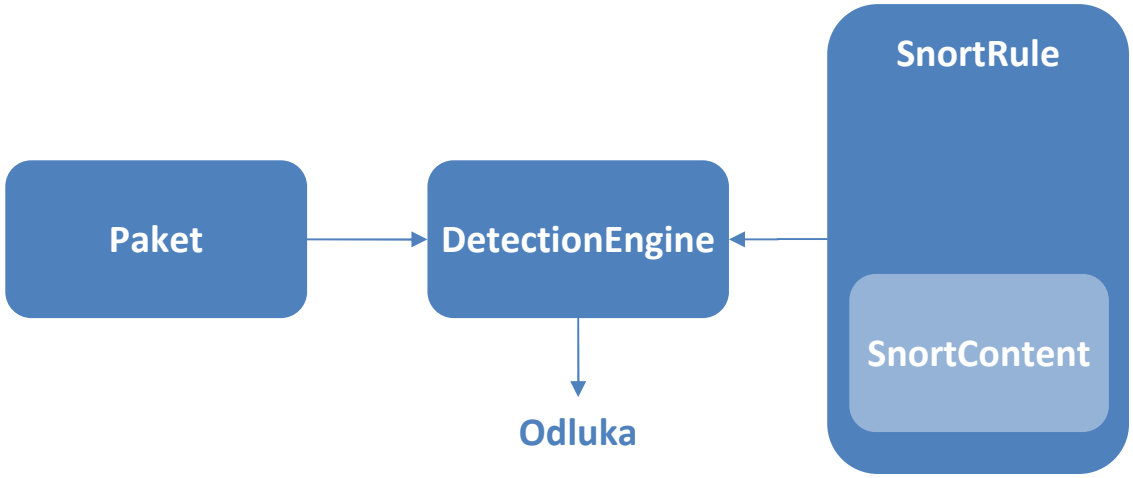
6.1.3. *SharpPcap*

Kako je *WinPcap* napisan u programskom jeziku *C++*, a ostvarenje ovog jednostavnog sustava za otkrivanje napada je napravljeno u programskom jeziku *C#*, radi lakoće korištenja *WinPcap* biblioteke funkcija korištena je posebna biblioteka funkcija koja funkcionira kao svojevrsno sučelje u programskom jeziku *C#* prema *WinPcap* biblioteci funkcija. Ta posebna biblioteka funkcija se zove *SharpPcap* i slobodno je dostupna na internetskim stranicama proizvođača. *SharpPcap* podržava većinu funkcija koje podržava i *WinPcap*, tako da aplikacije koje koriste ovu biblioteku umjesto izravnog korištenja *WinPcap* biblioteke funkcija nisu zakinute manjkom funkcionalnosti.

6.2. Model ostvarenog sustava

Mehanizam otkrivanja napada sastoji se od jedne klase nazvane `DetectionEngine` koja ima jednu javnu metodu `ProcessPacket` koja se poziva u trenutku dolaska paketa na mrežno sučelje na kojem se hvataju paketi. Parametri koji se prenose toj metodi su objektni model paketa koji je uhvaćen (objektni model paketa je implementiran u već spomenutoj biblioteci funkcija *SharpPcap*), popis pravila po kojima se obavlja otkrivanje te mrežni protokol na kojem se obavlja otkrivanje (TCP, UDP, IP ili ICMP). Osim te jedne javne metode, interna implementacija `DetectionEngine` klase sadrži i niz metoda koje služe za procesiranje paketa i prepoznavanje određenog sadržaja definiranog pravilima.

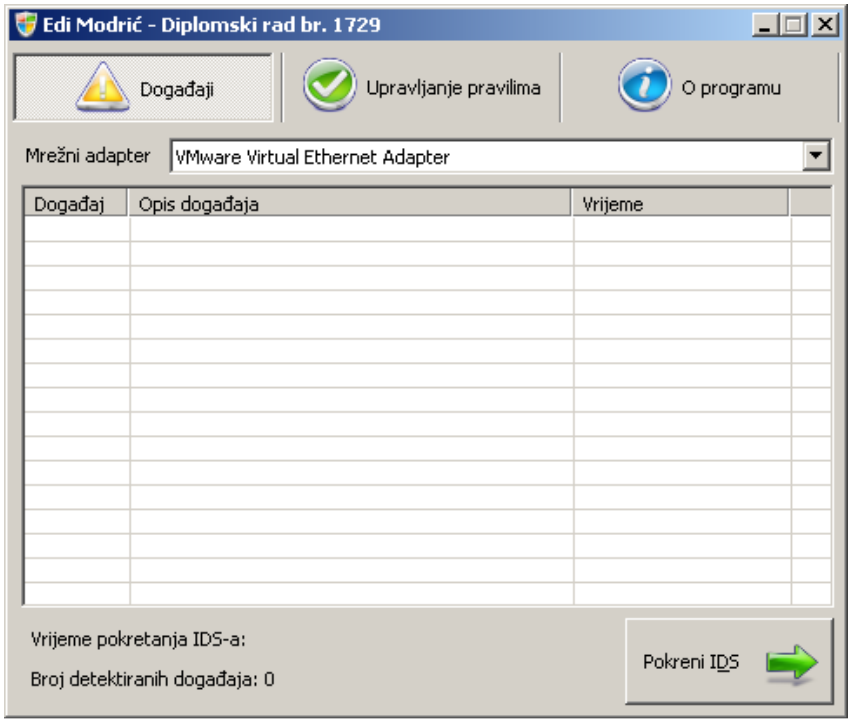
Kao sustav pravila je korištena sintaksa pravila iz javno dostupnog programskog paketa za otkrivanje napada *Snort*. Izgled i mogućnosti tih pravila su već opisane u poglavlju o *Snortu*. Implementiran je jednostavni parser koji čita pravila iz datoteke i stvara od njih objektni model za što lakšu komunikaciju sa klasom `DetectionEngine`. Za svako pravilo se stvara taj objektni model koji se sastoji od nekoliko klasa. Tu je glavna klasa `SnortRule` koja opisuje zaglavlje *Snort* pravila te `SnortContent` klasa koja opisuje tijelo *Snort* pravila. `SnortRule` i `SnortContent` klase su u takvom odnosu da svaki objekt tipa `SnortRule` sadrži objekt tipa `SnortContent`. Osim te dvije klase, objektni model pravila sadrži i četiri klase koje su u biti enumeracije te pobliže opisuju neke stvari u *Snort* pravilu. To su klase `ComparisonOperator`, `Direction`, `Protocol` i `RuleAction`. Enumeracija `ComparisonOperator` se od svih enumeracija najviše koristi jer govori kako će se vrijednosti opcija navedenih u pravilima uspoređivati s vrijednostima tih istih opcija dobivenih iz paketa (hoće li sustav za otkrivanje napada prijaviti sumnjivi paket ako je neka od opcija u paketu veća, manja ili jednaka referentnoj vrijednosti te opcije u pravilu). Enumeracija `Protocol` služi za definiciju mrežnog protokola na kojem će *Snort* pravilo biti aktivno. Enumeracija `Direction` govori u kojem će se smjeru prometa odvijati otkrivanje dok `RuleAction` enumeracija govori što će sustav za otkrivanje napada učiniti kad naiđe na paket koji odgovara nekom od pravila. Slika 6.1 prikazuje model ostvarenog sustava za otkrivanje napada.



Slika 6.1. Model ostvarenog sustava za otkrivanje napada

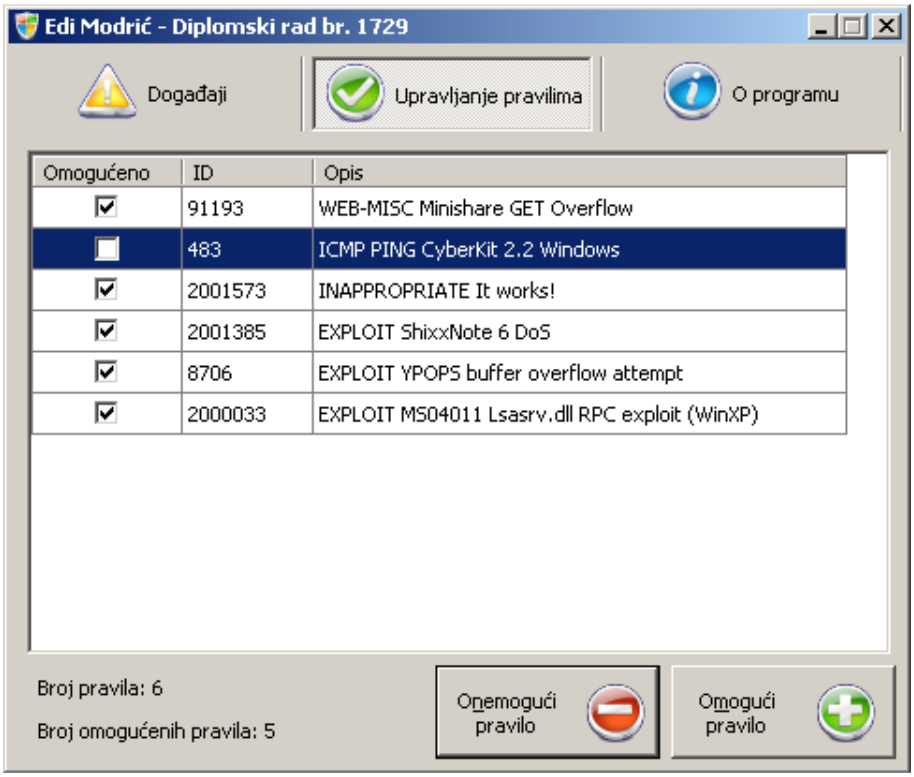
6.3. Korištenje ostvarenog sustava

Sustav se sastoji od dva jednostavna ekrana. Prilikom pokretanja, korisniku se prikazuje početni ekran. Na tom početnom ekranu se nalaze osnovne informacije koje mehanizam otkrivanja napada prosljeđuje sučelju tijekom svog rada. Za pokretanje procesa otkrivanja napada je dovoljno u listi na vrhu forme odabrati mrežni adapter na kojem će se hvatati paketi i pritisnuti na gumb "Pokreni IDS". Tijekom rada će se u sučelju prikazivati osnovne informacije o pronađenim sumnjivim paketima: identifikacijski broj pravila koji je zaslužan za otkrivanje paketa, tekstualni opis i vrijeme pojavljivanja paketa na mrežnom sučelju. Slika 6.2 prikazuje izgled početnog ekrana ostvarenog sustava.



Slika 6.2. Izgled početnog ekrana ostvarenog sustava

Drugi ekran programa je "Upravljanje pravilima" koji sadrži popis svih pravila učitanih iz datoteke i njihovih pripadnih detalja (identifikacijski broj pravila i tekstualni opis pravila). Za ta pravila su dostupne dvije opcije: privremeno onemogućavanje i omogućavanje pravila. Pravila se mogu omogućiti i onemogućiti samo za vrijeme kad mehanizam za otkrivanje napada nije aktivan. Slika 6.3 prikazuje izgled tog ekrana.



Slika 6.3. Izgled ekrana za upravljanje pravilima u ostvarenom sustavu

Da bi se mehanizam za otkrivanje napada mogao uspješno koristiti, potrebno je stvoriti nekoliko objekata koji su potrebni za rad mehanizma. Klase iz kojih se ti objekti stvaraju su definirane u već spomenutoj *SharpPcap* biblioteci funkcija. Prilikom inicijalnog pokretanja programa izvršava se sljedeći izvorni tekst:

```
PcapDeviceList devices = null;
devices = SharpPcap.GetAllDevices();
```

Slika 6.4. Isječak izvornog teksta koji dohvaća aktivna mrežna sučelja

Ovaj izvorni tekst dohvaća sva postojeća mrežna sučelja na računalu i njih prikazuje na glavnom ekranu. Prilikom odabira mrežnog sučelja na kojem će se obavljati otkrivanje napada izvršava se sljedeći izvorni tekst:

```
PcapDevice device = null;
device = devices[comboBoxAdapters.SelectedIndex];
device.PcapOnPacketArrival += new
       SharpPcap.PacketArrivalEvent(device_PcapOnPacketArrival);
device.PcapOpen(true, 1000);
```

Slika 6.5. Isječak izvornog teksta koji omogućava hvatanje paketa na mrežnom sučelju

Prvo se stvara novi objekt tipa `PcapDevice` koji predstavlja mrežno sučelje na računalu. Zatim se tom objektu pridjeljuje sučelje odabrano iz liste sučelja. Nakon toga se tom sučelju registrira metoda za upravljanje događajem (*eng. event handler*) koja se pokreće svaki put kad se na mrežnom sučelju pojavi paket. U zadnjoj liniji izvornog teksta, mrežno sučelje se proglašava spremnim za korištenje pozivom metode `PcapOpen` sa dva parametra. Prvi parametar označava način na koji će se obavljati hvatanje paketa (normalno ili promiskuitetno), dok drugi parametar zadaje veličinu privremene memorije koja služi za smještaj paketa koji čekaju na obradu. Prilikom pokretanja mehanizma otkrivanja napada se izvršava sljedeći izvorni tekst:

```
device.PcapStartCapture();
```

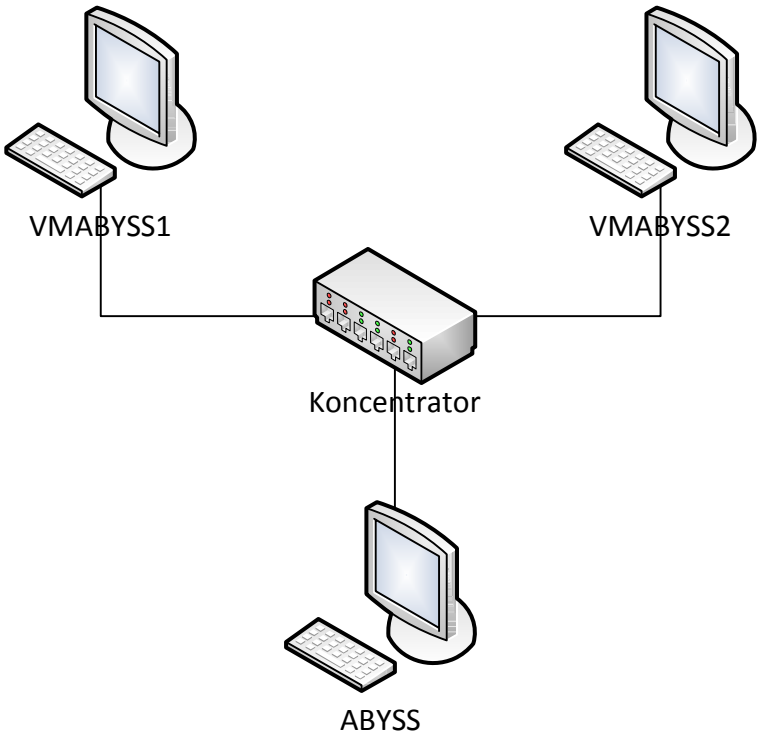
Slika 6.6. Isječak izvornog teksta koji pokreće mehanizam hvatanja paketa

Nakon pokretanja mehanizma za hvatanje paketa, ostvareni sustav za svaki uhvaćeni paket pokreće implementirani mehanizam za otkrivanje napada i prijavljuje pakete koji odgovaraju nekom od pravila definiranih u ulaznoj datoteci.

6.4. Konfiguracija ispitne mreže

Za ispitivanje rada programa je korišten programski paket *VMware Workstation* koji je namijenjen za rad s virtualnim strojevima. Unutar tog programskog paketa su stvorena dva virtualna stroja i na njih je instaliran operacijski sustav *Windows XP Professional* sa prvim servisnim paketom zakrpi (*eng. SP1 - service pack 1*). Ta dva stroja su povezana u mrežu preko koncentratora (*eng. hub*). Njima je u mrežu dodano i lokalno računalo na kojem su pokrenuti virtualni strojevi i na kojem je instaliran operacijski sustav *Windows XP Professional* sa drugim servisnim paketom zakrpi. Slika 6.7 prikazuje konačnu konfiguraciju ispitne mreže.

Računalo s nazivom ABYSS predstavlja lokalno računalo. Na tom računalu je pokrenut ostvareni sustav za otkrivanje napada. Na virtualnom računalu VMABYSS1 su pokretani napadi koji su iskorištavali ranjive programe i operacijski sustav na virtualnom računalu VMABYSS2. Upravo zbog činjenice da je mreža zasnovana na koncentratoru (*eng. hub*), a ne na preklopniku (*eng. switch*), je omogućila ostvarenom sustavu koji je pokrenut na lokalnom računalu da sluša promet između dva virtualna računala.



Slika 6.7. Konfiguracija virtualne mreže prilikom ispitivanja ostvarenog sustava

6.5. Korišteni napadi

6.5.1. Napad na aplikaciju *MiniShare 1.4.1*

MiniShare je mala i besplatna aplikacija koja služi za razmjenu datoteka HTTP protokolom. U verzijama 1.4.1 i nižima je otkriven sigurnosni propust koji udaljenom napadaču omogućava izvršavanje proizvoljnog programa na računalu na kojem je *MiniShare* instaliran. Sigurnosni propust se može iskoristiti slanjem dugačkog i posebno formatiranog zahtjeva HTTP GET metodom, što uzrokuje preljev spremnika.

Za napad je iskorištena aplikacija koja je napravljena u sklopu diplomskog rada br. 1703 na *Fakultetu elektrotehnike i računarstva* u Zagrebu, na *Zavodu za elektroniku, mikroelektroniku, računalne i inteligentne sustave*. Pravilo pomoću kojeg se napad otkriva u ostvarenom sustavu je prikazano ispod.

```
alert tcp any any -> $HOME_NET 80 (msg:"WEB-MISC Minishare GET Overflow";
flow:established,to_server; content:"GET "; within:4; content:!"/"; distance:0;
within:200; reference:cve,2004-2271; reference:bugtraq,11620; classtype:web-
application-attack; sid:91193; rev:1; )
```

Slika 6.8. *Snort* pravilo za otkrivanje napada na aplikaciju *MiniShare*

Napad se provodi tako da se u aplikaciji iz diplomskog rada br. 1703 u padajućem meniju odabere vrsta napada *MiniShare*, u za to namijenjeno polje upiše IP adresa računala na kojem se nalazi instalirana i pokrenuta aplikacija *MiniShare* te odabere opcija "Pokreni napad". U tom trenutku se udaljenom računalu šalje posebno kreirani HTTP GET zahtjev, koji ima za posljedicu pokretanje programa namijenjenog za dobivanje udaljenog pristupa

na računalo preko *Telnet* protokola otvaranjem pristupne točke na računalu. Spajanjem na tu pristupnu točku *Telnet* protokolom napadač dobiva administratorske ovlasti na napadnutom računalu te ima potpunu kontrolu.

Ovaj napad je uspješno otkriven implementiranim sustavom.

6.5.2. Napad na aplikaciju *YahooPOPs! 0.6*

YahooPOPs! je besplatna aplikacija koja omogućava korištenje POP3 i SMTP protokola za čitanje elektroničke pošte na *Yahoo! Mail* internetskoj usluzi. Kao i kod aplikacije *MiniShare*, sigurnosni propust u verziji 0.6 ove aplikacije se iskorištava slanjem posebno formiranog i dugog niza okteta i to na pristupnu točku 25 na udaljenom računalu što uzrokuje preljev spremnika u aplikaciji.

Za napad je iskorišten na Internetu javno dostupan izvorni tekst napisan u programskom jeziku *C*. Njegovim izvršavanjem i upisivanjem IP adrese udaljenog računala se tom računalu šalje potreban niz okteta. U tom trenutku se u aplikaciji dešava preljev spremnika koji je, kao i kod aplikacije *MiniShare*, iskorišten za dobivanje pristupa udaljenom računalu sa administratorskim ovlastima preko *Telnet* protokola. *Snort* pravilo koje otkriva ovaj napad je prikazano ispod.

```
alert tcp any any -> $HOME_NET 25 (msg:"EXPLOIT YPOPS buffer overflow attempt";
flow:established,to_server; content:"|EB 06 92|F"; content:"|EB 08 46 92 EB 08
46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92
EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08
46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92
EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08
46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92
EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08
46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92 EB 08 46 92
EB 08 46 92 EB 08 46 92|"; reference:bugtraq,11256; reference:cve,2004-1558;
classtype:attempted-admin; sid:8706; rev:2;)
```

Slika 6.9. *Snort* pravilo za otkrivanje napada na aplikaciju *YahooPOPs!*

Ovaj napad isprva nije bio otkriven ostvarenim sustavom jer je originalno pravilo sadržavalo regularni izraz koji je tražio niz okteta 0xEB, 0x08, 0x46, 0x92 ponavljan 49 puta. Kako je navedeno u poglavlju 6.6, ostvareni sustav ne podržava pretraživanje okteta pomoću regularnih izraza, pa je pravilo modificirano na način da se regularni izraz zamijeni blokom od 196 okteta koji odgovara regularnom izrazu. Korištenjem tako modificiranog pravila sustav je otkrio napad.

6.5.3. Napad na aplikaciju *ShixxNOTE 6.NET*

ShixxNOTE 6.NET je komercijalna aplikacija koja služi za definiciju i prikazivanje korisnički definiranog teksta i poruka na radnoj površini operacijskog sustava *Windows*. U ovoj verziji je otkriven sigurnosni propust koji iskorištavanjem pomoću napada preljevom spremnika napadaču omogućava izvršavanje proizvoljnog programa.

Za napad je korišten slobodno dostupan izvorni tekst preuzet s Interneta koji napadaču omogućuje udaljeni pristup pomoću *Telnet* protokola uz administratorske ovlasti. *Snort* pravilo koje detektira ovaj napad je prikazano ispod.

```
alert tcp any any -> $HOME_NET any (msg:"EXPLOIT ShixxNote 6 DoS";
flow:established,to_server; content:"|68 61 63 6b 75|"; offset:126; depth:5;
content:"|68 61 63 6b 90 61 61 61 61|"; offset:519; depth:9;
reference:url,aluigi.altervista.org/adv/shixxbof-adv.txt; classtype:shellcode-
detect; sid:2001385; rev:4; )
```

Slika 6.10. *Snort* pravilo za otkrivanje napada na aplikaciju *ShixxNOTE 6.NET*

Ostvareni sustav je pomoću navedenog pravila bez problema otkrio pokušaj izvršavanja ovog napada.

6.5.4. *Ping* udaljenog računala pomoću aplikacije *CyberKit 2.5*

Neki računalni sustavi mogu biti podešeni da svaki pokušaj otkrivanja prisutnosti nekog računala (npr. pomoću naredbe *ping* u operacijskom sustavu) smatraju nedopuštenim. Sustavi za otkrivanje napada tako moraju biti u mogućnosti prijaviti i takvu vrstu mrežne aktivnosti.

Za ovaj napad je korištena besplatna aplikacija *CyberKit* koja među ostalim, ima i mogućnost slanja *ping* zahtjeva preko mreže. *Snort* pravilo koje otkriva korištenje ove aplikacije je napisano tako da otkriva prisutnost *ping* zahtjeva generiranih ovom aplikacijom pomoću niza okteta koji se automatski šalju korištenjem aplikacije i preko kojih je moguće identificirati aplikaciju. Pravilo je prikazano ispod.

```
alert icmp any any -> $HOME_NET any (msg:"ICMP PING CyberKit 2.2 Windows";
itype:8; content:"|AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA|"; depth:32;
reference:arachnids,154; classtype:misc-activity; sid:483; rev:6;)
```

Slika 6.11. *Snort* pravilo za otkrivanje mrežnih aktivnosti aplikacije *CyberKit*

Ostvareni sustav je bez problema otkrio i ovu vrstu napada.

6.5.5. Simulacija otkrivanja nepoćudnog sadržaja

Kako je jedna od namjena bilo kojeg sustava za otkrivanje napada i izvještavanje o kršenju politika definiranih od strane institucije u kojoj se sustav koristi, ostvareni sustav za otkrivanje napada je ispitan jednostavnom simulacijom koja pokazuje način na koji se sigurnosne politike mogu provoditi.

Odabrana je simulacija otkrivanja nepoćudnog sadržaja kojeg korisnici računalnog sustava preuzimaju s Interneta, a u sigurnosnoj politici institucije je označen kao zabranjen. Za simulaciju je korišten HTTP server *Apache* koji nakon uspješne instalacije, uz korištenje tvornički isporučene konfiguracije, te primanja HTTP zahtjeva za početnim direktorijem od strane Internet pretraživača prikazuje korisniku jednostavnu HTML stranicu na kojoj piše "It works!". Ostvareni sustav koristi pravilo koje detektira niz okteta koji odgovara tekstu na toj HTML stranici. Izgled pravila je prikazan ispod.

```
alert tcp any 80 -> $HOME_NET any (msg:"INAPPROPRIATE It works!"; flow:
from_server,established; content:"It works!"; nocase; classtype: policy-
violation; sid: 2001573; rev:1; )
```

Slika 6.12. *Snort* pravilo za otkrivanje nepoćudnog sadržaja

Ova simulacija napada je također uspješno otkrivena ostvarenim sustavom.

6.5.6. Napad na *Windows* uslugu za primjenu sigurnosnih politika

Za ovaj napad je iskorišten sigurnosni propust u jednoj od usluga u operacijskom sustavu *Windows XP Professional*. Naziv usluge je *Local Security Authority Subsystem Service* i zadužena je za primjenu pravila definiranih u lokalnim sigurnosnim politikama operacijskog sustava *Windows*.

Sigurnosni propust u ovoj usluzi izaziva preljev spremnika prilikom slanja posebno formiranog niza okteta udaljenom računalu i omogućava izvršavanje proizvoljnog programa. Za napad je korišten slobodno dostupan izvorni tekst preuzet s Interneta koji pokretanjem omogućava pristup napadnutom računalu preko *Telnet* protokola uz administratorske ovlasti. *Snort* pravilo korišteno za otkrivanje ovog napada je prikazano ispod.

```
alert tcp any any -> any 445 (msg:"EXPLOIT MS04011 Lsasrv.dll RPC exploit
(WinXP)"; flow:to_server,established; content:"|95 14 40 00 03 00 00 00 7C 70
40 00 01|"; content:"|78 85 13 00 AB 5B A6 E9 31 31|";
reference:url,http://www.microsoft.com/technet/security/bulletin/MS04-011.mspx;
classtype:misc-activity; sid:2000033; rev:5;)
```

Slika 6.13. *Snort* pravilo za otkrivanje napada na uslugu za primjenu sigurnosnih politika

Ostvareni sustav je i ovaj napad uspješno prepoznao.

6.6. Nedostaci implementiranog sustava

Svi nedostaci implementiranog sustava su posljedica velike složenosti pravila za otkrivanje napada u programskom paketu *Snort*. Kako je cilj implementiranog sustava bio na što jednostavniji način pokazati kako je moguće obaviti otkrivanje napada, nisu implementirane sve mogućnosti pravila u programskom paketu *Snort*. To se prvenstveno odnosi na mogućnosti pravila da u podatkovnom dijelu paketa traži uzorke niza tekstualnih znakova i okteta pomoću regularnih izraza. Uz to, implementiranom sustavu nedostaje i predprocesor kojem je uloga da, kako je već rečeno u prethodnim poglavljima rada, obavlja sastavljanje (*eng. defragmentation*) paketa koji su preko mrežnih sučelja poslani u dijelovima i da obavlja dekodiranje znakova u uniformnom identifikatoru resursa (*eng. URI - Uniform Resource Identifier*). Posljedica nedostatka pretprocesora je da implementirani sustav ne može pratiti stanje TCP veze pa je pregledavanje paketa ograničeno na pojedinačne pakete. Uz to, analiza paketa na višim slojevima OSI mrežnog je također vrlo ograničena činjenicom da pretprocesor nije implementiran.

7. Zaključak

Ovaj diplomski rad je prikazao implementaciju jednostavnog sustava za otkrivanje napada. Naglasak pri implementaciji sustava je stavljen na izradu središnjeg modula sustava za otkrivanje napada, mehanizam otkrivanja napada. Ostali moduli nisu implementirani zbog svoje složenosti koja prelazi vremenske okvire za izradu rada. Implementirani modul se pokazao ispravnim i spremnim za korištenje upotrebom jednostavnijih pravila iz programskog paketa *Snort*.

Kako uvijek ima mjesta za napredak implementiranog sustava, mogućnosti daljnjeg razvoja su praktički beskonačne. Od važnijih stvari koje bi trebalo implementirati i usavršiti su mogućnosti koje nedostaju implementiranom sustavu, a spomenute su kao nedostaci implementiranog sustava. To se odnosi na implementaciju pretprocesora i podrške za regularne izraze, čime bi implementirani sustav postao dosta moćniji alat za otkrivanje napada. Implementacijom pretprocesora sustav bi se mogao promatrati i kao punokrvni mrežni sustav za otkrivanje napada. Za to je, kako je već spomenuto u radu, potrebna određena sklopovska podrška. Naime, sustav se pokazao sposobnim pratiti promet na mreži zasnovanoj na koncentratoru, tako da nema razloga da sustav ne bude sposoban pratiti mrežni promet i na mreži zasnovanoj na preklopnicima, uz već spomenutu sklopovsku podršku.

Konačno, nakon usavršavanja mehanizma za otkrivanje napada, implementirani sustav bi se mogao nadograditi da funkcionira i kao sustav za sprječavanje napada. Za to su potrebna određena znanja o programiranju pogonskih programa za *Windows* operacijski sustav, koja autoru rada u ovom trenutku nedostaju.

8. Literatura

1. An overview of computational science, dostupno na Internet adresi http://www.phy.ornl.gov/csep/ov/ov.html
2. Wikipedia.org: Computer security, dostupno na Internet adresi http://en.wikipedia.org/wiki/Computer_security
3. Klarić, Ivan: Načini zlouporabe ranjivosti računalnog sustava, Seminar, Fakultet elektrotehnike i računarstva, Zagreb, 2007. http://os2.zemris.fer.hr/ns/malware/2007_klaric/
4. Wikipedia.org: Buffer overflow, dostupno na Internet adresi http://en.wikipedia.org/wiki/Buffer_overflow
5. Antončić, Vlatka: Napadi s uskraćivanjem usluge (DoS napadi), Seminar, Fakultet elektrotehnike i računarstva, Zagreb, 2007. http://os2.zemris.fer.hr/ns/2007_Antoncic/
6. Wikipedia.org: Denial of service attack, dostupno na Internet adresi http://en.wikipedia.org/wiki/Denial-of-service_attack
7. Noonan, Wes; Dubrawsky, Ido: Firewall Fundamentals, Cisco Press, 2006.
8. Wikipedia.org: Social engineering, dostupno na Internet adresi http://en.wikipedia.org/wiki/Social_engineering_(security)
9. Wikipedia.org: Antivirus, dostupno na Internet adresi http://en.wikipedia.org/wiki/Antivirus
10. Wikipedia.org: Heuristic analysis, dostupno na Internet adresi http://en.wikipedia.org/wiki/Heuristic_analysis
11. Sullivan, Chad; Mauvais, Paul; Asher, Jeff: Advanced Host Intrusion Prevention with CSA, Cisco Press, 2006.
12. Wikipedia.org: Intrusion prevention, dostupno na Internet adresi http://en.wikipedia.org/wiki/Intrusion_prevention
13. Wikipedia.org: Intrusion detection, dostupno na Internet adresi http://en.wikipedia.org/wiki/Intrusion_detection
14. Wikipedia.org: Intrusion detection system, dostupno na Internet adresi http://en.wikipedia.org/wiki/Intrusion_detection_system
15. Snort IDS Homepage http://www.snort.org
16. Internet Security Systems - Product Documentation, dostupno na Internet adresi http://www.iss.net/support/documentation/
17. Požgaj, Igor: Raspodijeljeni sustav za sprječavanje mrežnih napada, Diplomski rad br. 1708, Fakultet elektrotehnike i računarstva, Zagreb, 2008
18. Ravnić, Dino: Otkrivanje i zaštita od napada u mrežnim sigurnosnim sustavima, Diplomski rad br. 1703, Fakultet elektrotehnike i računarstva, Zagreb, 2008.