KANDIDATUPPSATS

Nätverkssäkerhet med IPS

Författare
Michael Dubell & David Johansson

IT-forensik och Informationssäkerhet
Nätverkssäkerhet

Halmstad 2013-06

Nätverkssäkerhet med IPS

Förbättrad nätverkssäkerhet med Intrusion Prevention Systems

Kandidatuppsats

2013 06

Författare: Michael Dubell & David Johansson

Handledare: Olga Torstensson

Examinator: Urban Bilstrup

Sektionen för informationsvetenskap, data- och elektroteknik

Högskolan i Halmstad

Box 823, 301 18 HALMSTAD

Kandidatuppsats
Sektionen för informationsvetenskap, data- och elektroteknik
Högskolan i Halmstad

Förord

Till att börja med vill vi tacka Olga Torstensson för bra tips och god handledning genom arbetets gång. Vi vill även tacka vänner och familj för deras stöd och uppmuntran. Ett sista tack till Eson Pac AB som skänkte oss två stycken datorer vilket möjliggjorde uppbyggnad av en bra laborationsmiljö för att genomföra våra experiment!

Abstrakt

Att skydda sin IT-miljö mot olika typer av intrång och attacker som till exempel trojaner, skadliga Java applets eller DoS attacker med hjälp av brandväggar och antivirusprogram är två viktiga lager i skalskyddet.

I den här uppsatsen undersöks hur väl ett *Intrusion Prevention System* skulle kunna fungera som ett ytterligare lager i skalskyddet. Fokus ligger på hur väl IPS-systemet klarar av att avvärja attacker, hur mycket tid som går åt till konfigurering och drift för att få ett fungerande IPS samt hur prestandan i nätverket påverkas av implementationen. För att mäta hur väl IPS systemet klarar av att upptäcka och blockera attacker utförs två experiment där ett mindre nätverk attackeras på olika sätt. I det första experimentet skyddas infrastrukturen av en brandvägg och klienterna är utrustade med antivirusprogram. I det andra experimentet genomförs samma attacker igen fast med ett Snort IPS implementerat i nätverket.

Resultatet av de genomförda experimenten visar att en IPS klarar att blockera ca 87% av attackerna, men nätverksprestandan påverkas negativt. Slutsatsen är att endast brandväggar och antivirusprogram inte ger ett fullgott skydd.

Nyckelord: IPS, Intrusion Prevention System, Snort, Nätverkssäkerhet, Attacker

Innehållsförteckning

Figurförteckning

Tabellförteckning

Nätverkssäkerhet med IPS

1. Introduktion

Idag hanteras stora mängder känslig information i våra datorer och i våra nätverk, detta gäller information som till exempel personuppgifter, bankärenden och företagshemligheter. Detta är värdefull information som i fel händer kan utgöra stor skada för både privatpersoner och företag. Därför måste man med olika hjälpmedel försöka skydda denna känsliga information.

För att skydda datorer och nätverk från informationsläckage används traditionellt brandväggar och antivirusprogram. Brandväggar blockerar otillåten trafik till och från internet. Man specificerar till exempel portar, protokoll eller IP-adresser som bör tillåtas eller blockeras. Antivirusprogram arbetar lokalt på datorer och servrar för att lokalisera och oskadliggöra känd skadlig programvara som försöker infektera ändnoderna i nätverket. Frågan är om det räcker med det här skyddet?

Attacker som till exempel Nmap-skanning, brute force försök eller specialdesignade trojaner upptäcks eller blockeras normalt inte av en brandvägg eller av ett antivirusprogram[1]. Enligt en studie från Symantec[2] uppgick kostnaderna för de som utsatts för cyberbrott till ca 110 Miljarder dollar under 2012. En IPS(*Intrusion Prevention System*) som det ofta förkortas används idag av företag vilka har en IT-avdelning med tillräcklig kompetens och tid för att dra nytta av att använda ett IPS för att övervaka och analysera nätverkstrafik i syfte att förebygga attacker mot infrastrukturen. Nätverkssäkerheten borde inte begränsas på grund av budget eller företagsstorlek då informationen som lagras bör skyddas lika väl hos alla företag. Därför har vi valt att undersöka open-source alternativet Snort som IPS.

1.1 Mål

Vi vill i den här uppsatsen undersöka hur mycket nätverkssäkerheten skulle kunna förbättras genom att komplettera ett befintligt skydd bestående av brandvägg och antivirusprogram med ett IPS. Vi vill även undersöka hur mycket tid som behöver läggas på konfigurering och drift samt hur mycket ett IPS påverkar nätverkets prestanda.

1.2 Begränsningar

I våra experiment finns begränsningar i form av hur många datorer som används och den trafik de genererar, därav kan inte slutsatser dras för hur IPS-systemet hade fungerat i en riktig produktionsmiljö. Topologin ska efterlikna ett litet företagsnätverk men eftersom det är en laborationsmiljö med begränsade tillgångar så består den endast av två klienter och två servrar. I övrigt är även prestandan hos den server vi kör Snort på begränsad då hårdvaran inte är den nyaste men tillräcklig enligt de specifikationer som finns i hårdvarukraven för Snort.

Vi har även valt att begränsa attackerna till riktiga attacker som går att använda mot nyuppdaterade system och några automatiserade syntetiska testattacker används inte.

1.3 Etiska aspekter

Vi är väl medvetna om att de attacker som vi använt och beskrivit i våra experiment kan användas för olagliga syften. Men för att kunna förstå hur man kan förbättra

nätverkssäkerhet måste man känna till hur man kan kringgå säkerhetsimplementationer för att få en bättre förståelse om både tekniken samt hur man kan förbättra säkerhetsimplementationer.

För att kunna genomföra experimenten och visa hur viktigt det är att använda sig av IPS-system och inte bara brandväggar och antivirusprogram ansåg vi att information om hur attackerna genomfördes och vad de gav för resulterade var tvungna att publiceras.

2. Bakgrund

Till att börja med så finns det i huvudsak två olika intrångs detekterings system, det första är det system som vi fokuserar på i den här uppsatsen, Intrusion Prevention System. IPS är ett aktivt skydd som i realtid analyserar trafiken som passerar genom systemet och avgör om den ska tillåtas eller blockeras. Detta påverkar prestandan i nätverket då trafiken måste analyseras av IPS-systemet för att tillåtas passera. När ett hot upptäcks blockeras trafiken och administratören informeras om händelsen[1].

Det andra systemet kallas för IDS(*Intrusion Detection System*), detta är ett passivt skydd vilket man ofta speglar trafik från switchar till för att analysera trafik, trafiken behöver alltså inte passera systemet för att komma vidare i nätverket. IDS systemet kan även vara placerat som ett IPS där all trafik passerar IDS systemet och analyseras, eftersom det är ett passivt skydd påverkas inte nätverkets användbarhet på samma sätt som när ett aktivt IPS används. IDS system skickar sedan larm och rapporter till administratören när hot upptäcks i den analyserade trafiken men det saknar funktionen att aktivt blockera de upptäckta hoten[1].

2.1 Tidigare forskning

Det har under åren undersökts hur man utvärderar funktionaliteten hos IDS där man tittar på hur träffsäkra systemen är i detektering av olika typer av attacker. Till exempel har Amerikanska Department of Defense underorganisation inom avancerad forskning DARPA tagit fram en strategi med olika attacker för utvärdering av IDS, den publicerades 1999 och har senare undersökts och kritiserats av en grupp Indiska forskare[3] där den anses vara en bra grundläggande metod men utdaterad och behöver förnyas. År 2007 publicerades en artikel av Wei L.[4] där han tar upp en existerande utvärderingsstrategi av IDS och jämför mot hans egna föreslagna strategi, detta för att bättre kunna testa och utvärdera ett IDS så att man ska kunna förbättra funktionaliteten hos systemen och vara beredd för framtida attacker. Ett arbete liknade vårt eget gjordes 2007 av Markus Ringström Saltin[5] där han gjorde en utvärdering av Snort som IDS med resultat som visar att Snort är väl kapabelt att upptäcka de flesta av de olika attacker som testades.

2.2 Så fungerar ett IPS

Det finns idag flera olika IPS på marknaden som till exempel Ciscos olika IPS produkter[6], Suricata[7] eller Snort vilket är den mjukvaran som användes i den här uppsatsen. Huvudsyftet med ett nätverksbaserat Intrusion Prevention system är att analysera nätverkstrafik efter aktivitet som bryter mot de policys eller regler som används och sedan varna administratören och blockera trafiken [1]. Mer information om hur Snort fungerar finns i experimentdelen.

I huvudsak använder sig ett IPS av tre olika tekniker för att identifiera hot:

- Signaturbaserad
  Nätverkspaket analyseras efter signaturer för kända hot. Alltså finns samma svaghet som hos antivirusskydd, man måste känna till signaturen i förväg annars upptäcks inte hotet.
- Statistiskt avvikande trafik
  Systemen som använder den här funktionen ”tränas” först under en tid för att veta hur normal trafikbelastning av olika protokoll ser ut och hur bandbredden används under olika delar av dygnet, detta för att sedan kunna upptäcka avvikande mönster, skicka larm och blockera.
- Logg analys
  Analys av loggar från lokala enheter som brandväggar, switchar eller datorer.

Man ska även ha i baktanke att en IPS inte är tänkt att ersätta andra skyddsmekanismer som brandväggar och antivirusprogram utan är ett komplement för att få ett ytterligare lager i skyddskalet[8].

2.3 Utveckling

Under senare delen av 1970-talet började man fundera på hur man skulle kunna identifiera otillåten åtkomst av filer och datorsystem. På 1980-talet publicerades studier om hur man skulle upptäcka missbruk av datorsystem. Mellan 1984-1986 utvecklades de första IDS-systemen av Denning D. och Neuman P. detta kom att bli starten för utvecklingen av IDS-system. Systemen har under senare tid till att få nya metoder för att känna igen mönster med hjälp av statistik och inte bara på signaturer. En utveckling av IDS kom under 1990 talet och från att till en början varit en passiv övervakningsstation gått till att aktivt kunna blockera hot och det är detta som kallas IPS.[9]

3. Metod

För att mäta hur mycket en IPS skulle förbättra säkerheten i ett nätverk används en kvantitativ forskningsmetod för att statistiskt jämföra två experiment. Dessa experiment utförs i en isolerad labbmiljö, detta för att ha full kontroll på nätverket så att ingen yttre påverkan på något sätt skulle kunna påverka slutresultatet. Uppsatsen avser även att ge svar på hur mycket prestandan i nätverket påverkas samt hur mycket tid som behöver läggas på konfigurering och drift av IPS-systemet för att få ut bästa funktionalitet. Resultatet mäts genom att notera antalet lyckade attacker/intrång före och efter implementering av en IPS samt hur systemet påverkar prestanda och användande av nätverket.

För att ge en bakgrund till utförandet av experimenten som ska kunna svara på frågeställningarna angående IPS som komplement i skalskyddet så har Lei Wei's[4] artikel om utvärdering av IDS-system studerats. Den ger en bra inblick i hur en utvärdering går till och vad man bör ta hänsyn till för olika parametrar, dock fokuserar inte den här uppsatsen på att utvärdera förmågan hos IDS-systemet i sig utan hur mycket bättre säkerheten skulle bli med än utan en IPS.

I experimenten används SNORT[10] som IPS, Snort är ett projekt som bygger på öppen källkod och enligt deras hemsida är det ett av de populäraste IDS/IPS-systemen i världen.

3.1 Varför Snort IPS?

Anledningen till att det är just Snort som används som IPS i de utförda experimenten är för att det är fritt tillgängligt och skrivet med öppen källkod. Det har funnits på marknaden sedan år 1998 och därav finns det mycket god dokumentation tillgänglig.

Det finns även flera forum där olika problem och frågor diskuteras vilket gör felsökning betydligt enklare. En ytterligare motivation är att man snabbt och enkelt kan skriva egna regler då de har en mycket logisk uppbyggnad.

Anledningen till att uppsatsen är inriktad på IPS istället för IDS är att det automatiskt blockerar hot till skillnad mot IDS där systemadministratören måste analysera varningar och agera utefter dessa.

4. Experiment

För att ta fram resultat utförs två experiment där åtta stycken olika attacker utförs mot ett litet nätverk med en Pfsense brandvägg, två servrar och två klientdatorer. Det första experimentet utförs utan IPS integrerad i miljön och det andra med IPS integrerad.

4.1 Laborationsmiljö

Router och brandvägg

Som router och brandvägg används Pfsense[11] vilket är en router och brandväggs-distribution av FreeBSD, detta kördes på en fysisk dator.

Brandväggen är konfigurerad med NAT (*network adress translation*) och från WAN (*internet*) sidan är alla portar utom port 80 filtrerade. Detta för att tillåta kommunikation med webservern som finns på insidan. Trafik på port 80 till nätverkets externa IP dirigeras då med ”port forward” till webservern på insidan.

Övriga anslutningar som tillåts är de som är etablerade från någon dator på LAN-sidan till WAN-sidan(*internet*). Icke etablerade anslutningar från internet till en lokal dator blockeras, detta gäller ej för anslutningar till webbservern på port 80.

Servrar

På LAN-sidan fanns två stycken servrar. Den första är en Windows Server 2008r2 som används som DC(*domain controller*), AD(*active directory*), DNS(*domain name server*) och fil-server. Systemet använde senaste servicepack och uppdateringar som fanns tillgängliga i april 2013 som antivirusprogram användes Microsoft Security Essentials då det utan problem går att installera på Windows server operativsystem. Den andra servern var en Ubuntu server 12.04 LTS, denna användes som webbserver och körde Apache 2.2.2. Även detta system använde senaste uppdateringarna som fanns i april 2013.

Dessa två servrar var virtualiserade med hjälp av VMware Esxi version 5.0, detta för att enkelt kunna göra kopior på systemen så att de enkelt kunde återställas till ett tidigare läge.

Klienter

Klientdatorerna bestod av två stycken system med Windows 7 Pro x64. Klienterna använde senaste servicepack och uppdateringar som fanns tillgängliga i april 2013 och var utrustade med Avast Free 8 som antivirusprogram[12]. Anledningen till att just Avast valdes är att det är gratis och presterar generellt mycket bra i tester som publiceras. Dessa var fysiska datorer. Övriga applikationer som fanns installerade var Java 7 med senaste uppdateringar från april 2013.

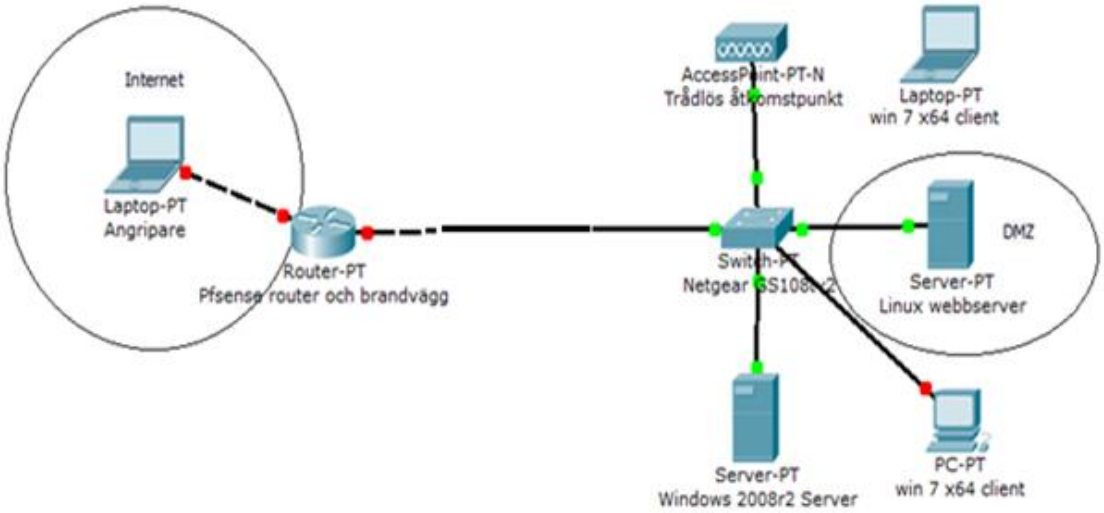
Webbsida

För att simulera ett företags hemsida skapads en hemsida där anställda kunde logga in med sitt användarkonto och skriva nyheter samt ladda upp bilder. Där fanns medvetet programmerade sårbarheter som gjorde det möjligt att utföra till exempel en SQL-injektion. Webbsidan användes som plattform för de attacker som gick via webben.

4.2 Experiment 1

I det första experimentet utfördes åtta stycken olika attacker som både privatpersoner och företag kan tänkas råka ut för. Attackerna är valda så att vissa utgår från sårbara webbapplikationer och några direkt på systemnivå. Intressant statistik på attacker under 2012 kan läsas i Verizons utredning av dataintrång under 2012[13]. De attacker som används i experimenten beskrivs i avsnitt 4.5 Attacker.

Målet var att penetrera nätverket och få kontroll över ett eller flera system. Med hjälp av resultatet från de olika attackerna kan man avgöra hur bra skydd brandvägg och antivirusprogram ger mot dessa attacker. För logisk nätverkstopologi i experiment 1 se figur 1.

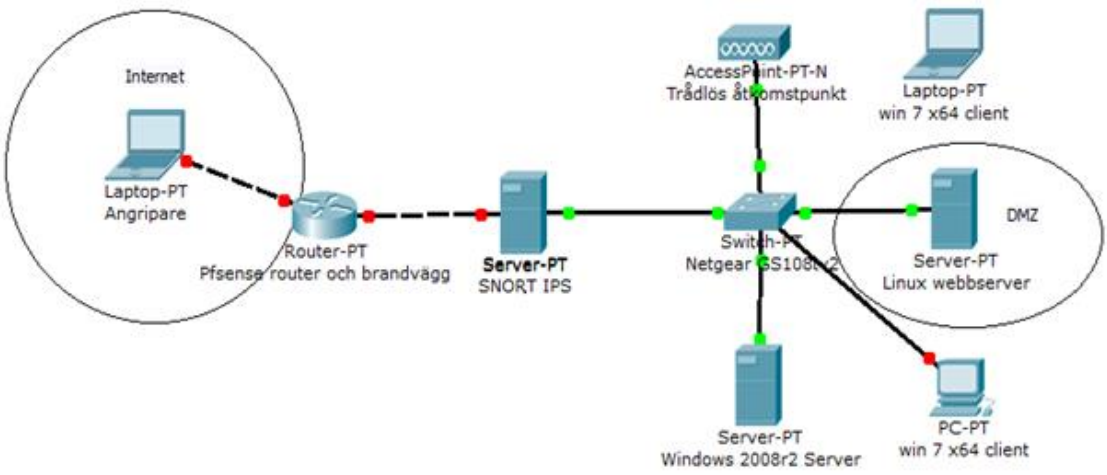


Figur 2. Logisk nätverkstopologi för experiment 1.

4.3 Experiment 2

I det andra experiment gjordes ett försök att förbättra nätverkssäkerheten genom att placera en Snort IPS i nätverket i hopp om att det skulle skydda nätverket och systemen från attackerna genom att blockera de hot som upptäcktes. Snort analyserar trafik för både ingående och utgående trafik

I experimentet genomfördes samma attacker som i experiment 1 igen för att mäta hur implementeringen av IPS-systemet påverkar resultaten. De attacker som användes finns beskrivna i avsnitt 4.5 Attacker. Hur Snort fungerar och är konfigurerat beskrivs i avsnitt 4.4 Snort IPS. För logisk topologi i experiment 2 se figur 2.



Figur 3. Logisk nätverkstopologi för experiment 2.

Mätning av hur IPS-systemet påverkar prestandan för trafik mellan WAN(internet) och LAN(det lokala nätverket) sker genom att vid flera tillfällen mäta uppladdningshastighet och nerladdningshastighet via tjänsten bredbandstest hos bredbandskollen.se[27].

4.4 Snort IPS

Snort utvecklades från början av Martin Roesch år 1998, och utvecklas nu av det kommersiella företaget *Sourcefire*. Snort är fortfarande fritt att använda gratis och det är skrivet med öppen-källkod. Den kommersiella delen består av ett grafiskt gränssnitt från *Sourcefire* och man får tillgång till nya regler när de släpps. Icke betalande användare måste vänta 30 dagar innan de får tillgång till nya regler från *Sourcefire*.

Snort finns till ett flertal operativsystem så som Linux, BSD och Windows(med vissa begränsningar)[10].

Hur Snort ska fungera och agera konfigureras med hjälp av snort.conf filen. Där finns ett stort antal variabler att ställa in efter behov. Hur man ställer in Snort för sina behov går att läsa i manualen för Snort[15].

I våra experiment användes Snort version 2.9.4.5 som släpptes 4 April 2013 installerat på Linux Debian 6, detta var en fysisk dator. Hårdvaran i systemet som användes var en Intel Pentium Dual Core processor med en klockfrekvens på 2,0GHz samt 2,0GB RAM-minne och två stycken PCI-express Intel Gigabit nätverkskort.

Nätverksinställningar

För att kunna låta nätverkstrafik passera genom Snort-datorn behöver man två stycken nätverkskort. Dessa används för att tillsammans skapa en nätverksbrygga så att trafiken kan passera genom datorn. Nätverkskorten måste konfigureras att körs i så kallat *Promiscous Mode* för att kunna hämta upp nätverkspaketen som skickas[16].

Regelverk

Regler till Snort består av specifikationer för varifrån trafiken kommer och vilken destination den har, vilket hjälper till att bestämma om trafiken ska analyseras. Reglerna

kan specificera allt från portar och protokoll till signaturer för kända mönster och hot[15]. För att ta del av de regler som Sourcefire Vulnerability Research Team utvecklar (*Snort VRT regler*) krävs det att man registrerar ett gratis-konto på Snorts hemsida. Om man vill ha nya regler så fort de publiceras måste man betala en avgift, annars släpps de fritt till registrerade användare efter 30 dagar.

Reglerna består av regel-filer där varje regel har ett id, detta id möjliggör för Snort att titta i en annan fil som medföljer där en beskrivning av vad regeln reagerar på finns, samt ibland CVE eller länkar till externa hemsidor med beskrivningar av det upptäckta hotet.

Det är inte bara VRT regler som finns att tillgå, det finns även andra företag som publicerar sina egna regelverk för Snort, i våra experiment användes de fritt tillgängliga VRT reglerna samt de fritt tillgängliga regler som publiceras av Emergingthreats.com. Man kan även mycket enkelt skriva och använda egna regler.

Ett exempel på en regel som blockerar all extern icmp trafik med destination till något system på det interna nätet kan se ut som följande:

```
drop icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"ICMP
block"; sid:10000001; rev:1;)
```

EXTERNAL_NET och *HOME_NET* är variabler som konfigureras i *snort.conf*, *any* är portnummer och *sid* är id numret för beskrivningen.

För att automatiskt hämta de senaste reglerna användes perl-scriptet *Pulledpork*[17] vilket man kör som ett *Cron* jobb så ofta som man tycker är lämpligt. Det VRT regelverk som används i vår uppsättning av Snort är version 2.9.4.1.

DAQ

För att få tillgång till nätverkstrafiken använder Snort från och med version 2.9 ett tillägg som heter DAQ (*Data Aquisition Library*), detta är ett bibliotek för att kunna kommunicera direkt med hårdvaran(*nätverkskort*) och på så sätt kunna analysera trafiken. Beroende på vilket läge man vill köra Snort i kan man välja mella olika DAQ:ar[15].

Netfilter, iptables och nfqueue

Netfilter är ett ramverk i Linux som ger operativsystemet möjlighet att filtrera och routa nätverkstrafik. Som ett verktyg för att skapa regler för nätverkstrafik och möjlighet att bygga en Linux-baserad brandvägg används ofta Iptables, vilket är en del av netfilter[18]. IPS-systemet i experimentet använde netfilter och iptables för att lägga all inkommande trafik i en netfilter-kö(*nfqueue*) som ger Snort möjligheten att analysera trafiken i kön.

Snort inline mode (IPS)

Snort är normalt konfigurerat för IDS bruk men kan med några ändringar i konfigureringsfilen snort.conf ändras till att köras som IPS. I Snort kallas detta för inline mode. I inline-läget har Snort möjlighet att påverka trafiken som analyseras vilket det inte har när det körs som IDS[15]. För att Snort ska kunna låta trafik passera genom sin analysmotor så behövs först och främst två stycken nätverkskort. Uppsättning i

experimentet använde Linux inbyggda brandvägg netfilter/iptables för att dirigera all inkommande trafik till Snort för analys av trafiken. Snort flaggar sedan trafik som ska blockeras vilket medför att trafiken inte tillåts passera och ej når sin slutdestination. För att kunna hämta information från iptables användes en DAQ som heter nfq(*netfilter-queue*) vilket tillåter snort att hämta trafiken som netfilter lägger i kön. Detta tillåter nu Snort att hämta trafik från netfilter-kön, analysera och flagga den för att netfilter ska kunna göra valet att acceptera eller blockera trafiken.

De regler som snort normalt använder sig av är oftast av typen ”alert”, detta betyder att varje gång regeln ger utslag så loggas en varning och information om händelsen. För att blockera anslutningar istället för att bara logga en varning konverterades hela regelverket till ”drop” regler genom att byta ut alla förekomster av ”alert” till ”drop”.

BASE

För att få en enkel grafisk överblick över de larm som Snort genererar användes det webbaserade gränssnittet BASE(*Basic Analysis and Security Engine*)[19]. Detta körs med APACHE2 och PHP5 lokalt på Snort datorn. Det finns även andra grafiska verktyg som visar Snort-genererade larm, BASE valdes då det är enkelt att installera och konfigurera.

4.5 Attacker

För att utföra attacker användes operativsystemet KALI-linux[20] (tidigare Backtrack-linux)vilket är en Linuxdistribution för penetrationstestning. KALI innehåller många användbara verktyg som till exempel Nmap[21] och Metasploit-framework[22].

De attacker som användes som var riktade mot Windows är noggrant framtestade för att lyckas ta sig förbi Windows 7 Pro SP1 med senaste uppdateringarna, antivirusprogram och vid tillfället senaste Java versionen. Nedan är en lista med de attacker som utförts och under dessa en närmare beskrivning.

System Attacker

- Trojan via USB

Webb Attacker

- Nmap spaning
- SQL Injektion
- Java driveby
- DoS attack
- Brute force
- Nedladdning av skadlig exe-fil
- Webb shell

Trojan via USB

Social ingenjörskonst är en teknik som används för att lura en användare att göra något som bereder attackeraren tillgång till dennes system. Scenariot i experimentet är tänkt att ett USB-minne med en intressant exekverbar fil placerats ut så att någon plockar upp det och sedan kör programmet på sin dator.

För att skapa en till synes harmlös exekverbar fil användes scriptet *Syringe*[28] för att undvika detektering av antivirusprogram. Filen som skapas ser ut att vara en seven-zip-installationsfil men innehåller egentligen Metasploits *Meterpreter reverse-tcp-shell*.

Meterpreter är en avancerad trojan som tillhör *Metasploit* ramverket. När meterpreter körs på mål-datorn injicerar meterpreter sig själv i en redan aktiv process i systemet. Meterpreter körs endast i minnet och skriver aldrig data till hårddisken vilket minskar forensisk bevisinsamling. Det finns även en funktion som gör det möjligt att migrera till en annan aktiv process vilket tillåter meterpreter att köras oförmärkt på ett system.

Meterpreter Reverse-tcp-shell trojanen anropar vid körning attackerarens dator och ger denne möjlighet att till exempel använda offrets webbkamera, spela in ljud via mikrofonen, spela in tangentbordstryckningar eller ladda upp eller ner filer från/till den infekterade datorn. Kommandot nedan skapar den exekverbara filen och sätter upp en så kallad *lyssnare* som väntar på inkommande anslutningar från infekterade datorer.

```
root@kali:~/Desktop/exjobb/syringe 0.1# ./syringe.sh
```

Nmap Spaning

För att genomföra en attack mot ett företag eller en privatperson är första steget att ta reda på information om miljön eller systemet som man vill attackera[10]. För att göra detta användes i experimenten verktyget Nmap som är en öppen-källkods baserad nätverksscanner. Detta steg som ofta kallas för *reconnaissance* (*spaning*, *informationsinsamling*) är i sig inget som behöver påverka systemen negativt men man kan få reda på mycket information som är värdefull för att kunna gå vidare med en attack mot ett specifikt system eller tjänst i miljön.

I experimentet utförs spaningsattacken från internet för att ta reda på om det finns ett eller flera system på andra sidan nätverkets brandvägg som med hjälp av NAT(*network adress translation*) gömmer det som finns på LAN-sidan. Om man hittar system genom sin spaning vill man ta reda på deras öppna portar, vilka operativsystem som används och vilka tjänster som körs. Ett kommando likt det nedanför kan användas för att ta reda på denna information.

```
root@kali:~/Desktop# nmap -sv –O 192.168.88.5
```

SQL Injektion

SQL är ett programmeringsspråk som används för att kommunicera med databaser som innehåller information. En SQL Injektion är en sårbarhet i hanteringen av indata som skickas till en databas. Dessa indata kan vara personuppgifter som till exempel en webbutik behöver för att skicka den beställda varan. Genom att skriva SQL kod som indata istället för namn och efternamn kan man manipulera databasen till att skriva ut information som normalt sätt ej tillåts.

För att utföra en SQL-injektion skapads en webbplats som ska simulera hemsidan till ett företag. Det programmerades medvetet in sårbarheter som gjorde det möjligt att utföra en SQL-injektion.

Genom SQL-injektion avslöjades användarnamn och hashade lösenord. I vårt fall användes SHA-256 hashalgoritmen.

Vi skapade alla konton och kände till alla lösenord men för att simulera en riktig attack försökte vi knäcka det hashade lösenordet med en *lösenordslista*. En lösenordslista består av en mängd olika lösenord och oftast dess respektive hashvärde. Dessa ordlistor brukar oftast vara väldigt stora, upp mot flera 100GB, eftersom de innehåller hundratusentals lösenord.

För att man ska lyckas knäcka ett hashat lösenord med hjälp av en lösenordslista måste lösenordet finnas i listan. En lista med topp 50 lösenord 2010 från Sophos[27] skapades. Programmet passwordspro användes för att hitta lösenordet med hjälp av vår lista. Det tog inte mer än en sekund för passwordspro att hitta lösenordet genom att jämföra hashsumman vi fick via SQL-injektionen med alla förgenererade hashsummor i listan.

Java driveby

För ändamålet användes en till synes trovärdig Java-applet som laddas vid besök av en hemsida. Inbakad i appleten finns en *metasploit meterpreter* trojan som injiceras direkt i datorns RAM-minne. Detta sker med hjälp av *pyinjenctor shellcode injector* vilket är ett verktyg som finns att tillgå i SET(social engineering toolkit). Tekniken med att injicera shellcode direkt till minnet gör att man enkelt kan ta sig förbi antivirusprogram[23].

Java-appleten, hemsidan och en lyssnare för inkommande anslutningar sätts enkelt upp genom att följa instruktionerna i SET.

Med användaruppgifter från SQL-injektionen loggade vi in på hemsidan och laddade upp vår webb shell som tillät oss att ändra alla filer som tillhör hemsidan.

Via vår webb shell kunde man injektera en iframe länk i hemsidans index fil som pekar mot en server som innehåller den skadliga java koden. Detta resulterar i att alla som besöker företagshemsidan kommer få upp en ruta på sin skärm som frågar användaren om java appleten skall köras. Om användaren väljer att köra filen kommer användaren att bli infekterad utan att antivirusprogrammet reagerar.

DoS Denial of Service attack

Denial of Service (*förkortat DoS*) är en form av attack vars syfte är att överbelasta ett eller flera system. Denna typ av attack är populär för att sätta hemsidor och viktiga system ur tjänst.

För attacken användes programmet *Slowloris*[24] skrivet av Robert Hansen. Slowloris använder en annan typ av metod för att utföra DoS attacker. Oftast brukar DoS verktyg fungera genom att skicka mängder med data till en server vilket resulterar i överbelastning. Man brukar utföra sådana attacker från flera tusen datorer samtidigt, detta kallas då för *Distribueted Denial of Service*(*förkortat DDoS*).

Slowloris fungerar på det sätt att den försöker öppna så många anslutningar som möjligt till en webbserver och hålla anslutningarna aktiva. Slowloris uppnår detta genom att skicka HTTP headers periodiskt men fullföljer aldrig HTTP förfrågan. Vilket betyder att webbservern kommer vänta på flera anslutningar samt acceptera nya. Detta resulterar i att webbservern kommer uppnå gränsen för att max antal anslutningar som den klarar av samtidigt vilket leder till att webbservern kommer blockera ytterligare anslutningar. Det som även utmärker Slowloris är att det inte kräver mycket bandbredd för att utföra en lyckad attack.

Slowloris fungerar på en mängd olika webbservrar, bland dem finns Apache2 som är den mest populära vilket är den som användes i experimenten. Slowloris kan köras med hjälp av följande kommando:

```
root@kali:~/Desktop# perl ./slowloris.pl -dns 192.168.88.5
```

Brute force

Brute force(*råstyrka*) är en metod som används för att testa mängder olika kombinationer av ett lösenord. Hemsidor brukar ofta använda sig av hashalgoritmer som SHA256 för att hasha lösenord. Hashalgoritmer är envägsfunktioner vilket betyder att det inte går att avkryptera en hashsumma.

För att med hjälp av brute force försöka knäcka ett hashat lösenord testar man miljontals olika kombinationer av ett lösenord som producerar samma hashsumma som lösenordet man försöker knäcka. Detta kan vara väldigt tidskrävande beroende på lösenordets längd och komplexitet.

När man ska utföra en brute force attack kan det underlätta att använda sig av en ordlista som består av förgenererade användernamn och lösenord. På detta sätt kan man testa flera tusentals olika inloggningsuppgifter på kortare tid. Nackdelen med att använda en ordlista är att om användarnamnet och lösenordet inte finns i ordlistan, kommer attacken uppenbarligen misslyckas, därför gäller det att man har en stor ordlista som även är anpassad åt målet för attacken.

I experimentet användes programmet *Hydra*[25] och en stor ordlista för att för att utföra attacken.. Ett exempel för hur ett kommando för en Hydra attack kan se ut visas nedan.

```
root@kali:~/Desktop/exjobb# hydra 192.168.88.5 http-form-post
"/index.php?page=login:username=^USER^&login=Logga+in&password
=^PASS^:Wrong username/password." -L
/root/Desktop/exjobb/users.txt -P
/root/Desktop/exjobb/pass.txt -t 10 -w 30 -o
/root/Desktop/hydra-http-post-attack.txt
```

Nedladdning av skadlig exe-fil

En enkel nedladdning av en till synes ofarlig .exe-fil. I själva verket är det en skadlig .exe-fil som skapats med hjälp av skriptet *Syringe*[28]. Attacken fungerar på samma sätt som den via USB-minnet, det som skiljer dem åt är hur filen når till systemet.

Webb shell

En webb shell är ett program som körs via en webbserver och används för att interagera med det underliggande operativsystemet. Angripare använder sig av webb shells för att skapa sig en bakdörr på en webbserver som de kan nå direkt från webben. Med hjälp av denna bakdörr kan angriparen utföra en rad olika funktioner beroende på hur avancerad bakdörren är. Grundläggande funktioner är att kunna utföra begränsade systemkommando, radera/ändra filer och ladda upp/ner filer.

Avancerade web shells tillåter till exempel en angripare att transformera webbserven till en proxy vilket betyder att angriparen kan styra all sin trafik att gå via webbservern.

En viktig aspekt att tänka på är att om systemet som webbservern är installerad på innehåller sårbarheter kan angriparen utnyttja dessa för att eskalera sina begränsade rättigheter till systemrättigheter, vilket betyder att angriparen har fullkontroll över systemet.

5. Resultat & Analys

Syftet med resultaten från våra experiment är att kunna visa om ett IPS system är en bra investering för att höja säkerheten i ett nätverk. De påverkningar som eventuellt en IPS kan medföra på prestanda och tillgänglighet i nätverket kommer även att evalueras.

5.1 Konfiguration och drift

Systemunderhållet består till största del av samma uppgift som för övriga datorsystem, konfiguration och uppderating av programvara. Att konfigurara Snort som en IPS kan vara väldigt tidskrävande och kräver mycket kunskap från teknikerns sida för att optimera systemet. Om man ej tidigare konfigurerat Snort, får man ägna mycket tid åt denna uppgift. Inför våra experiment lades ca 60 timmar på att konfigurera och driftsätta Snort.

Uppdateringar av Snorts regelverk som blockar/rapporterar attacker sker automatiskt, dock krävs det att man manuellt uppdaterar Snorts programvara när en ny version släpps.

5.2 Attacker

Nedan presenters resultat från samtliga av attackerna och på slutet visas en sammanfattande tabell.

SQL-Injektion – Utan IPS

SQL-injektionen lyckades och gav tillgång till användarnamn och hashade lösenord.

SQL-Injektion – Med IPS

Med en IPS installerad i nätverket som blockerar all illvillig trafik var det ej möjligt att utnyttja sårbarheten som gjorde det möjligt att göra en SQL-injektion. Snort detekterade ett försök till en SQL-injektion och blockerade trafiken omedelbart.

Flera metoder testades för att försöka kringgå detektering men Snort lyckades blockera alla försök. Det är fullt möjligt att det finns andra metoder för att undvika detektering som vi inte känner till, men i våra tester reagerade Snort på alla försök.

Denial of Service – Utan IPS

I experimentet lykades Slowloris göra webbservern oåtkomlig efter cirka en minut med endast en dator som utförde attacken.

Denial of Service – Med IPS

I experimentet med en IPS blev webbservern ej helt oåtkomlig men hemsidan upplevdes som väldigt långsam att surfa runt på samtidigt som attacken utfördes.

Brute force – Utan IPS

Brute force attacken lyckades och gav de rätta användaruppgifterna till hemsidans administratörs konto.

Brute force – Med IPS

Snort detekterade *Hydra*-aktivitet och blockerade inloggningsförsöken.

Web shell – Utan IPS

Eftersom man fick tillgång till inloggningsuppgifter både från SQL-injektionen och brute force attacken gavs möjligheten att logga in på hemsidan och försöka ladda upp filer. På test-hemsidan finns det en funktion för att ladda upp endast bildfiler. Funktionen använde samma kod som finns att hämta från *w3Schools*[26]. Kodens checkar som såg till att endast bildfiler gick att ladda upp gick att kringgå.

En egenutvecklad webb shell laddades upp, denna var skriven i PHP och tillåter attackeraren att utföra system kommandon på webbservern, ladda upp/ner filer samt ändra befintliga filer som tillhör hemsidan.

Web Shell – Med IPS

Snort blockerade attackerna som gjorde det möjligt att komma åt hemsidans administrator sida. För att testa denna attack även med en IPS loggade vi in på hemsidan med lösenordet vi tidigare skapat.

Snort detekterade till att börja med det *web shell* som försöktes laddas upp eftersom den upptäckte PHP taggar *<?php [..kod..] ?>*, men genom att ta bort texten *php* från taggarna lyckades gick det att kringgå Snorts detektering och filen laddades upp.

Spaning med Nmap – Utan IPS

Nmap skanningen lyckades och avslöjade en webbservern på port 80, vilken Apache version webbservern körde samt vilket operativsystem som användes.

Spaning med Nmap – Med IPS

Snort detekterade och blockerade omedelbart Nmap-aktiviteten och skannern misslyckades med att få fram värdefull information.

Java driveby – Utan IPS

Användaren fick en förfrågan att köra en till synes trovärdig *Java-applet*, vid alternativet *kör* injicerades den skadliga koden i datorns RAM-minne utan detektering. Attackeraren gavs då möjlighet att interagera med offrets dator på olika sätt.

Java driveby – Med IPS

Snort detekterade och blockerade den potentiellt skadliga *java-appleten* och användaren såg aldrig någon förfrågan om att köra någon applet.

Trojan via USB – Utan IPS

USB minnet kopplades in i datorn och den till synes ofarliga exekverbara filen kördes utan detektion av antivirusprogram. Attackeraren gavs då möjlighet att interagera med offrets dator på olika sätt.

Trojan via USB – Med IPS

Antivirus programmet upptäckte inte trojanen, men Snort reagerade på trafiken som skickades till angriparen när trojanen startades och blockerade den. Hade trafiken varit krypterad hade Snort inte reagerat.

Nedladdning av skadlig .exe fil – Utan IPS

En trojan laddades ner som skapades tidigare från angriparens dator för att simulera att nedladdningen skulle komma från internet. Precis som tidigare reagerade inte antivirus programmet och trojanen kördes utan problem.

Nedladdning av skadlig .exe fil – Med IPS

Snort blockerade den exekverbara filen. Antivirusprogrammet reagerade inte.

Sammanfattning av attacker

Tabell 1 nedan listar de attacker som testats och vilket resultatet av attackerna blev.

Attack	Experiment 1, utan IPS	Experiment 2, med IPS
Nmap spaning	Lyckad	Misslyckad
Trojan via USB	Lyckad	Misslyckad
Nedladdning av skadlig .exe	Lyckad	Misslyckad
Denial of Service	Lyckad	Misslyckad
Java driveby	Lyckad	Misslyckad
SQL-injektion	Lyckad	Misslyckad
Brute force	Lyckad	Misslyckad
Webb shell	Lyckad	Lyckad

Tabell 1. Experimentresultat

Efter att i experiment 1 ha genomfört åtta stycken lyckade attacker mot det givna nätverket varav flera är av allvarlig grad så ser man att IPS systemet lyckades med att detektera och blockera attackerna med gott resultat. Med allvarlig grad avses attacker som resulterar i att till exempel kunna exekvera kommandon på systemet, hämta ner och ladda upp filer, läsa av tangentbords tryckningar eller använda webbkameran.

Med ett IPS som tillägg för att förbättra nätverkssäkerheten genom att aktivt analysera trafiken detekterades och blockerades 87,5% av attackerna jämfört med 0% vid experiment utförda utan en IPS. Bland de blockerade attackerna ingår de allvarligaste vilka i det första experimentet gav direktåtkomst till de utsatta systemen.

5.3 Prestanda

Vi upptäckte att nätverksprestandan sjönk avsevärt med Snort IPS implementerat i nätverket. Vid de mätningar som utfördes innan användning av IPS-systemet blev medelvärdet ca 91Mbit/s i nedladdningshastighet och ca 11Mbit/s i uppladdningshastighet.

Med Snort IPS-systemet i nätverket sjönk medelvärdet till ca 12Mbit/s i nedladdningshastighet medan uppladdningshastighetens medelvärde förblev oförändrat med ca 11Mbit/s. För att optimera nätverksprestandan var vi tvungna att

ändra i konfigurationsfilen snort.conf för att inaktivera regler som stoppade attacker mot specifika tjänster som vi ej utnyttjade i våra experiment, som till exempel VoIP (*Voice over IP*) servrar eller SCADA protokoll.

Efter ett antal nya mätningar hade medelvärdet för nedladdningshastighet ökat till ca 17Mbit/s.

Att anpassa regler för hur miljön ser ut är något man kan lägga mycket tid på för att få ut optimal prestanda, analys av till exempel protokoll för SCADA-trafik är onödigt att aktivera om man ej använder dessa protokoll i sitt nätverk.

Eftersom Snort analyserar all trafik som passerar mellan internet och det lokala nätverket skulle problemet kunna lösas genom att lastbalansera trafiken mellan flera instanser av Snort och på så sätt minimera förlusten i bandbredd men fortfarande behålla det extra skyddet som IPS systemet ger. Ett ensamt Snort IPS ger alltså en stor negativ påverkan i nätverksprestanda samtidigt som det ger ett ökat skydd mot olika säkerhetshot.

5 Diskussion

Snort har efter genmförda experiment visat sig vara att bra kompletterande lager till skalskyddet. Trots detta extra skydd som hjälper till att skydda de övriga systemen i nätverket skapas det en ny sårbar länk som påverkar hela nätverket. En avancerad DoS attack mot ett eller flera system på insidan av vårt nätverk, till exempel webbservern skulle resultera i att Snort överbelastas och all trafik mellan internet och nätverket upphör.

En ytterligare sårbarhet som uppenbarar sig är den webbserver och webbgränssnittet(BASE) som användes för att grafiskt kunna övervaka Snort larm. Denna webbserver går ej att nå från internet utan endast från insidan av nätverket. Ponera att en angripare redan tagit sig in i nätverket och finner sårbarheter på webbservern eller i BASE, beroende på sårbarhetens grad skulle angriparen ha möjlighet att maskera sin väg in genom att ändra informationen som finns sparad i databasen på webbservern.

Denial of Service attacken var delvis lyckad när Snort var implementerad i nätverket eftersom hemsidan upplevdes som väldigt långsam. Hade fler datorer använts vid DoS attack hade hemsidan blivit otillgänglig och Snort hade förmodligen överbelastats.

Snort blockerade Nmap skanningen dock finns det andra metoder man kan använda sig av för att skanna webbservrar. Genom att använda ett program som heter Netcat lyckades vi få ut samma information som vi tidigare fick med Nmap, förutom vilken version av SSH som användes.

```
hax@crunchbang:~/hax$ nc 192.168.88.5 80
HEAD / HTTP/1.0

HTTP/1.1 200 OK
Date: Fri, 26 Apr 2013 09:37:45 GMT
Server: Apache/2.2.22 (Ubuntu)
X-Powered-By: PHP/5.3.10-1ubuntu3.5
Set-Cookie: PHPSESSID=esa2o5lm4v16udrge1ejfmq647; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Vary: Accept-Encoding
Connection: close
Content-Type: text/html
```

Figur 3: HEAD förfrågan som skickas till webbservern som svarar med värdefull information

I figur 3 ser man hur man kan använda Netcat för att extrahera meningsfull information från HTTP headers. Genom att skicka en HEAD förfrågan till webbservern fick vi reda på vilken version av Apache, PHP samt vilket operativsystem som användes.

Det går dock att motverka denna typ av spaning genom att ändra konfigurationen för Apache och PHP. Lägger man till följande inställning till Apaches konfigurationsfil *apache2.conf*

```
ServerTokens ProductOnly
ServerSignature Off
```

Kommer Apache inte visa vilken version av webbservern som används.

Ändrar man expose_php On till Off i konfigurationsfilen för PHP kommer webbserver inte berätta vilken version av PHP som används. I figur 4 kan man se HTTP headers från en webbserver där man har ändrat dessa inställningar.

```
hax@crunchbang:~/hax$ nc 192.168.88.5 80
HEAD / HTTP/1.0

HTTP/1.1 200 OK
Date: Fri, 26 Apr 2013 09:51:46 GMT
Server: Apache
Set-Cookie: PHPSESSID=l2gv2jt5t2f8g3m4qurpb8dco6; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Vary: Accept-Encoding
Connection: close
Content-Type: text/html
```

Figur 4: HEAD förfråga som skickas till webbservern som inte svarar med värdefull information.

I figur 4 går det se att verken Apaches eller PHPs signatur syns i HTTP headers. Server signaturer kan innehålla värdefull information därför är det rekommenderat att avaktivera dessa.

Snort blockerade inte vår webb-shell men för att motverka detta kan man ändra *short_open_tag = On* till *Off* i konfigurationen för PHP som heter *php.ini*. Detta hindrar inte en lyckad uppladdning av filen men hindrar webbservern från att köra filen vilket betyder att angriparen misslyckas med att få tillgång till ett fungerande shell.

6 Slutsats

Att använda en IPS för att ytterligare försöka öka säkerheten i ett nätverk har visat sig vara ett bra komplement som ger ett ytterligare skal i det skydd man byggt upp för att säkra sin infrastruktur. Vi blev positivt överraskade över hur väl Snort blockerade ca 87% av de olika attackerna i våra experiment utan vi själva skapat några specifika regler.

För att optimera prestandan behöver däremot en hel del tid läggas på att specialanpassa Snorts konfigureringsfil för vad man vill analysera samt vilka regelverk man känner att man behöver använda sig av. Själva använde vi oss av i stort sett alla de regler som fanns tillgängliga och som inte var utkommenterade. Prestandamässigt presterade inte vår konfiguration optimalt då vi endast fick ut ca 1/5 av bandbredden som fanns tillgänglig. Men att analysera all trafik som passerar mellan internet och alla datorer i ett LAN kräver mycket datorkraft för att inte påverka prestandan i nätverket. För att lösa detta tror vi att någon form av lastbalansering är nödvändig för att låta flera instanser av Snort hjälpas åt att analysera trafiken och på så sätt behålla prestandan samtidigt som man har det ökade skyddet av sitt nätverk kvar utan att låta trafik slippa igenom utan att analyseras. En annan svaghet med IPS är att krypterad trafik inte kan analyseras.

Efter genomförda experiment så anser vi att ett IPS är ett kompetent extra lager i skalskyddet för att hjälpa till att avvärja hot som inte blockeras av brandväggar och antivirusprogram.

6.1 Framtida forskning

Inför framtiden skulle vi vilja se mer forskning inom IPS-området då de positiva resultat vi sett från våra experiment visar att tekniken är bra. Exempel på framtida forskning skulle vara att lastbalansera trafik för IPS-systemen så att man uppnår bra prestanda i nätverken. Ett annat intressant område är hur man skulle kunna attackera eller kringgå en IPS utan att väcka misstankar om intrång.

Referenser

[1]LabCenter., Svenska IT-säkerhetshandboken 1.0(2009) Pagina ISBN 978-91-636-0953-4

[2]Symantec,2012 Norton Study, 2012, http://www.symantec.com/about/news/release/article.jsp?prid=20120905_02 Besöksdatum: 2013-06-03

[3]Ciza T. et al., "Usefulness of DARPA Dataset for Intrusion Detection System Evaluation", ????, http://eprints.iisc.ernet.in/26885/1/darpa.pdf Besöksdatum: Besöksdatum: 2013-06-03

[4]Wei L., "Evaluation of Intrusion Detection Systems", 2007, http://www.cs.auckland.ac.nz/compsci725s2c/archive/termpapers/lwei.pdf Besöksdatum: 2013-06-03

[5]Ringström Saltin M., "Intrusion Detection Systems: utvärdering av snort", 2009, http://www.diva-portal.org/smash/get/diva2:222880/FULLTEXT01 Besöksdatum: 2013-06-03

[6]Cisco IPS produkter, http://www.cisco.com/en/US/products/ps5729/Products_Sub_Category_Home.html Besöksdatum: 2013-06-03

[7]Suricata IDS/IPS, http://suricata-ids.org/ Besöksdatum: 2013-06-03

[8]Caswell B et al. Snort Intrusion Detection and Prevention toolkit(2007) SYNGRESS ISBN-10: 1597490997

[9]Bruneau G., "History and Evolution of intrusion Detection", 2001 http://www.sans.org/reading_room/whitepapers/detection/history-evolution-intrusion-detection_344 Besöksdatum: 2013-06-03

[10]Snort, http://www.snort.org/snort Besöksdatum: 2013-06-03

[11]Pfsense, http://www.pfsense.org/ Besöksdatum: 2013-06-03

[12]Avast, http://www.avast.com/en-se/index Besöksdatum: 2013-06-03

[13]Verizon, ”Data breach investigations report 2012”, 2012, http://www.verizonenterprise.com/resources/reports/rp_data-breach-investigations-report-2012_en_xg.pdf Besöksdatum: 2013-06-03

[14]Bredbandskollen, http://www.bredbandskollen.se/ Besöksdatum: 2013-06-03

[15]Snort Team., Snort User’s Manual, 2012, http://s3.amazonaws.com/snort-org/www/assets/166/snort_manual.pdf Besöksdatum: 2013-06-03

[16]Rouse M, Definition av promiscuous mode, http://searchsecurity.techtarget.com/definition/promiscuous-mode Besöksdatum: 2013-06-03

[17]Pulledpork hemsida, code.google.com/p/pulledpork/ Besöksdatum: 2013-06-03

[18]Netfilter, http://www.netfilter.org/ Besöksdatum: 2013-06-03

[19]BASE, http://base.secureideas.net/ Besöksdatum: 2013-06-03

[20]KALI-linux, http://www.kali.org/ Besöksdatum: 2013-06-03

[21]Nmap, http://nmap.org/ Besöksdatum: 2013-06-03

[22]Metasploit Framework, http://www.metasploit.com/ Besöksdatum: 2013-06-03

[23]Pyinjector shellcode injector, https://www.trustedsec.com/august-2012/new-tool-pyinjector-released-python-shellcode-injection/ Besöksdatum: 2013-06-03

[24]Slowloris.pl, http://ha.ckers.org/slowloris/ Besöksdatum: 2013-06-03

[25]Hydra, http://www.thc.org/thc-hydra/ Besöksdatum: 2013-06-03

[26]Filuppladdning för php, http://www.w3schools.com/php/php_file_upload.asp Besöksdatum: 2013-06-03

[27]50 vanligaste lösenorden 2010, http://nakedsecurity.sophos.com/2010/12/15/the-top-50-passwords-you-should-never-use/ Besöksdatum: 2013-06-03

[28]Syringe, https://code.google.com/p/syringe-antivirus-bypass/ Besöksdatum: 2013-06-03

Space for presentation of the author.
Photo of the author to be placed above.

Space for presentation of the author.
Photo of the author to be placed above.