Satakunnan ammattikorkeakoulu

OPINNÄYTETYÖ

Risto Sillberg

Satakunnan ammattikorkeakoulu

Risto Sillberg

TIETOVERKKOON TUNKEUTUMISEN HAVAITSEMINEN SNORTIN AVULLA

Tekniikka Pori

Tietotekniikan koulutusohjelma

2008

TIIVISTELMÄ

TIETOVERKKOON TUNKEUTUMISEN HAVAITSEMINEN SNORTIN AVULLA
Sillberg Risto

Satakunnan ammattikorkeakoulu
Tietotekniikan koulutusohjelma
Toukokuu 2008
Valvoja: Holm Hannele
UDK: 004.056, 004.41, 004.49
Sivumäärä: 50
Asiasanat: IDS, DDoS, Snort, IPS, tunkeutumisen havaitseminen

---

Tietoverkkojen käyttöön liittyy monenlaisia uhkia, jotka saattavat aiheuttaa tietoturvariskin. Opinnäytetyössä tutustuttiin tunkeutumisen havaitsemisjärjestelmiin ja niiden toimintaan. Työssä tutkittiin myös kuinka hyökkäyksiä voidaan toteuttaa, miten tunkeutumisia voidaan havaita tietoliikenneverkoissa ja miten niitä voidaan estää.

Työssä perehdyttiin myös siihen, kuinka yleistä on joutua hajautettujen palvelunestohyökkäysten (DDoS, Distributed Denial of Service) kohteeksi ja kuinka DDoS-hyökkäys tapahtuu.

Tunkeutumisen havaitsemisjärjestelmistä valittiin Snort-ohjelmisto, koska se on maailmanlaajuisesti suuren suosion saanut tehokas työkalu tunkeutumisia vastaan. Työn alussa tutkittiin Snort-ohjelmiston rakennetta ja toimintatapoja sekä sen osuutta tietoliikenneverkon tietoturvassa. Työssä tutkittiin myös sitä, miten tunkeutumisilta voidaan välttyä ja miten ennalta ehkäistä hyökkäysten tapahtuminen.

Ohjelmistot asennettiin laboratorio-oloissa. Snort on ilmainen avoimeen lähdekoodiin (Open Source) perustuva ohjelmisto. Testauksessa käytettiin lisäksi ilmaisia Nmap- ja Nessus-ohjelmia.

ABSTRACT

NETWORK INTRUSION DETECTION WITH SNORT
Sillberg Risto

Satakunta University of Applied Sciences
Degree Programme in Information Technology
May 2008
Supervisor: Holm Hannele
UDC: 004.056, 004.41, 004.49
Number of Pages: 50
Keywords: IDS, DDoS, Snort, IPS, intrusion detection

Usage of data networks have different kinds of threats, which can incur data security risks. The purpose of the thesis was to familiarize oneself in intrusion detection systems and how they work. The thesis also studied how attacks can be put into practice, how intrusions can be detected in data communication networks and how they can be prevented.

There was also a study about how common it is to become attacked by Distributed Denial of Service-attacks (DDoS) and how DDoS-attacks occur.

The Snort intrusion detection system was chosen for this thesis, because it is a powerful tool against intrusions and it has acquired substantial popularity worldwide. At the beginning of the thesis there is a study on the structure and operation modes of Snort as well as its role is in the data security of a data communication network. Also there was a study on the methods how intrusions can be avoided and how to prevent attacks from happening.

All programs were installed in laboratory conditions. Snort is a free, open source based program. The programs used in testing were Nmap and Nessus, which are free to use.

LYHENTEET

NIDS	Network Intrusion Detection System
BASE	Basic Analysis and Security Engine
ACID	Analysis Console for Intrusion Databases
IDS	Intrusion Detection System
NIDS	Network Intrusion Detection System
IPS	Intrusion Prevention System
(D)DoS	(Distributed) Denial of Service
DNS	Domain Name Server
ISP	Internet Service Provider
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
ICMP	Internet Control Message Protocol
HTTP	Hypertext Transfer Protocol
CSV	Comma Separated Value
XML	Extensible Markup Language
SNML	Simple Network Markup Language
ASCII	American Standard Code for Information Interchange
PHP	PHP: Hypertext Preprocessor

SISÄLLYS

1 JOHDANTO

Nykypäivänä tietokoneiden ja tietokonelaitteistojen turvajärjestelmät ovat kehittyneet hyvinkin paljon viimeisien kymmenien vuosien aikana. Tänä aikana on kehitetty monia uusia menetelmiä turvata tietoja ja erilaisia tunnistamismetodeja, joilla tunkeutujat voidaan pitää järjestelmän ulottumattomissa. Kuitenkin järjestelmiä on melko mahdotonta saada murtovarmaksi ja täten on mahdollisuus että järjestelmään voidaan tunkeutua erilaisia keinoja käyttäen. Jos tunkeutuja pääsee murtautumaan järjestelmään, tämä voi tehdä hyvinkin paljon tuhoa aikaan pääsemällä luvatta käsiksi esimerkiksi arkaluontoisiin dokumentteihin tai haittaamalla järjestelmän resursseja. Työssä käsitellään millaisia eri murtautumistapoja, ja miten niiltä voidaan suojautua käyttämällä Snort-tunkeutumisen havaitsemisohjelmaa sekä Snort-ohjelman rakennetta. Työn tarkoituksena on luoda Snort-ohjelman avulla tunkeutumisen havaitsemisjärjestelmä, joka osaa havaita hyökkäykset, joista järjestelmän ylläpitäjät voivat tehdä mahdolliset toimenpiteet hyökkäyksiä vastaan. Työ keskittyy paljolti hyökkäysten estämiseen, joka on hyvin ajankohtainen asia ympäri maailmaa vrt. palvelunestohyökkäykset Virossa vuoden 2007 huhtikuun lopulla.

2 YLEISTÄ ASIAA TUNKEUTUMISEN HAVAITSEMISESTA

2.1 Mitä tunkeutuminen tarkoittaa

Amoroson määritelmän mukaan ***tunkeutuminen on pahantahtoisen vastapuolen aiheuttama tapahtumaketju, joka koostuu toisiinsa liittyvistä tapahtumista, ja joka aiheuttaa tietoturvauhkien luvattoman realisoitumisen kohteena olevassa tietojärjestelmässä tai -verkossa***. Amoroso on perustellut määritelmäänsä ja siinä painotettuja asioita ja sanavalintoja muun muassa seuraavasti. /5/

Pahantahtoisen vastapuolen aiheuttama

Tunkeutuminen ei tapahdu vahingossa, vaan se on aina tunkeutujan tarkoitus murtautua järjestelmään. Määritelmässä ei oteta kantaa siihen, että onko tunkeutumisen aiheuttajana hakkerin määrätietoinen vai robotin aiheuttama toiminta. Kyse ei ole tunkeutumisesta, jos hyökkääjän tarkoitusperät eivät ole pahansuovat. Vahingossa tapahtuva tunkeutuminen kuuluu ”tunkeutumistieteen” sijaan johonkin muuhun tieteenalaan, esimerkiksi luotettavuuteen, turvallisuustekniikkaan, käytettävyyteen jne. Määritelmän mukaan sekä pahantahtoisen että ”viattoman” toiminnan seurauksena syntyvien tietoturvauhkien yhteisenä yläkäsitteenä on alettu käyttää termiä tiedon takaaminen (information assurance). /5/

Tapahtumaketju

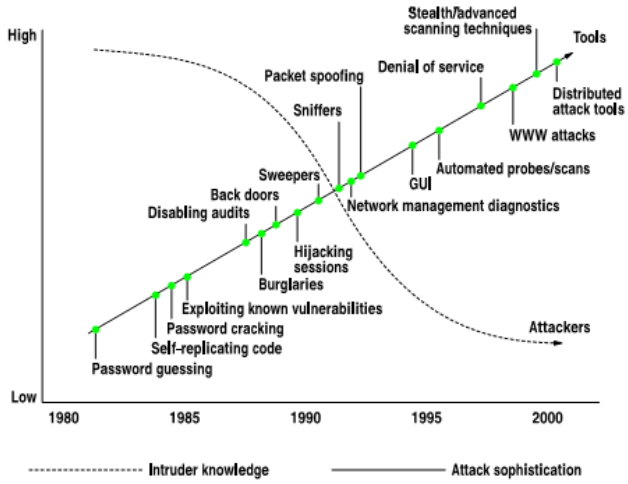
Tunkeutuminen on tapahtumaketju, joka voi ajallisesti jakaantua sekuntien, minuuttien, tuntien, kuukausien tai jopa vuosien mittaiselle ajanjaksolle. Tunkeutumisen ensimmäiset merkit voivat koostua esimerkiksi porttiskannauksista tai muista 'aikaisista varoituksista', näitä seuraamalla ja oikein tulkitsemalla tarkkaavainen käyttäjä voi ehtiä reagoimaan tunkeutumiseen jo hyvin varhaisessa vaiheessa. /5/

Toisiinsa liittyvistä tapahtumista

Vain tunkeutumiseen liittyvät tapahtumat ovat osa tunkeutumista. Tapahtumat, joilla ei ole mitään tekemistä tunkeutumisen kanssa, eivät liity tunkeutumiseen. Käyttäjän näkökulmasta on hyvä tietää, mikä liittyy tunkeutumiseen ja mikä ei. Tunkeutuja voi esimerkiksi ajoittaa tunkeutumisen ajallisesti hyvinkin pitkän ajan varrelle, jolloin tunkeutumiseen liittyvät tapahtumat hukkuvat muuhun liikenteeseen. Toisaalta hyökkääjä voi myös vetää puolustajan huomion muualle aiheuttamalla tapahtumia, jotka eivät liity mitenkään tunkeutumiseen. /5/

2.2 Tunne vihollisesi

Ei ole vain tärkeää tietää mitä on suojaamassa, vaan myös millaisia uhkia voi olla olemassa. Jos ei ymmärrä tunkeutujan toimintatapaa, on miltei mahdotonta rakentaa toimivaa suojaa sitä vastaan. Nämä tunkeutujat eivät piittaa siitä, ketä vastaan he hyökkäävät. He voivat aiheuttaa hyökkäyksen kohteelle erilaista haittaa kuten: Palvelu menee käyttökelvottomaksi (downtime) tai tietoja häviää (data loss). Alla olevassa kuvassa näkyy miten hyökkääjien ammattitaito ja hyökkäyksien kehitys on muuttunut. /1/



Kuva 1. Tunkeutujien ammattitaidon ja hyökkäyksien kehityksen muutos /3/

Onnenonkijat, varkaat ja vandaalit

Hyvin usein nämä hyökkääjät vain etsivät suuria määriä järjestelmiä, jotka voisivat olla vahinkoalttiita heidän tuntemilleen tekniikoille. Onnenonkijat yrittävät murtautua kohteeseensa yleensä vain haasteen vuoksi. Päästyään sisään, hyökkääjä mahdollisesti vain ”katselee” ympärilleen tai alkaa vandaaliksi. Tunkeutuja, joka ei välttämättä ole pahansuopainen, voi tehdä tahattomasti tuhoa tai aiheuttaa palvelunestoa toimillaan. Varkaat eivät yleensä ole hakemassa kohteen tallennettuja tietoja, vaan kohteen internet-yhteyttä, laskentatehoa tai käyttävät postijärjestelmää hyväkseen omiin tarkoituksiinsa. Vandaalit usein töhrivät kohteen verkkosivut niin, että kohde menettää mahdollisesti mainettaan.

Ammattilaiset

Yleensä hallituksien järjestelmiin kohdistuvien hyökkäyksien takana on joitakin terroristeihin liittyviä ryhmiä. Hyökkäysten todennäköisyys pieniin tietojärjestelmiin ovat yleensä melko pieni. Hyökkäysten vaikutukset voivat kuitenkin olla hyvinkin suuria.

Tyytymättömät nykyiset ja entiset työntekijät ja urakoitsijat

Uhat tältä ryhmältä on usein aliarvioitu. Nämä ihmiset tietävät kohteen verkkoympäristön ja osaavat usein ohittaa suojaukset. Heillä on motiivi, tilaisuus ja taito tehdä oikeaa vahinkoa kohteelleen. Tällaiselta uhalta on melko vaikea suojautua ja se vaatii harkintaa ja tarkkaavaisuutta.

Robotit ja madot

Nämä ovat usein automatisoituja hyökkäyksiä, jotka etsivät haavoittuvia järjestelmiä tiettyyn hyökkäykseen.

2.3 Hyökkäyksen anatomia: Viisi P:tä

Nämä viisi kohtaa seuraavat hyökkäyksen kulkua, oli hyökkäyksen lähde mikä tahansa. Nämä viisi P:tä ovat Probe/Tiedustelu, Penetrate/Tunkeutuminen, Persist/Itsepintaisuus, Propagate/Eteneminen ja Paralyze/Lamauttaminen. /1/

2.3.1 Tiedustelu (Probe)

Tässä vaiheessa hyökkääjä kerää tietoja mahdollisesta kohteesta. Kohdistetussa hyökkäyksessä skannaus voi olla rajoitettu kohteen IP-osoitteisiin, kun taas kohdistamattomassa hyökkäyksessä skannaus on laajempi. Usein ensimmäisten tiedustelujen yhteydessä ei lähetetä yhtään pakettia kohdeverkkoon. Tämän vaiheen päämääränä on kartoittaa kohteen verkko ja määrittää mitä järjestelmiä kohteessa on. Tämä antaa hyökkääjälle edun räätälöidä hyökkäystä seuraavaa vaihetta varten.

2.3.2 Tunkeutuminen (Penetrate)

Kun järjestelmät ja mahdolliset haavoittuvat palvelut on havaittu, seuraava askel on hyökkäys. Hyökkäys voi koostua monenlaisista muodoista, Esim. järjestelmä suorittaa hyökkääjän tekemän haittakoodin. Automatisoidut hyökkäykset, kuten madot ja skriptit, käyttävät itse asiassa tiedustelun ja tunkeutumisen kombinaatiota yksinkertaisesti aloittamalla hyökkäyksiä moneen osoitteeseen. Joskus hyökkäys (haittaohjelma) on naamioitu johonkin toiseen ohjelmaan troijalaiseksi hevoseksi.

Jatkuvat sisäänkirjautumisyritykset (Authentication grinding)

Jos tiedusteluvaiheessa löydetään jollekin palvelulle käyttäjänimi/salasana -kehote, hyökkääjä voi käyttää erilaisia tapoja testatakseen salasanoja. Näitä voi olla ns. ”Bruteforce”-menetelmä (Raaka voima, yrittää jokaista mahdollista kirjainta niiden kombinaatioita ja pituutta) tai sanakirjamainen menetelmä (testaa kaikki mahdolliset sanat, jotka esiintyvät sanakirjassa). Käyttäjänimet ovat melko helppoja selvittää, koska sähköpostiosoitteet ovat usein samat kuin käyttäjänimet tai käytetään käyttöjärjestelmien ja ohjelmien oletus-käyttäjänimiä. Jos järjestelmä ei lukitse käyttäjätiliä, kun määrätty lukumäärä vääriä kirjautumisyrityksiä on tehty, voi

hyökkääjä arvata salasanaa niin kauan kunnes se on selvillä. Lukitus voidaan kuitenkin murtaa web-pohjaisella käyttöliittymällä, joka ei käytä samaa suojausta käyttäjän tunnistamiseen.

Puskurin ylivuodot (Buffer overflows)

Kun ohjelma on käynnissä, se varaa muistiin tietoja, joita ohjelma tarvitsee. Hyvin usein ohjelma kysyy käyttäjältä tiedon kuka ohjelman ”omistaa”, tämä tieto tallennetaan näihin muistilohkoihin. Esim. ohjelma kysyy käyttäjältä nimeä ja varaa tähän 20 merkkiä muistista. Tunkeutuja pakottaa ohjelman ylikirjoittamaan varattuun muistiin, jota taas toiset ohjelmat käyttävät. Näin järjestelmää huijataan siten, että hyökkääjä pääsee käynnistämään omia ohjelmiaan. Useimmat puskurin ylivuodot aiheuttavat palvelun tai järjestelmän kaatumisen.

Ohjelman käyttäytyminen rajapintavirheissä (Application behavior boundary flaws)

Hyvä esimerkki tästä on tekniikka, joka huijasi Windows IIS (Internet Information Server) web-palvelimia käyttämällä ”../” merkkijonoa. Useimmissa käyttöjärjestelmissä merkkijono käskee komentokehotetta menemään yläkansioon tiedostojärjestelmässä. Käsittelemällä URL:a näillä merkeillä, oli mahdollista päästä web-sivuston juurihakemistoon ja sitä kautta järjestelmähakemistoon.

2.3.3 Itsepintaisuus (Persist)

Kun hyökkääjä on löytänyt haavoittuvan järjestelmän, ja onnistuneesti hyökännyt kohteeseen, olisi harmillista tehdä toistamiseen koko prosessi alusta asti uudelleen pyrkiessään järjestelmään. Hyökkääjä voi mahdollisesti tehdä pääkäyttäjän ja salasanan, jonka vain hyökkääjä tietää tai hankkii käyttäjä/salasana -tietokannan järjestelmästä ja purkaa salasanat auki. Hyökkääjät voivat piilottaa todisteet toimistaan muokkaamalla tai poistamalla systeemi/palomuuri -lokeja ja käyttävät mahdollisesti työkaluja, jotka piilottavat hakemistoja. Jos hyökkääjä on robotti tai verkkomato, voivat ne kopioida itsensä systeemitiedostoihin selviytyäkseen uudelleenkäynnistyksistä.

2.3.4 Eteneminen (Propagate)

Kun hyökkääjä on murtautunut järjestelmään, seuraava askel on tarkistaa, mitä muuta on saatavilla. Tämä vaihe alkaa siitä, että uhrikone toimii hyökkäyksen lähteenä. Seuraavaksi hyökkääjä yrittää kartoittaa sisäisen verkon. Jos uusia kiinnostavia koneita löytyy, ovat ne myös kohteita.

2.3.5 Lamauttaminen (Paralyze)

Hyökkääjän tarkoituksena on murtautua järjestelmään. Päämääränä voi olla varastaa tai tuhota tietoa, kaataa järjestelmä tai käyttää hyväkseen järjestelmää toiseen järjestelmään hyökkäämiseksi. Toisin sanoen viaton järjestelmä saadaan näyttämään varsinaiselta hyökkääjältä. Koneita käytetään ns. zombi-koneina.

2.4 Mikä on tunkeutumisen havaitsemisjärjestelmä (IDS) ja tunkeutumisen estojärjestelmä (IPS)?

IDS (Intrusion Detection System) on tunkeutumisen havaitsemisjärjestelmä, joista ollaan siirtymässä toisen sukupolven IPS-järjestelmiin ja samalla yhä automaattisempaan verkkoturvaan. Yleensä tunkeutumisen estojärjestelmällä IPS (Intrusion Prevention System) tarkoitetaan edelleen kehitettyä versiota IDS:stä. Josta voidaan sanoa, että siinä, missä IDS varoittaa tunkeutumisyrityksestä, IPS menee pidemmälle sillä se myös estää hyökkääjää pääsemästä eteenpäin. Kyseessä on siis tietomurtojen ehkäisyyn tarkoitettu työkalu, joka tunnistaa alkavan hyökkäyksen ja estää sen onnistumisen. /6/

2.4.1 Tunkeutumisen estojärjestelmä (IPS)

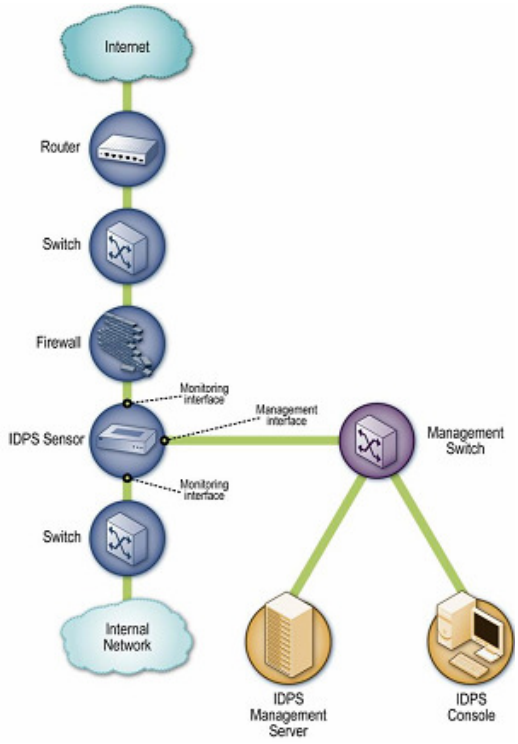
Tunkeutumisen estojärjestelmä on tietokoneella oleva suojausohjelma, joka tarkkailee verkon toimintaa ja/tai järjestelmän toimintaa mahdollisilta hyökkäyksiltä, ohjelma osaa reagoida tähän hyökkäykseen reaaliajassa torjumalla tai estämällä hyökkäävän toiminnan. Esimerkiksi verkkopohjainen IPS toimii ns. “inline” tilassa eli järjestelmä on kytketty ”linjalle”, jossa se tarkkailee kaikkea verkkoliikennettä havaitakseen haitallista koodia tai hyökkäyksiä. Kun hyökkäys on havaittu,

hyökkäävät paketit voidaan pudottaa pois samalla antaen laillisen pakettivirran päästä läpi. Tunkeutumisen estojärjestelmä kehitettiin 1990-luvun loppupuoliskolla selventämään epäselvyyttä passiivisten verkkojen monitorointiin. IPS:n huomattava parannus palomuurien teknologiasta on se, että IPS tekee pääsynvalvonnan päätökset ennemmin ohjelmakohtaisesti kuin IP-osoitteen tai porttien perusteella, kuten normaalit palomuurit ovat tehneet.

Tunkeutumisen estojärjestelmä voi myös toimia tietokonetasolla verkkopohjaisenjärjestelmän sijaan torjumaan mahdollista epäilyttävää toimintaa. Näiden teknologioiden välillä on monia hyötyjä ja haittoja, mutta monissa tapauksissa on niiden ajateltu täydentävän toisiaan. Tunkeutumisen estojärjestelmän pitää myös olla hyvä tunkeutumisen havaitsemisjärjestelmä, jotta saadaan pienennettyä väärien hälytysten määrää. /7/

“Inline” (IPS-järjestelmä)

Yleensä “inline”-sensori sijoitetaan samaan tapaan kuin palomuurit ja muut laitteet verkkotopologiassa niin että verkkoliikenteen on kuljettava sensorin läpi. Päämotiivi sijoittaa IPS-sensorit ”inline”-tilaan on se että halutaan pysäyttää hyökkäykset estämällä niiden pääsy sisäverkkoon. Alla olevassa kuvassa on esimerkki tyypillisestä IPS:n sijoittamisesta, sensori voidaan sijoittaa myös ennen palomuuria. /25/



Kuva 2. IPS-sensori Inline-tilassa /25/

2.4.2 Tunkeutumisen havaitsemisjärjestelmä (IDS)

Yleisesti tunkeutumisen havaitsemisjärjestelmä havaitsee ei-toivottuja järjestelmän muutosyrityksiä. Nämä muutosyritykset saattavat olla osa hyökkäystä, joita hyökkääjä valmistelee.

Tunkeutumisen havaitsemisjärjestelmää käytetään havaitsemaan useita mahdollisesti haitallista käyttäytymistä, joka saattaisi vaarantaa tietoturvallisuutta ja luotettavuutta tietokonejärjestelmissä. Tämä sisältää hyökkäyksiä esim. haavoittuviin palveluihin, ohjelmien ylivuoto-ongelmiin, konekohtaisiin hyökkäyksiin, kuten käyttäjätunnusten saaminen, luvattomiin kirjautumisiin, pääsyn arkaluontoisiin tiedostoihin ja mahdollisten virusten/matojen ajaminen kohdekoneessa.

IDS-järjestelmä koostuu useista komponenteista: Sensoreista, jotka havaitsevat erilaisia hyökkäyksiä. Konsolista, jolla monitoroidaan hälytyksiä ja hallitaan sensoreita. Palvelimesta, joka tallentaa sensoreiden havaitsemat tapahtumat tietokantaan ja hälyttää niistä. Monissa yksinkertaisissa IDS-järjestelmien toteutuksissa nämä komponentit ovat samassa laitteessa tai koneessa. /8/

Passiivisten järjestelmien ero reagoiviin järjestelmiin

Passiivisissa järjestelmissä tunkeutumisen havaitsemisjärjestelmän sensorit havaitsevat mahdolliset tietoturvauhat, tallentavat tiedot hyökkäyksestä ja lähettää hälytyksen konsolille ja/tai järjestelmänvalvojalle. Kun taas reagoivassa järjestelmässä IPS reagoi epäilyttävään toimintaan lopettamalla hyökkääjän yhteyden tai ohjelmoimalla palomuurin estämään verkkoliikenteen haitallisesta lähteestä. Tämä voi tapahtua joko automaattisesti tai käsin.

IDS:n hyviä puolia on se, että se pystyy havaitsemaan myös hyökkäykset sisäverkon puolelta, jota taas palomuurit eivät havaitse. /8/

2.5 Palvelunestohyökkäys (Denial-of-Service)

Palvelunestohyökkäys (Denial of Service) on yritys saada tietokone sellaiseen tilaan, ettei sen palveluihin pääse käsiksi. Syyt DoS-hyökkäyksiin tiettyyn kohteeseen voivat vaihdella, mutta yleensä se käsittää sinnikkäitä ja pahansuopia yrityksiä estää Internet sivuston tai palvelun toimimasta kunnolla tai ollenkaan, väliaikaisesti tai ennalta määräämättömäksi ajaksi. DoS-hyökkäysten kohteina ovat yleensä sivustot tai palvelut, jotka ovat sijoitettuna suurille web-palvelimille.
Yleisin tapa on tukkia kohteen tietoliikennekanavat (saturating) ulkoisten yhteyspyyntöjen avulla. Kohde vastaa hyvin hitaasti, tai ei voi vastata ollenkaan. Tämä johtaa siihen, ettei palvelua voida käyttää. Yleensä DoS-hyökkäykset on toteutettu seuraavilla tavoilla: /21/

- Pakottaa hyökätyn kohteen tietokoneet nollautumaan tai käyttämään resursseja niin, ettei sille tarkoitetut palvelut enää toimi
- Häiritä yhteyttä käyttäjien ja kohteen välillä siten, ettei yhteyttä voida käyttää.
- ”Floodata” (Flooding) eli tukehduttaa verkko, estäen oikean verkkoliikenteen toimimisen.
- Häiritä palvelinta lähettämällä enemmän pyyntöjä kuin palvelin pystyy käsittelemään estäen samalla pääsyn palveluun.
- Estää tiettyä henkilöä pääsemästä palveluun.
- Häiritä tietyn palvelun saatavuutta järjestelmille tai henkilöille.

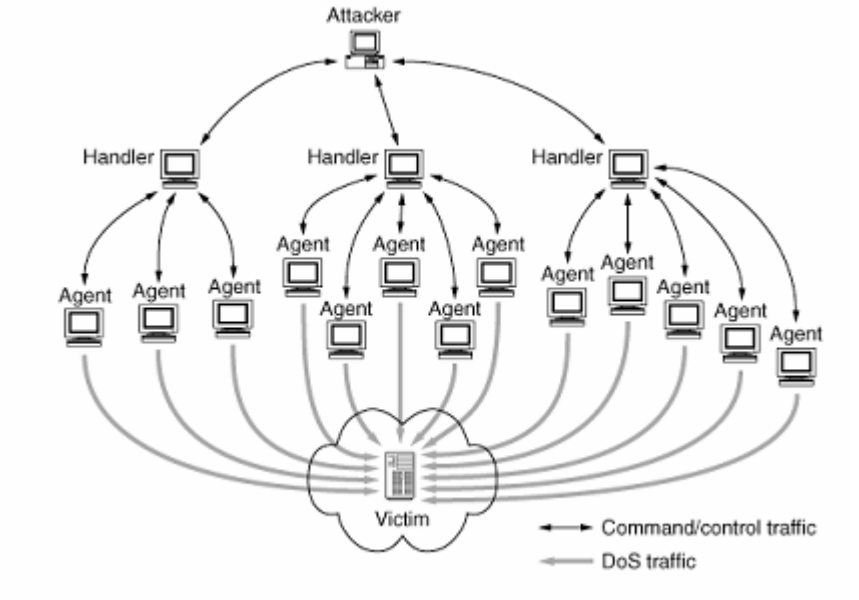
Hyökkäyksiä voidaan myös kohdistaa melkein mihin tahansa, esim. reitittimiin, web-

sähköposti- tai DNS-palvelimiin (Domain Name Server). DoS hyökkäys voidaan tehdä monella eri tavalla. Seuraavaksi on esitelty neljä perustyyppiä hyökkäyksistä: /21/

- Resurssien kulutus, esim. Internet-yhteyden, levytilan tai prosessorin käyttö.
- Konfiguraatiotietojen häirintä, esim. reititystiedot.
- Tilatietojen häirintä, esim. TCP-istunnon ennenaikainen katkaiseminen.
- Fyysisten verkkokomponenttien häirintä.

DoS hyökkäys voi sisältää myös haittaohjelman suorittamisen saadakseen: /21/

- Prosessointitehon maksimiin, estäen samalla palvelun käytön.
- Käynnistämään koneen mikrokoodissa olevat virheet.
- Käynnistämään virheet käskyjen toiminnassa, joka johtaa koneen epästabiiliin tilaan tai kaatumiseen.
- Hyödyntää virheet käyttöjärjestelmässä, joka aiheuttaa ns. 'resurssien alijäämän' ja/tai 'roskaantumisen', jotka käyttävät kaikki saatavilla olevat resurssit estäen oikeiden töiden suorittamisen
- Käyttöjärjestelmän kaatumaan.

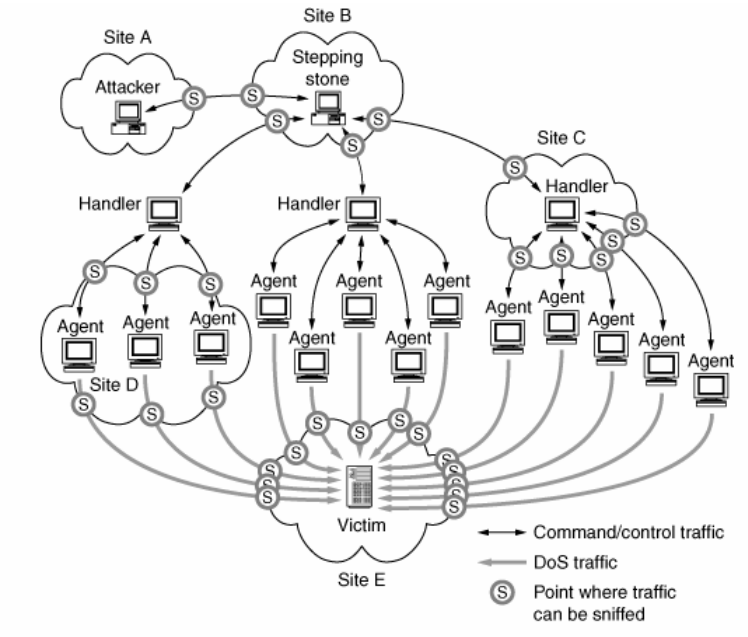


Kuva 3. DoS hyökkäyksen arkkitehtuuri, jossa hyökkääjä piilottaa identiteettinsä monien kerrosten avulla. /2/

2.6 Hajautettu palvelunestohyökkäys (Distributed Denial-of-Service attack)

Hajautetussa palvelunestohyökkäyksessä (DDoS) hyökkääjä voi käyttää kohteen tietokonetta hyökätäkseen toiseen kohteeseen. Tämä tapahtuu siten, että hyökkääjä hyödyntää järjestelmän haavoittuvuuksia tai heikkouksia ottaessaan tietokonetta haltuunsa. Hyökkääjä saattaa pakottaa tietokoneen lähettämään suuren määrän dataa Web-sivustoon tai johonkin tiettyyn sähköpostiosoitteeseen. Hajautetussa hyökkäyksessä tunkeutuja käyttää montaa eri tietokonetta käynnistääkseen palvelunestohyökkäyksen. DDoS-hyökkäyksen aikana monet saastuneet tietojärjestelmät hukuttavat kohteen verkkoyhteyden tai järjestelmän resurssit täysin. Nämä järjestelmät ovat saastuneet hyökkääjän toimesta monin eri tavoin. /23/

- Haittaohjelmat voivat sisältää DDoS-hyökkäysten mekanismeja, hyvä esimerkki tällaisesta haittaohjelmasta on MyDoom-mato, jonka DoS-mekanismi oli säädetty käynnistymään tiettyyn päivään ja kellonaikaan. /21/
- Järjestelmä voi saastua troijalaisella, antaen tunkeutujalle mahdollisuuden ladata tietokoneelle ns. ”haamuohjelman” (zombie agent), joka lähettää käyttäjän huomaamatta esim. roskasähköposteja (e-mail spam). Tunkeutujat voivat myös murtautua järjestelmään käyttäen automatisoituja työkaluja. Ne hyödyntävät tietoliikennettä tarkkailevien ohjelmien virheitä. Toiminta hoidetaan yleensä tarkkailemalla etäkoneita. /21/



Kuva 4. (DDoS) Hajautetun palvelunestohyökkäyksen arkkitehtuuri. /2/

Miksi DDoS on vaikea ongelma

Yleensä DDoS hyökkäysten kohdekoneet joko kaatuvat, lukkiintuvat (deadlock) tai osa tärkeistä resursseista on käytössä. DDoS hyökkäykset tarvitsevat vain muutaman paketin ollakseen tehokas ja siten hyökkäys voidaan käynnistää yhdestä tai hyvin vähäisestä määrästä zombi-agentteja. Kun taas floodauksessa (paketteja lähetetään koko ajan viemällä lähes kaikki järjestelmän resurssit), resurssit ovat käytössä niin kauan kuin hyökkäys kestää. Täten floodaus tarvitsee jatkuvaa pakettivirtaa kohteeseen ollakseen tehokas.

Hyökkäykset kohdistuvat yleensä protokollaan tai kohdejärjestelmän tietoturva-aukkoihin, joiden kautta hyökkääjä voi tunkeutua järjestelmään aiheuttaen erinäistä haittaa palvelulle tai kaataen koko järjestelmän toimimattomaksi. Tietoturvamekanismeja parannellaan koko ajan tällaisia uhkia vastaan, kuten ohjelmistojen paremmalla ohjelmointistandardoinnilla ja päivittämällä ohjelmistot saadaan tätä ongelmaa huomattavasti pienennettyä. Kuitenkin kekseliäät hyökkääjät voivat silti ohittaa suojaukset etsimällä tietoturva-aukkoja uusista ohjelmistojen päivityksistä. Tämäntyyppinen “hienostunut” hyökkäys vaatii kuitenkin paljon taitoa ja paneutumista hyökkääjältä ja siten ollen hyvin harvinaista, koska palvelun estäminen on usein paljon helpompaa.

Koska floodauksessa ei tarvita tiettyjä paketteja, voivat hyökkääjät kehittää vaihtelevaa liikennettä, joka sekoittuu oikeaan liikenteeseen. Tällöin voidaan myös käyttää IP-spoofausta (IP-osoitetta vaihdellaan ajoittain) saadakseen vielä suuremman vaihtelun verkkoliikenteeseen ja täten piilottaa paremmin hyökkäyksen alkuperän. Näin kohdejärjestelmä havaitsee suuren määrän palvelupyyntöjä (mahdollisesti myös oikeita) ja yrittää palvella kaikkia, siten käyttäen kaikki resurssit pudottaen samalla ylimääräiset palvelupyynnöt, joita ei pystytä käsittelemään. Seuraavat DDoS:n floodauksen tunnusmerkit tekevät hyökkäyksestä tehokkaan ja hyvin vaikean ongelman suojautua niiltä:

- Yksinkertaisuus. On olemassa monia DDoS-ohjelmia, joita voidaan hyvin helposti ladattavissa ja käytettävissä. Tällaiset ohjelmat ovat erittäin yksinkertaisia, joista osa on ollut olemassa vuosia silti kehittäen tehokkaita hyökkäyksiä hyvin pienellä vaivalla.
- Verkkoliikenteen vaihtelevuus. Samankaltaisuus oikean ja hyökkäävän

verkkoliikenteen tekee tunkeutumisen havaitsemisesta erittäin hankalaa.

- IP-spoofaus. Tämä tekee hyökkäävän verkkoliikenteen näyttämään siltä, että ne tulisivat monilta oikeilta käyttäjiltä.
- Suuret määrät verkkoliikennettä. Suuret määrät hyökkäävää verkkoliikennettä ei vain tukahduta kohteen resursseja vaan myös tekee siitä vaikeasti seurattavaa. Suurella pakettiliikenteellä turvamekanismit, kuten palomuurit voivat vain tehdä yksinkertaisen pakettitarkastuksen.
- Monet zombi-tietokoneet. DDoS-hyökkäyksen tehokkuus piilee siinä että se käyttää suurta joukkoa zombi-koneita eri puolilla Internetiä. Hyökkääjä voi lähettää hyvin pienen määrän paketteja jokaiselta zombi-koneelta saadakseen kaatumaan jopa suuret tietoverkot. Tällaisia hyökkäyksiä vastaan on hyvin vaikea saada tieto siitä, mistä hyökkäys alun perin on lähtöisin.
- Internet-topologiassa olevat heikot lenkit. Nykyisellä Internet-topologialla on vain muutamia suurkaistaisia yhteyksiä, jotka jakavat liikenteen muualle Internetiin. Nämä yhteydet ovat määritelty käsittelemään raskasta liikennettä, mutta jos ne kaadetaan hyökkääjän toimesta, voivat seuraukset olla vakavia.

Näyttäisi siis siltä, että DDoS-hyökkäys olisi täydellinen rikos Internet-maailmassa. On olemassa runsaasti keinoja (ohjemat) ja rikoskumppaneita (zombi-koneet), joita on saatavilla helposti. Myös riittävä määrä vaihtelevaa liikennettä voi saada kohteen polvilleen käyttäen IP-spoofausta hyväkseen siten hyökkääjä voi piiloutua ja näin ollen saaden hyvin pienen riskin jäädä kiinni. /2/

Miten voi välttyä olemasta ”osa ongelmaa”?

Tähän ongelmaan ei ole tehokasta tapaa estää olemasta DoS- tai DDoS-hyökkäyksen uhri, mutta on olemassa tapoja, joilla tätä uhkaa voidaan vähentää: /22/

- Asenna ja pidä viruksentorjuntaohjelmisto ajan tasalla (päivitykset)
- Asenna palomuuri ja konfiguroi se rajoittamaan tulevaa ja lähtevää tietoliikennettä.
- Älä anna sähköpostiosoitettasi kaikkialle. Ottamalla sähköpostisuodattimet käyttöön voi vähentää ei-toivottua liikennettä.

Mistä tietää, että hyökkäys on käynnissä?

Häiriöt palvelussa eivät välttämättä aina ole seurausta DoS-hyökkäyksestä. Häiriöitä voi aiheuttaa laitteisto-ongelmat tai järjestelmän ylläpitäjät ovat suorittamassa huoltoa. Kuitenkin seuraavat oireet ***saattavat*** merkitä DoS- tai DDoS-hyökkäystä:

- Epätavallisen hidas verkon suorituskyky (tiedostojen avaaminen tai pääsy web-sivustoon).
- Käytettävyys tietyllä web-sivustolla.
- Kyvyttömyys päästä millekään web-sivustolle.
- Yllättävästi kasvaneet roskasähköpostien määrät.

Mitä tehdä, kun hyökkäys on käynnissä?

Edes silloin kun on tunnistanut tulleensa DoS- tai DDoS-hyökätyksi, on hyvin epätodennäköistä määrittää hyökkäyksen varsinaista kohdetta tai lähdettä.

- Jos huomaat, että et pääse käsiksi omiin tiedostoihin tai pääse millekään ulkoiselle web-sivustolle työkoneelta, ota yhteys verkon ylläpitäjiin. Tämä voi merkitä sitä, että koneellesi tai yrityksen verkkoon on hyökätty.
- Jos havaitset samanlaista ilmiötä kotikoneella, ota yhteys Internet palvelun tarjoajaasi (ISP, Internet Service Provider). Jos ongelmia löytyy, ISP voi neuvoa, miten edetä tilanteessa.

2.7 Väärät positiiviset ja negatiiviset (False positive & False negative)

Termi 'väärä positiivinen' (false positive) on hyvin laaja ja hieman epämääräinen termi, sillä se kuvaa tilannetta, jossa NIDS-laite (Network Intrusion Detection System) käynnistää hälytyksen kun se huomaa pahansuopaa liikennettä tai hyökkäyksen. Muut yleiset termit tätä tilannetta kuvaamaan ovat 'väärät hälytykset' (false alarms) ja 'harmittomat hälytykset' (benign trigger). Väärä hälytys on paras termi kuvaamaan tällaisen tilanteen luonnetta, koska väärä positiivinen antaa kuvan siitä, että IDS-järjestelmä olisi pohjimmiltaan viallinen. Harmittomat hälytykset taas antavat kuvan siitä, että vääriä positiivisia ei olisi olemassakaan. Väärät hälytykset

ovat ongelmallisia, koska ne ovat aiheettomia hälytyksiä ja vähentävät todellisten hälytysten merkitystä ja tärkeyttä.

Termiä väärä negatiivinen käytetään kuvaamaan IDS-järjestelmän kyvyttömyyttä tunnistaa todellisia hälytyksiä tietyissä olosuhteissa. Toisin sanoen, pahansuopaa toimintaa ei havaita ja hälytetä.

Väärien hälytysten kategoriat:

Väärät hälytykset voidaan jakaa muutamaan enemmän merkitsevään ja tarkempiin ryhmiin. Yleiset ryhmät joihin väärät hälytykset voidaan jakaa sisältää: /24/

- Taantumukselliset verkkoliikennehälytykset: Liikenne, joka on aiheutunut tapahtumista, jotka useimmin eivät ole pahansuopia. Esim. NIDS havaitsee ICMP-floodausta, jossa useat kohteet ovat pakettien saavuttamattomissa, tämä voi johtua laiterikosta jossakin kohtaa Internet ”pilveä”.
- Laitteistopohjaiset hälytykset: Hälytykset, jotka ovat lauenneet omituisista, tunnistamattomista paketeista, joita tietty verkkolaite tuottaa. Usein kuormituksenjakajat laukaisevat tämän tyyppisiä hälytyksiä.
- Protokollarikkomukset: Hälytykset ovat aiheutuneet tunnistamattomista verkkoliikenteistä. Usein nämä ovat aiheutuneet huonosti tehdyillä ohjelmilla.
- Todelliset väärät hälytykset: Hälytykset, jotka IDS on luonut ilman mitään syytä. Nämä ovat usein aiheutuneet IDS-ohjelmiston ohjelmointivirheistä.
- Pahansuovattomat hälytykset: Hälytys on luotu jonkin tapahtuman jälkeen, jolla ei ole pahansuopaista luonnetta.

Mikä on sallittu määrä vääriä hälytyksiä?

Riippuen verkkoliikenteestä ja IDS:n sijoittamisesta, IDS-sensori, jonka sääntöjä ei ole muokattu, vääriä hälytyksiä saa olla vain 10 % todellisten hälytysten määrään verrattuna. Kaikki tämän luvun ylittävät eivät ole hyväksyttäviä. Kyseenalaista on se, mitä voidaan pitää hyväksyttävänä lukuna vääristä hälytyksistä. Sääntöjen kunnollisella muokkaamisella voidaan sanoa että 60 % tai parempi luku todellisia hälytyksiä on mahdollisesti normaaleissa oloissa. /24/

Väärät positiiviset (Väärät hälytykset, False positives)

Väärät positiiviset ovat hälytyksiä, jotka ovat lähtöisin harmittomista lähteistä, mutta tunnistuvat tunkeutumisyrityksiksi ja täten voidaan jättää huomioimatta. Paras merkki vääristä hälytyksistä on se, että sääntö havaitsee niitä paljon enemmän kuin normaalisti. Seuraavat kohdat auttavat tunnistamaan vääriä hälytyksiä: /24/

- Tarkista lähteen ja kohteen IP-osoitteet
- Tarkista sääntö, joka havaitsee hälytykset. Sääntö voi olla yksinkertaisesti liian yleinen tai liian laaja ja täten havaitsee oheisliikennettä. Tarkista myös sormenjäljen osat, jotka saattavat osua hyökkäävään liikenteeseen.
- Tarkista fyysisesti koneet, jotka tuottavat hälytyksiä. Voi olla että sisäverkossa on kone, jonka verkkoliikenne aiheuttaa vääriä hälytyksiä. Voi myös olla, että sisäverkon koneet lähettävät kyselyjä internetiin ja niiden vastaukset on havaittu hälytykseksi.
- Tarkista etteivät koneet, joihin hälytykset vaikuttavat, ole vaarantuneet hyökkäyksistä.
- Tarkista hälytykset ja varmista että ne voidaan jättää huomioimatta. Joskus väärät hälytykset saattavat ilmaista jonkin tietoturvaan liittymättömän ongelman kuten: viallinen laitteisto tai väärin konfiguroidut verkkolaitteet
- Älä ota sääntöä pois käytöstä tai jätä huomioimatta hälytyssarjaa ilman, että tutkit sen pääsyytä.

Havaitsematta jäävät hälytykset (False negatives)

Väärät negatiiviset ovat hyökkäyksiä, joita järjestelmä ei huomaa ja täten ovat hyvin suuri ongelma. Vääriä negatiivisia on vaikea määrittää, koska ei välttämättä tiedä mitä tehdä. On kuitenkin olemassa toimenpiteitä, joilla saadaan vähennettyä väärien negatiivisten määrää lisäämättä väärien hälytysten määrää. Joskus muut komponentit, jotka kuuluvat hyökkäyksien puolustukseen näyttää merkkejä hyökkäyksestä, joita IDS ei havaitse.

Yleiset syyt vääriin negatiivisiin, ja malleja miten niiltä vältytään

- Liikenteen salaus, IDS-sensorin oikea sijoittaminen on hyvin tärkeää. Salattu

liikenne ei aiheuta hälytyksiä, koska sormenjälkien näytteet eivät täsmää. Tämän estämiseksi on hyvä sijoittaa siten, että sensori näkee liikenteen salaamattomana. Tässäkin tapauksessa voi olla liikennettä, jota ei voi seurata.

- Verkon konfiguraatio-ongelmat. Virheet sensorin sijoittamisessa tai verkon rakenteen monimutkaisuus voi aiheuttaa, ettei sensori kaappaa kaikkea liikennettä. Tai jos on monta reittiä Internetiin, sensori ei välttämättä näe kaikkia lähtevää ja tulevaa liikennettä.
- Snort on sormenjälkiin perustuva IDS, jos hyökkäys tapahtuu ja siihen ei ole sormenjälkeä, pääsee hyökkäys läpi huomaamatta. Siksi on tärkeää pitää sormenjäljet jatkuvasti ajan tasalla.
- Virheelliset sormenjäljet. Voi olla, että sormenjälki on kirjoitettu väärin ja täten ei seuraa oikeaa sääntöä. Muunnelma hyökkäyksestä ei välttämättä aiheuta hälytystä samalla säännöllä kuin sen aikaisempi versio.
- Huono yhteydenpito järjestelmän muutoksista. Hyvin usein yrityksellä on eri osastot, jotka hoitavat eri asioita. Jos näiden osastojen välillä on huono yhteydenpito, voi syntyä ongelmia.
- Sensorin hallinnointiongelmat. Sensorin pääkäyttäjä saattaa ottaa pois käytöstä tärkeän säännön vähentääkseen vääriä hälytyksiä. Tämä voi aiheuttaa sen, että jokin hyökkäys voi päästä läpi huomaamatta. Järjestelmä saattaa kuormittua liikaa ja siten osat paketeista tippua pois tarkastuksesta.

3 SNORT

3.1 Yleistä Snortista

Snort on Network Intrusion Detection System (NIDS)-työkalu. Alkujaan Snort oli ohjelma, joka dekoodasi tcpdumpin tuottamaa dataa ihmisen luettavaan muotoon ja jota ei ollut tarkoitus julkaista. Sittemmin siitä on kehittynyt paljon käytetty ja toimiva tunkeutumisten havaitsemistyökalu.

Snort voidaan asentaa useisiin tietokonearkkitehtuureihin, kuten:

- i386
- Sparc
- Motorola 68000/Power PC
- Alpha

Snort tukee seuraavia käyttöjärjestelmiä:

- Linux
- OpenBSD
- FreeBSD
- Solaris
- HP-UX
- AIX
- Mac OS X
- Win32

Tämän kaiken mahdollistaa pakettikaappari-kirjasto libpcap, joka on käännetty useille käyttöjärjestelmille. Snort käyttää tätä kirjastoa kaappaamaan verkkoliikennettä. Lisäksi Snort perustuu avoimeen lähdekoodiin (Open source), joten ohjelmaa voi vapaasti muokata. Tämän ansiosta uudet tunkeutumis-sormenjäljet voivat olla jaossa jo muutaman tunnin kuluttua. Kuka tahansa voi tehdä uuden tunkeutumis-sormenjäljen, jota käyttäjät voivat testata, muokata ja laittaa jakoon hyvinkin nopeasti kaupallisiin tuotteisiin verrattuna. Siksi Snort omaakin maailman laajimman ja kattavimman tietokannan tunkeutumis-sormenjäljistä. Vuonna 2003 tietokannassa oli noin 1500 sormenjälkeä ja nykyisin jo yli 10000 kappaletta. Sormenjäljet tulevat Snortin mukana sitä asennettaessa. Snortia voidaan ajaa kolmessa eri käyttötilassa: pakettinuuskija, pakettikirjaaja ja NIDS. /4/

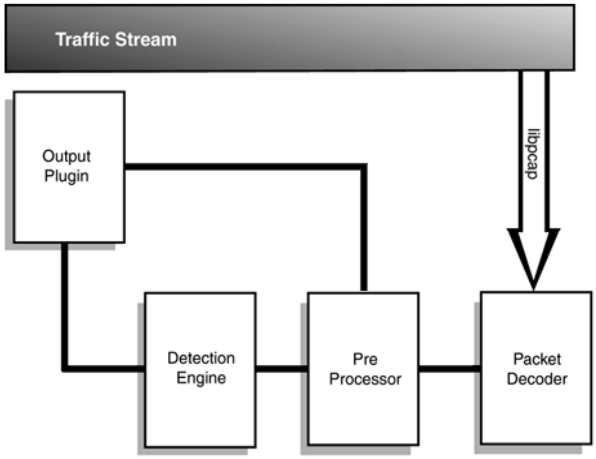
3.2 Snortin tunkeutumisen havaitsemismetodit

Snort käyttää tunkeutumisien havaitsemiseen kahta metodia, sormenjälkien ja poikkeamien havaitsemista. Sormenjälkien havaitseminen perustuu siihen, että jollakin tunkeutumisella on jokin yleinen tunnusmerkki, näitä voidaan havaita tarkastelemalla verkossa kulkevia paketteja. Snort tekee Sormenjäljistä sääntöjä, jotka taasen ladataan havaitsemisytimeen (Detection Engine), joka on vastuussa sormenjälkien havainnoinnista. Säännöillä voidaan tutkia epäilyttävää kuormaa (payload), pakettiotsikoita ja sekä myös tiettyjä protokollaelementtejä esim. HTTP URL:ssa ".ida?"-merkkijonoa. Kaikilla tunkeutumisilla ei välttämättä ole erottuvaa

sormenjälkeä, jolloin tarvitaan esiprosessori, joka havaitsee tämäntyyppiset tunkeutumiset. Kuitenkin voidaan tietää tietyn liikenteen olevan epänormaalia, esim. Web-palvelimelle tehdyt TCP-kyselyt muihin portteihin kuin 80 tai 443 luokitellaan heti epänormaaliksi. /4/

3.3 Komponentit ja datavuo

Snort voidaan jakaa viiteen pääkomponenttiin, jotka ovat kriittisiä tunkeutumisen havaitsemisessa (Kuva 5): Pakettikaappari (libpcap), Pakettidekooderi (Packet Decoder), Esiprosessori (Pre Processor), Havaitsemisydin (Detection Engine) ja Raportointi (Output Plugin). Alla oleva kuva esittää datavuon, miten verkosta kaapatun paketin kulkua Snortissa. /4/



Kuva 5. Komponentit ja datavuo /4/

3.3.1 Pakettikaappari (libpcap)

Saadakseen paketit esiprosessoreille ja siitä havaitsemisytimelle, ne pitää ensin kaapata verkkoliitynnästä muuttamattomina (Traffic Stream) ja ohjaa ne pakettidekooderille. Snort käyttää libpcap-kirjastoa tähän tehtävään, koska Snortilla ei vielä ole omaa pakettikaappaukseen tarkoitettua työkalua.

3.3.2 Pakettidekooderi (Packet Decoder)

Pakettidekooderi tallettaa pakettien protokollaelementit Snortin sisäiseen tietorakenteeseen, jonka jälkeen tiedot ovat valmiita analysoitavaksi esiprosessorissa. Snort purkaa erikseen linkki-, verkko- ja siirtokerroksen protokollat. Pakettidekooderi ohjaa datan edelleen esiprosessorille. /4/

3.3.3 Esiprosessori (Pre Processor)

Esiprosessorit ovat liitännäisiä, joiden avulla Snort on laajennettavissa. Paketit ajetaan kaikkien esiprosessoreiden läpi, koska jotkut hyökkäykset havaitaan vasta usean esiprosessorin jälkeen. /4/

3.3.4 Havaitsemisydin (Detection Engine)

Havaitsemisytimellä on kaksi ensisijaista toimintaa: sääntöjen parsiminen ja sormenjälkien havaitseminen. Sormenjäljet koostuvat säännöistä, jotka ladataan käynnistyksen yhteydessä. Säännöt luetaan rivi riviltä ja muutetaan Snortin sisäiseksi tietorakenteeksi. Tietoliikenne kulkee sääntöjen läpi, joita voidaan priorisoida ja järjestellä tarpeen mukaan. Sääntö on jaettu otsikko- ja optio-osaan. Otsikko-osa asettaa säännölle ehtoja, kuten protokollan, lähteen ja kohteen IP-osoitteet, portin ja tallennus-muodon. Optio pitää sisällään sormenjäljen, hälytysprioriteetin ja lyhyen selostuksen tunkeutumisesta. /4/

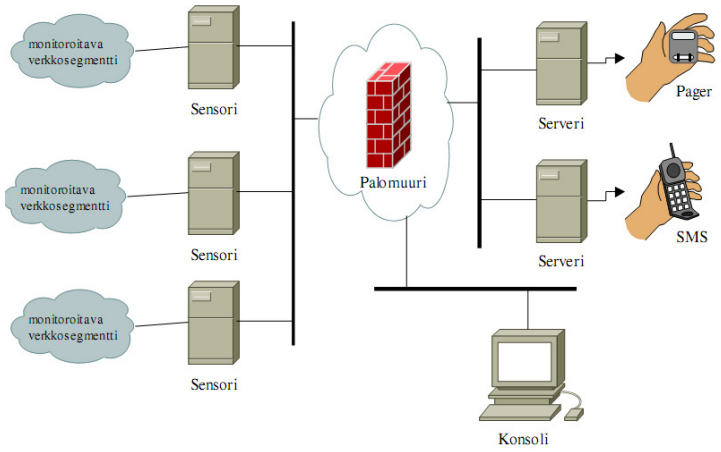
3.3.5 Raportointi (Output Plugin)

Raportoinnin avulla hälytykset voidaan antaa käyttäjälle, joko suoraan esiprosessorilta tai havaitsemisytimen tunnistettua tunkeutumisen. Snortilla on useita raportointitapoja joita voidaan suorittaa samanaikaisesti, näitä tapoja on kaikkiaan 12 kappaletta. Näitä formaatteja ovat esimerkiksi: CSV (Comma Separated Value), XML (Simple Network Markup Language, SNML) Syslog, tietokannat (MySQL, Postgresql, Oracle) ja Unified. Raportointi voi olla pullonkaulana, sillä Snort

prosessoi paketit nopeasti, mutta hidastuu huomattavasti yrittäessään kirjoittaa hitaaseen tietokantaan tai verkon yli muualle. Unified binäärinen lokiformaatti ja Barnyard tietokanta on kehitetty nopeisiin verkkoihin. Tällöin Snort kirjoittaa binääristä lokia tiedostoon suurella nopeudella ja Barnyard välittää tiedot tietokantaan maksiminopeudella. Täten Snort ei hukkaa yhtään verkossa kulkevaa pakettia raportoinnin hitauden takia. /4/

3.4 Kolmitasoinen arkkitehtuuri

Snort NIDS-järjestelmä voidaan rakentaa 3-tasoiseksi arkkitehtuuriksi, jolloin järjestelmässä esiintyy Sensori-, Palvelin- ja Konsolitaso. Eri tasoilla on eri tietoturvallisuuden tarpeet, joten ne erotetaan toisistaan palomuureilla. /4/



Kuva 6. Kolmitasoinen arkkitehtuuri /5/

3.4.1 Sensori

Sensorilla ajetaan Snort-ohjelmaa. Sensori kaappaa paketit verkosta, tulkitsee niitä ja ohjaa hälytystiedot palvelimelle. Sensori täytyy sijoittaa samaan verkkosegmenttiin tutkittavan liikenteen kanssa ja siinä ajetaan vain Snortia sekä sen vaatimia tukiohjelmia. Sensorissa on yleensä kaksi verkkoliityntää. Ensimmäinen verkkoliityntä on pakettien kaappausta varten, tässä liitynnässä ei ole IP-osoitetta (Promiscuous mode) joten liityntä toimii ”piilossa” muulta verkolta. Toinen hallintayhteyttä varten, jolla voidaan hallita sensoria ja on sijoitettu eri verkkoon kuin kaappausverkkoliityntä,. Hallintaverkko pitää suojata ulkoverkolta palomuurin avulla. Sensoreissa on suositeltavaa käyttää vakaita ja turvallisia Linux- tai BSD-

käyttöjärjestelmiä. ohjelmistot: Snort, libpcap, OpenSSL, OpenSSH, MySQL-client. /4/

3.4.2 Palvelin

Palvelimen ensisijaisena toimintona on koota hälytysdataa sensoreilta ja rakentaa siitä tietokantaa. Serverin tehtävä on esittää hälytystiedot luettavassa muodossa esim. BASE-käyttöliittymän avulla. BASE vaatii Web-palvelimen. palvelimen tehtävä on myös luoda reaaliaikaisia hälytyksiä ylläpitäjille. Palvelimien verkkosegmentti tulee suojata ulkoverkolta palomuurin avulla. Tarvittavat ohjelmistot: OpenSSL, OpenSSH, Web-palvelin Apache, BASE ja MySQL-tietokanta. /4/

3.4.3 Konsoli

Kolmas taso on konsoli, jolla ei ajeta Snortia, vaan sillä tutkitaan hälytysdataa ja hallitaan sensoreita ja palvelimia. Minimivaatimuksena konsolissa tulee olla Web-selain, jonka avulla toimintoja suoritetaan. Hälytystietoja tutkiessa konsolilta otetaan yhteys palvelimen BASE:n. Ylläpitäjän tehtäväksi jää reagoida BASE:n näyttämiin hälytyksiin. Lisäksi tarvitaan seuraavat ohjelmistot: OpenSSH ja Web-selain. /4/

3.5 Vaihtoehtoiset arkkitehtuurit

Snort voidaan asentaa myös 1-tasoiseksi arkkitehtuuriksi. 1-tasoisessa arkkitehtuurissa sensori ja palvelin ovat samalla koneella, tällainen kone on ns. hybridi. Hybridi-arkkitehtuuri on huonosti skaalautuva ja koneen suorituskyky voi olla rajoitus Snortin toiminnalle, koska sama kone joutuu myös tallentamaan tiedot tietokantaan, joka kuluttaa koneen resursseja. Hybridi on myös melko tietoturvaton, sillä hälytystiedot sijaitsevat samassa koneessa, joka sijaitsee ”turvattomassa” verkon osassa, jolloin hyökkääjä koneelle tunkeutuessaan voi päästä käsiksi myös koko hälytystietokantaan. Suositeltavaa onkin asentaa Snort aina 3-tasoiseksi todelliseen tietojärjestelmään ja käyttää hybridiä ainoastaan Snortin toimintaan tutustuessa. /4/

4 SNORT – KÄYTTÖ, SÄÄTÖ, SÄÄNNÖT JA REAALIAIKAISET HÄLYTYKSET

4.1 Käyttötilat

Snortin mahdollisia komentorivi-optioita on lukematon määrä eivätkä ne kaikki sovi yhdessä käytettäviksi. Komentorivi-optioiden ohjeet löytyvät joko Snortin sivuilla olevasta oppaasta /18/ tai Linuxilla komennolla *man snort*. Snortia voidaan käyttää kolmessa eri tilassa; pakettinuuskija (Sniffer), pakettikirjaaja (Packet Logger) ja Network Intrusion Detection System (NIDS). /5/

4.1.1 Pakettinuuskijatila (Sniffer)

Pakettinuuskijatilassa Snort kaappaa paketit verkosta ja tulostaa pakettitiedot jatkuvana virtana. Snort tulostaa näytölle (-v) TCP/UDP/ICMP ja IP otsikkokentät. Sovelluksen data (-d) ja linkkikerroksen otsikkokenttä (-e) saadaan tulostumaan komennolla *snort -vde*. /9/

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

08/16-14:21:19.409530 0:13:72:8B:8D:8F -> 0:F:EA:1D:DF:C1 type:0x800 len:0x86
192.168.10.84:4978 -> 192.168.10.59:22 TCP TTL:128 TOS:0x0 ID:57816 IpLen:20 Dgm
Len:120 DF
***AP*** Seq: 0x80619DE3  Ack: 0xA15E2497  Win: 0x4470  TcpLen: 20
D4 CC C7 7F 34 90 1C 05 87 60 3B 9F E9 F0 F8 12  ....4....`;.....
47 18 82 05 59 AE DD CD 09 F7 A4 1E 52 83 E7 46  G...Y.......R..F
5F B8 A9 38 AF DB 59 DB C7 ED D9 AF 47 D4 34 72  _..8..Y.....G.4r
D1 40 A1 05 62 87 6C EF 1C 02 D7 C2 72 70 6F 10  .@..b.l.....rpo.
2B B2 E2 51 B9 80 A4 7B D4 2D 9B 48 48 99 B7 DD  +..Q...{.-.HH...

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Kuva 7. Snortin kaappaama paketti

4.1.2 Pakettikirjaajatila (Packet logger)

Pakettikirjaajatilassa Snort tallettaa paketit levylle joko IP-osoitteiden mukaiseen hakemistorakenteeseen ASCII-teksti muodossa tai yhteen tiedostoon binäärisessä tcpdump-muodossa. Binäärinen talletusmuoto sopii paremmin suurille liikennemäärille, jolloin suorituskyky voi olla pakettien tallentamisessa rajoittava tekijä. Esim. komennolla *snort -bl ./log* Snort tallentaa pakettitiedot yhteen

tiedostoon ”log”-hakemistoon binäärisessä tcpdump-muodossa. /9/

4.1.3 Tunkeutumisten havaitsemistila (NIDS)

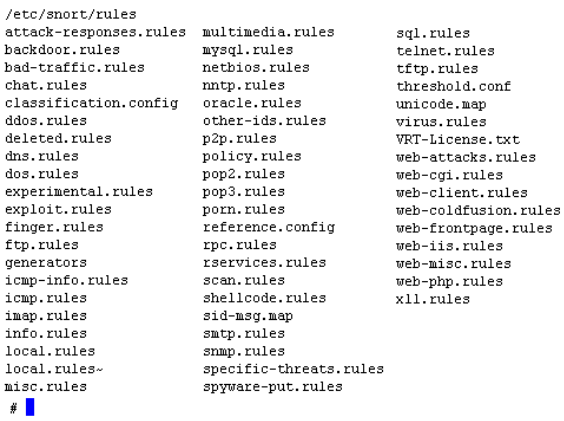
NIDS-tila on käytetyin ja se on kaikista muokattavin. Siinä Snort tallettaa sääntöihin osuvat paketit levylle ja muodostaa käyttäjän määrittelemän hälytystiedoston. NIDS-tilassa Snort toimii siis viritettynä pakettinuuskijana, joka havaitsee paketin epäilyttävät tai pahansuovat piirteet ja tuottaa hälytyksen tämän perusteella. Esim. komennolla snort -dc snort.conf -l ./log sääntöihin osuvat paketit tallentuvat ASCII-tekstinä ”log”-hakemistoon IP-osoitteiden mukaiseen hakemistorakenteeseen. /9/

4.2 Säätäminen

Yleisin tapa Snortin säätämiseksi on muuttaa tulevan tiedon määrää ja laatua. Verkkolaitteiden kanssa voidaan esim. multicast-liikenne sulkea verkkosegmentistä, jolloin Snortiin tulevien pakettien määrää saadaan vähennettyä. Toinen tapa muuttaa Snortin toimintaa on säätää sen esiprosessoreita, näitä muokkaamalla saadaan Snortin toimintaa muutetuksi ennen sääntöihin vertailua. Kolmas ja tärkein tapa Snortin säätämisessä on tunkeutumisien havaitsemissääntöjen muokkaaminen, näitä muuttamalla saadaan esim. väärien hälytysten ja havaitsematta jäävien tunkeutumisten määrää vähennettyä. /4/

4.3 Säännöt

Snortin mukana on yli 10000 valmista sääntöä tunkeutumisen havaitsemiseen. Säännöt voidaan kuvata joko suoraan snort.conf tiedostossa tai sitten siinä määritellään sääntötiedostot sisältävä hakemisto ja sieltä mukaan otettavat sääntötiedostot.



Kuva 8. Snortin sääntötiedostot sisältävä hakemisto

Snortin tunkeutumisen havaitsemiseen käyttämät säännöt muodostuvat kahdesta osasta; säännön otsikko-osasta (Rule Header) sekä säännön optio-osasta (Rule Option). Otsikko-osa määrittelee havaitsemisen jälkeisen toiminnan sekä tarkkailtavan protokollan, tarkkailun suunnan sekä IP-osoitteet ja portit. Optio-osa määrittelee tarkkailtavan uhan tunnusmerkin sekä havaitsemisen prioriteetin. Optio-osan tunnusmerkkiosuus sisältää yhden tai useamman optioavainsanan. Kun Snort huomaa sääntöön osuvan paketin, se voi toimia kolmella tavalla;

- Hälytys ja paketin tallentaminen (alert)
- Pelkkä paketin tallentaminen (log)
- Jätä paketti huomioimatta (pass)

Lisäksi Internetistä löytyy valmiita sääntöjä ladattaviksi /17/. Nämä säännöt voidaan ottaa joko suoraan käyttöön tai sitten niitä voidaan muokata enemmän omaan verkkoon sopiviksi. /4/

4.4 Priorisointi

IDS:ää käytettäessä on tarve luokitella sen tuottamat hälytykset ja laittaa ne tärkeys-järjestykseen, sillä kaikkia hälytyksiä ei tarvitse tutkia samalla tarkkaavaisuudella ja huolellisuudella. Priorisointeja on kolmenlaisia:

- ei priorisointia
- kovakoodattu (esim. high, medium ja low)
- muunneltava priorisointi

Snort käyttää näistä viimeistä ja sillä on 32 ennalta määriteltyä hälytyskategoriaa. Kullekin Snortin havaitsemalle tunkeutumiselle määritellään hälytysprioriteetti, ja sen avulla saadaan lähetetyksi eritasoisia hälytyksiä. Prioriteetti määritellään classification.config tiedoston avulla tai sääntöön liitettävän priority-option kanssa. Snortin mukana tuleva valmis classification.config tiedosto näyttää seuraavalta: Erilaisia hälytyksiä on jaoteltu kolmeen hälytyskategoriaan, 1-, 2- ja 3-tason hälytyksiin. /4/

```
# $Id: classification.config,v 1.11 2003/10/20 15:03:03 chrisgreen Exp $
# The following includes information for prioritizing rules
# Each classification includes a shortname, a description, and a default
# priority for that classification.
# This allows alerts to be classified and prioritized.  You can specify
#  what priority each classification has.  Any rule can override the default
#  priority for that rule.
#
# config classification: shortname,short description,priority
#
config classification: not-suspicious,Not Suspicious Traffic,3
config classification: unknown,Unknown Traffic,3
config classification: bad-unknown,Potentially Bad Traffic, 2
config classification: attempted-recon,Attempted Information Leak,2
config classification: successful-recon-limited,Information Leak,2
config classification: successful-recon-largescale,Large Scale Information Leak,2
config classification: attempted-dos,Attempted Denial of Service,2
config classification: successful-dos,Denial of Service,2
config classification: attempted-user,Attempted User Privilege Gain,1
config classification: unsuccessful-user,Unsuccessful User Privilege Gain,1
config classification: successful-user,Successful User Privilege Gain,1
config classification: attempted-admin,Attempted Administrator Privilege Gain,1
config classification: successful-admin,Successful Administrator Privilege Gain,1
```

Kuva 9. classification.config tiedosto /5/

4.5 Reaaliaikaiset hälytykset

Snort on tehty suorittamaan yksi tehtävä, ja se tekee merkittävän työn tunkeutumisia havaitessaan. Kaikki muu tunkeutumisten havaitsemisen lisäksi on jätetty IDS:stä vastaavan henkilön harteille, kuten reaaliaikaisten hälytysten toteuttaminen. /4/

5 OHJELMISTOJEN ASENNUS

Alusta, jota käytettiin, oli Mandriva Linux 2007.1 versio. Tälle koneelle asennettiin vain tarvittavat ohjelmistot. Esim. Snort-ohjelmisto, MySQL, Web-palvelin, kääntäjät (C, C++, ym.), graafinen käyttöliittymä.

Monet ohjelmat, jotka Linuxiin löytyi, olivat suoraan RPM-pakettina. Tällaisen paketin voi suorittaa komentoriviltä komennolla: ”rpm -iv [tiedosto].rpm”, ”urpmi [asennuspaketin nimi]” tai graafisesta ympäristöstä voidaan tuplaklikata pakettia, jolloin ohjelma asentaa itsensä.

- rpm → Paketti manageri ohjelma, jolla voidaan asentaa, poistaa, ym. ohjelmia rpm-paketeista
  - -i → Näyttää paketin tiedot, joka sisältää paketin nimen, versionumeron ja kuvauksen.
  - -v → Näyttää ruudulla mitä tapahtuu.
- urpmi → Hakee ja asentaa automaattisesti rpm-paketit ja sen tarvittavat ohjelmistot, joko asennusmedialta tai määritellystä asennuslähteestä. Esim. Mirrors.tp.spt.fi.

Jotkut paketit olivat Source-Tar-pakettina, jossa ohjelma on kääntämättömänä, eli se pitää ensin kääntää ja siitä asentaa. Tämä tapahtuu seuraavanlaisesti:

”tar zxvf [tiedosto].tar.gz” → Mene kansioon johon tar purki tiedostot (yleensä nimetty kuten tar-paketti ilman tar.gz päätettä) → ./configure (mahdolliset lisämääreet esim. Snortia asentaessa –enable-mysql) → make && make install

Jossa:

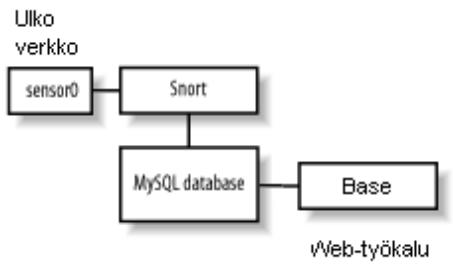
- tar → Purkuohjelma, joka on sisällytetty kaikkiin Linux-käyttöjärjestelmiin.
  - -z → Määrittää tar ohjelmalle että paketti, joka aukaistaan on pakattu Gzip:ksi.
  - -x → Purkaa paketin.
  - -v → Näyttää ruudulla mitä tapahtuu.
  - -f → Määrittää mikä paketti aukaistaan.
- ./configure → Valmistelee paketin lähdetiedostot niin, että sen voisi asentaa.
- make → Kääntää ohjelman.
- make install → Lopuksi ohjelma asennetaan koneelle.

Ohjelmat, jotka asennetaan koneelle, näistä lyhyt ohje miten kyseiset ohjelmat

asennetaan

- Snort&Snort-MySQL (Versio 2.7.0-1, IDS-ohjelmisto, Asennetaan RPM-paketeista /9/)
    - LibPcap (Versio 0.9.4-10, Asennetaan RPM-paketista /16/)
    - MySQL Support, Inline Support
- Oinkmaster (Versio 2.0, Automaattinen Sääntöjen lataustyökalu)
- MySQL (Versio 5.0.37–2, Tietokanta Snortin hälytyksiä varten /12/)
- Apache (Versio 2.2.4–6, Web-palvelin BASE:a varten /13/)
    - GD Support, OpenSSL Support, MySQL Support
- PHP(Asennetaan suoraan urpmi komentoja käyttäen)
    - GD Support, PEAR Support, OpenSSL Support, MySQL Support
- BASE(Versio 1.3.6 (Louise), Hallinta/Tarkkailu-ohjelma /14/)
    - ADOdb (Versio 4.91, Database abstraction library for PHP /15/)
    - Jpgraph/phplot, GD

5.1 Snort



Kuva 10. Kokoonpano

Snort:a asentaessa, /10/ videosta voi olla suuri apu, vaikkakin kyseinen asennusohje oli tehty Fedora 5 & 6:lle (Red Hat). Tässä videossa on kuvattu kohta kohdalta eri asennuksien vaiheet, jolla Snortin saa toimimaan perusasetuksilla. Seuraavissa kohdissa on lueteltu asetukset Snortiin (snort.conf). Jotkut asetuksista ovat valmiiksi asetettu:

- var HOME_NET [Oman sisäverkon osoite esim. 192.168.1.0/24]
- var EXTERNAL_NET !$HOME_NET (Määrittää ulkoverkon siten, ettei se ole koskaan sama kuin sisäverkon osoite)
- var RULE_PATH [Hakemisto johon Snortin säännöt ovat tallennettu, yleensä valmiiksi määritettynä /etc/snort/rules]
- preprocessor stream4: detect_scans, disable_evasion_alerts, ttl_limit 10
  - detect_scans, Havaitsee piilotetut porttiskannaukset (stealth portscans) ja tekee niistä hälytyksen
  - disable_evasion_alerts, Poistaa käytöstä mahdollisesti päällekkäin olevat "äänekkäät" hälytykset
  - ttl-limit 10 (Time To Live), Määrittää sen, kuinka kauan yhteyden "elossa"-oloaika saa muuttua ennen kuin siitä tehdään hälytys (ttl-arvo pienenee joka kerta, kun paketti on kulkenut reitittimen/kytkimen läpi)
- preprocessor stream4_reassemble: both, ports default (Verkkoliikenteen jälleenkokoamisesiprosessori, both määrittää sen, että molempien puolien yhteyksien liikenne kootaan uudelleen ja ports määrittää sen, mitkä portit ovat esiprosessorin käsiteltävänä)
- preprocessor sfportscan: proto {all} scan_type {all} memcap {10000000} sense_level {low}
  - proto, Määrittää mitä kaikkia protokolla skannauksia halutaan tarkkailla (tcp, udp, icmp, ip, all)
  - scan_type, Määrittää mitä kaikkia skannaustyyppejä halutaan tarkkailla (port-scan, portsweep, decoy_portscan, distributed_portscan, all)
  - memcap, Määrittää kuinka paljon muistia on varattu porttiskannauksen tunnistamiseen, mitä suurempi luku, sitä enemmän näitä voidaan seurata
  - sense_level, Määrittää sen, että kuinka herkästi se havaitsee porttiskannaukset (low, medium, high)
- output database: log, mysql, user =, password =, dbname =, host =
  - user = Käyttäjänimi, joka luodaan MySQL:ään
  - password = Salasana, joka luodaan MySQL:ään

- dbname = Tietokanta, joka luodaan MySQL:ään
- host = localhost (Hälytyksiä varten määritellään tietokanta, johon Snort voi tallentaa ne). Viittaukset MySQL:ään on selitetty kappaleessa 5.3

5.2 Oinkmaster

Oinkmaster pitää Snortin säännöt ajan tasalla ja tekee sääntöjen muutokset automaattisesti, hakemalla uusimmat päivitykset esim. www.snort.org sivuilta. Crontab käyttää tietojen muokkaamiseen vi-editoria, komennot osoitteessa /11/
Oinkmaster ohjelman asennus:

- Kopioi oinkmaster.pl tiedosto /usr/local/bin/ kansioon. Kopioi oinkmaster .conf tiedosto /etc/ tai /usr/local/etc/ kansioon (Oinkmaster hakee konfiguraatio tiedot oletuksena näistä).
- Editoi oinkmaster.conf tiedostoa.
  - url = http://www.snort.org/pub-bin/oinkmaster.cgi/<oinkcode>/snortrules-snapshot-CURRENT.tar.gz (Määrittää mistä säännöt haetaan, oinkcode kohtaan määritellään koodi, jonka saa kun on rekisteröitynyt www.snort.org-sivustolle)
- Lisää Oinkmaster crontab:iin komennolla crontab -u [user] -e > aukeaa vi-editori > aloittaaksesi kirjoittaa, paina a-näppäintä ja lisää seuraava rivi tähän tiedostoon, lopuksi paina shift pohjaan ja paina kaksi kertaa z-näppäintä tallettaaksesi muutokset:
  30 2 * * * oinkmaster.pl -o /etc/snort/rules/ -b /etc/snort/backup 2>&1 |logger -t oinkmaster. (nyt crontab käynnistää oinkmasterin joka päivä 2:30, ottaa varmuuskopion vanhoista säännöistä ja ilmoittaa syslogiin jos jotain muutoksia on tehty)

5.3 MySQL:n konfigurointi

MySQL:n asennus on hyvin yksinkertainen jos käyttää käyttöjärjestelmään sisällytettyä ohjelma-pakettia. Ohjeessa käydään läpi miten Snortille asennetaan MySQL-tietokanta, jolloin Snort osaa tallentaa tietokantaan hälyttämänsä tiedot.

- Käynnistä MySQLd ja lisää se automaattisesti käynnistymään komennolla

#chkconfig --level 3 mysqld

- Käynnistä MySQL-komentokehote '***mysql***' komennolla
- Salasana root-käyttäjälle: mysql>set password for root@localhost=password ('salasana');
- Luo myös Snortille oma tietokanta: mysql>create database snort;

5.3.1 Taulujen ja käyttöoikeuksien lisääminen

Tässä osiossa selitetään miten tietokantaan lisätään käyttöoikeuksia ja tauluja.

- Määritetään ne oikeudet joita eri käyttäjät voi muokata tietokannassa
  - mysql>***grant insert, select on root.* to snort@localhost;***
  - mysql>***set password for snort@localhost=password('salasana');***
  - mysql***>grant create, insert, select, delete, update on snort.* to snort@localhost;***
  - mysql***>grant create, insert, select, delete, update on snort.* to snort;***
  - mysql>***exit***
- Seuraavaksi luodaan snort-tietokantaan taulut, tämä tapahtuu suorittamalla Snortin kansiosta skripti, joka sisältää komennot MySQL:lle.
  - ***# mysql -u root -p < ~/snort-2.7.0-1/schemas/create_mysql snort***
    - 'mysql -u root', Määrittää mikä käyttäjä kirjautuu mysql tietokantaan
    - -p, Määrittää sen että käyttäjältä kysytään salasana
    - < ~/snort-2.7.0-1/schemas/create_mysql snort, Määrittää mistä skripti haetaan ja mihin tietokantaan se tallennetaan
- Seuraavaksi tehdään mysql-tietokanta hieman turvallisemmaksi poistamalla ”tyhjät” käyttäjät mysql.user ja mysql.db taulusta, jotta vain sallitut käyttäjät pääsevät kirjautumaan mysql-palvelimelle. Taulun saa näkyviin komennolla:

  mysql>select user,host from mysql.user

```
+---------+--------------------+
|  user   |       host         |
+---------+--------------------+
|  snort  |         %          |
|         |    localhost       | <- Poistetaan tämä  rivi
|  root   |    localhost       |
|  snort  |    localhost       |
+---------+--------------------+
käyttämällä komentoa:
mysql>delete from mysql.user where user="";
Jonka jälkeen taulu näyttää tältä

+---------+--------------------+
|  user   |       host         |
+---------+--------------------+
|  snort  |         %          |
|  root   |    localhost       |
|  snort  |    localhost       |
+---------+--------------------+
```

Kuva 11. MySQL tietokannan taulu

5.4 Apache Web-palvelimen konfigurointi

Koneelle asennettiin Apachen HTTP-palvelin, Mandrivan omalta asennuslevyltä, jossa oli valmiina SSL-suojattu yhteyskäytäntö, joka lisää hieman tietoturvaa.

Koska IDS-kone (Intrusion Detection System) on niin hyvin suojattu kuin koneen vähiten suojattu ohjelma, jota koneessa käytetään. Koska tällainen kone on yleensä sijoitettu palomuurin ulkopuolelle, täten web-palvelimesta tehdään hieman paremmin suojattu. Tehdään httpd.conf tiedostoon seuraavat muutokset:

- ***Serversignature Off***, Tämä muuttaa palvelimen toimintaa siten, jotta se ei näytä tämän jälkeen mitä versiota palvelin käyttää
- ***#Listen 80***, Web-palvelin ei kuuntele 80 porttia

```
SSLCertificateFile /etc/httpd/conf/ssl.crt/server.crt
SSLCertificateKeyFile /etc/httpd/conf/ssl.key/server.key
<Directory /var/www/>
AuthType                basic
AuthName                "Snort"
AuthBasicProvider       file
AuthUserFile            conf/protected.passwd
Require                 valid-user
</Directory>
```

Kuva 12. lisäys httpd.conf tiedostoon

Koska Web-liikennettä ei haluta seurattavan, asetetaan SSL-salaus käyttöön:

- Poistetaan testi-varmenne ja -avain koneelta, nämä löytyvät kansioista: /etc/pki/tls/certs/ ja */etc/pki/tls/private/*
  - #cd /etc/pki/tls/certs → # rm localhost.crt
  - #cd /etc/pki/tls/private → # rm localhost.key

- Varmenne tehdään seuraavanlaisesti:

```
# openssl req -new -x509 -nodes -out server.crt -keyout server.key

Generating a 1024 bit RSA private key
.....................++++++
.....................++++++
writing new private key to 'server.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [GB]:
State or Province Name (full name) [Berkshire]:
Locality Name (eg, city) [Newbury]:
Organization Name (eg, company) [My Company Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:
Email Address []:
```

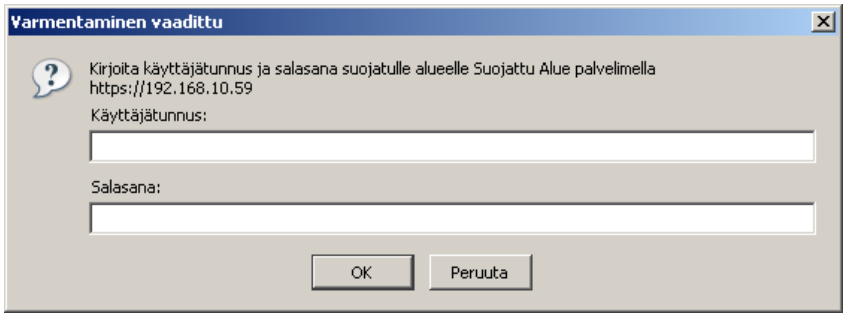
Kuva 13. Varmenteen tekeminen

Kun kaikki tiedot on annettu, ohjelma palaa komentoriville. Tämän jälkeen siirretään varmenteet yllä oleviin kansioihin. Seuraavaksi luodaan salasana vapaasti valittavalle käyttäjänimelle, jolla on tämän jälkeen oikeudet päästä IDS-koneella olevaan web-palvelimeen. Tämän jälkeen käynnistetään palvelin uudelleen, jotta uudet asetukset tulisivat voimaan.

```
[root@localhost conf]# htpasswd -c protected.passwd[käyttäjä]
New password:
Re-type new password:
Adding password for user[käyttäjä]
[root@localhost conf]# service httpd restart
Shutting down httpd:                                       [  OK  ]
[root@localhost conf]# service httpd restart

Starting httpd:                                            [  OK  ]
```

Kuva 14. salasanan luominen WWW-sivulle

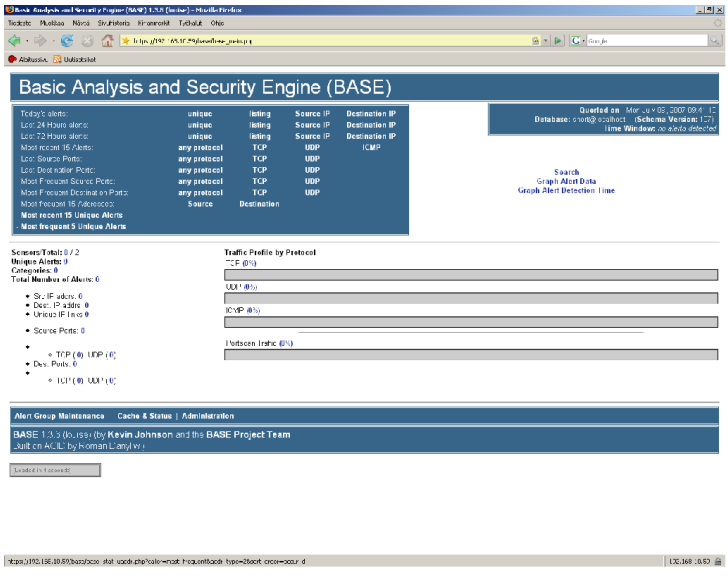


Kuva 15. WWW-palvelin kysyy käyttäjää/salasanaa

5.5 BASE Analyysi-työkalu

Kohtuullisen hyvä komento on tail -f, kun tarkkaillaan Snortin hälytyksiä, mutta jos halutaan paremman analysoinnin ja suojan verkkoon on hyvä olla olemassa tarkkailu työkalut. BASE (Basic Analysis and Security Engine) kerää kaikki tiedot hälytyksistä käyttäen hyväksi MySQL-tietokantaa, johon Snort tallentaa hälytykset. BASE:n ohjelmointikoodi perustuu ACID:n projektiin (Analysis Console for Intrusion Databases). BASE:lla on webbipohjainen käyttöliittymä, joka käyttää PHP-koodia hyväkseen ja analysoi hälytyksiä, joita Snort havaitsee. Tämän hallintakonsolin tarkoituksena on auttaa verkon ylläpitäjää hallitsemaan hälytyksiä, joita esim. Snort havaitsee. Jotta BASE toimisi, pitää ensin /var/www/html/base/ hakemistosta muokata base_conf.php tiedostoa niin, että se hakee tiedot MySQL-tietokannasta. Websivusto voidaan avata selaimella muuttamalla seuraavilla riveillä olevat tiedot.

- $Use_Auth_System = 1; (Ottaa käyttöön käyttäjän tunnistuksen)
- $BASE_urlpath = ''; → $BASE_urlpath = '/base'; (Määrittää missä osoitteessa BASE on kyseisellä palvelimella Esim. Https://127.0.0.1/base)
- $DBlib_path = ''; → $DBlib_path = 'Kansioon johon ADOdb on asennettu Esim. /var/www/adodb/';
- $alert_dbname = ''; → $alert_dbname = 'Tietokanta joka luotiin MySQL:ään';
- $alert_user = ''; → $alert_user = 'Käyttäjänimi joka luotiin MySQL:ään';
- $alert_password =''; → $alert_password = 'Salasana joka luotiin MySQL:ään';



Kuva 16. BASE:n käyttöliittymä

6 SNORT – KÄYTÄNNÖN TESTAUS

Snortin toimintaa voidaan testata monin tavoin netistä ladattavilla ohjelmilla, jotka simuloivat hyökkäyksiä. Näitä ovat esimerkiksi Nmap ja Nessus, nämä ovat ilmaisia ohjelmia, joilla voidaan tutkia mitä portteja tai palveluita on ”avoinna”. Käynnistä Snort testauksen ajaksi komennolla ***snort -c /etc/snort/snort.conf***, jossa –c tarkoittaa sitä, että Snort lataa käynnistysparametrit konfigurointitiedostosta.

Testauksessa ollut kytkentä on seuraavanlainen, joka simuloi hyökkäystä ”ulkoverkosta”. Tämä siksi koska koulun verkossa ei tällaista kytkentää voinut helposti toteuttaa. Kuitenkin asiasta saa hyvän käsityksen siitä, kuinka Snort havaitsee hyvin erilaisia hyökkäystyyppejä, joilla ohjelmat yrittävät tiedustella tai tunkeutua kohteeseen.

Oikeassa tilanteessa olisi Snort-sensori asennettu palomuurin taakse ja sitä kontrolloitu konsolin avulla palomuurin sisäpuolelta (ks. sivu 30).

Kuva 17. Testikäytössä ollut kytkentä.

Yllä olevassa kuvassa käytetty kytkentä, Snort- ja hyökkääjä koneen IP:t

Snort: 192.168.10.59

Attacker, jossa Nmap ja Nessus ohjelmat: 192.168.10.84

```
        --== Initialization Complete ==--

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.7.0 (Build 35)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/team.html
           (C) Copyright 1998-2007 Sourcefire Inc., et al.

           Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 1.6  <Build 11>
           Preprocessor Object: SF_SSH  Version 1.0  <Build 1>
           Preprocessor Object: SF_SMTP  Version 1.0  <Build 7>
           Preprocessor Object: SF_DNS  Version 1.0  <Build 2>
           Preprocessor Object: SF_FTPTELNET  Version 1.0  <Build 10>
           Preprocessor Object: SF_DCERPC  Version 1.0  <Build 4>
Not Using PCAP_FRAMES
```

Kuva 18. Snort ladattu käyntiin.

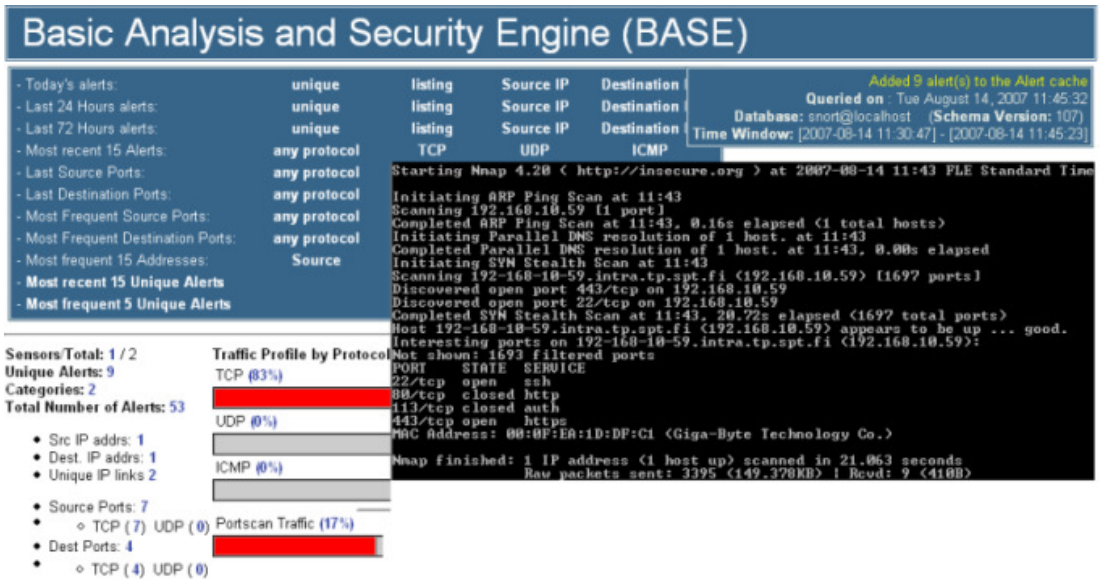
Määrittelyt snort.conf tiedostoon, joilla poistetaan monta yleistä väärää positiivista hälytystä [LIITE 1]. Väärät havainnot vaikuttavat siihen, miten hyvin hyökkäyksiä voidaan torjua. Tapahtumia saattaa tulla suuria määriä, joita ei ole mahdollista ihmisvoimin käydä läpi niin, ettei vahingossa voisi tulla poistaneeksi oikeaa hälytystä. Väärät positiiviset ovat yksi perusongelmista, väärien hälytysten ongelman ratkaiseminen on yksi pääasioista.

6.1 Nmap

Nmap ("Network Mapper") on ilmainen Open Source lisenssin alainen työkalu tietoliikenneverkon tutkimiseen tai verkon turvallisuuden tarkasteluun. Nmap käyttää ”raakaa” IP-pakettia järjestelmien tutkimiseen, jolla se saa selville järjestelmän tarjoamat palvelut (sovelluksen nimi ja versio), kuten myös käyttöjärjestelmän tiedot ja kymmenittäin muita ominaisuuksia. Nmap on kehitetty tutkimaan nopeasti suuria tietokoneverkkoja, mutta Nmap toimii hyvin myös yhden koneen tutkimisessa. Nmap toimii kaikissa hyvin tunnetuissa käyttöjärjestelmissä, joihin on konsoli ja graafinen käyttöliittymä saatavilla. /20/

6.1.1 Porttiskannauksen vaikutus Snorttiin

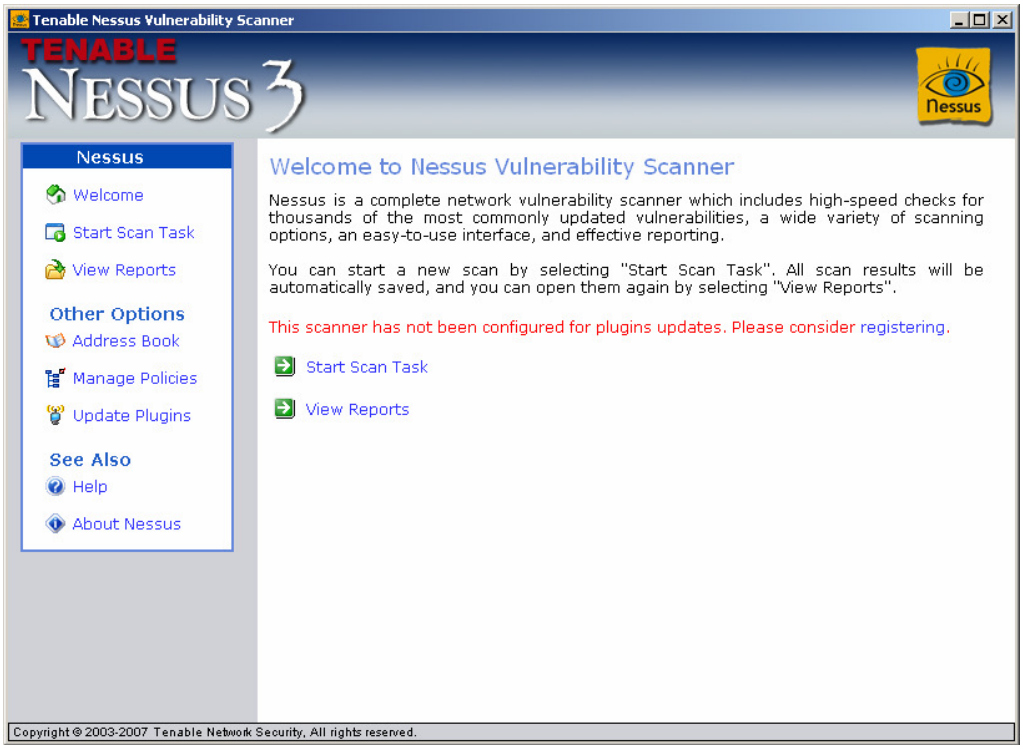
nmap -sT -sU -O -v -v 192.168.10.59



Kuva 19. Nmap:n löytämät avoimet portit.

6.2 Nessus

Nessus Projekti käynnistettiin Renaud Deraisonin toimesta vuonna 1998 tarjoamaan Internet yhteisölle ilmaisen, tehokkaan, päivitetyn ja helppokäyttöisen Internet-turvallisuutta tarkastelevan skannerin. Nessus on ohjelma, jolla voidaan tarkastaa tietokoneverkon täydellisen analyysin nykyisestä turvallisuustasosta /19/



Kuva 20. Nessus ohjelman käyttöliittymä.

6.3 Skannaus

```
[**] [1:1444:3] TFTP Get [**]
[Classification: Potentially Bad Traffic] [Priority: 2]
08/14-12:01:36.259418 192.168.10.84:6729 -> 192.168.10.59:69
UDP TTL:64 TOS:0x0 ID:28145 IpLen:20 DgmLen:51
Len: 23

[**] [1:524:8] BAD-TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
08/14-12:02:24.260830 192.168.10.84:4019 -> 192.168.10.59:0
TCP TTL:64 TOS:0x0 ID:11942 IpLen:20 DgmLen:40
******S* Seq: 0x12DB  Ack: 0x0  Win: 0x200  TcpLen: 20

[**] [1:634:2] SCAN Amanda client version request [**]
[Classification: Attempted Information Leak] [Priority: 2]
08/14-12:02:34.260295 192.168.10.84:3756 -> 192.168.10.59:10080
UDP TTL:128 TOS:0x0 ID:46261 IpLen:20 DgmLen:94

[**] [1:1504:6] MISC AFS access [**]
[Classification: Misc activity] [Priority: 3]
08/14-12:02:57.148771 192.168.10.84:3769 -> 192.168.10.59:7001
UDP TTL:128 TOS:0x0 ID:46611 IpLen:20 DgmLen:60
Len: 32
[Xref => http://cgi.nessus.org/plugins/dump.php3?id=10441]

[**] [1:2049:4] MS-SQL ping attempt [**]
[Classification: Misc activity] [Priority: 3]
08/14-12:03:08.777731 192.168.10.84:3784 -> 192.168.10.59:1434
UDP TTL:128 TOS:0x0 ID:46862 IpLen:20 DgmLen:29
Len: 1
[Xref => http://cgi.nessus.org/plugins/dump.php3?id=10674]

[**] [1:221:5] DDOS TFN Probe [**]
[Classification: Attempted Information Leak] [Priority: 2]
08/14-12:08:53.009706 192.168.10.84 -> 192.168.10.59
ICMP TTL:255 TOS:0x0 ID:9 IpLen:20 DgmLen:32
Type:8  Code:0  ID:678   Seq:1  ECHO
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=2000-0138][Xref => http:/
/www.whitehats.com/info/IDS443]

[**] [1:469:4] ICMP PING NMAP [**]
[Classification: Attempted Information Leak] [Priority: 2]
08/14-12:08:54.283022 192.168.10.84 -> 192.168.10.59
ICMP TTL:64 TOS:0x6 ID:18247 IpLen:20 DgmLen:28 DF
Type:8  Code:123  ID:28019   Seq:28019  ECHO
[Xref => http://www.whitehats.com/info/IDS162]

[**] [122:1:0] (portscan) TCP Portscan [**]
[Priority: 3]
08/14-12:09:11.505786 192.168.10.84 -> 192.168.10.59
PROTO:255 TTL:0 TOS:0x0 ID:0 IpLen:20 DgmLen:166 DF
```

Kuva 21. Eräitä hyökkäyksiä joita Snort on havainnut.

7 YHTEENVETO

Opinnäytetyön tarkoituksena oli perehtyä IDS-järjestelmiin ja niiden tekniikkaan havaita tunkeutumiset. Alussa tutkittiin miten toteuttaa IDS-järjestelmän asemointi verkossa, mihin käyttöjärjestelmään se asennetaan ja miten IDS-järjestelmä havaitsee tunkeutumisia verkossa. Lisäksi tutkittiin kuinka DDoS-hyökkäykset tapahtuvat ja miten niitä voidaan estää.

Tutkittavaksi ohjelmaksi valittiin Snort, koska se on ilmainen, maailman parhaaksi kutsuttu IDS-työkalu, jolla voidaan havaita tunkeutumisia tietoliikenneverkossa. Aluksi tutustuttiin Snort-ohjelmaan miten se toimii ja mikä olisi paras vaihtoehto käyttöjärjestelmäksi. Testauksessa käytettiin Nmap ja Nessus ohjelmia, joilla voidaan tutkia mitä palveluita on kohdekoneessa käytössä.

Testauksessa käytettiin pientä verkkoa jossa oli yksi Linux ja Windows kone. Näistä Linux-koneeseen oli asennettu Snort ja Windows-koneeseen Nmap ja Nessus.

LÄHTEET

WWW-Sivut tarkistettu Heinäkuussa 2007

/1/ Kerry J. Cox, Christopher Gerg: Managing Security with Snort and IDS Tools, August 2004

/2/ Jelena Mirkovic, Sven Dietrich, David Dittrich, Peter Reiher: Internet Denial of Service Attack and Defense Mechanisms. Prentice Hall Professional Technical Reference, December 2004

/3/ Allen, J., McHugh, J., Christie, A., Defending Yourself: The Role of Intrusion Detection Systems. IEEE Software September/October 2000

/4/ Jack Koziol: Intrusion Detection with Snort, Sams 2003

/5/ Koskinen Jukka: Tunkeutumisen havaitseminen, 2004, TTY http://www.cs.tut.fi/kurssit/8306000/TL-TT/2004/ids.pdf

/6/ IPS ja IDS ohjelmistojen eroavaisuuksia http://searchnetworking.techtarget.com/tip/0,289483,sid7_gci1133204,00.html

http://www.stillsecure.com/docs/StillSecure_CyberDefense_IPS_v_IDS_0304.pdf

http://www.digitoday.fi/page.php?page_id=14&news_id=200313939

/7/ http://en.wikipedia.org/wiki/Intrusion-prevention_system

/8/ http://en.wikipedia.org/wiki/Intrusion_detection_system

/9/ Snort-tunkeutumisen havaitsemisjärjestelmä http://www.snort.org/ http://www.snort.org/docs/snort_htmanuals/htmanual_2615/

/10/ Yksinkertainen Snortin asennusvideo http://geeknextdoor.net/members/video.asp?TutorialID=4

/11/ Vi-editor cheat sheet http://bullium.com/support/vim.html

/12/ MySQL-tietokanta http://www.mysql.com/

/13/ Web-palvelin http://www.apache.org/

/14/ Basic Analysis and Security Engine http://base.secureideas.net/

/15/ database abstraction library for PHP http://adodb.sourceforge.net/

/16/ Snortin vaatima verkkoliikenteen kaappaus-kirjasto, (Kaikki versiot) http://www.tcpdump.org (http://www.tcpdump.org/release/)

/17/ Snort Rules http://www.snort.org/pub-bin/downloads.cgi, Jotta saa uusimmat säännöt ladattua, pitää rekisteröityä sivustolle

/18/ Uusin Snort käyttöopas löytyy Tar-paketista /docs/snort_manual.pdf http://www.snort.org/docs/snort_manual/2.6.1/snort_manual.pdf

/19/ Nessus http://www.nessus.org, seuraavasta linkistä voi ladata vanhemman version ilman rekisteröitymistä:

http://www.download.com/Nessus/3000-2085_4-10668771.Html?tag=lst-0-1

/20/ Nmap http://www.insecure.org/nmap/

/21/ Wikipedia, Denial of Sevice
http://en.wikipedia.org/wiki/Denial-of-service_attack

/22/ CERT What is Denial of Sevice

http://www.us-cert.gov/cas/tips/ST04-015.html

/23/ www.cpni.gov.uk/Docs/re-20021025-00481.pdf

/24/ http://www.securityfocus.com/infocus/1463

/25/ Guide to Intrusion Detection and Prevention Systems (IDPS)
http://csrc.ncsl.nist.gov/publications/nistpubs/800-94/SP800-94.pdf

Suositellut asetukset snort.conf tiedostoon, jos havaitaan hyvin suuria määriä vääriä hälytyksiä

Suurin osa hälytyksistä on vääriä jättämällä nämä asetukset päälle.

config disable_decode_alerts
config disable_tcpopt_experimental_alerts
config disable_tcpopt_obsolete_alerts
config disable_tcpopt_ttcp_alerts
config disable_tcpopt_alerts
config disable_ipopt_alerts

Tämän asetuksen voi jättää päälle jos halutaan seurata porttiskannauksia.

(#) preprocessor flow: stats_interval 0 hash 2

Jos Snort sensori on palomuurin takana tai reitittimen, joka osaa koota uudelleen pirstoutuneet paketit, tämä asetus voidaan ottaa pois päältä.

 # preprocessor frag2

Osa non-security-related liittyvistä ongelmista voi aiheuttaa Stream4 uudelleen-kokoamis virheitä, tämän estämiseksi kerrotaan esiprosessorille jättää hälytykset tekemättä

preprocessor stream4_reassemble: noalerts

jotkut verkon asetuksista saattavat aiheuttaa vääriä hälytyksiä, tämän estämiseksi kerrotaan esiprosessorille jättää hälytykset tekemättä

preprocessor http_inspect_server: server default profile all ports {80 8080 8180} oversize_dir_length 500 no_alerts

Useimmille application, on parasta jättää nämä pois käytöstä (valmiiksi pois päältä)
 # web-attacks.rules, # backdoor.rules, # shellcode.rules, # policy.rules, # porn.rules, # info.rules
 # icmp-info.rules, # virus.rules, # chat.rules, # multimedia.rules, # p2p.rules

Seuraavat ovat väärien hälytysten lähteitä ja ne voidaan ottaa pois käytöstä useimmissa asennuksissa.

 # icmp.rules, # misc.rules, # nntp.rules, # finger.rules

Seuraavia sääntöjä voidaan ottaa käyttöön jos sellainen palvelu on olemassa (#-merkki pois):

Coldfusion-palvelin: *# web-coldfusion.rules*
FrontPage laajennukset: *# web-frontpage.rules*
PHP ohjelma palvelut: *# web-php.rules*
Tietokannat: Oracle: *# oracle.rules,* MySQL: *# mysql.rules*, Microsoft SQL Server: *# sql.rules*
Järjestelmät: Unix *# rpc.rules, # rservices.rules, # x11.rules,* Windows # *netbios.rules*
Web-palvelimet: Apache *# web-cgi.rules,* Internet Information Services (IIS)*# web-iis.rules*

Seuraavat säännöt ovat käytännöllisiä ja ne aiheuttavat hyvin harvoin vääriä hälytyksiä, nämä on hyvä jättää päälle:
exploit.rules, dos.rules, ddos.rules, attack-responses.rules