jamk.fi

Keskitetty lokienhallintajärjestelmä

Nagios Log Server

Samu Tuomala

Opinnäytetyö
Toukokuu 2015

Tietotekniikan koulutusohjelma
Tekniikan ja liikenteen ala

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

Kuvailulehti

Tekijä(t)	Julkaisun laji	Päivämäärä
Samu Tuomala	Opinnäytetyö	25.05.2015
	Sivumäärä 85	Julkaisun kieli Suomi
		Verkkojulkaisulupa myönnetty: (x)

Työn nimi

Keskitetty lokienhallintajärjestelmä

Nagios Log Server

Koulutusohjelma

Tietotekniikan koulutusohjelma

Työn ohjaaja(t)

Mika Rantonen

Antti Häkkinen

Toimeksiantaja(t)

Qvantel Finland Oy

Saku Pietilä

Tiivistelmä

Opinnäytetyön toimeksiantajana toimi Qvantel Finland Oy. Työn tarkoituksena oli perehtyä Nagios Log Server -sovellukseen ja toteuttaa sillä lokienhallintajärjestelmä. Lokienhallinnalla pyritään lisäämään tietoa verkon tapahtumista, tehostamaan verkon toimintaa ja parantamaan yrityksen tietoturvaa.

Lokitieto on tiettynä hetkenä tapahtuneen tapahtuman dokumentti. Erilaiset järjestelmät, kuten työasemat, palvelimet ja verkkolaitteet tuottavat lokitietoa. Lokeja on erilaisia, kuten Windows-käyttöjärjestelmien Event log ja Syslog, jota käytetään Linux-käyttöjärjestelmissä. Lokit sisältävät usein arkaluotoista tietoa organisaatiosta tai sen asiakkaista, joten lokeja on käsiteltävä varoen.

Nagios Log Server on Nagios Enterprisesin uudehko tuote lokien hallintaan ja analysointiin. Sen tarkoitus on kerätä, tallentaa ja suodattaa lokitietoja ja muodostaa niistä havainnollisia raportteja. Sovellus mukautuu monenlaisiin ympäristöihin ja on yhteensopiva eri laitteiden kesken.

Nagios Log Serverin avulla pystytään toimeksiantajan verkon tapahtumista suodattamaan haluttuja tietoja ja havaitsemaan mahdollisia vikatilanteita tai tietomurtoja.

Avainsanat (asiasanat)

Event log, Loki, Nagios Log Server, Syslog

Muut tiedot

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

Description

Author(s)	Type of publication	Date
Samu Tuomala	Bachelor's thesis	25.05.2015
	Number of pages 85	Language of publication Finnish
		Permission for web publication: (x)

Title of publication

Centralized log management system

Nagios Log Server

Degree programme

Information Technology

Tutor(s)

Mika Rantonen
Antti Häkkinen

Assigned by

Qvantel Finland Oy
Saku Pietilä

Abstract

This bachelor's thesis was assigned by Qvantel Finland Oy. The purpose of this thesis was to evaluate Nagios Log Server application and implement a log management system. Log management aims to increase information about network events, enhance networks functionality and increase company's information security.

Log is a document of an event that happens at a given time. Various systems such as workstations, servers and network devices generate logs. There are different kinds of logs, such as Event log, which is used in Windows operating systems and Syslog, which is used in Linux operating systems. Logs must be handled with care, since they often contain sensitive information about the organization or its customers.

Nagios Log Server is a newish product of Nagios Enterprise's for log management and analysis. Its purpose is to collect, store and filter the log data and generate graphic reports. The application adapts to a wide range of environments and it is compatible between different devices.

Nagios Log Server is able to filter required information from network events and to identify possible system faults or intrusions.

Keywords/tags (subjects)

Event log, Log, Nagios Log Server, Syslog

Miscellaneous

Sisältö

Kuviot

Taulukot

Lyhenteet

AD	Active Directory
BSS	Business Support Systems
CLF	Common Log Format
CSV	Comma-Separated Value
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
DS	Domain Services
ELF	Extended Log Format
FQDN	Fully Qualified Domain Name
HTTP	Hypertext Transfer Protocol
IETF	Internet Engineering Task Force
IIS	Internet Information Services
IP	Internet Protocol
JSON	JavaScript Object Notation
KATAKRI	Kansallinen turvallisuusauditointikriteeristö
NLS	Nagios Log Server
NRDP	Nagios Remote Data Processor
NTP	Network Time Protocol
OSE	Open Source Edition
PE	Premium Edition

RELP	Reliable Event Logging Protocol
REST	Representational State Transfer
RFC	Request for Comments
SEM	Security Event Management
SIEM	Security Information and Event Management
SIM	Security Information Management
SNMP	Simple Network Management Protocol
SSL	Security Socket Layer
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
XML	Extensible Markup Language
Xpath	XML Path Language

1 Lähtökohdat

1.1 Toimeksiantaja

Työn toimeksiantajana toimi Qvantel Finland Oy. Qvantel on kansainvälinen ICT-palveluita tarjoava yritys, joka työllistää yhteensä noin 250 ammattilaista viidessä EU-maassa, Suomessa, Ruotsissa, Virossa, Sveitsissä ja Espanjassa sekä Intiassa. Qvantel tarjoaa kustomoituja, pilvipohjaisia BSS-järjestelmiä (Business Support Systems) ja konsultointia pääasiassa ICT-palveluntarjoajille, mutta myös sähköisen laskutuksen tarjoajille ja mediayhtiöille. (Company 2015.)

Qvantelin tavoitteena on olla asiakkaiden ensimmäinen valinta tarvittaessa yhteistyökumppania yrityksensä kriittisille tukijärjestelmille. Yli 20 vuoden kokemuksella Qvantelilla on vankka ja mukautuva runko, jonka pohjalta voi helposti ja tehokkaasti muokata yrityksille sopivia ratkaisuja. Qvantelin asiakkaita ovat mm. DNA, Basware ja espanjalainen teleoperaattori Yoigo. (Company 2015.)

1.2 Tavoitteet

Opinnäytetyön tavoitteena oli luoda järjestelmä yrityksen lokitietojen hallintaan lisäämään tietoa verkon tapahtumista, parantamaan tietoturvaa ja tehostamaan verkon toimintaa. Lokitietoja tulisi kerätä verkon eri laitteista ja järjestelmistä, mm. työasemista, palvelimista ja verkkolaitteista, minkä jälkeen niitä voitaisiin tallentaa, yhtenäistää ja luokitella riippuen lokitietojen merkityksestä.

Kerättyjä lokitietoja oli tarkoitus myös pystyä tarkastelemaan eri arvojen perusteella, tuottaa hälytyksiä kriittisistä tapahtumista verkon ylläpitäjille ja muodostaa niistä helposti tarkasteltavia raportteja. Lokien hallintaan ja analysointiin käytettäväksi järjestelmäksi toimeksiantaja oli ennalta valinnut Nagios Log Serverin.

Nagios Log Server on yritystason sovellus lokien hallintaan ja analysointiin. Sovelluksen avulla lokitietojen kerääminen onnistuu eri laitteista ja järjestelmistä, ja sillä pystytään

visualisoimaan monipuolisesti verkon tapahtumia. Sovelluksesta on saatavana ilmainen kokeiluversio, mutta se perustuu maksulliseen tuotteeseen, jonka mukana tulee lisenssi sekä kattava tuki sovellukseen.

2 Lokit

2.1 KATAKRI

KATAKRIn (Kansallinen turvallisuusauditointikriteeristö) päätavoitteena on määrittää yhtenäinen kriteeristö sille, kun viranomainen toteuttaa yrityksessä tai muussa yhteisössä turvallisuustason todentavan tarkastuksen. Sen toinen päätavoite on auttaa yrityksiä, muita yhteisöjä ja viranomaisia omassa sisäisessä turvallisuustyössään. KATAKRIn ensimmäinen versio otettiin käyttöön vuonna 2009 ja versio 2.0 julkaistiin vuonna 2011. Koska kriteeristä on tarkoitettu myös muille kuin viranomaisille, se sisältää erilliset elinkeinoelämän suositukset, joiden kautta voidaan tarpeen vaatiessa edetä viranomaisvaatimusten tasolle. (Kansallinen turvallisuusauditointikriteeristö (KATAKRI) 2015.)

Kriteeristö jakautuu neljään osa-alueeseen: hallinnollinen turvallisuus, henkilöstöturvallisuus, fyysinen turvallisuus ja tietoturvallisuus. Vaatimukset on puolestaan luokiteltu kolmeen turvallisuustasoon: perustaso (IV), korotettu taso (III) ja korkea taso (II). (Kansallinen turvallisuusauditointikriteeristö (KATAKRI) 2015.)

Lokeja kriteeristössä käsitellään kohdassa I 504.0, joka sisältää seuraavat kysymykset:

”Pääkysymys: Ovatko organisaation teknisten laitteiden ja palveluiden lokimenettelyt kunnossa?”

”Lisäkysymys: Kerätäänkö verkoista, laitteista ja järjestelmistä keskeiset lokitiedot ja käsitelläänkö niitä asianmukaisesti?”

Viranomaisvaatimukset perustasolla (IV):

1) Tallenteiden kattavuus on riittävä tietomurtojen tai niiden yritysten jälkikäteiseen todentamiseen.

2) Keskeisiä tallenteita säilytetään 6 kk tai erillisessä sopimuksessa määrätty aika.
3) Suojattavaa tietoa sisältävät lokitiedot on suojattu asianmukaisesti (pääsynvalvonta, käsittely, poisto).

Elinkeinoelämän suositukset:

1) Tallenteiden kattavuus on riittävä tietomurtojen tai niiden yritysten jälkikäteiseen todentamiseen.
2) Keskeisiä tallenteita säilytetään riskienarvioinnissa määritetty aika.

(Kansallinen turvallisuusauditointikriteeristö 2011.)

Vaatimuksia on tarkemmin käsitelty myöhemmässä vaiheessa otsikossa lokien käsittely. Tässä työssä KATAKRI on otettu osaksi teoriaa ja sitä pyrittiin noudattamaan myös toteutuksessa.

2.2 Lokien määritelmä

”Lokitieto on dokumentti jonkin tapahtuman toteutumisesta jonakin tiettynä hetkenä. Loki dokumentoi tapahtumia, jotka ovat tapahtuneet organisaation järjestelmissä, verkoissa tai muussa ympäristössä ja toiminnassa.” (Lokiohje 2009, 13.)

Lokit ovat olemassa tiettyä tarkoitusta varten tietyn ajan, ja niitä käsitellään niin erityiskuin normaalitilanteissa. Lokeja tarvitaan normaalitilanteissa toiminnan seuraamiseen ja varmistamiseen, kun taas erityistilanteissa lokit auttavat palauttamaan tilanteen normaaliksi ja selvittämään tapahtumien osapuolia, syitä ja vaikutuksia. Järjestelmien ylläpitäjät saavat lokeista tärkeää tietoa, joka edesauttaa esim. tietoturvapoikkeamien selvittämisessä, verkon ja järjestelmien optimoinnissa sekä eri tahojen oikeusturvan toteutumisessa. Lokitietojen suunnitelmallinen suojaaminen, seuranta ja säilyttäminen on tärkeää, sillä niillä on erittäin suuri merkitys tietoturvallisuuden ja tietosuojan kannalta. Lokijärjestelmästä tulisi pystyä todentamaan aukottomasti eri tapahtumien kulku alusta loppuun saakka. (Lokiohje 2009, 13.)

Lokitietojen käsittelyllä pyritään varmistamaan tapahtumien osapuolet, kiistämättömyys ja kulku. Osapuolilla tarkoitetaan henkilöitä, jotka ovat olleet osallisena tapahtu-

missa. Kiistämättömyydellä tarkoitetaan sitä, ettei kukaan osallisena ollut pysty kiistämään osallisuuttaan. Lokit voidaan järjestää aikajärjestykseen, jolloin tapahtumien kulku saadaan selville. Lisäksi lokit auttavat ylläpitäjiä havaitsemaan tunkeutumisia, poikkeamia ja suorituskykyongelmia, ja niiden avulla varmistetaan käyttäjien oikeusturvaa. Tapahtumista, jotka mahdollisesti johtavat jatkotoimenpiteisiin, pystytään selvittämään, onko henkilö ollut osallisena vai ei. (Lokiohje 2009, 15.)

Usein lainsäädäntö asettaa omat vaatimuksensa lokien käsittelylle. Ainakin seuraavat lait liittyvät lokien käsittelyyn asettamalla vaatimuksia tai antamalla oikeuksia: henkilötietolaki, julkisuuslaki, laki yksityisyyden suojasta työelämässä ja sähköisen viestinnän tietosuojalaki. Lait vaikuttavat erityisesti silloin, kun käsittelyn kohteena on tunnistamis- tai henkilötietoja. Jos näitä tietoja esiintyy osana lokitietoja, tulisi jo käsittelyn suunnitteluvaiheessa ottaa huomioon tarvittavat vaatimukset. (Lokiohje 2009, 20–21.) Myös eräät tietoturvastandardit, ISO 27001 ja ISO 27002, antavat vaatimuksia, ohjeita ja suosituksia lokienhallinnan eri osa-alueille (Lokiohje 2009, 15).

2.3 Lokienhallinnan merkitys

Perusteita lokienhallintaan on ainakin kolme:

- Liiketoiminnan jatkuvuus
- Toiminnan tehostaminen
- Parempi tietoturva

(Lokienhallinta 2015).

Liiketoiminnan jatkuvuus tarkoittaa toiminnan jatkumista häiriöistä huolimatta. Lokienhallintajärjestelmä mahdollistaa poikkeustilanteissa ongelmien nopean selvittämisen ja niistä palautumisen. Mahdollisten SLA- eli palvelutasosopimusten vaikuttaessa liiketoimintaan on erityisen tärkeä palauttaa sovittu palvelutaso sanktioiden välttämiseksi. Muun muassa KATAKRI ja aiemmin mainitut standardit jopa velvoittavat tietyn tyyppisiä organisaatioita lokienhallintaan.

Organisaation toimintaa voidaan tehostaa lokienhallinnan avulla. Järjestelmät tuottavat suuren määrän lokitietoa, joista osa on hyödyllistä ja osa hyödytöntä. Lokienhallintajärjestelmällä pystytään ensinnäkin tunnistamaan hyödylliset lokitiedot ja lisäksi ne pystytään ottamaan tehokkaaseen tarkasteluun, jolloin tietomäärästä saadaan poimittua tärkeät tapahtumat. Kun tiedetään tapahtumien todellinen vaikutus ja merkitys, voidaan toimintaa muokata tehokkaampaan suuntaan.

Lokienhallinta parantaa tietoturvaa. Kuten jo aikaisemmin mainittiin, lokit mahdollistavat tietoturvapoikkeamien ja tapahtumien kulun selvittämisen. Lokienhallintajärjestelmiä voidaan kutsua SIEM-järjestelmiksi (Security Information and Event Management). Nimi tulee SIM-järjestelmän (Security Information Management) ja SEM-järjestelmän (Security Event Management) yhteisnimityksestä. SIM-järjestelmät hoitavat keskitetysti verkon lokien ja viestien hallinnan. SEM-järjestelmät puolestaan tarjoavat reaaliaikaista monitorointia ja tapahtumien hallintaa. Lokienhallintajärjestelmän tarkoituksena on lokien keskitetty kerääminen ja niiden reaaliaikainen analysointi.

2.4 Lokien käsittely

2.4.1 Kerääminen

KATAKRI kohdan I 504.0 lisäkysymyksen ja viranomaisvaatimuksen perustason (IV) ensimmäisessä kohdassa todetaan lokien keräämisen olevan riittävää, kun tallenteiden kattavuus on riittävä tietomurtojen tai niiden yritysten jälkikäteiseen todentamiseen (Kansallinen turvallisuusauditointikriteeristö 2011).

Yleisiä kerättäviä tietoja ovat kirjautumiset järjestelmiin ja siellä tehtävät toimenpiteet, laitteiden tilamuutokset, vikatilanteet ja suorituskyky. Kirjautumis- ja toimenpidelokeista pystytään jäljittämään tekijä, joka muutoksia on tehnyt, sekä se, mitä muutoksia on tehty. Esimerkiksi luvaton potilastietojen tarkastelu on rangaistava teko, ja teon tultua ilmi pystytään lokitiedostoista selvittämään rikkomuksen tekijä. Kytkimen toiminnan ollessa epävakaata pystytään sen lähettämistä vikatilanne- ja suorituskykylokeista havaitsemaan virheellinen toiminta ja puuttumaan ongelmiin ajoissa, mahdollisesti jopa ennen varsinaisia vikatilanteita.

Lokitietoja tulisi kerätä monipuolisesti eri lähteistä, jotta verkon tapahtumista saadaan riittävästi tietoa ja sitä pystytään paremmin analysoimaan. Keräyskohteita ovat Windows- ja Unix-työasemat, eri tarkoituksiin räätälöidyt palvelimet sekä aktiivilaitteet kuten reitittimet, kytkimet, palomuurit jne. Toisaalta lokien suuresta määrästä johtuen on tärkeää suodattaa tarpeettomat viestit jo keräysvaiheessa, jolloin vältytään oleellisen tiedon hukkumiselta epäolennaisten viestien sekaan. Lokienhallintajärjestelmästä riippuen kerätyt lokit siirretään yksittäisistä lähteistä yleensä keskitetylle palvelimelle, jossa varsinainen lokienhallinta tapahtuu.

2.4.2 Analysointi

Lokitietojen analysointi on usein vaativa, mutta todella tärkeä osa-alue. Se kuvataan yleensä työlääksi ja tehottomaksi, mutta hyvillä työkaluilla ja ympäristöllä voidaan automatisoida lokien käsittely ja analysointi. Säännöllisellä, jopa päivittäin tapahtuvalla lokitietojen tarkastelulla saadaan käsitys normaaleista tapahtumista, jolloin poikkeamia on helpompi havaita. Suurimman osan merkinnöistä muodostavat muutamat tapahtumat, joiden seasta opitaan vähitellen erottamaan epätavalliset ja poikkeavat merkinnät. (Lokiohje 2009, 47.)

Tarkastelun tuloksena voidaan rakentaa suodattimia, joiden avulla tunnistetaan epätavallisia tai haitallisia toimia ja niihin voidaan paremmin reagoida. Kun lokimerkinnöistä on suodatettu pois tavallisimpia tapahtumia, on analysointi paljon helpompaa ja tehokkaampaa. (Lokiohje 2009, 47–48.)

Analysoinnissa tulisi ottaa myös huomioon yhteydet toisiin erillisiin lokimerkintöihin tai järjestelmiin. Epätavallinen merkintä voi viitata mahdolliseen hyökkäykseen, jolloin vastaavien poikkeaminen etsiminen muista lokeista voi vahvistaa epäilyn. Toisinaan lokit ovat helppoja ymmärtää, mutta asiayhteydestä riippuen lokin tulkitsemiseksi voi tarvita ulkopuolisen asiantuntijan apua. Siksi onkin tärkeää tunnistaa, mikä on lokeja tuottavan lähteen normaalia toimintaa ja mikä puolestaan ei ole. (Lokiohje 2009, 48–49.)

2.4.3 Säilytys ja poisto

Lokien säilytykseen kohdistuu usein eri säädöksiä ja standardeja riippuen lokin tyypistä. Säilytyksen tulisi perustua lokin käyttötarkoitukseen eli siihen, miksi sitä ylipäätään kerätään. (Lokiohje 2009, 59.)

Jotkin lait edellyttävät, että lokeja tulee säilyttää tarpeen mukaan niin kauan, että tiedon alkuperä, eheys ja luotettavuus pystytään todentamaan. Esimerkiksi rikoksen tapahtuessa tulee lokitiedot olla saatavilla koko käsittelyjen ajan. Uuden tietojärjestelmän testauksesta kerätään yleensä virhetilanteiden ja kuormituksen selvittämiseksi lokitietoja, joita ei säilytetä kovin kauaa. (Lokiohje 2009, 60.) KATAKRIn kohdassa I 504.0 on viranomaisvaatimuksen mukaan määritetty keskeisten laitteiden ja palveluiden lokien säilytysajaksi perustasolla (IV) kuusi kuukautta tai erillisessä sopimuksessa määrätty aika (Kansallinen turvallisuusauditointikriteeristö 2011).

Lokitiedot pitää tuhota, kun niiden säilytysaika umpeutuu. Säilytysajan umpeutuessa on lokien tuhoaminen voitu automatisoida; vähimmäisajan tultua täyteen lokit tuhotaan alkuperäisestä tallennuspaikasta ja mahdollisesti arkistoidaan. Lokien tuhoamisessa tulee ottaa huomioon mahdolliset varmuuskopiot, jotka myös tuhotaan asianmukaisella tavalla. (Lokiohje 2009, 61.)

2.4.4 Suojaus

Lokien suojaaminen on tärkeää, sillä ne voivat sisältää arkaa tietoa organisaation henkilöistä, asiakkaista, järjestelmistä ja verkoista. Lokeja tulisi suojata tietoturvan kolmen pääperiaatteen mukaisesti: luottamuksellisuus, eheys ja saatavuus. Luottamuksellisuus tarkoittaa sitä, että lokitietoihin pääsee käsiksi vain sellainen henkilö, jolla on tarve käsitellä tietoja ja on siihen oikeus. Eheydellä pyritään varmistamaan tiedon tahaton tai tahallinen muuttumattomuus. Tiedon pitäisi myös olla aina saatavilla, kun sitä tarvitaan.

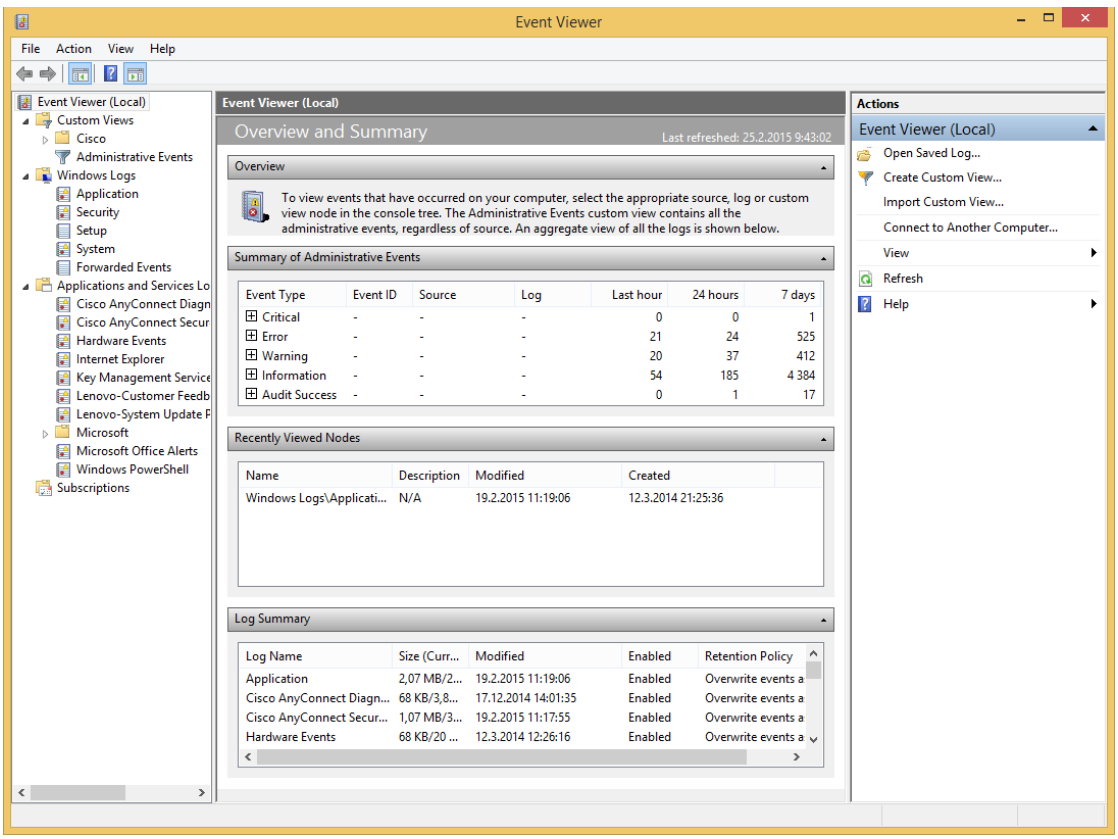
KATAKRIn kysymyksen I 504.0 kolmannessa kohdassa on käsitelty lokitietojen suojaamista: suojattavaa tietoa sisältävät lokitiedot on suojattu asianmukaisesti (pääsynvalvonta, käsittely, poisto) (Kansallinen turvallisuusauditointikriteeristö 2011). Lokien suojaaminen on tärkeää koko lokin elinkaaren ajan. Suojaaminen korostuu erityisesti liiku-

teltaessa lokeja paikasta toiseen sekä niitä säilytettäessä. Kaikissa käsittelyvaiheissa on pidettävä huoli siitä, että pääsynhallinta on kunnossa, jotta asiaankuulumaton henkilö ei pääse käsiksi lokitietoihin.

3 Windows Event log

3.1 Yleistä

Event log on Microsoft Windows -käyttöjärjestelmien oma lokijärjestelmä, joka on ollut käytössä aina 1990-luvun alkupuolelta Windows NT:n syntyajoilta asti. Event log -lokiviestejä ei voi tarkastella ilman erillistä sovellusta, koska se tallentaa ne järjestelmään binäärimuodossa. Lokien hallintaan ja tarkasteluun on Windows-käyttöjärjestelmissä sisäänrakennettu *Event Viewer* -sovellus (ks. kuvio 1), komentorivityökalu *wevtutil* sekä lisäksi on saatavilla myös ulkopuolisten tekemiä ohjelmia lokien tarkasteluun (Event Logs 2015).



Kuvio 1. Event Viewer -päänäkymä

Event log koki suuren uudistuksen Windows Vistan julkistamisen yhteydessä. Uudistuksella pyrittiin helpottamaan lokitietojen seulontaa ja seurantaa, parantamaan suorituskykyä ja skaalautuvuutta suurien tietomäärien varalle sekä tietoturvallisuutta. Yhtenä uudistuksena oli lokitapahtumien tietojen mukautuminen XML-merkintäkielen (Extensible Markup Language) rakenteeseen. Yhdessä XPath-hakukielen (XML Path Language) avulla se mahdollistaa tiettyjen kyselyiden luomisen suodattaen samalla epäolennaiset tapahtumat pois. XPath-hakukiellellä voidaan XML-pohjaisesta dokumentista hakea yksittäisiä osia tai tapahtumia tietyillä valintakriteereillä, jolloin tapahtumia voidaan tarkastella hyvinkin yksityiskohtaisesti. (Menn 2006.)

3.2 Lokityypit

Event logit voidaan jaotella kahteen kategoriaan, Windows-lokeihin sekä sovellus- ja palvelulokeihin. Windows-lokien tarkoitus on tallentaa tapahtumia legacy-sovelluksista ja tapahtumista, jotka vaikuttavat koko järjestelmään. Sovellus- ja palvelulokit tallentavat tapahtumia yksittäisistä sovelluksista ja komponenteista koko järjestelmään vaikuttavien tapahtumien sijaan. (Event Logs 2015.)

3.2.1 Windows-loki

Windows-lokit on jaoteltu viiteen eri osaan: application-, security-, setup-, system- ja forwardedevents-kategorioihin. Application-loki sisältää tietoja sovellusten ja ohjelmien tapahtumista. Security-loki kertoo tietoja kirjautumisista ja resurssien käytöstä, kuten tiedostojen luomisesta, avaamisesta tai poistamisesta. Setup-loki sisältää tiedot sovellusten asennuksista. System-loki kattaa tapahtumia Windows-järjestelmän komponenteista, kuten jonkin ajurin virheestä käynnistyksen yhteydessä. Forwardedevents-lokiin kuuluvat tiedot muista järjestelmistä. (Event Logs 2015.)

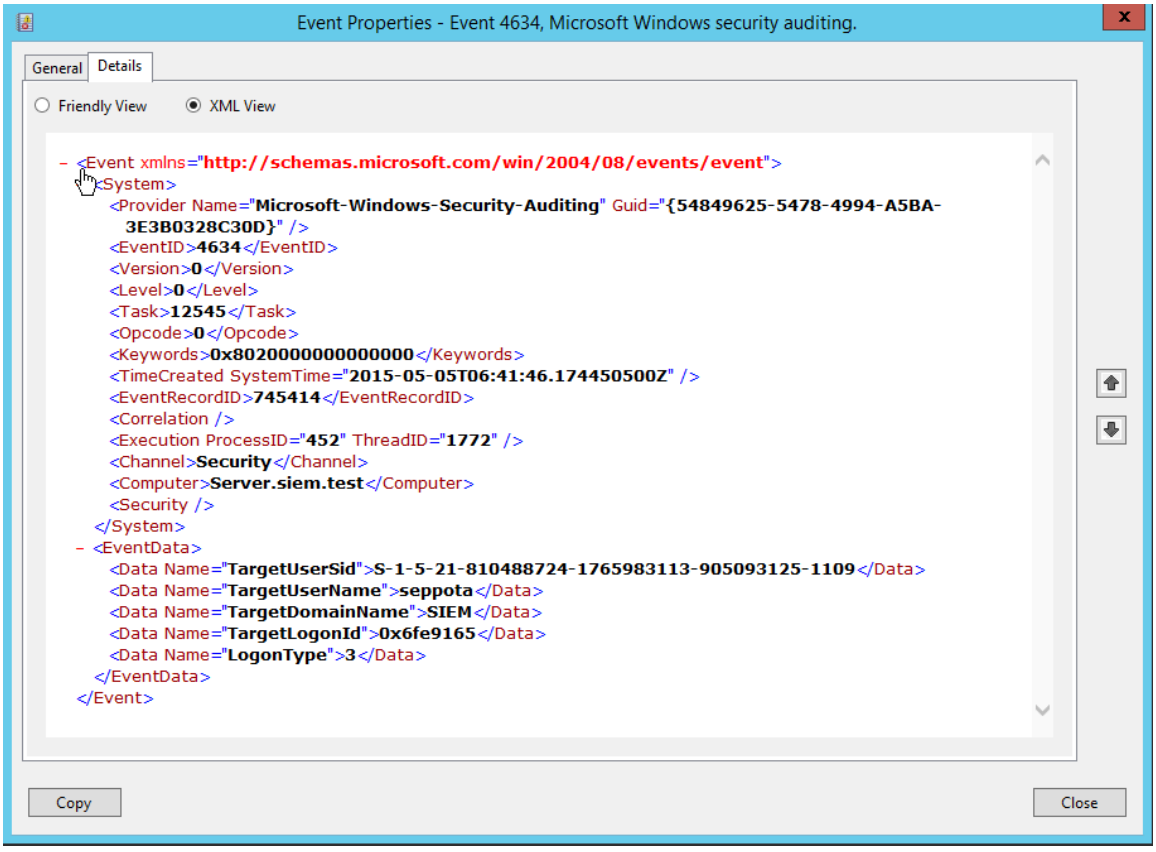
3.2.2 Sovellus- ja palveluloki

Sovellus- ja palvelulokit ovat puolestaan jaoteltu neljään eri osaan: admin, operational, analytic ja debug. Admin-tapahtumat on tarkoitettu ylläpitäjille ja tukihenkilöille, mutta myös loppukäyttäjille. Ne sisältävät ratkaisuja ongelmien vianmääritykseen ja antaa

suoria ehdotuksia ongelman korjaamiseksi, esim. kun sovellus ei saa yhteyttä skanneriin. Operational-tapahtumat on myös hyödyllinen IT-ammattilaisille, mutta ne vaativat todennäköisesti enemmän tulkitsemista. Tietoja käytetään ongelman tai tapahtuman analysointiin ja diagnosointiin, esim. kun skanneri poistetaan järjestelmästä. Analytic-tapahtumia syntyy suuria määriä, ja ne kuvaavat ohjelmien toimintaa ja osoittavat ongelmia, joihin käyttäjän toimilla ei voi vaikuttaa. Debug-tapahtumat on tarkoitettu sovelluskehittäjille ongelmienratkontaan. (Event Logs 2015.)

3.3 Rakenne

Event log muodostuu kahdesta osasta: System- ja EventData-osasta. System-osan tiedot koostuvat yleisistä tiedoista samanlaisten tapahtumien kesken sekä julkaisun yhteydessä kerätyistä järjestelmäparametreista. EventData-osa puolestaan sisältää jäsenneltyjä tietoja sovelluksista ja on laajennettavissa. Kuviossa 2 on XML-muotoinen kirjautumisloki, ja EventDatan kohdalta selviää mm. käyttäjätunnus (TargetUserName) ja domain (TargetDomainName). (Menn 2006.)



Kuvio 2. Event Viewer XML -näkymä

System-osa sisältää monia eri tietoja (ks. kuvio 2) tapahtumasta, mm. SystemTime, EventRecordID, ProcessID, ThreadID, Computer jne. EventID- ja Version-kentät määrittelevät keskenään tapahtuman, ja kaikki saman arvon tapahtumat ovat rakenteeltaan samanlaisia. Level-kenttä kuvaa tapahtuman vakavuutta tai laajuutta. Sen arvot ovat yleensä 1-5 taulukon 1 mukaisesti, mutta halutessaan tapahtuman tuottaja laajuudesta riippuen antaa omia arvojaan lukuun 255 asti. (Menn 2006.)

Taulukko 1. Event login Level -arvot

Level	**Value**
Critical	1
Error	2
Warning	3
Info	4
Verbose	5

Task-kentän arvo kuvastaa yleisellä alueella tapahtuman tuottajan toiminnallisuutta tai sovelluksen alikomponenttia. Opcode-kenttää käytetään tyypillisesti kuvaamaan ohjelman tiettyä toimintoa tai sen osaa ja yleisimmät arvot ovat joko 1 (Start) tai 2 (Stop). Keywords-kentässä on 56 lippua, jotka helpottavat samanlaisten tapahtumien ryhmittelyä. Yhdessä tapahtumassa voi olla monia lippuryhmiä osoittaen tapahtuman kuuluvan useisiin eri ryhmiin. (Menn 2006.)

4 Syslog

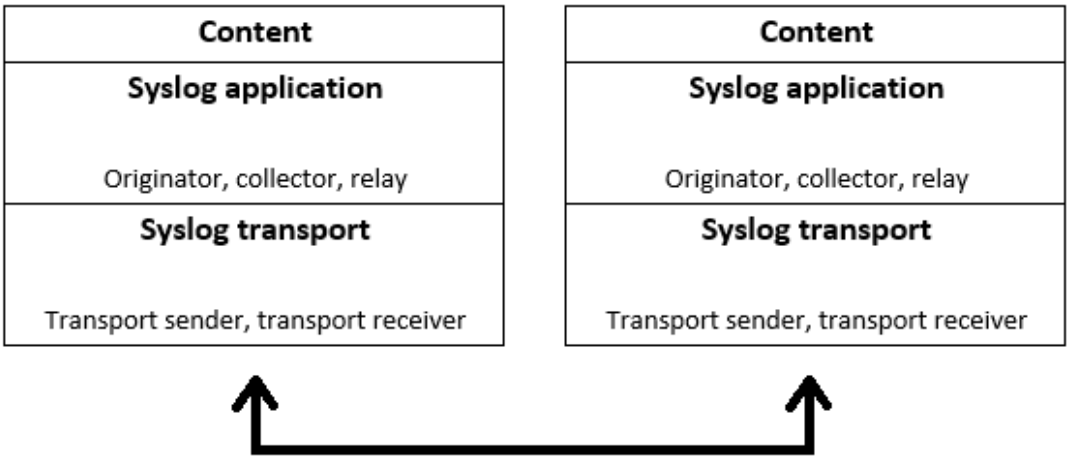
4.1 Yleistä

Syslog on lokiviestien hallintaan ja kuljetukseen tarkoitettu protokolla, ja sen on alun perin kehittänyt tietokoneohjelmoija Eric Allman (Lonvick 2001, 25). Protokolla yleistyi eri laitteissa, ja siitä oli olemassa monia eri versioita. Elokuussa vuonna 2001 IETF (Internet Engineering Task Force) julkaisi ensimmäinen RFC (Request for Comments) 3164-dokumentin ”The BDS Syslog Protocol”, joka standardoi Syslog-protokollan. Maaliskuus-

sa 2009 IETF julkaisi toisen RFC 5424-dokumentin ”The Syslog Protocol”, joka edelliseen verrattuna paransi protokollan tietoturvaa.

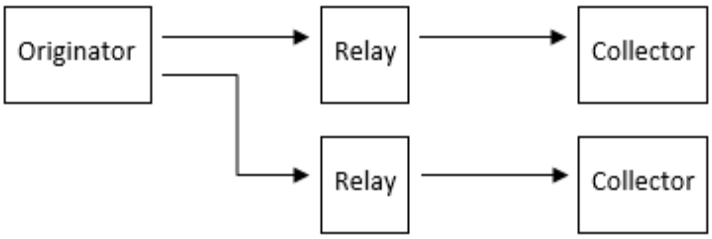
4.2 Rakenne

Syslog-protokolla pohjautuu 3-kerroksiseen arkkitehtuuriin, jossa viestin sisältö ja kuljetustapa on tarkoitus erottaa toisistaan ja samalla taata eri kerroksien helppo laajennettavuus. Kuviossa 3 on näkyvillä protokollan kolme eri tasoa ja niissä tapahtuvat toiminnot: content (sisältö), application (sovellus) ja transport (kuljetus). Content-kerros sisältää Syslog-viestin informaation, application-kerros käsittelee viestien tuottamista, tulkintaa, reititystä ja varastointia, ja transport-kerros huolehtii viestien kuljettamisesta paikasta toiseen. (Gerhards 2009, 4.)



Kuvio 3. Syslog-tasot ja -toiminnot

Eri kerroksilla tapahtuu erilaisia toimintoja. Application-kerroksessa originator (luoja) muodostaa viestin sisällön, collector (kerääjä) nimensä mukaisesti kerää sisällön analysointi varten ja relay (välittäjä) välittää ja hyväksyy viestejä luojilta ja muilta välittäjiltä ja lähettää ne eteenpäin kerääjille tai muille välittäjille. Transport-kerroksessa transport sender (lähettäjä) siirtää viestit kuljetusprotokollalle ja transport receiver (vastaanottaja) vastaanottaa viestit. Luojat ja välittäjät voidaan konfiguroida lähettämään sama viesti useille keräilijöille ja välittäjille ja nämä kaikki toiminnot voivat sijaita samassa järjestelmässä. Kuviossa 4 luoja lähettää viestin kahdelle eri välittäjälle, jotka välittävät sen eteenpäin keräilijöille. (Gerhards 2009, 5.)



Kuvio 4. Esimerkki Syslog-toiminnosta

Syslog-protokolla ei tarjoa kuittausta viestien toimituksista. Vaikka jotkut kuljetukset voivat antaa tilatietoja, on Syslog pelkästään kommunikaatioprotokolla. Sen sijaan Syslogin kuljetusprotokollat UDP (User Datagram Protocol) ja TCP (Transmission Control Protocol) on määritetty omissa RFC-dokumenteissaan. Jotta viestit kulkevat muuttumattomana lähettäjältä vastaanottajalle, on tärkeää, etteivät kuljetusprotokollat muokkaa viestejä. Jos viestejä kuitenkin täytyy muokata, tulevat muutokset palauttaa ennalleen ennen luovuttamista eteenpäin. (Gerhards 2009, 5-7.)

4.3 Viesti

4.3.1 Yleistä

Syslog-viestin minimipituus määritetään RFC 5424-dokumentissa 480 tavun mittaiseksi. Maksimipituutta ei ole määritetty ja se riippuu käytettävästä kuljetusprotokollasta. Syslog-viesti rakentuu kolmesta osasta, joita ovat PRI, HEADER ja MSG. (Gerhards 2009, 9.)

Kuviossa 5 on pfSense-palomuurin lokiviesti. Viesti sisältää paljon tietoa ja suurimmaksi osaksi se vastaa tässä osiossa tarkemmin esitettyä RFC-standardin viestimuotoa.

```
<134>Apr 29 10:22:13 filterlog:
59,16777216,,1000001581,em0,match,block,in,4,0x0,,128,787,0,none,
17,udp,78,10.x.x.x,10.x.x.x,137,137,58
```

Kuvio 5. pfSensen Syslog-viesti

4.3.2 PRI

PRI-osa kertoo viestin facility- ja severity-arvoista. Facility-arvo määrittää, mistä sovelluksesta tai prosessista viesti on peräisin. Severity-arvo puolestaan määrittää tapahtuman vakavuuden. PRI-osa on kolme, neljä tai viisi merkkiä, ja se sijoitetaan < > (suurempi ja pienempi kuin) -merkkien väliin. Merkkien sisällä oleva numero sisältää sekä facility- että severity-arvot, ja sitä sanotaan Priority-arvoksi (PRIVAL). Priority-arvo voi olla enintään kolme desimaalia pitkä, ja se lasketaan kertomalla facility-arvo kahdeksalla ja lisäämällä tulokseen severity-arvo. Molempien saadessa arvon nolla (0) tulee priority-arvoksikin nolla, jota ei saa käyttää. Taulukossa 2 on esiteltynä facility-arvot ja taulukossa 3 severity-arvot. (Gerhards 2009, 9-11.)

Taulukko 2. Syslog-viestin facility-arvot

Koodi	Facility
0	Kernel messages
1	User-level messages
2	Mail system
3	System daemons
4	Security/authorization messages
5	Messages generated internally by syslog
6	Line printer subsystem
7	Network news subsystem
8	UUCP subsystem
9	Clock daemon
10	Security/authorization messages
11	FTP daemon
12	NTP subsystem
13	Log audit
14	Log alert
15	Clock daemon (note 2)
16	Local use 0 (local0)
17	Local use 1 (local1)
18	Local use 2 (local2)
19	Local use 3 (local3)
20	Local use 4 (local4)
21	Local use 5 (local5)
22	Local use 6 (local6)
23	Local use 7 (local7)

Taulukko 3. Syslog-viestin severity-arvot

Koodi	Severity
0	Emergency: system is unusable
1	Alert: action must be taken immediately
2	Critical: critical conditions
3	Error: error conditions
4	Warning: warning conditions
5	Notice: normal but significant condition
6	Informational: informational messages
7	Debug: debug-level messages

4.3.3 HEADER

HEADER-osan formaatti on suunniteltu tarjoamaan yhteensopivuutta, ja sen tulisi RFC 5424-dokumentin mukaan koostua VERSION-, TIMESTAMP-, HOSTNAME-, APP-NAME-, PROCID- ja MSGID-kentistä. Toteutus vaihtelee kuitenkin melko paljon.

VERSION-kenttä ilmaisee protokollan käytössä olevan version. Jos johonkin HEADER-osaan tulee muutoksia, tulee VERSION-kenttää aina kasvattaa. TIMESTAMP-kenttä kertoo viestin aikaleiman. Aikaleiman tulee RFC 5424-dokumentin mukaan noudattaa muutamia rajoituksia. Karkaussekunteja ei saa käyttää, T-merkin käyttö on pakollista ja sekä T- että Z-merkit pitää merkitä suuraakkosin. Jos mahdollista, aikaleima tulisi esittää sekunnin murto-osien tarkkuudella tai NILVALUE-arvona, jos sovellus ei voi selvittää järjestelmäaikaa. (Gerhards 2009, 11–12.)

HOSTNAME-kenttä ilmaisee alkuperäisen Syslog-viestin lähettäjän koneen nimen. HOSTNAME-kentän tulisi sisältää lähettäjän host- ja domainname, joista käytetään nimitystä FQDN (Fully Qualified Domain Name). Kaikki Syslog-sovellukset eivät pysty tarjoamaan FQDN-nimeä, jolloin vaihtoehtoisestoisesti voidaan käyttää suositusjärjestyksessä lähettäjän staattista IP (Internet Protocol) -osoitetta, hostnamea, dynaamista IP-osoitetta tai NILVALUE-arvoa. (Gerhards 2009, 13.)

APP-NAME-kenttä tunnistaa laitteen tai sovelluksen, josta viesti on peräisin. Sen on tarkoitus suodattaa viestejä välittäjille (relay) tai kerääjille (collector). Jos viestin lähdettä ei tiedetä eikä voida selvittää, käytetään NILVALUE-arvoa. PROCID-kenttää käytetään usein

ilmaisemaan prosessin nimi tai Syslog-järjestelmän prosessi-id. Jos prosessi-id ei ole saataville, voidaan käyttää NILVALUE-arvoa. Prosessi-id:tä voidaan käyttää myös havaitsemaan viestejä, jotka kuuluvat samoihin ryhmiin. MSGID-kenttä tunnistaa viestin tyypin ja sillä voidaan suodattaa erityyppisiä tapahtumia. Esimerkiksi palomuuri voi käyttää saapuvalle TCP-liikenteelle MSGID-arvoa ”TCPIN” ja lähtevälle ”TCPOUT”. Viestien, joilla on sama MSGID-arvo, tulisi viitata aina samantyyppisiin tapahtumiin. NILVALUE-arvoa voidaan käyttää, kun sovelluksesta ei ole saatavilla mitään tietoa. (Gerhards 2009, 14–15.)

4.3.4 MSG

MSG-osa sisältää vapaamuotoisen viestin, joka tarjoaa tietoa lokitapahtumasta. Viestissä käytettävä merkistö tulisi olla UTF-8 koodattua. Jos Syslog-sovellus ei tue UTF-8 koodausta, on mahdollista käyttää muita koodauksia. (Gerhards 2009, 18.)

4.4 Toteutukset

4.4.1 Syslog-ng

Syslog-ng on Syslog-toteutus, jonka on kehittänyt BalaBit IT Security ja se on tarkoitettu UNIX-järjestelmille. Siitä on tarjolla kaksi versiota, joista ensimmäinen on ilmainen avoimen lähdekoodin OSE (Open Source Edition) ja toinen maksullinen, samaan pohjaan perustuva, ja lisäominaisuuksia sisältävä PE (Perium Edition). (The Foundation of Log Management 2015.)

Syslog-ng perustuu Syslog-standardien RFC 3164 ja RFC 5424 toimintoihin ja lisäksi mukautuu protokollien useisiin variaatioihin. Arkaluontoiset viestit pystytään salaamaan TLS (Transport Layer Security) -protokollan avulla ja TCP-protokollan avulla siirtämään epäluotettavienkin verkkojen yli. Syslog-ng avulla viestejä voidaan suodattaa ja lajitella niiden sisällön ja parametrien perusteella ja tallentaa tietokantatauluihin tai tiedostoihin. Vertailemalla lokiviestejä tiedettyihin malleihin pystytään tunnistamaan viestityyppejä ja sitä kautta luokittelemaan, räätälöimään ja merkitsemään tapahtumia. (Product features and benefits 2015.)

Syslog-ng OSE on saatavilla monille eri järjestelmäarkkitehtuureille, kuten x86, x86_64, SUN Sparc, PowerPC 32 ja 64, Alpha, ARM ja MIPS sekä käyttöjärjestelmille, kuten Linux, BSD, Solaris, IBM AIX, HP-UX jne. Lisäksi kaupallinen PE tarjoaa binääritiedostoja yli 40 alustalle, mm. Linux- ja UNIX-varianteille ja Windows-käyttöjärjestelmille. (Reliable log management 2015.)

4.4.2 Rsyslog

Rsyslog on avoimen lähdekoodin Syslog-toteutus, joka on tarkoitettu UNIX-tyyppisille järjestelmille. Vuonna 2004 Rainer Gerhards päätti tehdä uuden toteutuksen kilpailemaan Syslog-ng kanssa lisäämään ”monikulttuurisuutta ja valinnan vapautta” (Gerhards, R. 2007). Rsyslog perustuu ilmaisversioon, mutta sille on saatavilla maksullista tukea erilaisten pakettien muodossa (Professional Services 2015).

Syslog-ng tapaan Rsyslog perustuu Syslog-stardardeihin. TLS- ja TCP-tuen lisäksi SSL (Security Socket Layer) ja RELP (Reliable Event Logging Protocol), suodattaminen, lajittelu, tallentaminen ja joustavuus kuuluvat Rsyslog ominaisuuksiin. (Features 2015b.) Monet UNIX-pohjaiset käyttöjärjestelmät ja alustat tukevat Rsyslogia. Ainakin Debian, Ubuntu ja Red Hat käyttävät sitä oletuksena. Se saatavilla myös Solaris-, AIX-, BSD- ja HP-UX-alustoille. (Platforms 2009.)

5 Common Log Format ja Extended Log Format

Common Log Format (CLF) on standardisoitu lokiformaatti, jota käytetään yleensä web-palvelimissa lokitietojen kirjaamiseen. Standardoinnin ansiosta CLF soveltuu suurimpaan osaan työkaluista, joilla lokitietoja analysoidaan. Kuviossa 6 on Apache-palvelimen Combinet Log Format -tyyppinen lokiviesti, joka CLF-tyypin lisäksi sisältää kaksi lisäkenttää, asiakkaan URL (Uniform Resource Locator), joka liittyy sivuun sekä tietoja asiakkaan järjestelmästä. Ennen lisäkenttiä viestissä näkyvät asiakkaan IP-osoite, aikaleima, HTTP (Hypertext Transfer Protocol) -pyyntö, status ja vastauksen koko. (Hallam-Baker & Behlendorf n.d.)

```
192.168.1.101 - - [25/Mar/2015:11:09:55 +0200] “GET / HTTP/1.1”
200 3594 “-“ “Mozilla/5.0 (Windows NT 6.3; WOW64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.101
Safari/537.36”
```

Kuvio 6. Combined Log Format -viesti

Extended Log Format (ELF) on lokiformaatti, joka kehitettiin laajentamaan CLF-tyyppiä ja tarjoamaan enemmän tietoa ja joustavuutta. Samaan tapaan kuin CLF, ELF on ymmärrettävissä useimmissa analysointisovelluksissa. Jokaisen lokitiedoston alussa on selitetty kenttien tarkoitus, joka parantaa sen käytettävyyttä myös erilaisissa kustomoiduissa muodoissa. Kuviossa 7 on Windows-palvelimen Internet Information Services (IIS) -palvelun tuottama ELF-lokiviesti, joka sisältää tietoja, kuten aikaleima, sivu, koneen nimi, asiakkaan IP-osoite jne. (Hallam-Baker & Behlendorf n.d.)

```
#Software: Microsoft Internet Information Services 8.5
#Version: 1.0
#Date: 2015-03-25 11:37:02
#Fields: date time s-sitename s-computername s-ip cs-method cs-
uri-stem cs-uri-query s-port cs-username c-ip cs-version cs(User-
Agent) cs(Cookie) cs(Referer) cs-host sc-status sc-substatus sc-
win32-status sc-bytes cs-bytes time-taken
2015-04-22 06:46:26 W3SVC1 Server 192.168.1.10 GET / - 80 -
192.168.1.101 HTTP/1.1
Mozilla/5.0+(Windows+NT+6.3;+WOW64)+AppleWebKit/537.36+(KHTML,+li
ke+Gecko)+Chrome/42.0.2311.90+Safari/537.36 - - 192.168.1.10 304
0 0 141 471 685
```

Kuvio 7. Extended Log Format -viesti

6 Elasticsearch

6.1 Yleistä

Elasticsearch on avoimeen lähdekoodin perustuva hakupalvelin. Sen on kehittänyt Shay Banon vuonna 2010, kun hän oli aikeissa tehdä kolmannen version Compass-hakukoneesta. Banon halusi uuteen versioon ominaisuuksia, jotka olisivat vaatineet suurelta osin Compassin uudelleenkirjoittamisen, joten oli kannattavampi tehdä kokonaan uusi toteutus, Elasticsearch. (Banon 2010.)

Elasticsearch on käytännössä tietokantapalvelin, joka on kirjoitettu Java-ohjelmointikielellä ja se perustuu laajalti käytössä olevaan Apache Lucene -hakusovellukseen. Elasticsearchin voidaan sanoa olevan Lucenen ympärille rakennettu osa, joka käyttää samoja algoritmeja tiedon hakemiseen ja tallentamiseen, mutta se tarjoaa paremman ohjelmointirajapinnan, skaalautuvuuden ja toimintavälineet. Elasticsearch on suunniteltu REST (Representational State Transfer) -ohjelmistorajapintaan ja kommunikoi HTTP-protokollan avulla, mahdollistaen tietojen luomisen, lukemisen, päivittämisen ja poistamisen. (Cholakian 2013.)

Elasticsearch tallentaa tiedot optimoituun muotoon, johon on helppo tehdä hakuja. Se soveltuukin käytettäväksi tarkoituksiin, jossa suuresta tietomäärästä halutaan tehdä nopeita hakuja. Hauista palautetaan likimääräinen vastaus, jolloin vastauksen ei tarvitse tarkasti vastata hakulauseketta. Sillä pystytään myös tarkkoihin hakuihin. Ero perinteisen tietokannan ja Elasticsearchin välillä onkin juuri likimääräiset vastaukset sekä tarkkuus ja eheys. (Cholakian 2013.)

6.2 Rakenne

Elasticsearch-tietokannan pienin yksikkö on kenttä, joka sisältää tietylle tyypille yhden tai useamman arvon. Kenttä voi olla esim. numero tai merkkijono. Dokumentti on perustallennusyksikkö, joka sisältää kokoelman kenttiä. Vaikka kenttä on ohjelmointirajapinnan kannalta pienin yksikkö, saa dokumentti perustallennusyksikkö-tittelin ollessaan

pienin yksikkö varastoinnin kannalta. Jokaisen dokumentin tulee vastata käyttäjän asettamaa tyyppikartoitusta. Se määrittää kenttien tyypin ja tavan, jolla niiden ominaisuuksia indeksoidaan. Taulukossa 4 on esitelty Elasticsearch-tietokannan perustietotyypit. (Cholakian 2013.)

Taulukko 4. Elasticsearch tietotyypit

Tyyppi	**Määritelmä**
string	teksti
integer	32 bittinen kokonaisluku
long	64 bittinen kokonaisluku
float	liukuluku
double	64 bittinen liukuluku
boolean	tosi/epätosi
date	UTC päivämäärä/aika
null	nolla-arvo

JSON (JavaScript Object Notation) -formaattia käytetään Elasticsearch-tietokannassa tiedon muotoiluun. JSON on tiedostomuoto tiedonvälitykseen, jota ihmisten on helppo lukea ja kirjoittaa sekä koneiden helppo jäsentää ja tuottaa. Nimestään huolimatta JSON on ohjelmointikielestä riippumaton ja se kostuu kahdesta rakenteesta: kokoelmasta nimi/arvo-pareja ja järjestetystä listasta arvoja. Joitakin kenttiä Elasticsearch varaa omaan käyttöönsä, esim. _id-kentän, joka tekee dokumentista ainutlaatuisen. Kuviossa 8 on esimerkki JSON-muotoilusta. (Cholakian 2013.)

```
{
”_id”: 1,
”handle”: ”samppa”,
”age”: 25,
”hobbies”: [“rinkball”],
“computer”: {“cpu”: “intel i5”, “mhz”: 2300}
}
```

Kuvio 8. JSON-esimerkki

Suurin yksikkö Elasticsearch-tietokannassa on indeksi. Indeksit ovat fyysisiä ja loogisia osioita Elasticsearchin sisällä. Dokumentit ja dokumenttityypit ovat ainutlaatuisia indek-

sien sisällä. Tietoa pystytään hakemaan yhdestä tai useammasta indeksistä ja niiden rakenteen ansiosta mahdollistetaan tehokkaat ja nopeat haut. (Cholakian 2013.)

7 Logstash

7.1 Yleistä

Logstash on työkalu, joka on tarkoitettu tapahtumien- ja lokienhallintaan. Sillä voi kerätä, jäsentää ja tallentaa lokiviestejä useista eri lähteistä. Logstash perustuu avoimeen lähdekoodiin ja on saatavilla ilmaiseksi Apache 2 -lisenssin mukana, joten käytännössä se on toteutettavissa vapaasti lähes missä tahansa. Toimiakseen Logstash vaatii ainoastaan Javan, jonka alustariippumattomuuden ansiosta myös Logstash toimii useilla eri käyttöjärjestelmillä. (Logstash 2015.)

Logstashin toiminta perustuu kolmeen eri osaan, joita ovat inputs, filters ja outputs. Inputs eli sisääntulot määrittelevät sen, mistä ja miten lokiviestejä otetaan vastaan. Filters eli suodattimet jäsentelevät ja muotoilevat viestejä. Viesteistä voidaan erotella haluttuja kenttiä, niitä voidaan muotoilla tai halutessa jopa kokonaan poistaa. Outputs eli ulostulot määrittelevät viestien jatkokäsittelyn, jos viestejä halutaan esim. tallentaa tai lähettää eteenpäin seuraaville osapuolille. Kaikki osat sisältävät valmiita lisäosia. Lisäosia voi olla käytössä yhtä aikaa useita ja niillä pystytään tietyn tyyppisistä viesteistä erottelemaan tietyntyyppisiä tapahtumia.

7.2 Kibana

Kibana on verkkokäyttöliittymä, joka on sisäänrakennettu mm. Logstash-lokienhallintatyökaluun. Se käyttää myös Elasticsearch-tietokantaa hyödykseen viestien hakemisessa. Kibanan avulla voidaan lokiviesteistä hakea ja suodattaa sisällön tai nimen perusteella haluttuja tapahtumia ja muodostaa saaduista tuloksista helposti tulkittavia graafisia kuvaajia. Tapahtumia voidaan seurata reaaliajassa tai haut voidaan rajata halutulle aikavälille. Viestien sisällöstä voidaan tehdä erilaisia päätelmiä, mistä lähteestä

saapuu paljon tapahtumia ja ovatko ne mahdollisesti jonkun tietyn sovelluksen. Viestimäärien muutokset pystytään graafeista havaitsemaan, joka auttaa mahdollisten ongelmatilanteiden selvittämisessä. (Kibana 2015.)

8 Nagios Log Server

8.1 Yleistä

Nagios Log Server (NLS) on Nagios Enterprisesin lokakuussa vuonna 2014 julkistama sovellus lokien hallintaan ja analysointiin. Nagios Enterprises tarjoaa yritystason tuotteita, palveluja ja ratkaisuja IT-infrastruktuurin valvontaan ja omaa vahvan aseman monitorointityökalujen tarjoajana. NLS perustuu maksulliseen versioon, mutta siitä on saatavilla ilmainen 60-päivän kokeiluversio, jonka aikana voi tehdä päätöksen mahdollisesta hankinnasta.

Nagios Log Server on tehty ELK (Elasticsearch – Logstash – Kibana) -tuotteiden pohjalta, jossa Elasticsearch on lokitietojen kerääjä ja tallennuspaikka, Logstash jäsentelee ja suodattaa niitä ja Kibana tarjoaa verkkokäyttöliittymän. Tuotekehityspäällikkö Scott Wilkersonin mukaan monien eri ilmaisohjelmien sijasta he halusivat tehdä valmiin lokienhallintatuotteen, jossa ongelmien syntyessä on saatavilla kaupallinen tuki. Vaikka NLS onkin maksullinen sovellus, Wilkerson muistuttaa ilmaisohjelmien vaatimista työtunneista konfiguroinnin, resurssien ja ylläpidon osalta ja suosittelee heidän valmista, lähes kaikenlaisia lokilähteitä tukevaa monikäyttösovellusta. (Wilkerson 2014.)

8.2 Ominaisuudet

NLS on suunniteltu keräämään, analysoimaan ja varastoimaan lokitietoja ja tarjoamaan käyttäjilleen hyvän käsityksen verkkonsa tapahtumista. NLS tarjoaa yhtiön omien sanojen mukaan käyttäjilleen monipuolisia etuja:

- **Helppokäyttöisyys**. NLS yksinkertaistaa lokitietojen etsintää. Hälytyksien avulla voi havaita uhkia tai suodattamalla dataa voi nopeasti tarkastaa järjestelmän.

NLS:n avulla kaikkien laitteiden lokitiedot ovat helposti saatavilla yhdessä paikassa.

- **Skaalautuvuus**. Yrityksen kasvaessa pystyy NLS skaalautumaan sen mukana. Klusteriin eli toimintaryppääseen voidaan lisätä instansseja, toimintayksiköitä, joka mahdollistaa enemmän tehoa, nopeutta, tallennustilaa ja luotettavuutta monitorointijärjestelmään. NLS on suunniteltu niin pienille kuin suurillekin organisaatioille.
- **Reaaliaikaisuus**. NLS näyttää lokitiedot kaikista laitteista reaaliajassa ja tarjoaa käyttäjälle mahdollisuuden nopeisiin analyyseihin ja ratkaisuihin ongelmien ilmetessä.
- **Turvallisuus**. Ongelmien ilmetessä pystytään kyselyiden avulla tärkeimmistä tapahtumista tekemään hälytyksiä, joita voidaan ohjata eteenpäin esim. Nagios XI tai Nagios Core -monitorointisovelluksille, SNMP (Simple Network Management Protocol) -trap-viestillä, sähköpostilla tai halutessa voidaan suorittaa skriptejä ongelman nopeaksi ratkaisemiseksi.
- **Käyttäjien hallinta**. Monikäyttäjän ominaisuudet mahdollistavat työryhmien työskentelyn tehokkaasti yhdessä. Ylläpitäjä voi lisätä, muokata ja poistaa käyttäjiä tai asettaa rajoituksia, jolloin käyttäjä pystyy muokkaamaan vain omaa profiiliaan. Vastaavasti ylläpitäjä voi antaa oikeuksia, jotta käyttäjä pääsee muokkaamaan työryhmän yhteistä profiilia ja muutokset ovat yhteisesti nähtävillä.
- **Dashboardit**. Verkkokäyttöliittymässä on helppo luoda dashboardeja (kojelautoja), joilla visualisoidaan kyselyjen ja filtterien avulla halutuista lokiviesteistä havainnollisia graafeja. Käyttäjät voivat tallentaa omia dashboardeja yleisiksi, jolloin toiset käyttäjät pystyvät myös hyödyntämään niitä.
- **Tietoisuus**. NLS tarjoaa käyttäjälle paljon tietoa verkon infrastruktuurista; mm. tapahtumat, lokit ja suorituskyky. Lisäksi se tarjoaa todisteita turvallisuusuhkista ja haavoittuvuuksien nopeaa ratkaisemista jo aikaisemmin mainittujen hälytyksien ja ilmoitusten avulla.

(Overview 2015.)

Etujen lisäksi NLS tarjoaa myös muutamia mainitsemisen arvoisia ominaisuuksia. Palvelinklusteroinnin avulla lokitietoja tallennetaan automaattisesti ja ehkäistään tietojen hä-

viämistä vikatilanteissa ja varmistetaan niiden eheys ja saatavuus. Konfiguraatiovelhon avulla on helppo lisätä laitteita tai palveluita monitoroitavaksi ja yhtiö väittääkin NLS:n olevan yhteensopiva lähes kaikkien järjestelmien kanssa. Lokitietojen sijaitessa yhdessä paikassa, on tietyn tapahtuman etsiminen suodattimien ja kyselyiden avulla yksinkertaisempaa. NLS sopeutuu kaikenlaisiin ympäristöihin ja kykenee yhteistyöhön niin Nagioksen omien kuin kolmansien osapuolien tuotteiden kanssa. (Features 2015a.)

8.3 Hinnoittelu

Toisin kuin useimmilla muilla lokienhallintasovelluksilla, joiden hinnoittelu perustuu kerättävään lokimäärään, NLS:n hinnoittelu perustuu klusterissa käytettävien instanssien määrään. Jos käytössä on vain yksi instanssi ja kerättävät lokitiedot ovat kooltaan keskimäärin päivässä alle 500MB, on tuote ilmainen. Kokeilukäytössä yksi instanssi on varmasti riittävä, mutta yritys, joka aikoo hankkia lokienhallintasovelluksen, haluaa varmasti varmistaa käytettävyyden ja vikasietoisuuden käyttämällä vähintään kahta instanssia. Taulukossa 5 on nähtävillä NLS:n hintatiedot. (Pricing 2015.)

Taulukko 5. Nagios Log Serverin hintatiedot

Instanssien määrä	Hinta
2	995 $
3	1495 $
4	1995 $
5	2495 $

Lisenssin mukana tulee pääsy tukifoorumille, joka on tarkoitettu vain maksaville asiakkaille. Instanssien määrästä riippuen tarjolla on tukea tietylle määrälle ongelmatapauksia sähköpostitse ja lisäksi sähköpostitukea voi ostaa kolmen ongelmatapauksen tukipakettina hintaan 750$. Yhtiö tarjoaa lisämaksusta myös puhelintukipaketteja, jotka ovat käytössä koko lisenssin voimassaoloajan. Viiden ongelmatapauksen puhelupaketti maksaa 995$ ja kymmenen 1495$. (Pricing 2015.)

Splunk Enterprise on vastaavanlainen lokienhallintasovellus kuin NLS ja sen ominaisuudet ovat melko samat. Splunkista on tarjolla muuttamia eri lisenssejä: Free, Light ja En-

terprise. Free on ilmainen, mutta melko karsittu versio, Light on tarkoitettu pienille yrityksille ja Enterprise kattava hallintasovellus ja samalla kallein vaihtoehto. Splunk Enterprisen hinnoittelu perustuu käänteisesti lokimääriin ja niitä on esitelty taulukossa 6. (Splunk Pricing 2015.)

Taulukko 6. Splunk Enterprisesin hintatiedot

Lokimäärä/päivä	Ikuinen lisenssi	Vuotuinen lisenssi
1GB	4500$	1800$
10GB	2500$	1000$
50GB	1900$	760$
100GB	1500$	600$

Splunk-hinnat ovat yhdelle instanssille ja niiden päälle lisätään vuosittaiset tukimaksut. Jos NLS:n ja Splunkin välillä vertaa hintoja pienillä viestimäärillä, on niissä selvä ero NLS:n eduksi. Suurissa organisaatioissa ja suuremmilla lokimäärillä hintaero pienenee ja jopa tasoittuu, mutta jos asiaa tarkastellaan instanssien lukumäärän kannalta, tulee Splunk usealla lisenssillä varustettuna melko kalliiksi.

Jotta sovellusten todelliset erot, toimivuus ja ominaisuudet pelkän hintavertailun lisäksi tulisivat esille, pitäisi myös Splunk ottaa tarkempaan tarkasteluun ja kokeiluun. Siihen ei tässä työssä kuitenkaan ryhdytty.

8.4 Asentaminen

Tällä hetkellä Nagios Log Serverin asentaminen on mahdollista vain RHEL- ja CentOS -käyttöjärjestelmille, mutta tarkoituksena on saada sovellus toimimaan myös muilla alustoilla. RHEL on Red Hat -ohjelmistoyhtiön kaupallinen käyttöjärjestelmä ja CentOS puolestaan RHEL:iin pohjautuva avoimen lähdekoodin käyttöjärjestelmä. CentOS oli entuudestaan tuttu, ilmainen ja helposti saatavilla, joten valinta kohdistui siihen.

Taulukossa 7 on esitetty laitteistovaatimukset NLS-järjestelmälle. Minimivaatimuksena ovat 1-ydin-suoritin, 2 gigatavua keskusmuistia ja 40 gigatavua kovalevytilaa. NLS-järjestelmälle kuitenkin suositellaan 4-ydin-suoritinta, 8 gigatavua tai enemmän keskusmuistia ja 1000 gigatavua tai enemmän kovalevytilaa. Ympäristöstä johtuen, jota kä-

sitellään toteutus-osiossa, NLS-järjestelmälle ei saatu edes minimi-kohdassa määritettyä laitteistoa. NLS-järjestelmä toimi 1-ydin-suorittimella, 1 gigatavulla keskusmuistia sekä 20 gigatavun kovalevyllä.

Taulukko 7. NLS:n laitteistovaatimukset

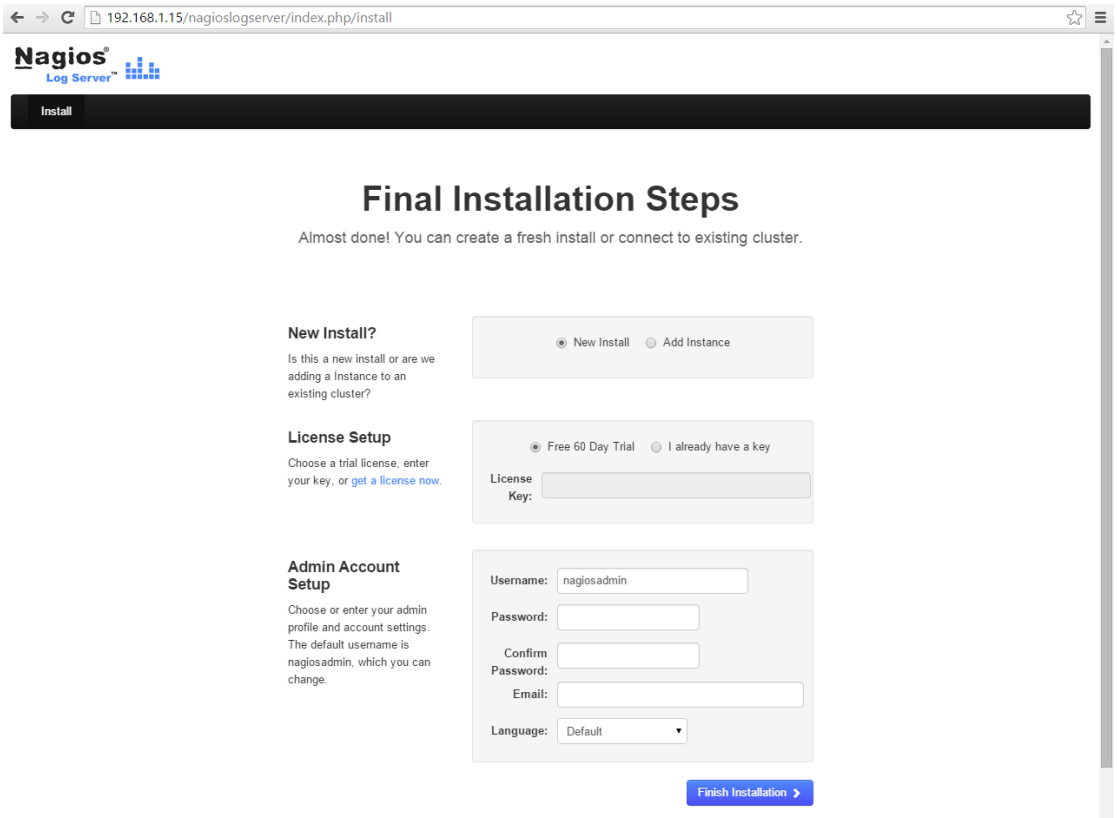
	Suoritin	**Muisti**	**Levytila**
Minimi	1-ydin	2 GB	40 GB
Suositus	4-ydin	8+ GB	1+ TB

Kun CentOS 6.6 oli asennettu, päivitetty ja valmiina käyttöön, pystyttiin myös NLS asentamaan komentokehotetta käyttäen.

```
cd /tpm
wget http://assets.nagios.com/downloads/nagios-log-
server/nagioslogserver-latest.tar.gz
tar xzf nagioslogserver-latest.tar.gz
cd nagioslogserver
./fullinstall
```

Ensiksi ladattiin NLS:n viimeisin versio Nagioksen internetsivuilta */tmp*-kansion alle. Seuraavaksi ladattu paketti purettiin, siirryttiin nagioslogserver-kansioon ja lopuksi käskettiin sovellusta asentumaan itsenäisesti.

Asennuksen viimeistelemiseksi piti selaimella navigoida osoitteeseen *http://192.168.1.15/nagioslogserver/*, jossa IP-osoite 192.168.1.15 on toteutus-osiosta löytyvän taulukon 8 mukaisesti NLS-palvelimen IP-osoite. Selaimessa aukesi kuvion 9 mukainen näkymä, jossa asennus oli tarkoitus viimeistellä.



Kuvio 9. Nagios Log Server – asennuksen viimeistely

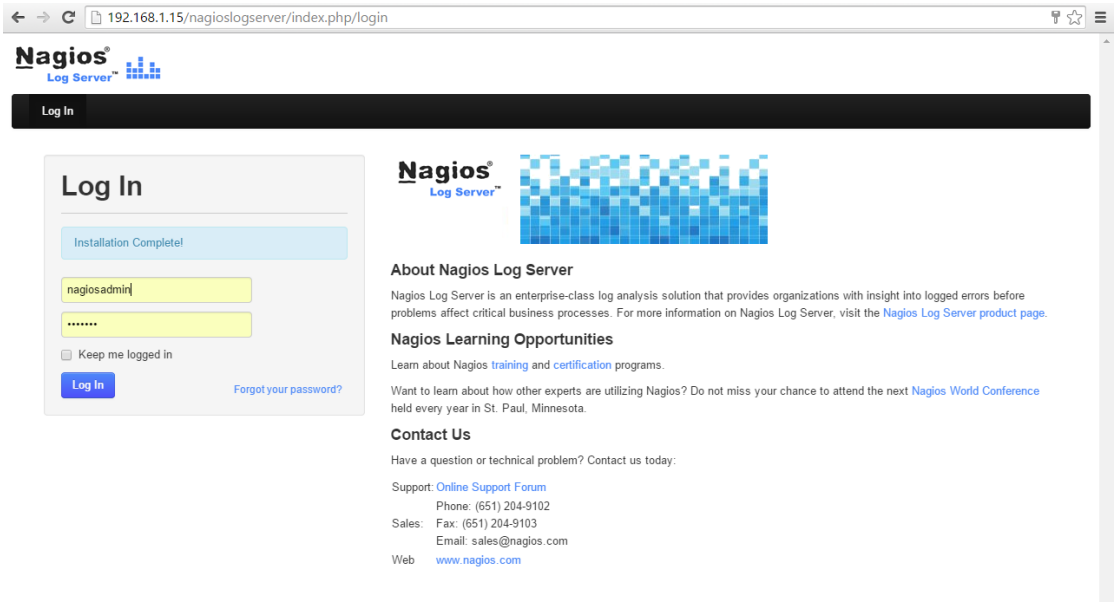
Viimeistelyssä pystyttiin valitsemaan, oliko kyseessä uusi asennus (New Install) vai lisättiinkö jo olemassa olevaan klusteriin uusi instanssi (Add Node). Tässä kohtaa valittiin uusi asennus, eikä toista instanssia keritty lopulta asentamaan. Seuraavaksi valittiin 60-päivän kokeiluversio. Jos käytössä oli maksettu lisenssi, pystyi sen syöttämään lisenssiavain-kenttään. Lopuksi täytettiin admin-käyttäjän asetukset, joita olivat käyttäjänimi, salasana, sähköposti ja käyttöliittymän kieli. Tämän jälkeen NLS oli asennettu ja valmiina käytettäväksi.

8.5 Käyttöliittymä

Nagios Log Serveriä käytetään verkkokäyttöliittymän avulla. Siellä nimensä mukaisesti tapahtuu lähes kaikki lokien hallintaan ja analysointiin liittyvät toimenpiteet. Käyttöliittymän kautta lisätään lähteitä, joista lokeja halutaan kerätä ja niitä pystytään tarkastelemaan, sieltä pääsee Nagios Enterprisesin tarjoamille tukisivustoille, nähdään palvelimesta tärkeitä tietoja sekä sen kautta tehdään konfiguraatiot mm. sisääntuloille, filtte-

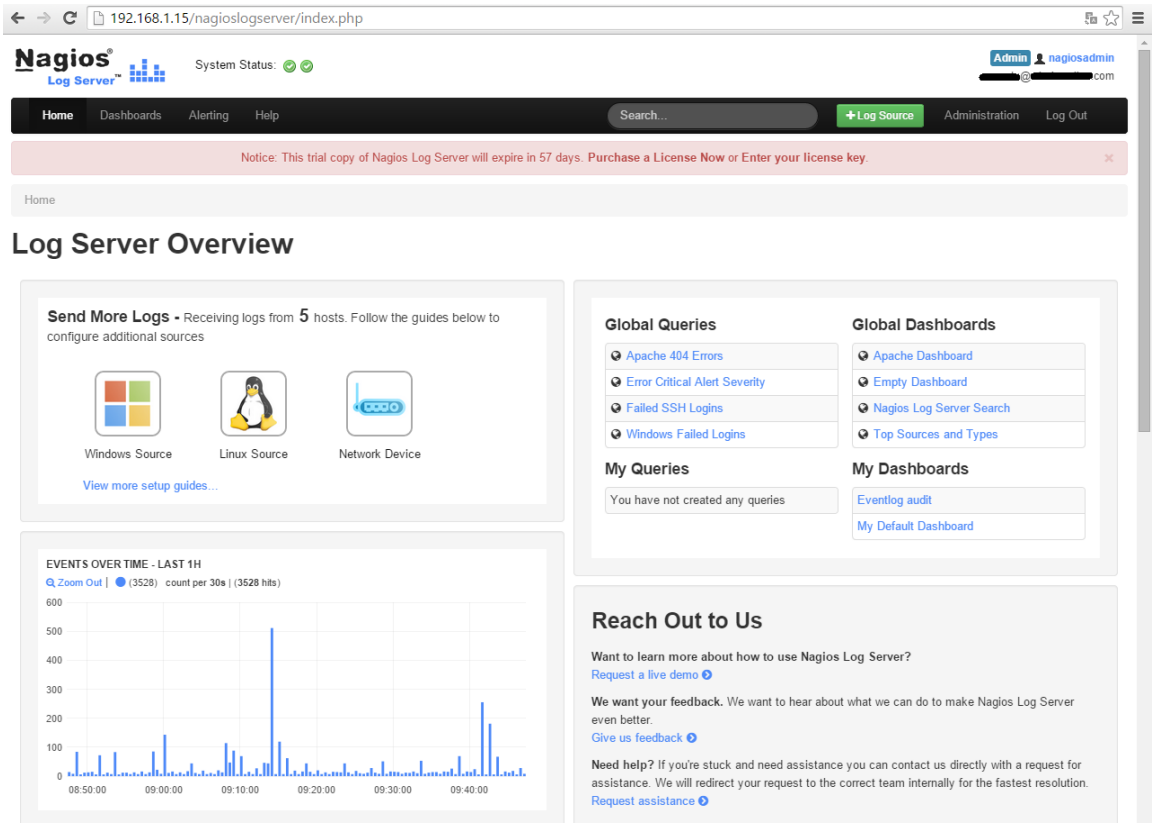
reille ja ulostuloille. Seuraavaksi on esitelty tarkemmin oleellisia näkymiä käyttöliittymästä.

Käyttöliittymään päästään käsiksi selaimella. Samaan tapaan kuin asennuksen viimeistelyssä, navigoidaan osoitteeseen *http://192.168.1.15/nagioslogserver/*, jossa NLS:n IP-osoite saadaan taulukon 8 mukaisesti. Asennuksen viimeistelyn onnistumisen jälkeen osoitteesta löytyy kuvion 10 mukainen kirjautumissivu. Ensimmäisen kirjautumiskerran ollessa kyseessä, sisään päästään asennuksessa luoduilla admin-käyttäjän tunnuksilla. Admin-käyttäjä pystyy lisäämään käyttäjiä, jolloin myöhemmin kirjautuminen onnistuu myös muilla tunnuksilla.



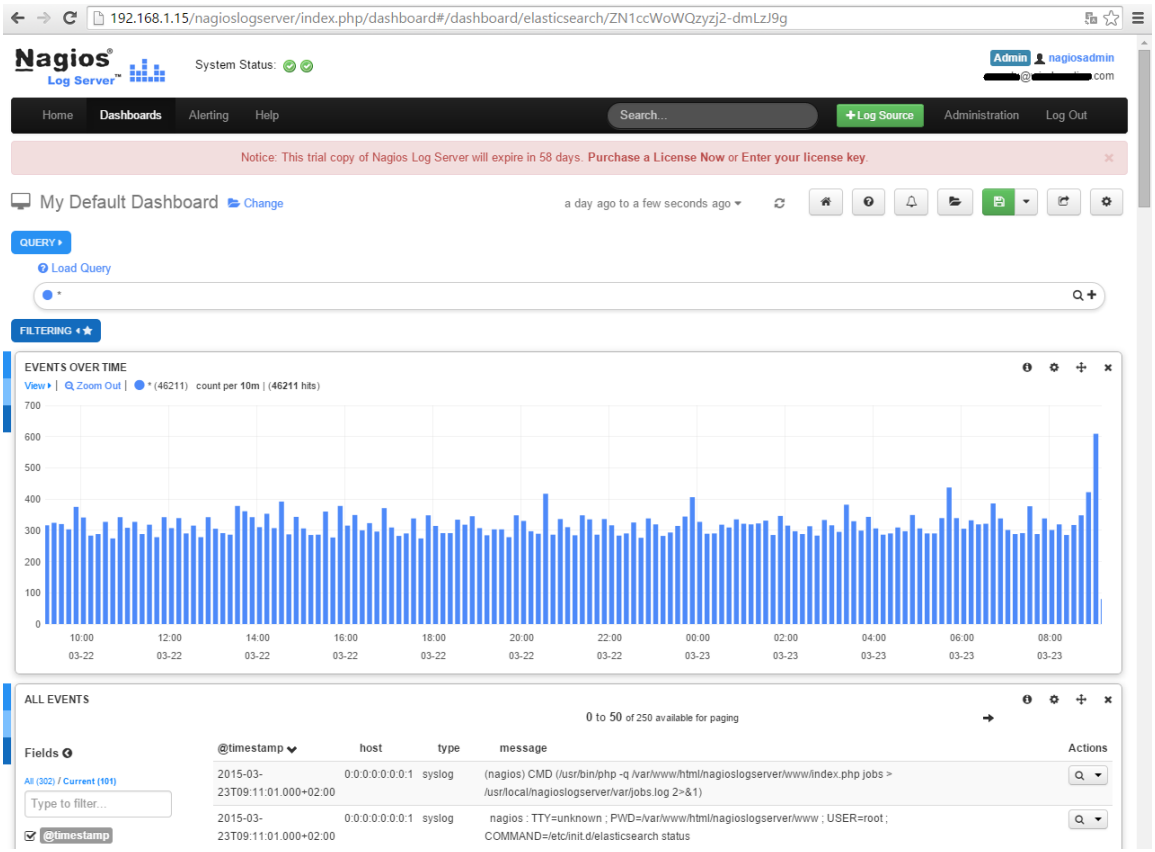
Kuvio 10. Nagios Log Server – kirjautumissivu

Onnistuneen kirjautumisen jälkeen eteen aukeaa kuvion 11 mukainen aloitussivu, joka näyttää yleiskatsauksen NLS:n toiminnasta. Se kertoo, kuinka monesta lähteestä lokeja kerätään ja miten lähteiden lisääminen onnistuu. Grafiikka näyttää saapuneiden lokiviestien määrän viimeisen tunnin ajalta. Aloitussivulta pystyy helposti siirtymään jaettuihin tai itse tallentamiin kyselyihin ja dashboardeihin, joissa lokiviestejä pystytään tarkastelemaan paremmin ja yksityiskohtaisemmin. Aloitussivulta näkee myös ajankohtaiset päivitykset, Nagios Enterprisesin uusimmat uutiset sekä yhteydenottomahdollisuuksia ongelmien ilmetessä tai jos käyttäjä haluaa saada lisätietoa sovelluksesta tai antaa palautetta.



Kuvio 11. Nagios Log Server – aloitussivu

Kuviossa 12 näkyvä dashboard-välilehti on todella tärkeä ja oleellinen osa lokien analysoinnin ja visualisoinnin kannalta. Siinä nähdään reaaliajassa saapuvat lokiviestit ja niiden sisältö. Lokeja pystytään tarkastelemaan esim. viimeisen viiden minuutin, 15 minuutin, tunnin, kahden tunnin jne. ajalta tai tietyltä aikaväliltä jopa kuukausia taaksepäin, riippuen siitä kuinka kauan lokitietoja säilytetään palvelimella.



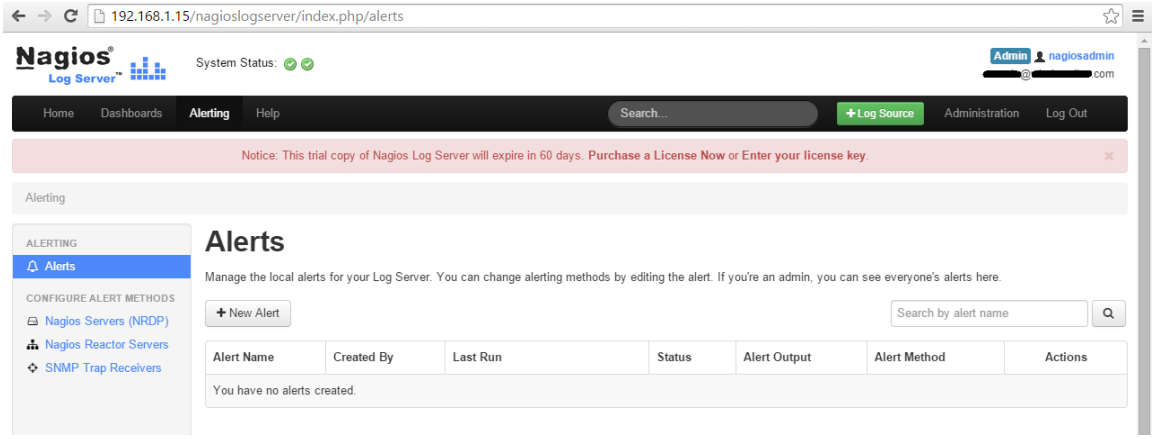
Kuvio 12. Nagios Log Server – dashboards-välilehti

Viestejä voidaan suodattaa eri tavoin ja saada näkymään vain halutut tapahtumat, esim. viestin tyypin, lähettäjän tai eri tunnisteiden perusteella. Suodatettuja viestejä voidaan kyselyiden avulla edelleen erotella ja lajitella arvojensa perusteella. Suodattimia ja kyselyitä on mahdollista tallentaa, jolloin haluttuihin viesteihin on helppo palata ja tarkastella niitä. Suodattimista ja kyselyistä muodostuvat dashboardit, joita NLS sisältää valmiina muutamia. Lokiviestejä saa niiden avulla havainnollistettua eri tavoin, kuten listaamalla lokiviestien lähteet, vertaamalla lokiviestien määrää tai tyyppejä lähteiden kesken ja tekemällä niistä erilaisia diagrammeja, kuvioita tai karttoja. Dashboardeja on helppo tehdä myös itse lisää ja niitä on mahdollista tallentaa sekä itselle että muille käyttäjille nähtäväksi.

Suodattimista, kyselyistä ja dashboardeista on mahdollista muodostaa hälytyksiä. Hälytykselle annetaan nimi, tarkistamistiheys ja -aikaväli, raja-arvot milloin lähetetään tapahtumasta varoitus ja milloin kriittinen ilmoitus sekä millä tavalla raja-arvojen ylittämisestä raportoidaan. Kuviossa 13 on alerting-välilehti, jossa tehdyt hälytykset näkyvät. Hälytyksestä näkee, kuka sen on tehnyt, milloin se on viimeksi ajettu, mikä on se tila,

onko se täsmännyt hakuihin ja millä tavalla siitä ilmoitetaan eteenpäin. NLS tarjoaa useita eri vaihtoehtoja, joilla voidaan ilmoittaa tapahtuneista hälytyksistä:

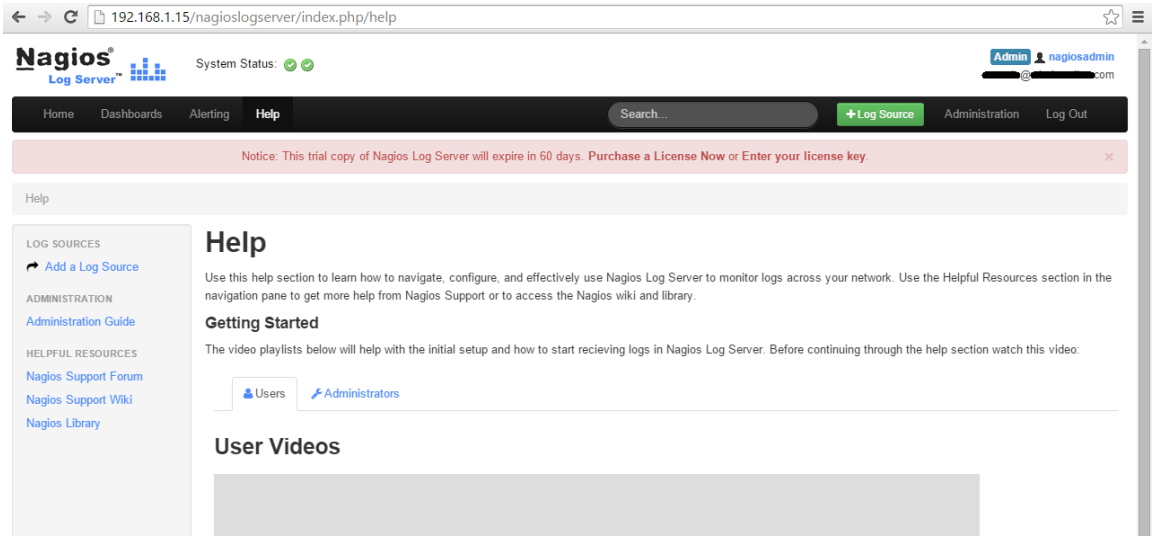
- Ei mitään. Hälytyksestä ei ilmoiteta eteenpäin.
- NRDP (Nagios Remote Data Processor). NRDP on Nagioksen tietojen kuljetusmekanismi. Sen avulla Nagios XI ja Nagios Core -monitorointityökalut voivat tiedustella NLS:n tilaa ja saavat tietoja hälytyksestä.
- Nagios Reactor. NLS lähettää hälytyksen valvontapalvelimelle, jonka tarkoitus on reagoida erinäköisiin ongelmiin.
- SNMP-trap. NLS lähettää SNMP-trap-vastaanottajalle viestin, jossa kerrotaan hälytyksestä. SNMP on tietoliikenneprotokolla, jonka avulla voidaan kysellä verkkolaitteen tilaa tai antaa hälytyksiä.
- Sähköposti. Lähetetään sähköpostiviesti valituille käyttäjille.
- Skripti. Suoritetaan tietty skripti hälytyksen ilmaantuessa.



Kuvio 13. Nagios Log Server – alerting-välilehti

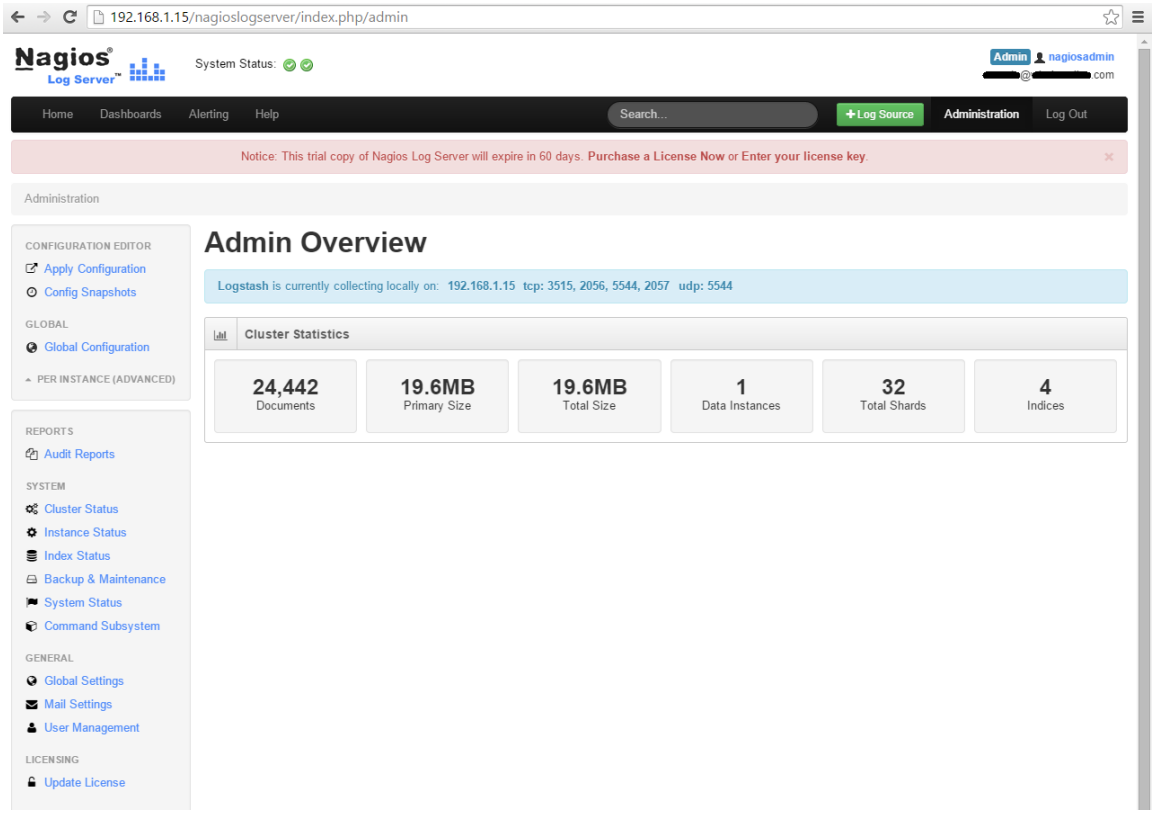
Kuviossa 14 näkyvä help-välilehti sisältää muutamia käyttäjille tarkoitettuja opetusvideoita lokilähteiden, hälytyksien ja kyselyiden lisäämisestä sekä ylläpitäjille tarkoitettuja videoita mm. käyttäjien, klustereiden ja instanssien hallinnasta. Lisäksi välilehdellä on linkkejä erilaisille tukisivustoille. Ylläpitäjän oppaassa (Administration Guide) on varsin kattava dokumentaatio aina NLS:n asentamisesta varmuuskopioiden ottamiseen saakka. Nagioksen tukifoorumilla (Nagios Support Forum) on kaksi osiota, yleinen osio ja asiakkailla tarkoitettu osio. Yleinen osio on tarkoitettu kaikille aiheesta kiinnostuneille ja asiakasosio luonnollisesti asiakkaille ja sinne on pääsy vain erillisillä käyttäjätunnuksilla. Nagios tarjoaa tuotteilleen tukea myös wiki- ja kirjastosivustoiden (Nagios Support Wiki,

Nagios Library) muodossa. NLS:n ollessa varsin uusi tuote, ei siitä viimeksi mainituissa sivustoissa ole juuri mitään aiheita. Sivustojen mukaan, niitä olisi kuitenkin pikapuolin tulossa.



Kuvio 14. Nagios Log Server – help-välilehti

Administration-välilehdeltä (ks. kuvio 15) näkee paljon NLS-järjestelmään ja ylläpitoon liittyviä tietoja ja sitä kautta tehdään muutoksia järjestelmään.



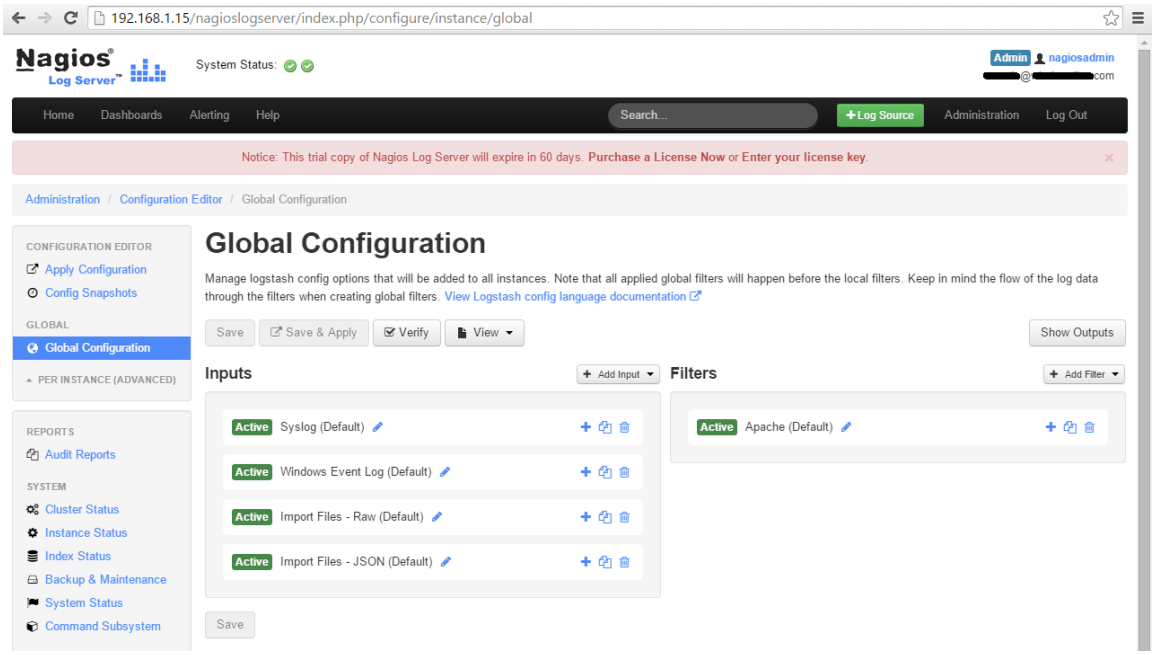
Kuvio 15. Nagios Log Server – administration-välilehti

Administration-välilehden kautta pystyy tekemään konfiguraatioita, jotka vaikuttavat yksittäisiin instansseihin tai konfiguraatioita, jotka vaikuttavat koko klusteriin. Konfiguraatioilla (ks. kuvio 16) määritetään sisääntulot (inputs), suodattimet (filters) ja ulostulot (output) ja ne vaikuttavat siihen, miten Logstash kerättäviä lokitietoja käsittelee. Kaikki NLS:n konfiguraatiot löytyvät liitteestä 1.

NLS sisältää oletuksena muutamia sisääntuloja mm. Syslog ja Windows Event log ja niillä määritetään, minkälaisia lokitietoja vastaanotetaan. Sisääntuloja voi muokata ja tehdä itse lisää. Logstashin verkkosivuilla on esimerkkejä tavallisimmille sisääntuloille ja ne sisältävät ohjeet, miten niitä tulisi käyttää.

Suodattimilla määritetään, miten vastaanotettuja lokitietoja käsitellään. Niitä voidaan muotoilla eri tarvoin: tietoja voi mm. lisätä, poistaa, korvata ja muuntaa, jotta välillä sekaviakin lokiviestejä saataisiin jäsenneltyä ja niistä saataisiin oleelliset tiedot helposti luettavaan muotoon. Myös suodattimista löytyy Logstashin verkkosivuilta valmiita esimerkkejä ja lisäksi eri internetsivustoilla käyttäjät ovat jakaneet itse tekemiään suodattimia erilaisiin tarpeisiin. Ehkä yleisin ja hyödyllisin suodatin on *grok*, jonka avulla jäsentymättömistä lokiviesteistä pystytään muodostamaan jäsenneltyjä kenttiä, joita on helppo indeksoida ja hakea. Toinen yleinen suodatin on *mutate*, jonka avulla viestien kenttiä voidaan lisätä, poistaa, uudelleennimetä, korvata ja muuttaa.

Ulostulot määrittävät sen, lähetetäänkö lokitietoja keräämisen ja suodattamisen jälkeen mahdollisesti eteenpäin. Myös ulostuloja löytyy valmiina ja ne ovat lähes samankaltaisia kuin sisääntulotkin. NLS:ssä kaikki lokien käsittelyyn liittyvä tapahtuu periaatteessa yhdellä palvelimella, joten toisin kuin monista eri osista rakennettu lokienhallintajärjestelmä, ulostuloille ei välttämättä ole tarvetta.



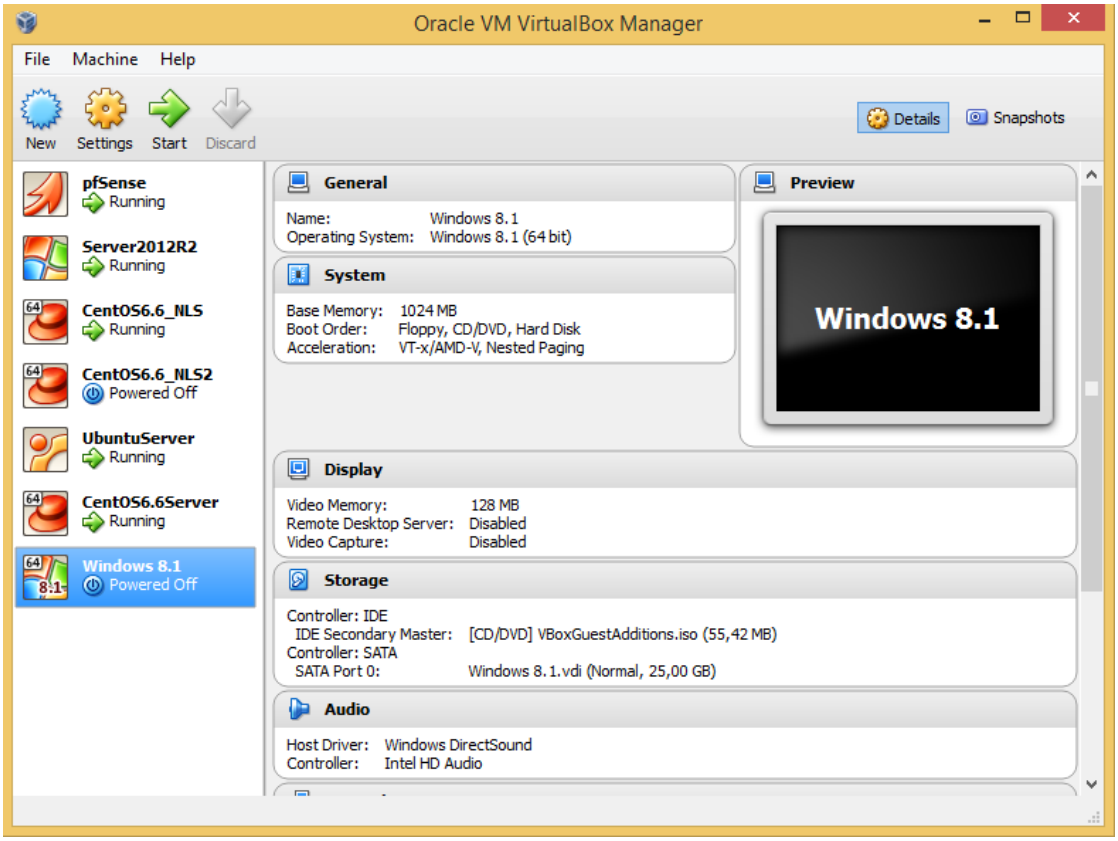
Kuvio 16. Nagios Log Server – konfiguraatiot

Konfiguraatioiden lisäksi administration-välilehdeltä näkee koko klusterista statuksen, eli kuinka paljon klusterista on dokumentteja, kuinka paljon ne vievät tilaa, montako instanssia on käytössä sekä tietoja klusterin tilasta. Jokaista instanssia pystyy myös erikseen tarkastelemaan. Instanssien status kertoo tietoja palvelimen levytilasta, muistin ja suorittimen käytöstä sekä muita järjestelmätietoja. Indeksistatus-kohdasta näkee päivittäin paketoidut lokitiedot ja miten niitä on käsitelty. Varmuuskopiointi-kohdassa pystyy määrittämään, milloin indeksejä optimoidaan, suljetaan ja poistetaan ja milloin niistä tehdään varmuuskopioita. Varmuuskopioille määritetään säilytyspaikka sekä kuinka kauan niitä säilytetään. Järjestelmästatus-kohdassa voi todeta Elasticsearchin ja Logstashin statuksen ja niitä voi käyttöliittymän kautta käynnistää, pysäyttää tai uudelleenkäynnistää. Lisäksi administration-välilehdeltä voi lisätä, poistaa ja muokata käyttäjiä, muuttaa yleisiä asetuksia ja sähköpostiasetuksia sekä näkee tietoja lisenssiin liittyen.

9 Toteutus

9.1 Ympäristö

Lokienhallintajärjestelmä toteutettiin virtualisoimalla VirtualBox-sovellusta apuna käyttäen. VirtualBox on Oracle Corporationin kehittämä avoimen lähdekoodin sovellus, joka on tarkoitettu niin yrityksille kuin kotikäyttäjillekin. Se mahdollistaa virtualisoinnin eli tässä tapauksessa usean tietokoneen suorittamisen isäntäkoneen sisällä sen resursseja hyödyntäen. Kuviossa 17 on esillä VirtualBox-sovellus. (Welcome to VirtualBox.org! 2015.)

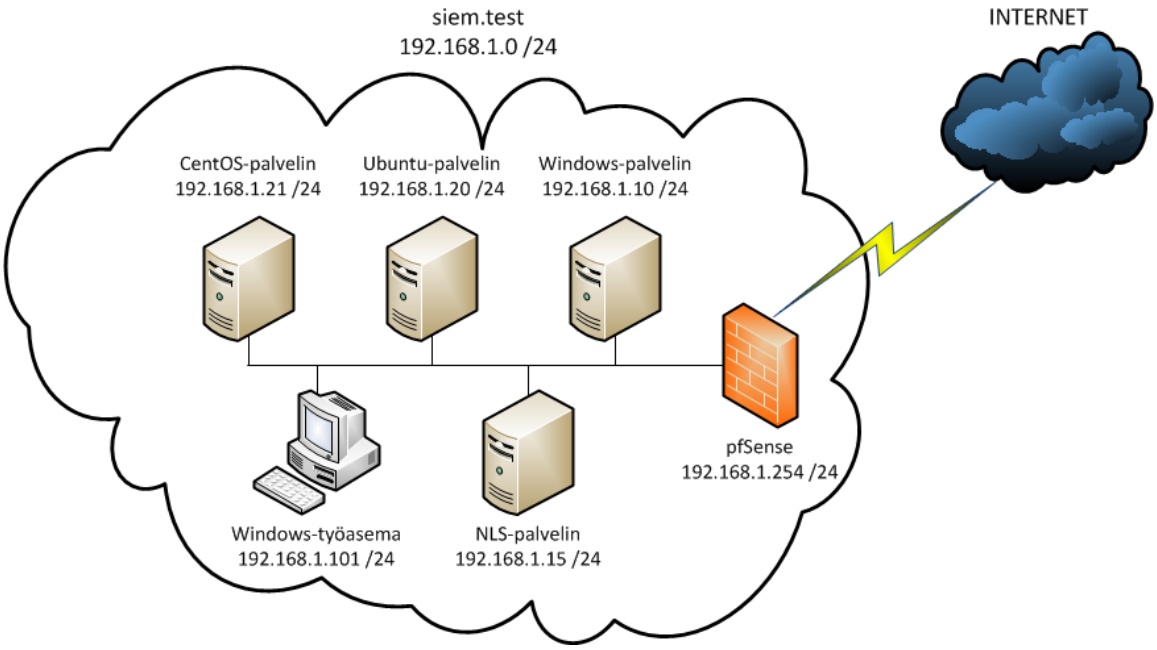


Kuvio 17. VirtualBox-sovellus

VirtualBoxissa jokaiselle virtuaalikoneelle pystytään määrittämään sen käyttämät resurssit mm. prosessorien, muistin ja kovalevytilan määrä ja ne vaihtelivat koneiden käyttötarkoituksen mukaan. Yhteistä kaikille koneille olivat kuitenkin verkkoasetukset, jotka määritettiin sisäiseksi verkoksi (internal network). Sisäisellä verkolla pystyttiin isäntäko-

neen sisään luomaan virtuaalinen lähiverkko, jolla pystyttiin havainnollistamaan yrityksen verkkoa ja sitä kautta toteuttamaan lokienhallintajärjestelmä. Ainoana poikkeuksena oli pfSense-kone, joka toimi yhdyskäytävänä sisäverkon ja internetin välillä. PfSense on avoimen lähdekoodin perustuva reititin- ja palomuurijärjestelmä, joka pohjautuu FreeBSD-käyttöjärjestelmään. Sisäisen verkon lisäksi pfSense-koneelle määritettiin toinen verkkokortti, joka käytti siltaavaa (bridged adapter) yhteyttä mahdollistaen internet-yhteyden lähiverkossa toimiville koneille.

Kuviossa 18 näkyy VirtualBoxilla luotu testiverkko siem.test, joka käyttää IP-osoitealuetta 192.168.1.0 /24. PfSense-koneen lisäksi verkkoon asennettiin yksi Windows-palvelin, kolme Linux-palvelinta sekä yksi Windows-työasema. Ympäristön koneet, IP-osoitteet ja palvelut näkyvät taulukossa 8.



Kuvio 18. Testiverkko siem.test

Taulukko 8. Laitteiden IP-osoitteet ja palvelut

Laite	**IP-osoite**	**Palvelut**
pfSense 2.2	192.168.1.254 /24	
Windows Server 2012 R2	192.168.1.10 /24	AD, DNS, DHCP, IIS, NTP
CentOS 6.6 / NLS	192.168.1.15 /24	NLS
Ubuntu 14.04.1 Server	192.168.1.20 /24	Apache
CentOS 6.6 Server	192.168.1.21 /24	Nginx
Windows 8.1 Enterprice	192.168.1.101 /24	

Windows-palvelimen käyttöjärjestelmänä toimi Windows Server 2012 R2 Standard ja sille annettiin staattinen IP-osoite 192.168.1.10 /24. Palvelimelle asennettiin AD DS (Active Directory Domain Services) -rooli, jonka avulla käyttäjiä ja työasemia pystyttiin hallitsemaan sekä määritettiin toimialue siem.test. Lisäksi Windows-palvelimeen määritettiin DNS- (Domain Name System), IIS-, DHCP- (Dynamic Host Configuration Protocol) ja NTP-palvelut (Network Time Protocol).

DNS- eli nimipalvelujärjestelmä muuntaa verkkotunnuksia IP-osoitteiksi ja toisinpäin, jolloin päästäkseen esimerkiksi Googlen internetsivuille, voidaan selaimen osoiteriville kirjoittaa www.google.com sen sijaan, että siihen kirjoitettaisiin Googlen IP-osoite. IIS on Microsoftin kehittämä web-palvelinohjelmisto, joka toimii Windows-käyttöjärjestelmissä. DHCP puolestaan jakaa IP-osoitteita verkkoon kytkettäville laitteille. NTP- eli aikapalvelu varmistaa sen, että verkon laitteiden aika määräytyy Windows-palvelimen mukaan. Lokienhallinnassa laitteiden yhteinen aikatieto on tärkeää, jotta tapahtumista voidaan esim. todentaa niiden tapahtumisjärjestys.

Linux-palvelimia verkossa oli yhteensä kolme, joissa kahdessa käyttöjärjestelmänä toimi CentOS 6.6 Basic Server ja yhdessä Ubuntu 14.04.1 Server. Kuten jo aikaisemmin mainittiin, toisella CentOS-palvelimista toimi Nagios Log Server ja sen käytettävä versio oli 2015R1.3. NLS-palvelimessa iptables-palomuuri otettiin alussa pois käytöstä viestien perillepääsyn varmistamiseksi, eikä loppuvaiheessa asiaan jäänyt aikaa paneutua. NLS-palvelin sai staattisen IP-osoitteen 192.168.1.15 /24 Windows-palvelimella toimivalta DHCP-palvelulta.

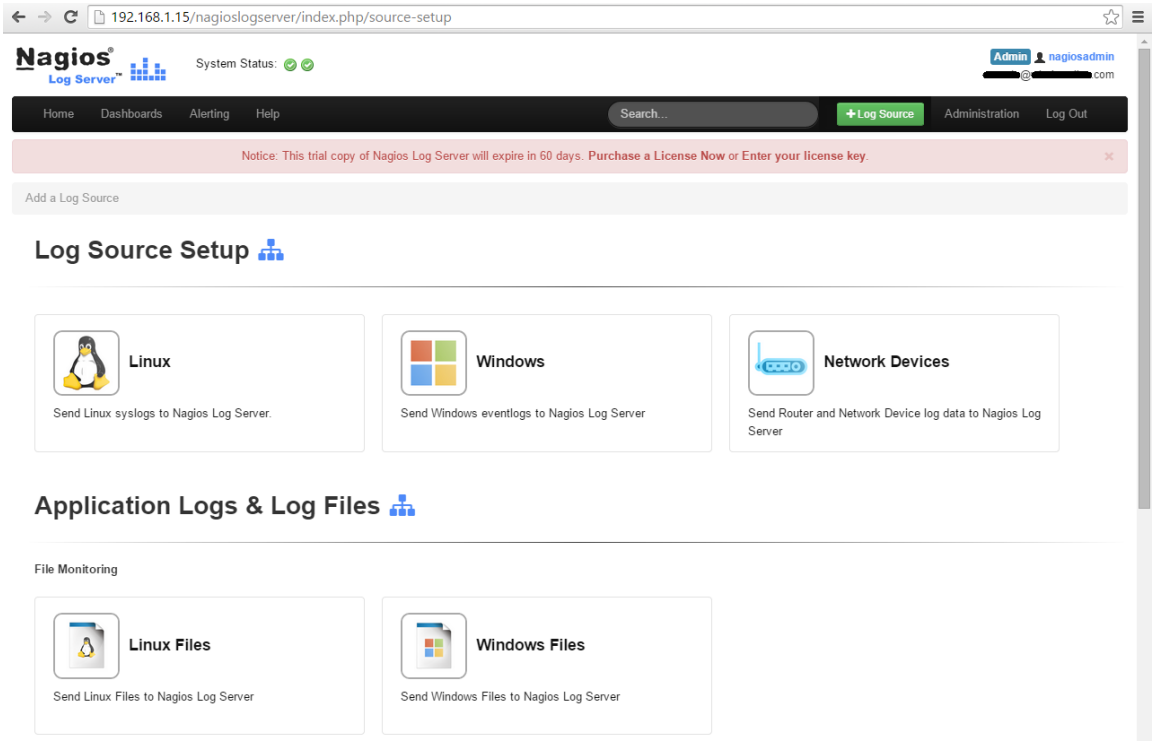
Toiselle CentOS-palvelimista asennettiin Nginx, joka on avoimen lähdekoodin web-palvelin ja jota käytetään myös välityspalvelimena sekä kuormantasaajana. Myös Nginx-palvelimelle annettiin staattinen IP-osoite 192.168.1.21 /24 DHCP:llä

Kolmannen Linux-palvelimen käyttöjärjestelmänä toimi Ubuntu. Se perustuu avoimeen lähdekoodiin ja se pohjautuu Debian-käyttöjärjestelmään. Ubuntu on yksi käytetyimmistä Linux-jakeluista työpöytäkäytössä ja sen osuus palvelimenakin on kasvussa. Ubuntu-palvelimeen asennettiin Apache, joka on maailman laajimmin käytetty avoimen lähdekoodin web-palvelin. Muiden palvelimien tapaan myös Ubuntulle jaettiin DHCP:llä staattinen IP-osoite 192.168.1.20 /24.

Windows-työaseman käyttöjärjestelmä oli Windows 8.1 Enterprise ja sitä käytettiin lähinnä eri palveluiden todentamiseen. Se sai dynaamisen IP-osoitteen DHCP:llä määritetystä osoiteavaruudesta 192.168.1.100 - .200 /24.

9.2 Lokilähteiden lisääminen

Uusien lokilähteiden lisääminen Nagios Log Serveriin on melko yksinkertaista. Lokilähteen lisääminen on dokumentoitu ylläpitäjän oppaassa, jonne pääsee esim. käyttöliittymäesittelyssä sijaitsevan kuvion 14 help-välilehden kautta. Ohjeet löytyvät helposti myös käyttöliittymästä löytyvän erillisen välilehden (+ Log Source) kautta. Välilehdellä (ks. kuvio 19) on eri osioita Linux-, Windows- ja verkkolaitteiden sekä yksittäisten palveluiden lisäämiseksi.



Kuvio 19. Nagios Log Server – lokilähteen lisääminen

Windows-palvelimelta päätettiin kerätä lokitiedot toimialueen kirjautumistapahtumista ja DNS- ja IIS-palveluiden tuottamat lokitiedostot. Myös Linux-palvelimilta kerättiin kirjautumistapahtumat sekä palvelimen tarkoituksesta riippuen sille asennetun palvelun lokitiedot. Vaikka pfSense ei virallisesti ympäristössä palomuurina toiminutkaan, huo-

mattiin sen tuottavan palomuurilokeja. Lokien monipuolistamiseksi myös siitä kerättiin lokitiedot analysoitavaksi.

Seuraavissa luvuissa on tarkemmin esitelty eri laitteiden ja palveluiden lisääminen osaksi lokienhallintajärjestelmää ja niiden toiminnan todentaminen.

9.3 Windows-palvelin

9.3.1 Event log

Nagios Log Server suosittelee *Nxlog*-sovelluksen käyttöä Event log -viestien lähettämiseen. Nxlog on avoimen lähdekoodin sovellus lokiviestien keräämiseen ja niiden edelleen lähettämiseen. Sillä pystyy myös suodattamaan ja parsimaan lokiviestejä. Nxlog on saatavilla Windows-, Unix-, Linux-, BDS- ja Android-käyttöjärjestelmille ja se tukee useita viestiformaatteja, kuten Syslog, Event log, XML, JSON, CSV (Comma-Separated Value) jne. (About 2015.)

Nxlog ladattiin NLS-käyttöliittymän kautta ja asennettiin oletuksena *C:\Program Files (x86)* -kansioon. *Program Files (x86)\nxlog\conf* -kansion alla on sovelluksen asetustiedosto *nxlog.conf*. NLS-käyttöliittymästä löytyy valmis konfiguraatio, joka kopioitiin asetustiedostoon. Lopullinen konfiguraatio löytyy kokonaisuudessaan liitteestä 2. Ensimmäisenä tiedostossa määritetään sovelluksen käyttämät tiedostopolut.

```
define ROOT C:\Program Files (x86)\nxlog
define CERT %ROOT%\cert

Moduledir %ROOT%\modules
CacheDir %ROOT%\data
Pidfile %ROOT%\data\nxlog.pid
SpoolDir %ROOT%\data
LogFile %ROOT%\data\nxlog.log
```

Seuraavaksi otettiin käyttöön *JSON*-moduuli. JSON-muotoilua käytettiin viestien lähettämiseen, koska se on helposti tuotettava tiedostomuoto ja sitä on vastaanottopäässä puolestaan helppo tulkita.

```
<Extension json>
    Module    xm_json
</Extension>
```

Tämän jälkeen määritettiin input eli sisääntulo, joka kertoo, mistä Nxlog lukee viestejä. Sisääntuloksi annettiin nimeksi *eventlog*. Moduulia *im_msvistalog* käytetään uudistuksen myötä Event logien lukemiseen Windows Vista/Server 2008 ja sitä uudemmissa Windows-käyttöjärjestelmissä. Windows Xp/Server 2003 ja sitä vanhemmissa käyttöjärjestelmissä käytetään moduulia *im_mseventlog*.

```
<Input eventlog>
    Module    im_msvistalog
 </Input>
```

Seuraavaksi on vuorossa output eli ulostulo, joka määrittää, minne kerätyt lokiviestit lähetetään. Lähetykseen käytettiin *TCP*-protokollaa, määränpääksi asetettiin NLS:n IP-osoite ja portiksi *3515*. Lisäksi muuttuja *tmpmessage* uudelleennimettiin muuttujaksi *message* ja muotoiluksi määritettiin JSON-formaatti.

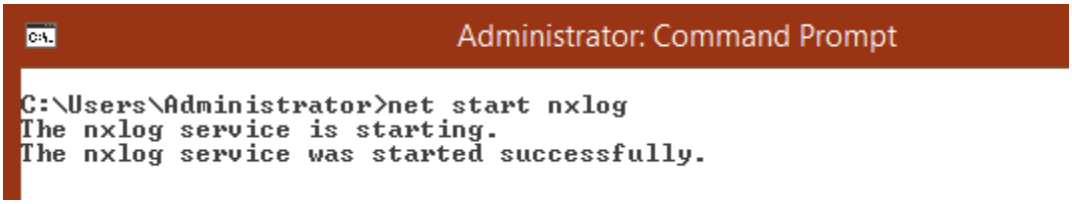
```
<Output out>
    Module    om_tcp
    Host      192.168.1.15
    Port      3515

    Exec    $tmpmessage = $Message; delete($Message);
rename_field("tmpmessage","message");
    Exec    $raw_event = to_json();
</Output>
```

Route- eli reittimääritys määrittää sen, missä järjestyksessä ja miten eri moduuleja viestien käsittelemiseksi käytetään. Tässä sisääntulomoduuli *eventlog* ohjattiin ulostulomoduuliin *out*.

```
<Route 1>
    Path    eventlog => out
</Route>
```

Nxlog oli nyt konfiguroitu ja lopuksi sovellus piti vielä käynnistää. Se onnistui komentokehotteen kautta, jonne syötettiin kuvion 20 mukainen komento: *net start nxlog*.



Kuvio 20. Nxlog-sovelluksen käynnistys

NLS:llä viestien vastaanotto määritetään käyttöliittymäesittelyssä sijaitsevan kuvion 16 konfiguraatio-osiossa. Event logille oli alun perin määritetty valmis sisääntulo. Viestit lähtevät Windows-palvelimelta ja Nxlog-sovellukselta *TCP*-protokollalla, joten luonnollisesti viestit myös vastaanotetaan sillä. Tyyppi (*eventlog*), portti (*3515*) ja koodekki (*JSON*) eivät myöskään muutu ja lisäksi JSON-formaatille määritetään latinalaisten aakkosten -merkistö.

```
tcp {
    type => "eventLog"
    port => 3515
    codec => json {
        charset => "CP1252"
    }
}
```

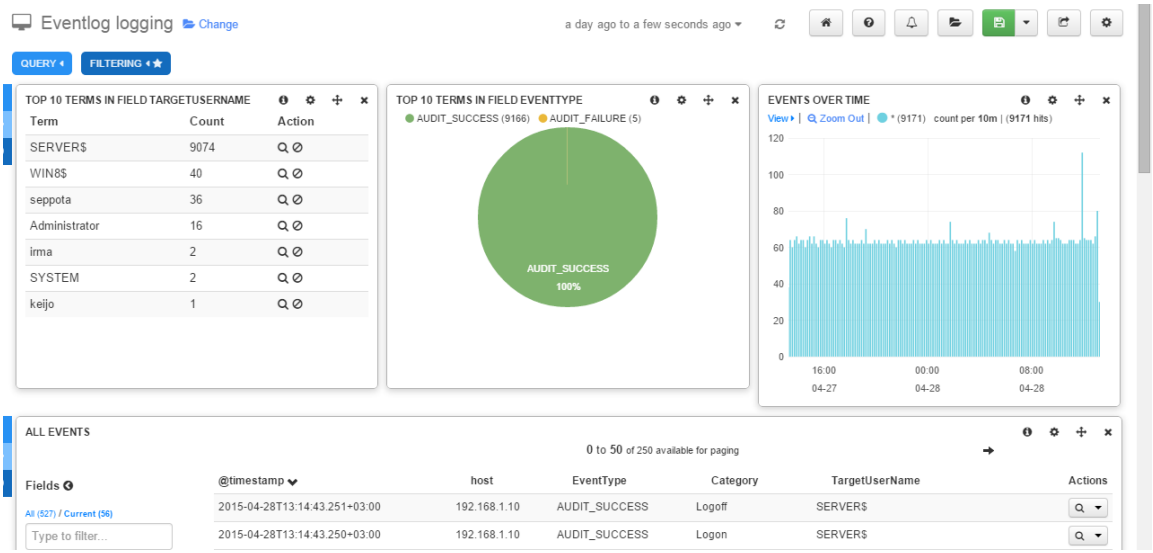
Kuvion 16 konfiguraatio-osiossa tapahtuu myös viestien suodattaminen ja muotoilu. NLS:lle luotiin uusi suodatin ja sille annettiin nimeksi *Event Log-filter*. Suodatin perustuu if-lauseeseen ja EventID:n ollessa 4624, 4625, 4634, 4771 tai 4768 (sisältäen EventType: AUDIT_FAILRE), siihen lisätään tapahtumaa kuvaava tagi. EventID 4624 kuvaa onnistunutta kirjautumista toimialueeseen ja EventID 4634 onnistunutta uloskirjautumista. EventID 4771, 4625 sekä 4768 omine ehtoineen kuvaavat epäonnistunutta sisäänkirjautumista.

```
if [type] == "eventlog" {
    if [EventID] == 4624 {
        mutate {
            add_tag => [ "logon-success" ]
        }
    }
    if [EventID] == 4634 {
        mutate {
            add_tag => [ "logoff-success" ]
        }
```

```
    }
    if [EventID] == 4771 or [EventID] == 4625 {
        mutate {
            add_tag => [ "logon-failure" ]
        }
    if [EventID] == 4768 {
        if [EventType] == "AUDIT_FAILURE" {
            mutate {
                add_tag => [ "logon-failure" ]
            }
        }
    }
}
```

Konfiguraatioiden tekemisen jälkeen ne tallennettiin ja hyväksyttiin, jonka jälkeen Windows-palvelimen lokitiedot saapuivat NLS:lle ja olivat valmiita tarkasteltaviksi. NLS:lle tehtiin *Eventlog logging* -niminen dashboard (ks. kuvio 21), joka näyttää konfiguraatiossa määritettyjen tagien mukaiset kirjautumiset. Dashboard siis suodattaa kaikkien Event log -viestien seasta ne viestit, joihin NLS merkkaa tagin *logon-success*, *logoff-success* tai *logon-failure*.

Kuviossa 21 vasemmalla ylhäällä näkyy listattuna käyttäjiä, joilla on eniten kirjautumistapahtumia. Suurennuslasi-kuvakkeella voidaan tarkastella tiettyä käyttäjää tai vastaavasti poisto-kuvakkeella poistaa käyttäjä tarkkailtavien joukosta. Keskellä näkyvä piirakkadiagrammi kuvastaa onnistuneiden ja epäonnistuneiden kirjautumisien suhdetta. Jos halutaan tarkastella epäonnistuneita kirjautumisia, klikataan epäonnistuneiden kirjautumisien aluetta. Näkyviin saadaan vain epäonnistuneet kirjautumiset, jolloin niitä voidaan tarkastella tarkemmin. Oikeassa reunassa on pylväsdiagrammi, josta näkyy kirjautumisien määrä valittuna ajanjaksona. ”Tavallisen määrän” joukosta on helppo havaita mahdolliset poikkeustilanteet. Grafiikoiden alapuolella näkyvät yksittäiset tapahtumat, joita klikkaamalla avautuvat tapahtuman yksityiskohtaiset tiedot.



Kuvio 21. Eventlog logging -dashboard

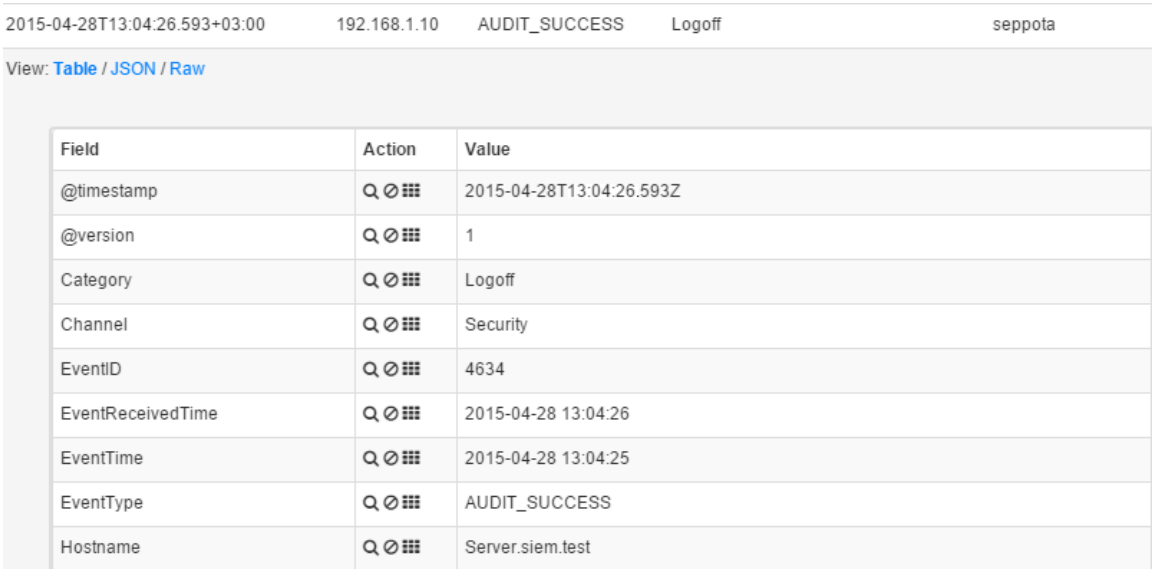
Kuviossa 22 on havainnollistettu järjestyksessä alhaalta ylöspäin kirjautumisien tapahtumasarja, jossa käyttäjä *seppota* oli kirjautuneena Windows 8.1-työasemalle. Ensin käyttäjä *seppota* kirjautui työasemasta ulos. Sen jälkeen käyttäjä *irma* yritti kirjautua sisään siinä onnistumatta. Seuraavaksi käyttäjä *seppota* yritti kirjautua sisään myöskään siinä onnistumatta. *Seppota* yritti kirjautumista uudelleen ja lopulta onnistuikin siinä.

Alimmassa tapahtumassa (EventID 4634) näkyy, kun käyttäjä *seppota* kirjautuu ulos onnistuneesti. Seuraavaksi on tapahtuma (EventID 4768), käyttäjän *irma* epäonnistunut kirjautuminen. Käyttäjää *irma* ei Windows-palvelimella käyttäjätietokannassa ole, jolloin ei luonnollisesti löytynyt sopivaa tiliä käyttäjä-salasana yhdistelmälle. Käyttäjän *seppota* epäonnistunut kirjautuminen (EventID 4771) sen sijaan johtuu tässä tapauksessa väärästä salasanasta. Viimeinen tapahtuma (EventID) kertoo käyttäjän *seppota* onnistuneesta sisäänkirjautumisesta.

@timestamp	host	EventType	Category	TargetUserName	EventID
2015-04-28T13:04:54.829+03:00	192.168.1.10	AUDIT_SUCCESS	Logon	seppota	4624
2015-04-28T13:04:50.807+03:00	192.168.1.10	AUDIT_FAILURE	Kerberos Authentication Service	seppota	4771
2015-04-28T13:04:44.754+03:00	192.168.1.10	AUDIT_FAILURE	Kerberos Authentication Service	irma	4768
2015-04-28T13:04:26.593+03:00	192.168.1.10	AUDIT_SUCCESS	Logoff	seppota	4634

Kuvio 22. Eventlog logging -kirjautumiset

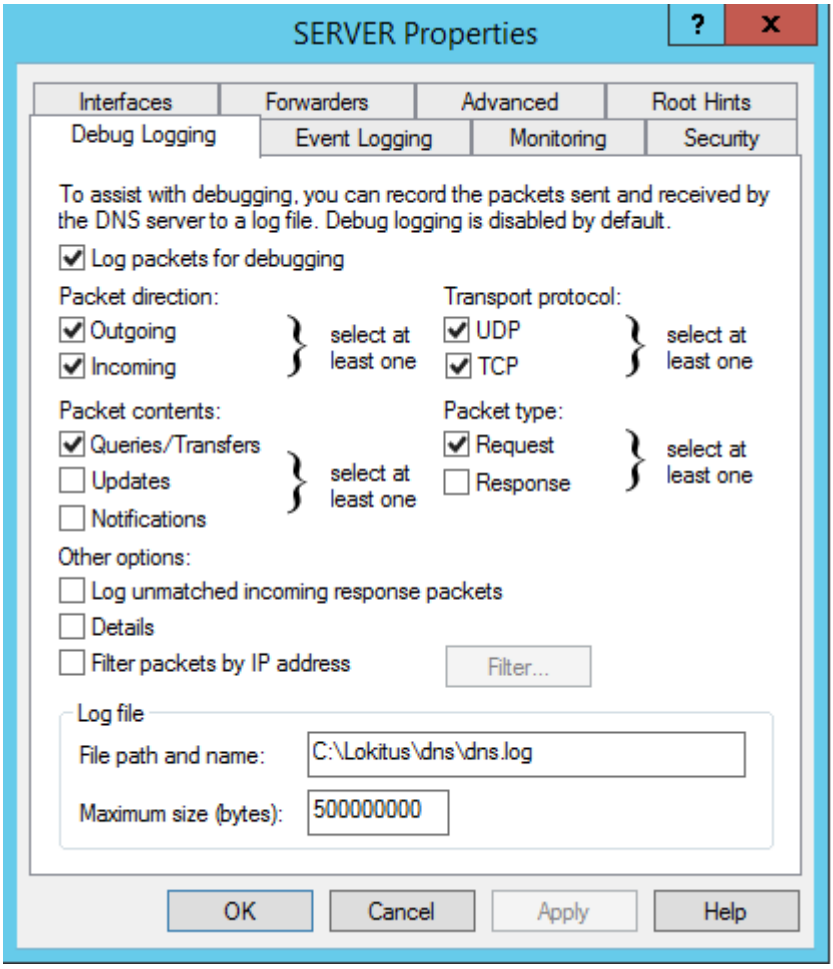
Kuviossa 23 on avattu käyttäjän *seppota* uloskirjautumistapahtuma. Jokaisesta lokitapahtumasta saa sitä klikkaamalla näkyviin tapahtuman kaikki tiedot mm. aikaleiman, kategorian, tapahtumatyypin, viestin, sovelluksen jne.



Kuvio 23. Eventlog logging -uloskirjaus

9.3.2 DNS

Windows-palvelimella toimivasta DNS-palvelusta kerättiin lokitiedot NLS:lle. DNS ei automaattisesti kirjaa tapahtumia lokiin. DNS-työkalun asetuksista Debug Logging -välilehdeltä löytyy kuvion 24 mukainen valikko, josta voidaan valita lokitettavien pakettien suunta, kuljetusprotokolla, paketin sisältö, pakettityyppi, muita valintoja sekä lokitiedoston määrityksiä. Paketin suunnaksi valittiin sekä lähtevät että saapuvat paketit, protokollaksi UDP- ja TCP-protokollat, paketin sisällöksi kyselyt ja pakettityypiksi pyynnöt. Lokitiedosto *dns.log* ohjattiin *C:\Lokitus\dns* -kansioon ja maksimikooksi sille annettiin *500* megatavua.



Kuvio 24. DNS-lokiasetukset

DNS-palvelun lokitiedot lähetettiin Event logien tapaan *Nxlog*-sovelluksella. DNS-lokien monitorointiin käytettiin NLS-käyttöliittymästä löytyvää yleistä ohjetta Windows-tiedostojen lisäämiseksi. Ensimmäiseksi *nxlog.conf* -tiedostoon lisättiin DNS-lokien sisääntulo. Sisääntulolle annettiin nimeksi *dnslogs* ja moduuliksi määritettiin *im_file*, joka tarkoittaa tiedostoa. Tiedostolle määritettiin DNS-lokitiedoston polun lisäksi ReadFromLast- ja SavePos-parametrit, jotka kertovat sovellukselle lukusuunnan ja viimeksi lähetetyn tiedon paikan. Lopuksi sovellukselle kerrottiin tiedoston suoritettavan rivi kerrallaan.

```
<Input dnslogs>
    Module      im_file
    File        'C:\Lokitus\dns\dns.log'
    ReadFromLast TRUE
    SavePos     TRUE
    Exec        $Message = $raw_event;
</Input>
```

DNS-lokeille tehtiin oma ulostulo ja reitti, jotta ne olisivat vastaanottopäässä helpompi erotella Event logien kanssa. Ulostulolle annettiin nimeksi *dnsout* ja Event logien tapaan lokit lähetettiin *TCP*-protokollalla NLS:n IP-osoitteeseen ja portiksi määritettiin *3513*. Lopuksi sisääntulo *dnslogs* ohjattiin omassa reittimäärityksessä ulostuloon *dnsout*.

```
<Output dnsout>
    Module    om_tcp
    Host      192.168.1.15
    Port      3513
</Output>
<Route 3>
    Path    dnslogs => dnsout
</Route>
```

NLS:llä DNS-lokit vastaanotettiin *TCP*:nä, tyypiksi määritettiin *dnslog*, portti *3513* määritettiin DNS-lokeille *Nxlog*-sovelluksessa ja koodekkina toimi *JSON*.

```
tcp {
    type => "dnslog"
    port => 3513
    codec => json
}
```

Kuviossa 25 on esimerkki Windows-palvelimen tuottamasta DNS-lokitapahtumasta. Lokiviestin formaatti on muotoa:

päivämäärä aika thread-id konteksti tunniste protokolla suunta IP-osoite xid sisältö [liput liput vastaus] kyselytyyppi kyselynimi

```
22.4.2015 10:25:45 02EC PACKET  0000007B37E10BA0 UDP Snd
192.43.172.30   f2c5   Q [0000       NOERROR] A
(3)www(9)nettiauto(3)com(0)
```

Kuvio 25. Alkuperäinen DNS-lokiviesti

DNS-viestin selventämiseksi NLS:lle tehtiin ala puolella näkyvä suodatin, jolle annettiin nimeksi *DNS-filter*. NLS:lle sisääntulevan lokiviestin tyypin ollessa *dnslog*, ohjautuu se *grok*-suodattimen läpi, jonka avulla DNS-viestin sisältö pystytään parsimaan kentiksi. *Match*-parametrilla pyritään löytämään tekstistä merkkijonoja, jotka vastaavat valmiisiin

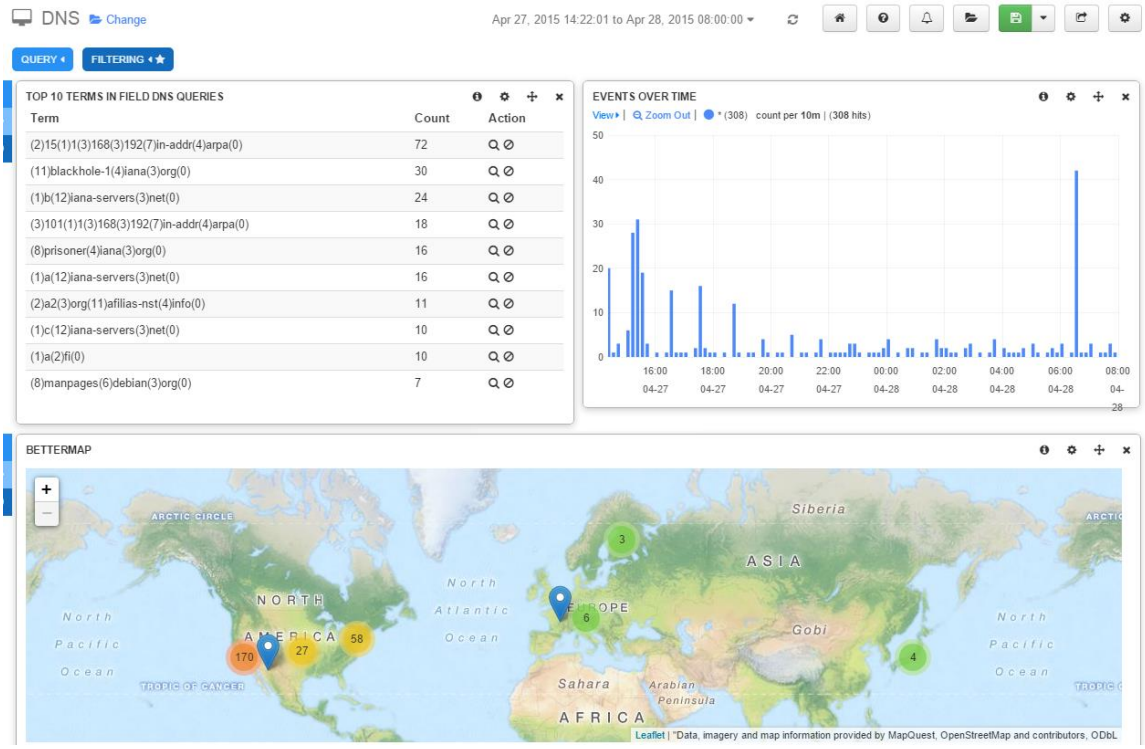
malleihin. Logstash sisältää malleja yli 120 ja niitä on mahdollista tehdä itse lisää. Match-parametri määritetään %{SYNTAX:SEMANTIC}, jossa SYNTAX on käytettävä malli ja SEMANTIC mallille annettava muuttuja. Esimerkiksi kuvion 25 DNS-lokiviestin päivämäärä 22.4.2015 täsmää %{DATE_EU:date}, jossa DATE_EU on malli ja sille annettiin muuttujaksi *date*.

```
if [type] == "dnslog" {
    grok {
        match=> [ "message",
"%{DATE_EU:date}%{SPACE}%{TIME:time}%{SPACE}%{WORD:dns_thread_id}
%{SPACE}%{WORD:dns_context}%{SPACE}%{WORD
:dns_packet_id}%{SPACE}%{WORD:dns_ip_protocol}%{SPACE}%{WORD:dns_
direction}%{SPACE}%{IP:dns_client_ip}%{SPACE}%{WORD:dns_xid}
(?:%{WORD:dns_query_type}|\s*)
Q%{SPACE}\[%{NUMBER:dns_flags_hex}%{SPACE}(?:%{WORD:dns_flags_cha
rs}|\s*)%{SPACE}%{WORD:dns_response}\]%{SPACE}%{WORD:dns_question
_type}%{SPACE}%{GREEDYDATA:dns_question_name}" ]
    }
    geoip {
        source => "dns_client_ip"
    }
}
```

Grok-suodattimella ja kenttien parsimisella lokiviestistä poimittiin IP-osoite, johon DNS-kysely lähetettiin. *Geoip*-suodattimella saadaan selville muuttujaan *dns_client_ip* tallennetun IP-osoitteen maantieteellinen sijainti. Sitä voidaan hyödyntää lokien analysointiin DNS:n lisäksi myös muissa palveluissa selvittämään, mihin päin maailmaa viestit lähtevät tai puolestaan mistä niitä tulee. Tietoja voidaan käsitellä ja sijoittaa esim. kartalle, jolloin pystytään selkeästi havainnollistamaan liikennettä lähes kaupunkikohtaisesti.

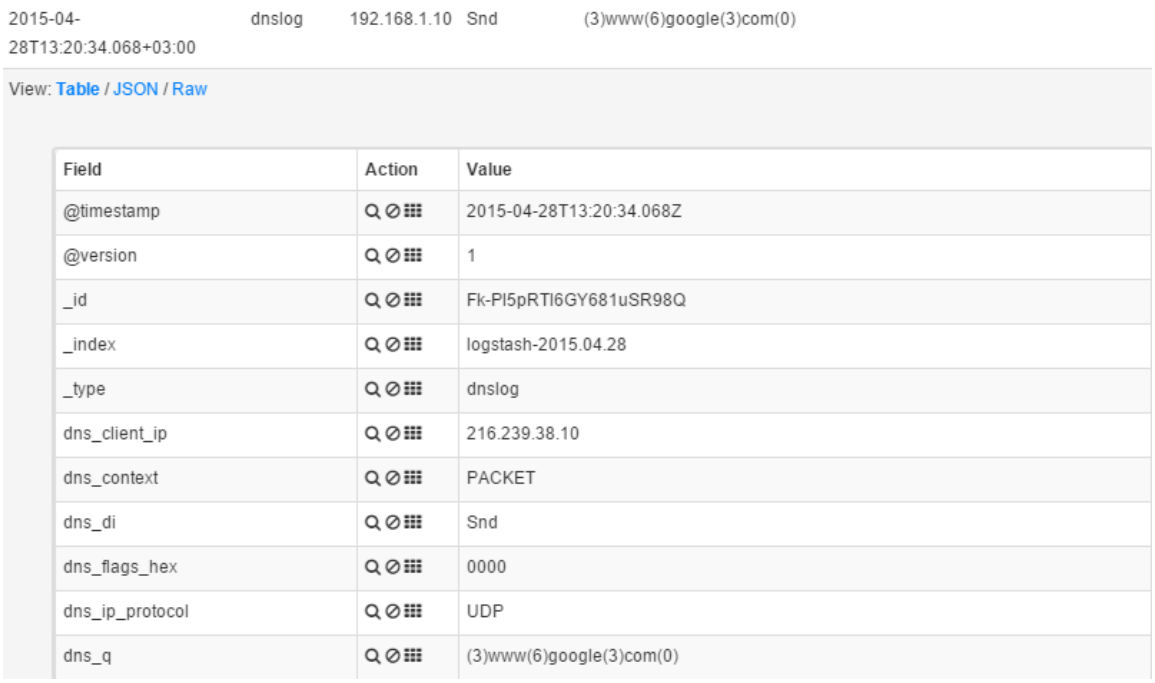
Suodattimien teon ja hyväksymisen jälkeen DNS-lokitiedostoja pystyttiin tarkastelemaan NLS:llä. DNS:lle luotiin oma *DNS*-dashboard (ks. kuvio 26), jossa tarkasteltiin *dnslog*-tyypin lokiviestejä ja tuona hetkenä vain lähetettyjä nimikyselyitä. Kuvion 26 vasemmassa yläkulmassa on listattuna kymmenen useimmin käytettyä osoitetta, joihin kyselyitä on lähetetty. Ensimmäisenä listassa on *in-addr.arpa*, joka on internetissä toimiva käänteisnimipalvelu ja jota käytetään muuntamaan IP-osoite verkkotunnukseksi. Oikeassa yläkulmassa on puolestaan kyselymäärät valittuna ajanjaksona. Alimpana on kartta, jossa on hyödynnetty geoip-suodatinta. Lähetetyistä nimikyselyistä poimituista IP-osoitteista saadaan kerättyä sijaintitiedot ja ne voidaan sijoittaa kartalle. Kuviosta 26

nähdään, että joitakin IP-osoitteita on paikannettu Eurooppaan, mutta suurin osa kyselyistä on lähetetty Pohjois-Amerikkaan.



Kuvio 26. DNS-dashboard

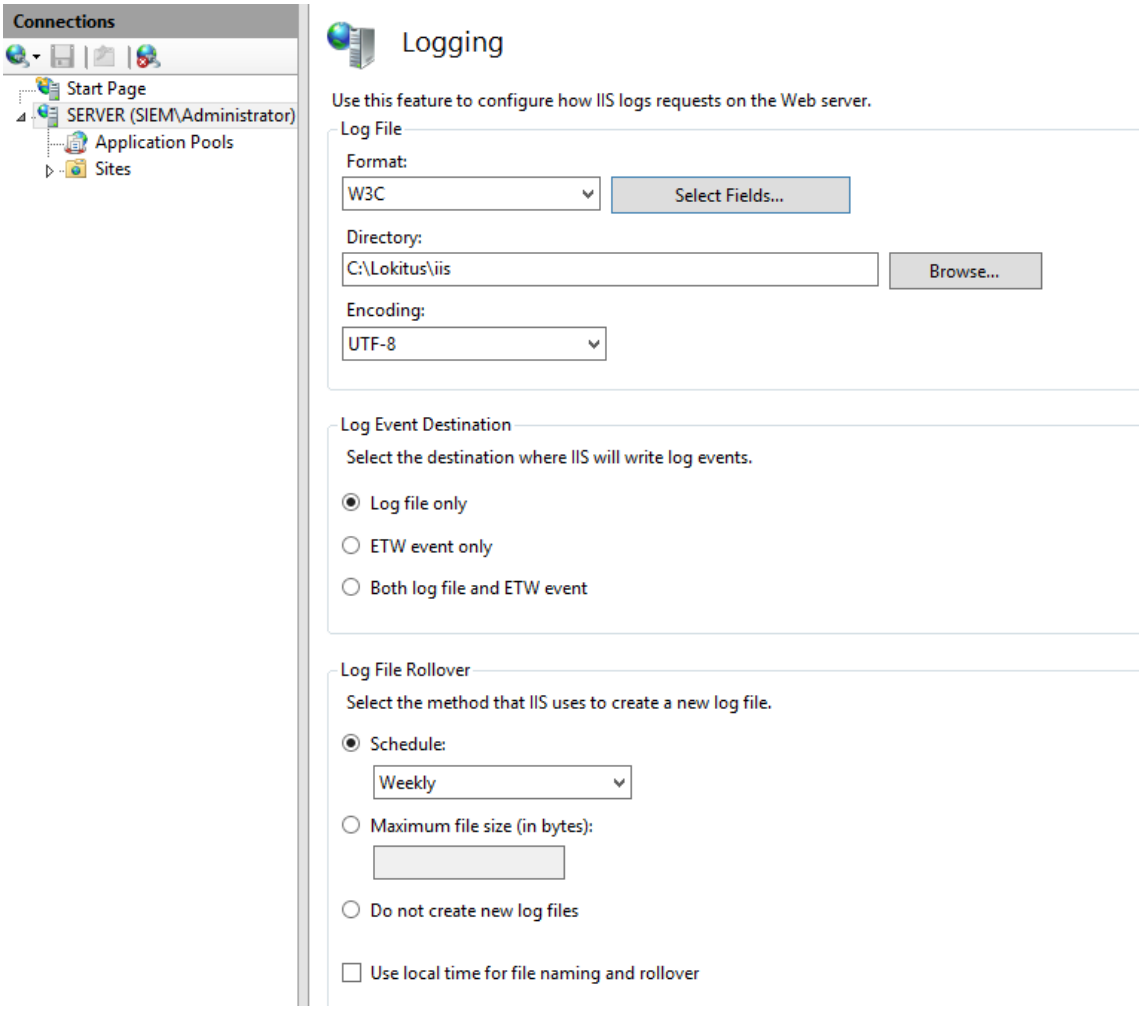
Kuviossa 27 on avattu yksi DNS-lokitapahtuma, jossa käyttäjä on todennäköisesti selaimella siirtynyt osoitteeseen *www.google.com*. Lokitapahtuma parsittiin DNS-suodattimella aikaisemmin kentiksi, joita kuviossa näkyy. Kenttiä ovat mm. *aikaleima*, *tyyppi*, *IP-osoite* ja *protokolla*, joita apuna käyttäen voi halutessaan etsiä lisää vastaavien arvojen tapahtumia tai vastavuoroisesti poistaa ne hakukriteereistä.



Kuvio 27. DNS-nimikysely

9.3.3 IIS

IIS-web-palvelu lisättiin myös tarkasteltavien palveluiden joukkoon. IIS-työkalun loki-osiossa (ks. kuvio 28) määritettiin lokiasetukset. IIS käyttää oletuksena *W3C*-formaattia (World Wide Web Consortium), joka on kansainvälisten yritysten ja yhteisöjen yhteenliittymän suositus ELF-formaatista. Tiedostolle määritettiin kansio *C:\Lokitus\iis*, koodaus (*UTF-8*) ja sen lisäksi asetukset, joissa lokit kirjoitettiin vain tiedostoon ja uusi tiedosto luotiin joka viikko.



Kuvio 28. IIS-lokiasetukset

Myös IIS-palvelun lokitiedot lähetettiin *Nxlog*-sovelluksella. Ensimmäisenä *nxlog.conf* -tiedostoon lisättiin *w3c*-niminen laajennus, jossa *CSV*-moduulin avulla määritetään, miten teoria-osiossa sijaitsevan kuvion 7 IIS-lokiviesti parsitaan kentiksi ja muuttujiin ja onko muuttuja joko merkkijono (*string*) vai kokonaisluku (*integer*). Kentät erotetaan toisistaan välilyönnillä.

```
<Extension w3c>
    Module      xm_csv
    Fields      $date, $time, $website, $hostname, $serverip,
$verb, $request, $querystring, $dstport, $user, $clientip,
$httpversion, $useragent, $cookie, $referrer, $fqdn, $status,
$substatus, $sc_win32_status, $sc_bytes, $cs_bytes, $time_taken
    FieldTypes  string, string, string, string, string, string,
string, string, string, string, string, string, string, string,
string, string, integer, integer, integer, integer, integer,
integer
    Delimiter   ' '
</Extension>
```

Seuraavaksi määritettiin sisääntulo, jolle annettiin nimeksi *iislogs*. DNS-lokin tapaan IIS-loki määritettiin tiedostoksi, annettiin tiedostopolku sekä parametrit ja sen jälkeen määritettiin if-lause. Jos rivi alkaa #-merkillä eli on ns. kommentti, se tiputetaan. Muussa tapauksessa rivi parsitaan *w3c*-laajennuksessa määritettyihin muuttujiin *CSV*-moduulin avulla. Lisäksi luotiin uusi muuttuja *$EventTime* muuttujien *$date* ja *$time* arvoista sekä viestin muotoiluksi määritettiin *JSON*-formaatti.

```
<Input iislogs>
    Module    im_file
    File      'C:\Lokitus\iis\W3SVC1\u_ex*.log'
    ReadFromLast  TRUE
    SavePos   TRUE
    Exec      if $raw_event =~ /^#/ drop();                \
    else                                                   \
    {                                                      \
        w3c->parse_csv();                                  \
        $EventTime = parsedate($date + " " + $time);       \
        to_json ();                                        \
    }
</Input>
```

IIS-lokeille tehtiin DNS:n tapaan oma ulostulo sekä reitti. Ulostulon nimeksi annettiin *iisout*, protokollaksi *TCP*, IP-osoitteeksi NLS:n IP ja portiksi *3514*. Reitissä sisääntulo *iislogs* ohjattiin ulostuloon *iisout*.

```
<Output iisout>
    Module    om_tcp
    Host      192.168.1.15
    Port      3514
</Output>
<Route 2>
    Path    iislogs => iisout
</Route>
```

Samoin kuin DNS, IIS-lokit vastaanotettiin NLS:llä TCP-protokollana, tyyppinä *iislog*, porttina *Nxlog*-sovelluksen määrittämä *3514* ja koodekkina *JSON*.

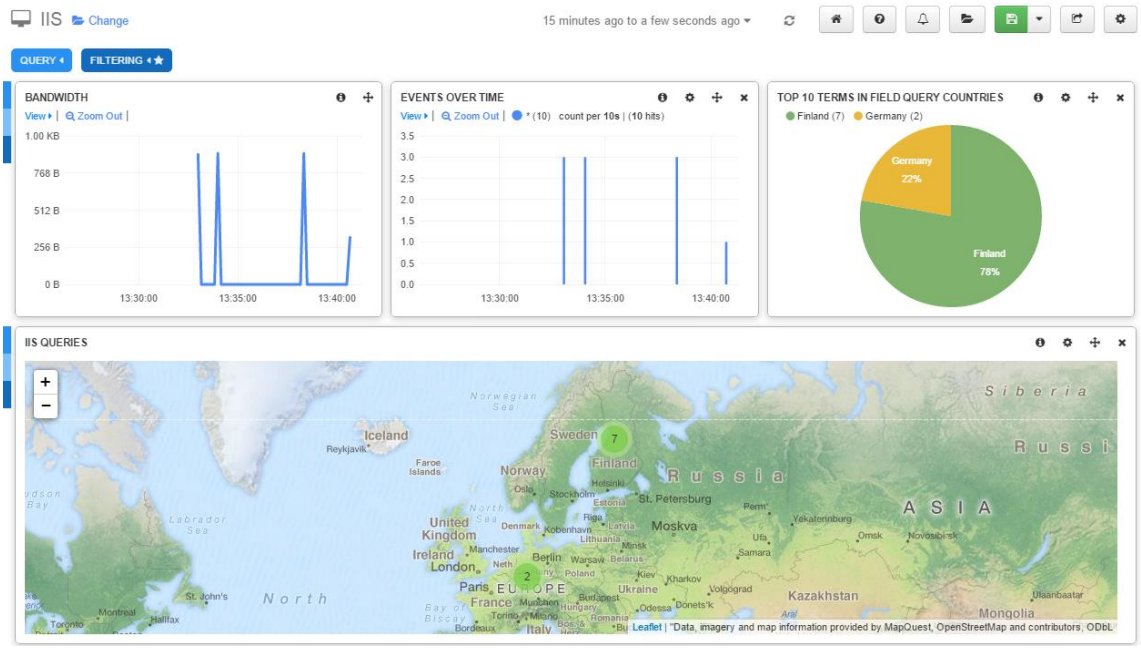
```
tcp {
    type => "iislog"
    port => 3514
    codec => json
}
```

NLS:llä tehtiin IIS-lokeille suodatin ja sille annettiin nimeksi *IIS-filter*. Jos lokiviestin tyyppi on *iislog*, niin se ohjautuu *mutate*-suodattimeen. Siinä *replace*-parametrilla lisättiin *@source_host*-muuttuja ja arvoksi annettiin Windows-palvelimen nimi *server.siem.test*. Myös *add_field*-parametrilla lisättiin uusi muuttuja *requesturl*, johon arvot otettiin muuttujista *fqdn*, *request* ja *querystring*. IIS-lokeista poimittiin myös verkkosivun kyselijän sijaintitiedot *geoip*-suodattimella.

```
if [type] == "iislog" {
    mutate {
        replace => [ "@source_host", "server.siem.test" ]
        add_field => { "requesturl" =>
"%{fqdn}%{request}%{querystring}" }
    }
    geoip {
        source => "clientip"
    }
}
```

IIS-konfiguraatioiden tallentamisen ja hyväksymisen jälkeen voitiin siirtyä tarkastelemaan lokitiedostoja. IIS-lokeille luotiin *IIS*-niminen dashboard, joka suodattaa viestit *iislog*-tyypin mukaan ja johon lisättiin olennaisimpia osioita kertomaan viestien sisällöstä. Virtuaaliympäristössä ja vain IIS-palveluun sisäänrakennettua kotisivua käytettäessä ei todellista kuvaa liikenteestä ja palvelun toiminnasta saanut, mutta tarkoitus olikin vain havainnollistaa IIS-lokien keräämistä ja niiden visualisointia. Esim. geoip-suodatin ei toimi sisäverkon IP-osoitteilla, joten lokitiedostoon lisättiin rivejä, joissa verkkosivun kyselijän sisäisen IP-osoitteen tilalle vaihdettiin satunnaisia julkisia IP-osoitteita.

Kuvion 29 vasemmanpuoleinen kuvaaja kertoo, kuinka paljon liikennettä web-sivun lataus aiheuttaa. Kuvaajasta huomataan, että liikennemäärä on jokaisella latauskerralla sama, joka johtuu lokitiedostoon lisättyjen rivien arvojen muuttumattomuudesta. Keskimmäisessä kuvaajassa näkyvät latausten määrät ja oikeanpuoleisessa piirakassa geoip-suodattimella poimitut maat, jotka ovat web-sivua ladanneet. Kuten piirakasta sekä kartastakin näkyy, tällä kertaa satunnaiset IP-osoitteet kohdentuivat pääasiassa Suomeen sekä muutama myös Saksaan.



Kuvio 29. IIS-dashboard

Kuviossa 30 näkyy osa yksittäisen lokiviestin kentistä. Konfiguraatiossa lisätty *@source_host* eli IIS-palvelimen *nimi* sekä *aikaleima*, *moduulityyppi* ja web-sivun kyselijän *IP-osoite* on kaikki eritelty omiksi kentikseen, joita voi jo aikaisemmin mainituin tavoin mm. tarkastella lähemmin tai poistaa kokonaan valinnoista.

2015-04-28T13:38:25.472+03:00 192.168.1.10 GET /favicon.ico

View: **Table** / JSON / Raw

Field	Action	Value
@source_host		server.siem.test
@timestamp		2015-04-28T13:38:25.472Z
@version		1
EventReceivedTime		2015-04-28 13:38:25
EventTime		2015-04-28 13:40:18
SourceModuleName		iislogs
SourceModuleType		im_file
_id		R2YYQNMnRlyG-WdYzU-LdQ
_index		logstash-2015.04.28
_type		iislog
clientip		193.96.134.26

Kuvio 30. IIS-web-kysely

9.4 Linux-palvelimet

9.4.1 Syslog

Linux-palvelimissa Syslog-viestien lähettämisestä NLS:lle vastasi *Rsyslog*. NLS-käyttöliittymän ohjeiden mukaan Linux-järjestelmien lisääminen lokilähteiksi on erittäin yksinkertaista skriptin avulla. Ensiksi skripti ladattiin lokilähteeksi lisättävälle laitteelle NLS:ltä *curl*-tiedonsiirtotyökalua apuna käyttäen ja sen jälkeen *bash*-komentotulkin avulla suoritettiin skripti, johon parametreiksi on lisätty NLS:n IP-osoite ja käytettävä portti.

```
curl -s -O http://192.168.1.15/nagioslogserver/scripts/setup-linux.sh
bash setup-linux.sh -s 192.168.1.15 -p 5544
```

Kuviossa 31 on skriptin lataus, suorittaminen ja tulosteet CentOS-palvelimella.

```
[root@centos6server ~]# curl -s -O http://192.168.1.15/nagioslogserver/scripts/setup-linux.sh
[root@centos6server ~]# bash setup-linux.sh -s 192.168.1.15 -p 5544
Detected rsyslog 5.8.10
Detected rsyslog work directory /var/lib/rsyslog
Destination Log Server: 192.168.1.15:5544
Creating /etc/rsyslog.d/99-nagioslogserver.conf...
==============================! WARNING !==================================
SELinux is enforcing. This may prevent rsyslog from forwarding messages.
If log messages do not reach Log Server from this host, ensure SELInux is
configured to allow rsyslog forwarding.
==========================================================================
rsyslog configuration check passed.
Restarting rsyslog service with 'service'...
Shutting down system logger:                               [  OK  ]
Starting system logger:                                    [  OK  ]
Okay.
rsyslog is running with the new configuration.
Visit your Nagios Log Server dashboard to verify that logs are being received.
```

Kuvio 31. Linux-palvelimen lisääminen skriptin avulla

Kuten kuviosta 31 näkyy, skriptin suorittaminen tekee */etc/rsyslog.d/* -kansioon tiedoston *99-nagioslogserver.conf*. Lisäksi skripti lisää */etc/rsyslog.conf* -tiedostoon rivejä, jotta *Rsyslog* osaa lukea ja käyttää juuri luotua tiedostoa. Lopuksi *Rsyslogin* konfiguraatiot tarkistetaan ja se käynnistetään uudelleen. Alla näkyvässä *99-nagioslogserver.conf* -tiedostossa määritetään mm. *Rsyslogin* työskentelytiedostojen *kansio*, *nimi* ja *maksimitila* sekä tietysti lokitiedostojen *lähetysprotokolla*, *-osoite* ja *-portti*.

```
# ### begin forwarding rule ### NAGIOSLOGSERVER
#
$WorkDirectory /var/lib/rsyslog # where to place spool files
$ActionQueueFileName fwdRule1 # unique name prefix for spool
files
$ActionQueueMaxDiskSpace 1g # 1gb space limit (use as much as
possible)
$ActionQueueSaveOnShutdown on # save messages to disk on shutdown
$ActionQueueType LinkedList # run asynchronously
$ActionResumeRetryCount -1 # infinite retries if host is down
# remote host is: name/ip:port, e.g. 192.168.0.1:514, port
optional
*.* @@192.168.1.15:5544
# ### end of the forwarding rule ###
```

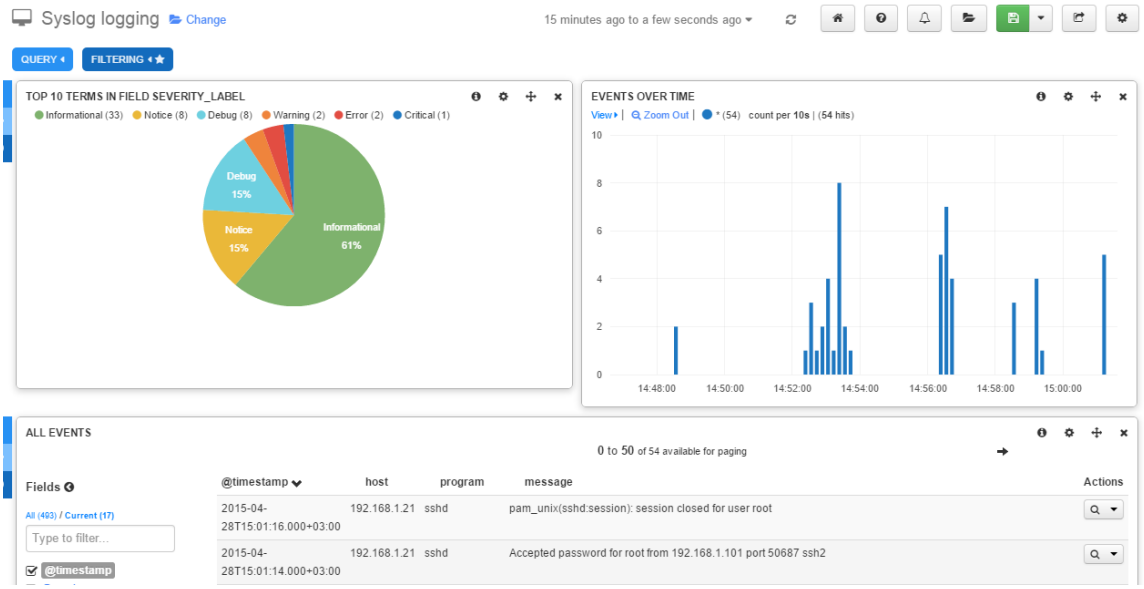
Linux-järjestelmien lisääminen neuvotaan käyttöliittymässä skriptin lisäksi myös manuaalisesti. Tässä vaihtoehdossa *rsyslog.conf* -tiedostoon kopioidaan itse yllä olevat määritykset ja arvot. CentOS-koneita lisättäessä huomattiin, että jos *Rsyslogin*-asetustiedostossa lähetysprotokollana oli TCP eli @@, eivät viestit saapuneet NLS:lle. Lähetysprotokolla määritetään yllä olevassa konfiguraatiossa @-merkein. Asian huomattiin korjautuvan käytettäessä lähetysprotokollana UDP:ta eli @. Viestien perille saapumiseen oltiin siinä vaiheessa tyytyväisiä, eikä asian selvittämiseksi loppuvaiheessa jäänyt enää aikaa.

Event login tapaan myös Syslogin sisääntulo oli valmiiksi määritetty NLS:llä (ks. kuvio 16 - konfiguraatiot). Tyyppi (*syslog*) ja portti (*5544*) ovat samat kuin lokilähteillekin asetetut.

```
syslog {
    type => "syslog"
    port => 5544
}
```

Syslog-viesteille ei yhteisesti erillistä suodatinta tehty, vaan myöhemmässä vaiheessa viestejä luokitellaan erikseen niitä tuottavien sovellusten perusteella. Linux-palvelimien kirjautumisien tarkkailuun luotiin *Syslog logging* -dashboard (ks. kuvio 32), joka suodattaa Syslog-viestien seasta kirjautumistapahtumat. Kuvion vasemmassa yläkulmassa näkyy piirakkadiagrammi, joka kertoo viestien Severity-arvosta eli tapahtuman vakavuudesta. Vakavuusosiota klikkaamalla saadaan näkyviin vain sen vakavuustason tapahtu-

mat tai ne voidaan puolestaan piilottaa näkymästä, jolloin voidaan keskittyä esimerkiksi error- tai critical-tason viesteihin, jota kuviossakin näkyy. Oikealla ylhäällä näkyvät tapahtumamäärät ja grafiikoiden alapuolella listattuna kaikki lokitapahtumat.



Kuvio 32. Syslog logging -dashboard

Kuviossa 33 näkyvät kirjautumistapahtumat CentOS-palvelimelle aikajärjestyksessä alhaalta ylöspäin. CentOS-palvelinta käytettiin etäyhteydellä Windows-työasemalta *PuTTY*-pääte-emulaattoria käyttäen. Kahdessa alimmassa tapahtumassa näkyvät paikallisen pääkäyttäjän *root* epäonnistunut kirjautuminen väärästä salasanasta johtuen. Seuraavaksi kirjautuminen on onnistunut ja istunto avattu käyttäjälle *root*. Lopuksi on vielä ilmoitus istunnon sulkemisesta.

@timestamp	host	program	message
2015-04-28T15:01:16.000+03:00	192.168.1.21	sshd	pam_unix(sshd:session): session closed for user root
2015-04-28T15:01:14.000+03:00	192.168.1.21	sshd	Accepted password for root from 192.168.1.101 port 50687 ssh2
2015-04-28T15:01:14.000+03:00	192.168.1.21	sshd	pam_unix(sshd:session): session opened for user root by (uid=0)
2015-04-28T15:01:11.000+03:00	192.168.1.21	sshd	Failed password for root from 192.168.1.101 port 50687 ssh2
2015-04-28T15:01:10.000+03:00	192.168.1.21	sshd	pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhost=192.168.1.101 user=root

Kuvio 33. Syslog logging -kirjautumiset

Kuviossa 34 näkyy AD-käyttäjätilin *seppota* kirjautuminen CentOS-palvelimelle. CentOS-palvelimelle asennettiin *Winbind*-sovellus, joka mahdollistaa AD- käyttäjätilien käytön Linux-laitteissa. Alimpina näkyvät *Winbind*-sovelluksen tekemät varmennukset käyttäjätili-salasana-yhdistelmästä sekä lopulta käyttäjän *seppota* onnistunut sisäänkirjautuminen. Kuviossa näkyy myös, kun käyttäjä *seppota* kirjautuu CentOS-palvelimen paikalliseksi pääkäyttäjäksi *root*. Tällä kertaa mitään toimenpiteitä ei suoritettu ja ylimpänä näkyvät käyttäjien *root* ja *seppota* istuntojen sulkeutuminen.

@timestamp	host	program	message
2015-04-28T15:03:49.000+03:00	192.168.1.21	sshd	pam_unix(sshd:session): session closed for user SIEM\seppota
2015-04-28T15:03:47.000+03:00	192.168.1.21	su	pam_unix(su:session): session closed for user root
2015-04-28T15:03:46.000+03:00	192.168.1.21	su	pam_unix(su:session): session opened for user root by SIEM\seppota(uid=16777216)
2015-04-28T15:03:42.000+03:00	192.168.1.21	sshd	pam_unix(sshd:session): session opened for user SIEM\seppota by (uid=0)
2015-04-28T15:03:41.000+03:00	192.168.1.21	sshd	pam_winbind(sshd:auth): user 'siem\seppota' granted access
2015-04-28T15:03:41.000+03:00	192.168.1.21	sshd	pam_winbind(sshd:auth): getting password (0x00000010)
2015-04-28T15:03:41.000+03:00	192.168.1.21	sshd	pam_winbind(sshd:auth): pam_get_item returned a password
2015-04-28T15:03:41.000+03:00	192.168.1.21	sshd	pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhost=192.168.1.101 user=siem\seppota
2015-04-28T15:03:41.000+03:00	192.168.1.21	sshd	pam_winbind(sshd:account): user 'SIEM\seppota' granted access
2015-04-28T15:03:41.000+03:00	192.168.1.21	sshd	Accepted password for siem\\seppota from 192.168.1.101 port 50709 ssh2

Kuvio 34. Syslog logging -kirjautumiset2

9.4.2 Nginx

Nginx-web-palvelu asennettiin taulukon 8 mukaisesti CentOS-palvelimelle. Ennen Nginxin asentamista, oli CentOS-palvelin hyvä päivittää. CentOS-palvelimen *yum*-paketinhallintaohjelma ei sisällä viimeisintä Nginx-versiota, joten siitä syystä asennettiin *EPEL*-pakettivarasto. Sen avulla saatiin käyttöön ajantasainen Nginx-versio, joka asennettiin ja käynnistettiin.

```
yum update
yum install epel-release
yum install nginx
/etc/init.d/nginx start
```

Seuraavaksi CentOS-palvelin lisättiin lähettämään lokiviestit *Rsyslogilla* Linux-palvelimet-kohdassa esitetyllä tavalla. *Rsyslog* ei kuitenkaan automaattisesti ymmärrä lähettää Nginxin *access*- ja *error*-lokitiedostoja, joihin palvelu lokittaa omat tapahtumansa. NLS:n käyttöliittymässä on ohje, joka neuvoo erillisten Linux-lokitiedostojen lähettämisen samaisen skriptin avulla, jota käytettiin lisätessä Linux-palvelimia. Skrtiptin suorittamiseen käytettiin lisäparametreja *-f* ja *-t* aikaisempaan verrattuna.

```
curl -s -O http://192.168.1.15/nagioslogserver/scripts/setup-
linux.sh
bash setup-linux-sh -s 192.168.1.15 -p 5544 -f
/var/log/nginx/error.log -t nginx_error
bash setup-linux-sh -s 192.168.1.15 -p 5544 -f
/var/log/nginx/access.log -t nginx_access
```

Lisäparametrit *-f* ja *-t* kertovat luettavan tiedoston tiedostopolun ja tagin, joka määrittää viestin program-kentän sisällön. Access- ja error-lokit luetaan kansiosta */var/log/nginx* ja tagiksi määritetään lokista riippuen joko *nginx_access* tai *nginx_error*. Skripti tekee */etc/rsyslog.d/* -kansioon tiedostot *90-nagioslogserver_var_log_nginx_access.log.conf* ja *90-nagioslogserver_var_log_nginx_error.log.conf*, jotka määrittävät tiedostojen lähettämisen *Rsyslogilla*.

Kuviossa 35 on *Nginx*-web-palvelimen *nginx_access*-muotoinen lokitapahtuma. Lokiviestin formaatti on muotoa:

asiakas-IP - käyttäjä [aikaleima] ”pyyntö” status lähetyskoko osoite ”käyttäjäagentti” kesto

```
192.168.1.101 - - [27/Apr/2015:10:02:50 +0300] ”GET / HTTP/1.1”
304 0 ”-” ”Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/42.0.2311.90 Safari/537.36” “-“
```

Kuvio 35. Alkuperäinen Nginx-lokiviesti

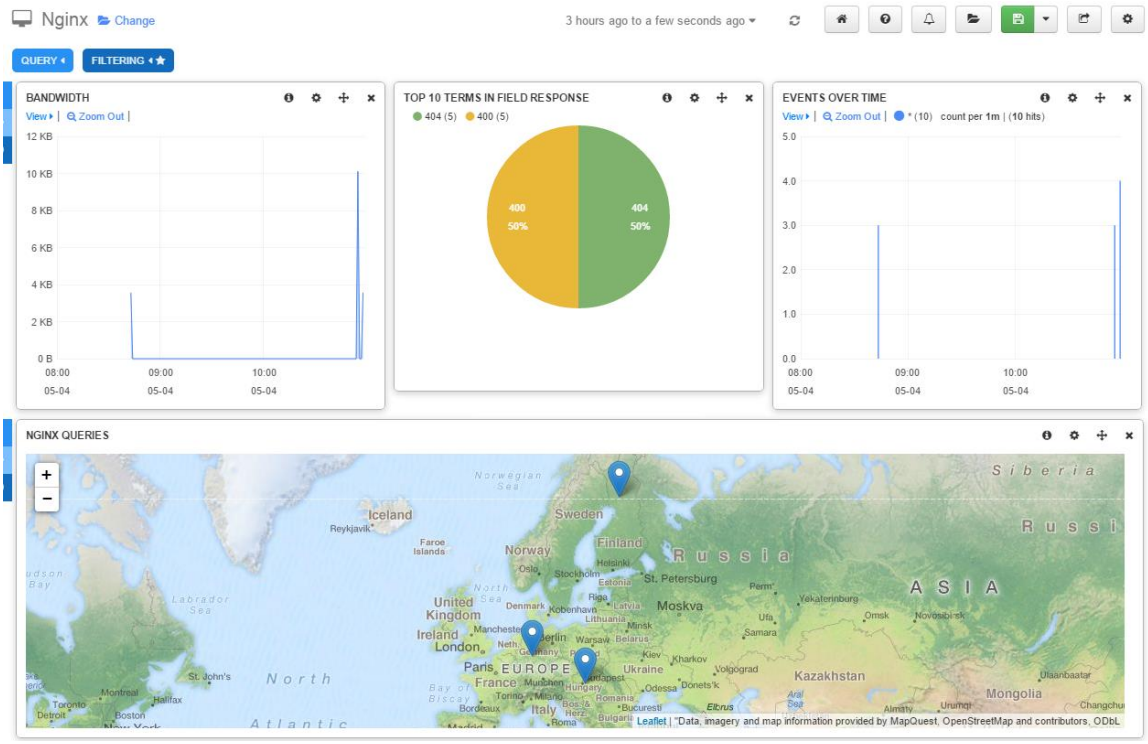
Nginxin lokitiedot otettiin NLS:llä vastaan Syslogina. Nginxille luotiin suodatin ja sille annettiin nimi *Nginx-filter*. Jos *program*-kenttä sisältää *Nginx*-palvelimella määritetyn tagin *nginx_access*, se ohjautuu suodattimeen. *Grok*-suodattimella ja *match*-parametrilla kuvion 35 lokiviesti parsitaan formaatin mukaan kenttiin, joihin viestin olennaiset osat tallennetaan. *Date*-suodattimen avulla puolestaan muuntaa viestin aikaleiman muotoon ”dd/MMM/yyyy:HH:mm:ss Z”, joka käytännössä tarkoittaa ”pvä/kk/vuosi:tunnit:min:sek aikavyöhyke”. *Mutate*-suodattimella viestin tyyppi-kenttään vaihdetaan *nginx_access* ja muuttujat *bytes* ja *response* vaihdetaan kokonaisluvuiksi. Lopuksi verkkosivun kyselijän IP-osoite poimitaan vielä *geoip*-suodattimella.

```
if [program] == "nginx_access" {
    grok {
        match => [ "message", "%{IPORHOST:clientip} -
%{USERNAME:auth} \[%{HTTPDATE:timestamp}\] \"(?:%{WORD:verb}
%{NOTSPACE:request}(?:
HTTP/%{NUMBER:httpversion})?|%{DATA:raw_http_request})\"
%{NUMBER:response} (?:%{NUMBER:bytes}|-) %{QS:referrer}
%{QS:agent} %{QS:timeduration}" ]
    }
    date {
        match => [ "timestamp", "dd/MMM/yyyy:HH:mm:ss Z" ]
    }
    mutate {
        replace => [ "type", "nginx_access" ]
        convert => [ "bytes", "integer" ]
        convert => [ "response", "integer" ]
    }
    geoip {
        source => "clientip"
    }
}
```

Nginx_error-viestien suodatus tapahtuu samassa *Nginx-filter*-suodattimessa *access*-viestien kanssa. *Program*-kenttä määrittää sen, kumpaan osioon viesti päätyy. *Program*-kentän sisältäessä *nginx_error*-tagin, se ohjautuu *mutate*-suodattimeen, jossa *nginx_error* viestejä ei ruvettu sen tarkemmin suodattamaan. Viestin tyyppi-kenttään vaihdetaan *nginx_error* ja muuten sen annettiin olla ennallaan.

```
if [program] == "nginx_error" {
    mutate {
        replace => [ "type", "nginx_error" ]
    }
}
```

Konfiguraatioiden tallentamisen jälkeen voitiin siirtyä tarkastelemaan Nginx-palvelun lähettämiä lokitietoja. Kuviossa 36 näkyy *Nginx*-dashboard, joka tehtiin keräämään vain *nginx_access*-tyypin tapahtumat. Dashboard on hyvin samankaltainen kuin IIS-palvelulla, johtuen samasta palvelutyypistä. Myös Nginx-palvelun lokitiedostoon lisättiin rivejä kuvaamaan liikennettä IIS-kohdan tapaan. Kuviossa oikeanpuoleinen osa kuvaa web-sivun latauksesta aiheutuvaa liikennettä ja vasemmanpuoleinen pylväikkö tapahtumien määrää. Ylhäällä keskellä on piirakka, joka kuvaa palvelimen antamaa vastausta sivun lataukseen. Yleisin virhe 404 (404 Not Found) esiintyy mm. vanhentuneiden linkkien kohdalla ja se kertoo epäonnistuneesta sivun lataamisesta. Geoip-suodattimella on poimittu karttaan maantieteellinen sijainti web-sivun lataajan IP-osoitteen perusteella.



Kuvio 36. Nginx-dashboard

Kuviossa 37 on avattu osa Nginx-kyselyä. Kuvion 35 lokiviesti on saatu parsittua siististi kenttiin, joilloin niistä saa selkeämmän käsityksen ja niitä pystytään hyödyntämään paremmin.

@timestamp	host	verb	response	request
2015-05-04T10:55:56.000+03:00	192.168.1.21	GET	404	/favicon.ico
2015-05-04T10:55:56.000+03:00	192.168.1.21	GET	404	/favicon.ico

View: Table / JSON / Raw

Field	Action	Value
@timestamp		2015-05-04T10:55:56.000Z
@version		1
_id		d_XSilcZRWiJ5A01TdvfHA
_index		logstash-2015.05.04
_type		nginx_access
agent		"Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/42.0.2311.135 Safari/537.36"
auth		-
bytes		4452
clientip		88.192.50.217

Kuvio 37. Nginx-web-kysely

9.4.3 Apache

Apache-web-palvelu asennettiin Ubuntu-palvelimelle, jota ennen palvelin kuitenkin päivitettiin.

```
apt-get update
apt-get install apache2
```

Asennuksen jälkeen Ubuntu-palvelin lisättiin lähettämään lokiviestit eteenpäin *Rsyslogilla*. Apache-palvelun lokitiedostot lähetettiin täysin samalla tapaa, kuin Nginx-palvelun lokiviestit. Parametreihin *-f* ja *-t* lisättiin */var/log/apache2* -kansiossa sijaitsevat *access*- ja *error*-lokitiedostot. Lisäksi Nginxistä poiketen tagiksi määritettiin *apache_access* tai *apache_error* tiedostosta riippuen.

```
curl -s -O http://192.168.1.15/nagioslogserver/scripts/setup-
linux.sh
bash setup-linux-sh -s 192.168.1.15 -p 5544 -f
/var/log/apache2/error.log -t apache_error
bash setup-linux-sh -s 192.168.1.15 -p 5544 -f
/var/log/apache2/access.log -t apache_access
```

Apache-viestien vastaanotto tapahtui Syslogina. NLS:llä oli valmiiksi tehty suodatin Apache-palvelua varten, jota ei ollut tarve muokata. *Program*-kentän sisältäessä *apache_access* tai *apache_error*, viesti ohjautui suodattimeen. *Apache_access*-osassa lokiviesti parsitaan ensin kenttiin *grok*-suodattimella ja *match*-parametrilla. Seuraavaksi viestin aikaleiman muotoa muutetaan samalla tapaa kuin Nginx-palvelussakin. Lopuksi *tyyppi*-kenttään vaihdetaan *apache_access* ja muuttujat *bytes* ja *response* vaihdetaan kokonaisluvuiksi.

```
if [program] == "apache_access" {
    grok {
        match => [ "message", "%{COMBINEDAPACHELOG}" ]
    }
    date {
        match => [ "timestamp", "dd/MMM/yyyy:HH:mm:ss Z" ]
    }
    mutate {
        replace => [ "type", "apache_access" ]
         convert => [ "bytes", "integer" ]
         convert => [ "response", "integer" ]
    }
}
```

Kuten suodattimesta näkee, lokiviesti parsitaan kenttiin vain yhdellä muuttujalla *COMPINEDAPACHELOG*. COMPINEDAPACHELOG on Logstash-malli, joka on varta vasten rakennettu vastaamaan Apache-palvelun apache_access lokiviestin muotoa. Malli löytyy kansion */usr/local/nagioslogserver/logstash/patterns/* -kansiosta ja tarkemmin grok-patterns-tiedostosta, joka sisältää myös muita grok-malleja. Seuraavassa on avattu COMPINEDAPACHELOG-muuttujan kenttiä ja samalla Apachen lokiformaattia:

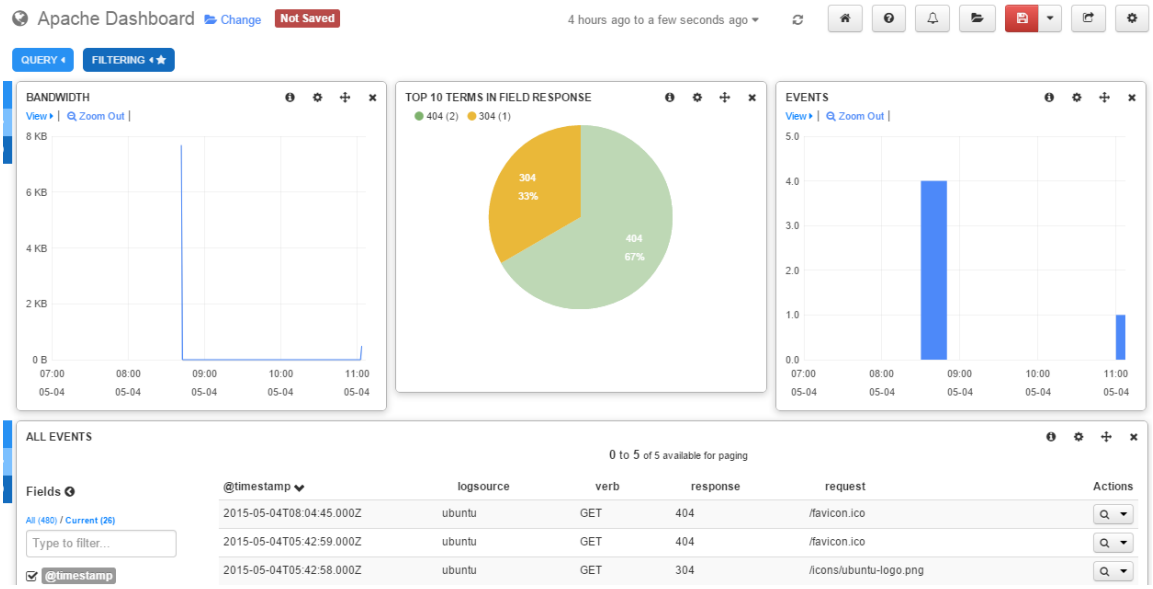
```
COMBINEDAPACHELOG %{IPORHOST:clientip} %{USER:ident} %{USER:auth}
\[%{HTTPDATE:timestamp}\] "(?:%{WORD:verb} %{NOTSPACE:request}(?:
HTTP/%{NUMBER:httpversion})?|%{DATA:rawrequest})"
%{NUMBER:response} (?:%{NUMBER:bytes}|-) %{QS:referrer}
%{QS:agent}
```

Apache_error-viestien suodatus tapahtuu samassa suodattimessa access-viestien kanssa. *Program*-kentän ollessa *apache_error*, viesti suodatetaan hieman eri tavalla. Error-viestiin ei Logstashilla ole samanlaista valmista mallia, vaan viesti parsitaan kentiksi osa

kerrallaan. Lopuksi viestin *tyyppi*-kenttä vaihdetaan tuttuun tapaan kuvaamaan paremmin tapahtumaa.

```
if [program] == "apache_error" {
    grok {
        match => [ "message", "\[(?<timestamp>%{DAY:day}
%{MONTH:month} %{MONTHDAY} %{TIME} %{YEAR})\] \[%{WORD:class}\]
\[%{WORD:originator} %{IP:clientip}\] %{GREEDYDATA:errmsg}" ]
    }
    mutate {
        replace => [ "type", "apache_error" ]
    }
}
```

NLS sisälsi valmiin suodattimen lisäksi myös valmiin Apache dashboardin (ks. kuvio 38). Dashboardi oli valmistajien toimesta räätälöity melko selkeäksi ja havainnolliseksi, josta otettiin myös mallia muihin web-palvelu-dashboardeihin. Karttaa ja pieniä yksityiskohtia lukuun ottamatta dashboard on identtinen Nginxin kanssa.



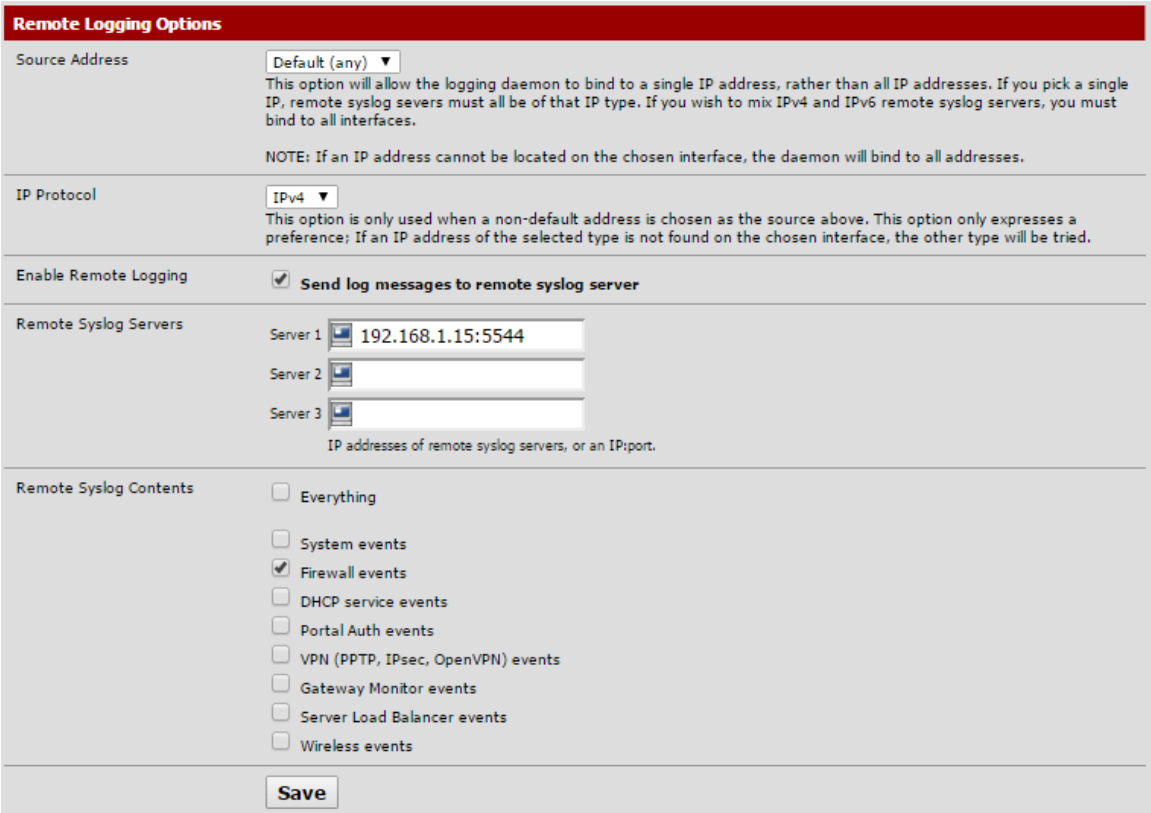
Kuvio 38. Apache-dashboard

9.5 Verkkolaitteet

Virtuaaliympäristön tuomista haasteista huolimatta yhden verkkolaitteen lokitietoja päästiin tarkastelemaan. NLS:llä oli yleiset ohjeet verkkolaitteiden lisäämiseen ja laitteesta riippumatta asetukset olivat kaikille samat:

Nagios Log Server IP: 192.168.1.15
Nagios Log Server Port (TCP/UDP): 5544

PfSensen palomuurilokien lähettäminen onnistui kuvion 39 käyttöliittymän kautta. Windows-työaseman selaimella siirryttiin pfSense-koneen IP-osoitteeseen ja sieltä lokiasetuksiin. NLS:n ohjeiden mukaan määritettiin lokipalvelimen IP-*osoite* sekä *portti* ja lähetettäviksi tapahtumiksi valittiin palomuuritapahtumat. Asetukset tallennettiin, jonka jälkeen lokitapahtumat alkoivat virrata NLS:lle.



Kuvio 39. pfSense-lokiasetukset

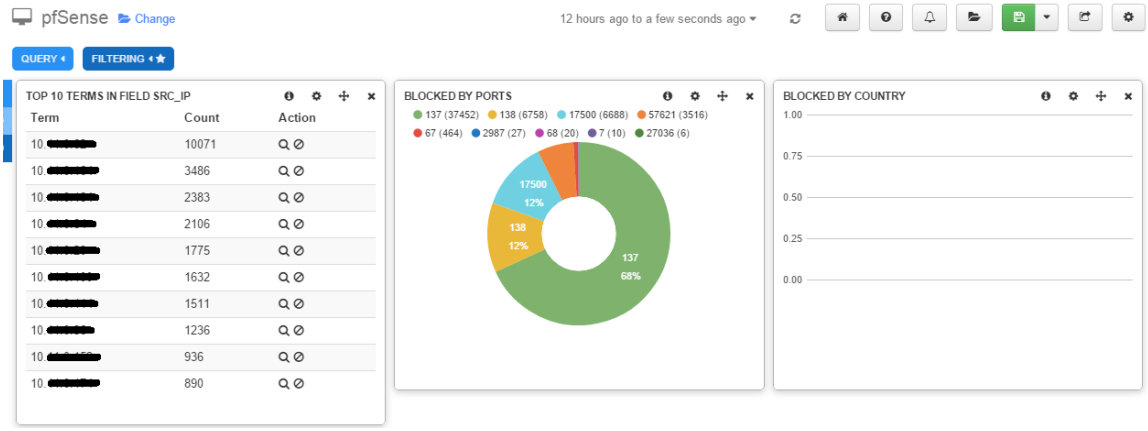
Viestit saapuivat NLS:lle teoria-osuudessa sijaitsevan kuvion 5 muotoisina, joten niille tehtiin suodatin nimeltä *pfSense-filter*. Koska pfSense-koneelta vastaanotettiin vain palomuuriloki, voitiin suodatin tehdä perustuen sen IP-osoitteeseen. Kaikki viestit, jotka saapuivat osoitteesta *192.168.1.254*, päätyivät tähän suodattimeen. Tutuksi tulleella *grok*-suodattimella lisättiin ensin tagi *pfsense* ja sitten viesti parsittiin kentiksi. Viestistä poimittiin *aikaleima* ja *ohjelma* ja viestin loppuosa tallennettiin muuttujaan *msg*. Muuttuja *message* korvattiin äsken luodulla muuttujalla *msg* ja sen jälkeen myös se parsittiin yksittäisiin kenttiin. Viesti olisi myös voitu parsia yhdellä kertaa, mutta tämä tapa sel-

keyttää toimenpidettä ja helpottaa mahdollista myöhempää muokkaamista, jos esim. pfSense-koneelta lokitettaisiin muitakin palveluita. Viestistä poimitaan vielä *geoip*-tieto ja lopuksi poistetaan ylimääräinen muuttuja *msg*.

```
if [host] == "192.168.1.254" {
    grok {
        add_tag => [ "pfsense" ]
        match => [ "message",
"<%{POSINT:syslog_pri}>(?<datetime>(?:Jan(?:uary)?|Feb(?:ruary)?|
Mar(?:ch)?|Apr(?:il)?|May|Jun(?:e)?|Jul(?:y)?|Aug(?:ust)?|Sep(?:t
ember)?|Oct(?:ober)?|Nov(?:ember)?|Dec(?:ember)?)\s+(?:(?:0[1-
9])|(?:[12][0-9])|(?:3[01])|[1-9]) (?:2[0123]|[01]?[0-9]):(?:[0-
5][0-9]):(?:[0-5][0-9])) (?<prog>.*?): (?<msg>.*)" ]
    }
    mutate {
        replace => [ "message", "%{msg}" ]
    }
    grok {
        match => [ "message",
"(%{INT:rule}),(%{INT:sub_rule}),,(%{INT:tracker}),(%{WORD:iface}
),(%{WORD:reason}),(%{WORD:action}),(%{WORD:direction}),(%{INT:ip
_ver}),(%{BASE16NUM:tos}),,(%{INT:ttl}),(%{INT:id}),(%{INT:offset
}),(%{WORD:flags}),(%{INT:proto_id}),(%{WORD:proto}),(%{INT:lengt
h}),(%{IP:src_ip}),(%{IP:dest_ip}),(%{INT:src_port}),(%{INT:dest_
port}),(%{INT:data_length})" ]
    }
    geoip {
        source => "src_ip"
    }
    mutate {
        remove_field => [ "msg" ]
    }
}
```

PfSense-suodattimen teon jälkeen viestejä oli selvästi mukavampi tarkastella. Palomuurilokeille luotiin oma *pfSense*-dashboard (ks. kuvio 40), joka näyttää kaikki viestit IP-osoitteesta 192.168.1.254. Kuvion 40 vasemmassa laidassa näkyy lista aktiivisimpien laitteiden IP-osoitteista, joilta tapahtumia palomuurille on saapunut. Keskellä näkyvä donitsi kertoo estettyjen tapahtumien portti-jakauman. Virtuaaliympäristöstä ja sisäverkon IP-osoitteista johtuen kuvion oikeanpuoleiseen taulukkoon ei saatu haluttuja tietoja näkymään. Siinä oli tarkoitus havainnollistaa sijaintitietojen avulla estettyjen tapahtumien maat pylväsdiagrammina. Palomuuri muutenkin olisi ollut hyvä laite sijaintitieto-

jen käyttämiseksi toimiessaan väylänä maailmalle, mutta tässä tapauksessa niitä ei saatu hyödynnettyä.



Kuvio 40. pfSense-dashboard

Kuviossa 41 on palomuuritapahtumia. Kaikki tapahtumat näyttäisivät olevan *estettyjä*, suuntana *sisäänpäin* ja protokollana *UDP*. Tapahtumaa klikkaamalla aukeavat tapahtuman konfiguraatioissa määritetyt kentät ja niiden sisällöt.

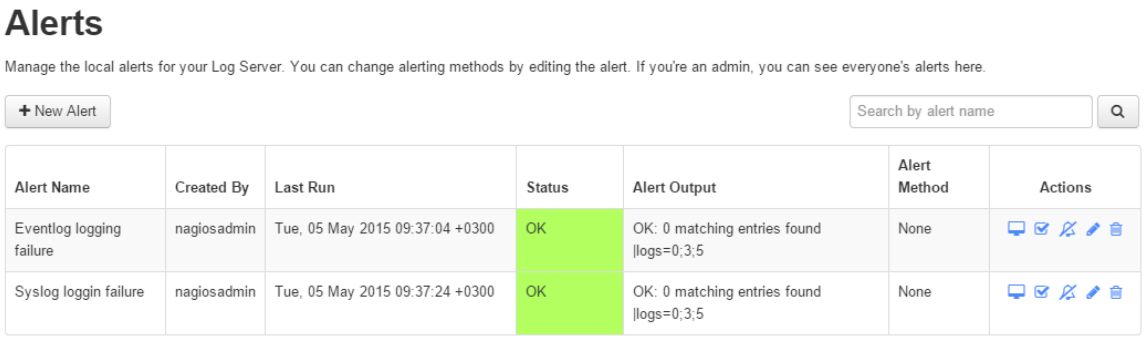
@timestamp	iface	action	direction	proto	syslog_pri	dest_port
2015-05-04T11:08:16.923+03:00	em0	block	in	udp	134	137
2015-05-04T11:08:13.937+03:00	em0	block	in	udp	134	17500
2015-05-04T11:08:13.937+03:00	em0	block	in	udp	134	137
2015-05-04T11:08:13.936+03:00	em0	block	in	udp	134	137
2015-05-04T11:08:13.928+03:00	em0	block	in	udp	134	137
2015-05-04T11:08:13.928+03:00	em0	block	in	udp	134	137

Kuvio 41. pfSense-tapahtumat

9.6 Hälytykset

Dashboardeista ja sen sisältämistä suodattimista ja kyselyistä oli mahdollista muodostaa hälytyksiä ja niitä pystyy tarkastelemaan NLS:llä alerting-välilehdellä. Annettujen raja-arvojen ylitys tietyllä aikavälillä laukaisee hälytyksen, josta on mahdollista ilmoittaa ylläpitäjille halutulla tavalla. Kuviossa 42 näkyvät tehdyt hälytykset, jota kokeiltiin epäonnistuneista Event log- sekä Syslog-kirjautumisista. Hälytyksille annettiin sitä kuvaava nimi ja raja-arvot. Varoitus-ilmoituksen (warning) raja-arvoiksi määritettiin kolme tapahtumaa ja kriittinen-ilmoituksen raja-arvoksi (critical) viisi tapahtumaa. Hälytyksien tarkistamis-

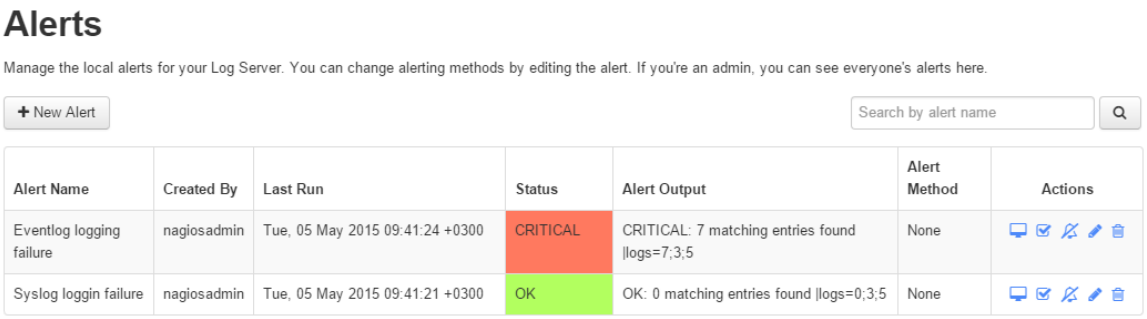
tiheydeksi asetettiin 5 minuuttia ja tarkistamisaikaväliksi 15 minuuttia. Hälytyksistä näkee viimeksi suoritetun ajankohdan sekä statuksen, joka määräytyy raja-arvojen ylittyessä. Alert output kertoo hälytyksien osumien tilanteen suhteessa raja-arvoihin. Jos hälytyksien rajaamia tapahtumia tapahtuu, laskuri näyttää tapahtumamäärät ja muuttaa statusta aina raja-arvojen ylittyessä.



Kuvio 42. NLS-hälytykset

Käyttöliittymäesittelyssä on kerrottu eri hälytystapojen mahdollisuuksista. Yhteistoimintaa muiden valvontasovellusten kanssa olisi ollut varsin mielenkiintoista kokeilla, mutta ympäristöstä johtuen vaihtoehtoja ei pystytty toteuttamaan, eikä hälytyksistä lopulta ilmoitettu mitenkään. Oikealla näkyvistä painikkeista hälytystä voi tarkastella dashboardissa, sen voi suorittaa saman tien, hälytyksen voi mitätöidä tai sitä voi muokata tai poistaa kokonaan.

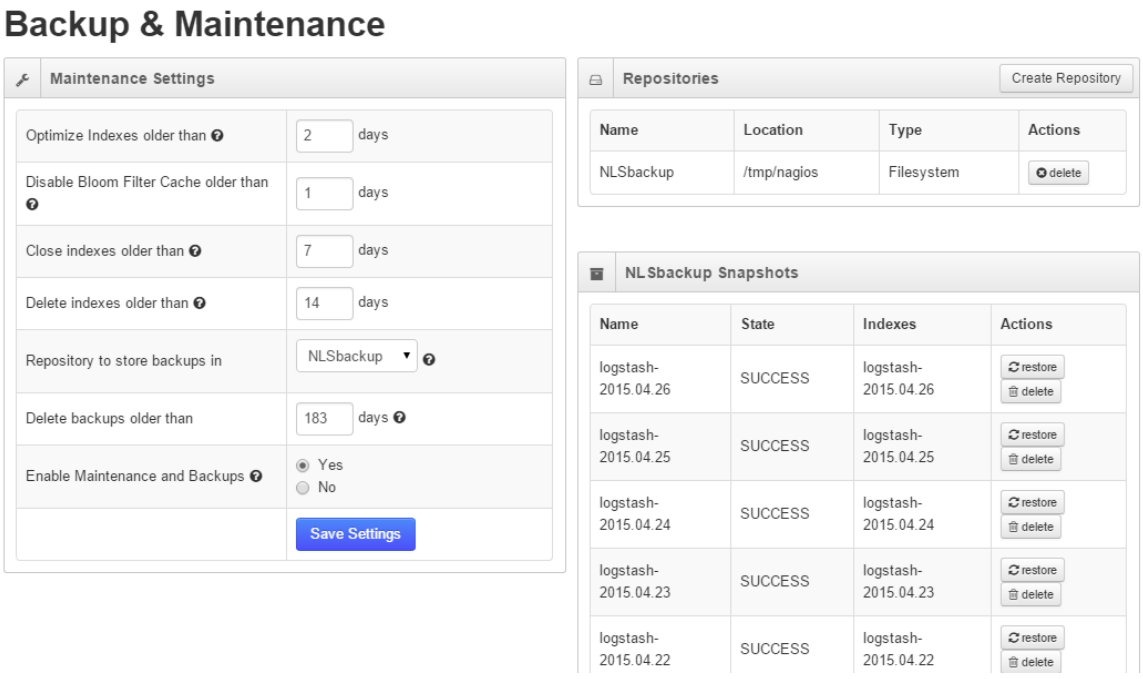
Event log -hälytystä kokeiltiin syöttämällä käyttäjälle seppota väärä salasana useaan kertaa Windows-työaseman kirjautumisikkunassa. Kuviossa 43 alert output näyttää seitsemän tapahtuman osuneen hälytyksen ehtoihin, jonka seurauksena hälytyksen statuksen väri on muuttunut punaiseksi ja tila kriittiseksi. Ilmoitus kohdan ollessa tyhjä hälytyksestä ei kuitenkaan ilmoiteta mitenkään. Syslog-hälytykseen ei ole tullut yhtään osumaa ja se on pysynyt muuttumattomana.



Kuvio 43. NLS-hälytykset2

9.7 Varmuuskopiointi

NLS:n administration-välilehdeltä löytyy varmuuskopiointi- ja huolto-osio, joka näkyy kuviossa 44. Sitä kautta määritetään lokitietojen säilyttäminen ja varmuuskopiointi. Toiminnot onnistuivat käyttöliittymän kautta ja ensimmäisenä lokiviesteille luotiin säilytyspaikka (repository), jonne varmuuskopiot tallennettiin. Säilytyspaikalle annettiin nimeksi *NLSbackup* ja tiedostopoluksi */tmp/nagios/*. Jos käytössä on useampi instanssi, tulee varmistaa sijainnin olevan jaettu verkkopolku, johon kaikki instanssit pääsevät käsiksi.



Kuvio 44. NLS-varmuuskopiointi-asetukset

Säilytyspaikan teon jälkeen kuvion 44 vasemmassa laidassa näkyvillä laatikoilla määritettiin asetukset varmuuskopiointiin. Päivän kaikki lokitiedot tallennetaan päivittäin omiin indekseihin ja ensimmäisellä laatikolla määritetään minkä ajan jälkeen indeksi ei enää hyväksy uusia tietoja ja optimoi vanhat. Toiselle laatikolla määritetään bloom filter -välimuistin toiminta, joka lisää muistin käyttöä. Seuraavilla laatikoilla määritetään, milloin indeksit suljetaan ja milloin ne poistetaan. Sulkemisen jälkeen indeksi ei vie muita resursseja kuin tallennustilaa, mutta tietojen tarkastelu vaatii indeksin uudelleen avaamisen. Indeksien poistaminen vapauttaa myös levytilan. Poistamisen jälkeen ainut tapa tarkastella tietoja on palauttaa ne arkistoiduista varmuuskopioista. Alasvetovalikosta valitaan säilytyspaikka ja sen jälkeen määritetään aika, jonka jälkeen varmuuskopiot poistetaan. Jos varmuuskopioita halutaan käyttää, tulee täppä olla kyllä-kohdassa ja lopuksi kuviossa on vielä asetuksien tallennuspainike.

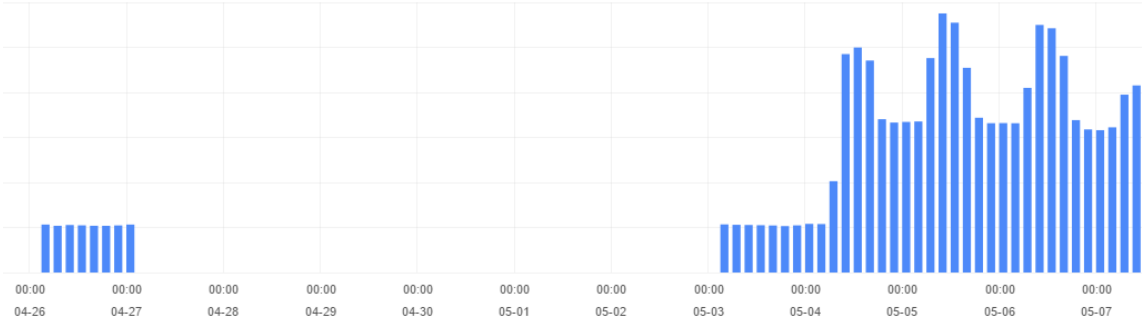
Varmuuskopiointi-asetukset määritettiin kuviossa 44 näkyvällä tavalla. Indeksien optimoinniksi asetettiin kaksi päivää, bloom filteriä käytetään yksi päivä, indeksit suljetaan yhden viikon ja poistetaan kahden viikon jälkeen. Säilytyspaikaksi valittiin aikaisemmin tehty NLSbackup ja varmuuskopioiden säilyttämisajaksi KATAKRI perustason (IV) vaatimusten mukaisesti asetettiin puoli vuotta.

Kuvion 44 vasemmassa alareunassa näkyvät varmuuskopioiden tilatiedot. Siinä näkyvät niiden nimet, tila, käytetty indeksi ja toimintopainikkeet. Varmuuskopioiden nimi määräytyy päivämäärän mukaan. Tila on kaikilla success, joka kertoo onnistuneesta varmuuskopioinnista. Indexes-kohdassa varmuuskopiosta näkee, mitä indeksiä se on käyttänyt. Varmuuskopio logstash-2015.04.26 on luonnollisesti käyttänyt saman päivän indeksiä logstash-2015.04.26. Kuviossa 45 on varmuuskopioiden säilytyspaikan eli NLS-palvelimen */tmp/nagios/* sisältö listattuna, joka käyttöliittymän lisäksi todentaa tiedostojen olemassaolon.

```
[root@nagiosls nagios]# ls
index                          snapshot-logstash-2015.04.05  snapshot-logstash-2015.04.17
indices                        snapshot-logstash-2015.04.06  snapshot-logstash-2015.04.18
metadata-logstash-2015.04.19   snapshot-logstash-2015.04.07  snapshot-logstash-2015.04.19
metadata-logstash-2015.04.20   snapshot-logstash-2015.04.08  snapshot-logstash-2015.04.20
metadata-logstash-2015.04.21   snapshot-logstash-2015.04.09  snapshot-logstash-2015.04.21
metadata-logstash-2015.04.22   snapshot-logstash-2015.04.10  snapshot-logstash-2015.04.22
metadata-logstash-2015.04.23   snapshot-logstash-2015.04.11  snapshot-logstash-2015.04.23
metadata-logstash-2015.04.24   snapshot-logstash-2015.04.12  snapshot-logstash-2015.04.24
metadata-logstash-2015.04.25   snapshot-logstash-2015.04.13  snapshot-logstash-2015.04.25
metadata-logstash-2015.04.26   snapshot-logstash-2015.04.14  snapshot-logstash-2015.04.26
snapshot-logstash-2015.03.19   snapshot-logstash-2015.04.15
snapshot-logstash-2015.04.04   snapshot-logstash-2015.04.16
```

Kuvio 45. NLS-varmuuskopiointi-tiedostot

Indeksien poistamiseksi asetettiin kaksi viikkoa, jonka jälkeen tapahtumat eivät olleet tarkasteltavissa dashboardeilla. Varmuuskopiointia kokeiltiin palauttamalla kuvioissa 44 sekä 45 näkyvä logstash-2015.04.26-varmuuskopio. Kuten kuvion 46 dashboardista näkee, varmuuskopion palauttaminen onnistui ja kyseisen päivän lokitiedot olivat jälleen luettavissa.



Kuvio 46. Varmuuskopion palauttaminen

10 Pohdinta

Alussa asetettujen tavoitteiden mukaisesti työssä saatiin toteutettua toimiva lokijärjestelmä, joka kerää, yhtenäistää ja analysoi erilaisia lokitietoja. Nagios Log Server -sovellus vastasi hyvin yhtiön myyntipuhetta ja toimi odotusten mukaisesti. Lokitietoja pystyttiin analysoimaan mm. lähteen, tyypin ja ajankohdan perusteella ja visualisoimaan lokitietojen tarjoamien tietojen rajoissa. Monipuoliset ominaisuudet, selkeä käyttöliittymä ja kohtuullinen hinnoittelu vakuuttivat sovelluksen olevan varsin kilpailukykyinen tuote.

Työn aihe oli entuudestaan melko tuntematon, joten liikkeelle lähdettiin tutustumalla ensin teoriaan. Tietopohjan hankkimisen jälkeen siirryttiin lokijärjestelmän toteutukseen. Työn selvästi eniten aikaa vaativin osio oli suodattimien teko ja niiden testaaminen eri lokeille. Myös itse ympäristön pystytys ja NLS-sovelluksen ominaisuuksiin tutustuminen veivät paljon aikaa, eikä toista instanssia keritty lokijärjestelmään asentamaan. Hälytyksiin, varmuuskopiointiin ja muihin ylläpitoon liittyviin ominaisuuksiin tutustuminen jäi vähäiseksi järjestelmän laajaan perustoimintaan tutustumisen vuoksi.

NLS julkaistiin syksyllä 2014, joten kyseessä on alle vuoden vanha sovellus. Sovelluksesta saatava tieto oli melko vähäistä ja yhtiön omilla tukisivustoilla osan dokumentaatioista kerrottiin olevan työn alla ja ilmestyvän lähiaikoina. Uskon kaupallisen tuotteen kehitystyön olevan aktiivista ja sen myötä sovelluksen tuki ja ominaisuudet varmasti paranevat ja lisääntyvät.

Sovelluksen siirtäminen tuotantoympäristöön ja vähemmälle jääneisiin aiheisiin tutustuminen olisi mielenkiintoista. Järjestelmä, joka toimi melko sujuvasti alle minimivaatimusten, toimisi asianmukaisessa laitteistossa varmasti laajassakin ympäristössä. Lokijärjestelmän liittäminen muiden tuotteiden kanssa osaksi organisaation valvontajärjestelmää parantaa organisaation tietoturvaa.

Lähteet

About. 2015. NXLOG Community Edition. Sourceforge, verkkosivut. Viitattu 20.4.2015. http://nxlog-ce.sourceforge.net/about

Banon, S. 2010. The Future of Compass & Elasticsearch. Viitattu 26.3.2015. https://web.archive.org/web/20130827121405/http://www.kimchy.org/the_future_of_compass/

Cholakian, A. 2013. Exploring Elasticsearch. Viitattu 25.3.2015. http://exploringelasticsearch.com/

Company. 2015. Qvantelin verkkosivut. Viitattu 5.3.2015. https://www.qvantel.com/company/

Docs. 2015. Logstash, verkkosivut. Viitattu 31.3.2015. http://logstash.net/docs/1.4.2/

Event Logs. 2015. Microsoft TechNet Library. Viitattu 25.2.2015. https://technet.microsoft.com/en-us/library/cc722404.aspx

Features. 2015a. Nagios Log Server Features. Nagios Enterprises, verkkosivut. Viitattu 1.4.2015. http://www.nagios.com/products/nagios-log-server/features

Features. 2015b. Rsyslog, verkkosivut. Viitattu 5.3.2015. http://www.rsyslog.com/features/

Gerhards, R. 2009. The Syslog Protocol. Viitattu 2.3.2015. http://tools.ietf.org/html/rfc5424

Gerhards, R. 2007. why does the world need another syslogd? (aka rsyslog vs. syslog-ng). Rainers's Blog. Viitattu 5.3.2015. http://blog.gerhards.net/2007/08/why-does-world-need-another-syslogd.html

Hallam-Baker, P. & Behlendorf, B. N.d. Extended Log File Format. W3C Working Draft. Viitattu 25.3.2015. http://www.w3.org/TR/WD-logfile

Kansallinen turvallisuusauditointikriteeristö. 2011. Puolustusministeriö. Viitattu 23.2.2015. http://www.defmin.fi/files/1870/KATAKRI_versio_II.pdf

Kansallinen turvallisuusauditointikriteeristö (KATAKRI). 2015. Suomen Puolustusministeriön verkkosivut. Viitattu 23.2.2015. http://www.defmin.fi/hallinnonala/puolustushallinnon_turvallisuustoiminta/kansallinen_turvallisuusauditointikriteeristo_%28katakri%29

Kibana. 2015. Kibana, verkkosivut. Viitattu 31.3.2015. http://kibana.org/

Logstash. 2015. Logstash, verkkosivut. Viitattu 31.3.2015. http://logstash.net/

Lokienhallinta. 2015. Nixu Corporation. Viitattu 6.5.2015. http://www.nixu.com/fi/palvelualueet/lokienhallinta

Lokiohje. 2009. Valtioinvarainministeriön verkkosivut. Viitattu 23.2.2015. https://www.vahtiohje.fi/c/document_library/get_file?uuid=775179cb-6c54-4dfb-b65d-e925d47c61d2&groupId=10128&groupId=10229

Lonvick, C. 2001. The BSD syslog Protocol. Viitattu 2.3.2015. https://tools.ietf.org/html/rfc3164

Menn, V. 2006. New tools for Event Management in Windows Vista. Microsoft TechNet Magazine. Viitattu 25.2.2015. https://technet.microsoft.com/en-us/magazine/2006.11.eventmanagement.aspx

Overview. 2015. Nagios Log Server Overview. Nagios Enterprises, verkkosivut. Viitattu 1.4.2015. http://www.nagios.com/products/nagios-log-server/overview

Platforms. 2009. Rsyslog wiki. Viitattu 5.3.2015. http://wiki.rsyslog.com/index.php/Platforms

Pricing. 2015. Nagios Log Server Pricing. Nagios Enterprises, verkkosivut. Viitattu 2.5.2015. http://www.nagios.com/products/nagios-log-server/pricing

Product features and benefits. 2015. BalaBit IT Security, verkkosivut. Viitattu 4.3.2015. http://www.balabit.com/network-security/syslog-ng/opensource-logging-system/features

Professional Services. 2015. Rsyslog verkkosivut. Viitattu 5.3.2015. http://www.rsyslog.com/professional-services/

Reliable log management. 2015. BalaBit IT Security, verkkosivut. Viitattu 4.3.2015. http://www.balabit.com/network-security/syslog-ng/opensource-logging-system

The Foundation of Log Management. 2015. BalaBit IT Security, verkkosivut. Viitattu 4.3.2015. http://www.balabit.com/network-security/syslog-ng

Welcome to VirtualBox.org! 2015. VirtualBox, verkkosivut. Viitattu 20.4.2015. https://www.virtualbox.org/

Wilkerson, S. 2014. Nagios Log Server vs. Elasticsearch - Logstash - Kibana. Nagios Labs. Viitattu 2.4.2015. http://labs.nagios.com/2014/10/19/nagios-log-server-vs-elasticsearch-logstash-kibana/?utm_source=twitterfeed&utm_medium=twitter

Liitteet

Liite 1. NLS konfiguraatiot

```
#
# Global Configuration
#

input {
    syslog {
        type => 'syslog'
        port => 5544
    }
    tcp {
        type => 'eventlog'
        port => 3515
        codec => json {
            charset => 'CP1252'
        }
    }
    tcp {
        type => 'import_raw'
        tags => 'import_raw'
        port => 2056
    }
    tcp {
        type => 'import_json'
        tags => 'import_json'
        port => 2057
        codec => json
    }
    tcp {
        type => 'iislog'
        port => 3514
        codec => json
    }
    tcp {
        type => 'dnslog'
        port => 3513
        codec => json
    }
}

filter {
    if [program] == 'apache_access' {
        grok {
            match => [ 'message', '%{COMBINEDAPACHELOG}']
```

```
        }
        date {
            match => [ 'timestamp', 'dd/MMM/yyyy:HH:mm:ss Z' ]
        }
        mutate {
            replace => [ 'type', 'apache_access' ]
             convert => [ 'bytes', 'integer' ]
             convert => [ 'response', 'integer' ]
        }
    }
    if [program] == 'apache_error' {
        grok {
            match => [ 'message', '\[(?<timestamp>%{DAY:day}
%{MONTH:month} %{MONTHDAY} %{TIME} %{YEAR})\] \[%{WORD:class}\]
\[%{WORD:originator} %{IP:clientip}\] %{GREEDYDATA:errmsg}']
        }
        mutate {
            replace => [ 'type', 'apache_error' ]
        }
    }

    if [type] == "eventlog" {
        if [EventID] == 4624 {
            mutate {
                add_tag => [ "logon-success" ]
            }
        }
        if [EventID] == 4634 {
            mutate {
                add_tag => [ "logoff-success" ]
            }
        }
        if [EventID] == 4771 or [EventID] == 4625 {
            mutate {
                add_tag => [ "logon-failure" ]
            }
        }
        if [EventType] == "AUDIT_FAILURE" {
            if [EventID] == 4768 {
                mutate {
                    add_tag => [ "logon-failure" ]
                }
            }
        }
    }

    if [type] == "iislog" {
        mutate {
            replace => [ "@source_host", "server.siem.test" ]
            add_field => { "requesturl" =>
"%{fqdn}%{request}%{querystring}" }
```

```
        }
        geoip {
            source => "clientip"
        }
    }

    if [type] == "dnslog" {
        grok {
                    match=> [ "message",
"%{DATE_EU}%{SPACE}%{TIME}%{SPACE}%{WORD:dns_thread_id}%{SPACE}%{
WORD:dns_context}%{SPACE}%{WORD
:dns_packet_id}%{SPACE}%{WORD:dns_ip_protocol}%{SPACE}%{WORD:dns_
direction}%{SPACE}%{IP:dns_client_ip}%{SPACE}%{WORD:dns_xid}
(?:%{WORD:dns_query_type}|\s*)
Q%{SPACE}\[%{NUMBER:dns_flags_hex}%{SPACE}(?:%{WORD:dns_flags_cha
rs}|\s*)%{SPACE}%{WORD:dns_response}\]%{SPACE}%{WORD:dns_question
_type}%{SPACE}%{GREEDYDATA:dns_question_name}" ]
          }
        geoip {
                    source => "dns_client_ip"
          }
    }

    if [program] == "nginx_access" {
        grok {
            match => [ "message", "%{IPORHOST:clientip} -
%{USERNAME:auth} \[%{HTTPDATE:timestamp}\] \"(?:%{WORD:verb}
%{NOTSPACE:request}(?:
HTTP/%{NUMBER:httpversion})?|%{DATA:raw_http_request})\"
%{NUMBER:response} (?:%{NUMBER:bytes}|-) %{QS:referrer}
%{QS:agent} %{QS:timeduration}" ]
        }
        date {
            match => [ "timestamp", "dd/MMM/yyyy:HH:mm:ss Z" ]
        }
        mutate {
            replace => [ "type", "nginx_access" ]
             convert => [ "bytes", "integer" ]
             convert => [ "response", "integer" ]
        }
          geoip {
                    source => "clientip"
          }
    }
    if [program] == "nginx_error" {
        mutate {
            replace => [ "type", "nginx_error" ]
          }
    }

    if [host] == "192.168.1.254" {
```

```
      grok {
        add_tag => [ "pfsense" ]
        match => [ "message",
"<%{POSINT:syslog_pri}>(?<datetime>(?:Jan(?:uary)?|Feb(?:ruary)?|
Mar(?:ch)?|Apr(?:il)?|May|Jun(?:e)?|Jul(?:y)?|Aug(?:ust)?|Sep(?:t
ember)?|Oct(?:ober)?|Nov(?:ember)?|Dec(?:ember)?)\s+(?:(?:0[1-
9])|(?:[12][0-9])|(?:3[01])|[1-9]) (?:2[0123]|[01]?[0-9]):(?:[0-
5][0-9]):(?:[0-5][0-9])) (?<prog>.*?): (?<msg>.*)" ]
      }
      mutate {
        replace => [ "message", "%{msg}" ]
      }
      grok {
        match => [ "message",
"(%{INT:rule}),(%{INT:sub_rule}),,(%{INT:tracker}),(%{WORD:iface}
),(%{WORD:reason}),(%{WORD:action}),(%{WORD:direction}),(%{INT:ip
_ver}),(%{BASE16NUM:tos}),,(%{INT:ttl}),(%{INT:id}),(%{INT:offset
}),(%{WORD:flags}),(%{INT:proto_id}),(%{WORD:proto}),(%{INT:lengt
h}),(%{IP:src_ip}),(%{IP:dest_ip}),(%{INT:src_port}),(%{INT:dest_
port}),(%{INT:data_length})" ]
      }
      geoip {
        source => "src_ip"
      }
      mutate {
        remove_field => [ "msg" ]
      }
    }
}

#
# Local Configuration
#
```

Liite 2. Nxlog.conf-tiedosto

```
## See the nxlog reference manual at
## http://nxlog.org/nxlog-docs/en/nxlog-reference-manual.html

## Please set the ROOT to the folder your nxlog was installed
into,
## otherwise it will not start.
#define ROOT C:\Program Files\nxlog
define ROOT C:\Program Files (x86)\nxlog
define CERT %ROOT%\cert

Moduledir %ROOT%\modules
CacheDir %ROOT%\data
Pidfile %ROOT%\data\nxlog.pid
SpoolDir %ROOT%\data
LogFile %ROOT%\data\nxlog.log

# Include fileop while debugging, also enable in the output
module below
#<Extension fileop>
#   Module    xm_fileop
#</Extension>

<Extension json>
    Module    xm_json
</Extension>

<Extension syslog>
    Module    xm_syslog
</Extension>

<Extension w3c>
#map iis log fields to Field Types
    Module    xm_csv
    Fields    $date, $time, $website, $hostname, $serverip,
$verb, $request, $querystring, $dstport, $user, $clientip,
$httpversion, $useragent, $cookie, $referrer, $fqdn, $status,
$substatus, $sc_win32_status, $sc_bytes, $cs_bytes, $time_taken
    FieldTypes  string, string, string, string, string, string,
string, string, string, string, string, string, string, string,
string, string, integer, integer, integer, integer, integer,
integer
    Delimiter   ' '
</Extension>

<Input internal>
    Module   im_internal
</Input>
```

```
# Watch your own files
<Input file1>
    Module   im_file
    File     '%ROOT%\data\nxlog.log'
    SavePos  TRUE
</Input>

# Windows Event Log
<Input eventlog>
# Uncomment im_msvistalog for Windows Vista/2008 and later
    Module im_msvistalog

# Uncomment im_mseventlog for Windows XP/2000/2003
#   Module im_mseventlog
</Input>

#Watch your IIS log files
<Input iislogs>
    Module    im_file
    File      'C:\Lokitus\iis\W3SVC1\u_ex*.log'
    ReadFromLast TRUE
    SavePos   TRUE
    Exec      if $raw_event =~ /^#/ drop();              \
        else                                             \
        {                                                \
            w3c->parse_csv();                            \
            $EventTime = parsedate($date + " " + $time); \
            to_json ();                                  \
        }
</Input>

#Watch your DNS log files
<Input dnslogs>
    Module    im_file
    File      'C:\Lokitus\dns\dns.log'
    ReadFromLast TRUE
    SavePos  TRUE
    Exec     $Message = $raw_event;
</Input>

<Output out>
    Module   om_tcp
    Host     192.168.1.15
    Port     3515

    Exec    $tmpmessage = $Message; delete($Message);
rename_field("tmpmessage","message");
    Exec    $raw_event = to_json();

          # Uncomment for debug output
```

```
        # Exec file_write('%ROOT%\data\nxlog_output.log',
$raw_event + "\n");
</Output>

<Output iisout>
    Module   om_tcp
    Host     192.168.1.15
    Port     3514
</Output>

<Output dnsout>
    Module   om_tcp
    Host     192.168.1.15
    Port     3513
</Output>

<Route 1>
    Path    internal, file1, eventlog => out
</Route>

<Route 2>
    Path    iislogs => iisout
</Route>

<Route 3>
    Path    dnslogs => dnsout
</Route>
```