jamk

Lokienhallintajärjestelmän suunnittelu ja toteutus

Jere Pesonen

Opinnäytetyö, AMK
Joulukuu 2022
Tietojenkäsittely ja tietoliikenne

jamk | Jyväskylän ammattikorkeakoulu
University of Applied Sciences

jamk

Kuvailulehti

Pesonen, Jere

Lokienhallintajärjestelmän suunnittelu ja toteutus

Jyväskylä: Jyväskylän ammattikorkeakoulu. Joulukuu 2022, 61 sivua.

Tietojenkäsittely ja tietoliikenne. Tieto- ja viestintätekniikka. Opinnäytetyö insinööri (AMK).

Julkaisun kieli: suomi

Julkaisulupa avoimessa verkossa: kyllä

Tiivistelmä

Verkkolaitteiden lokienhallinta on tärkeä osa tietoverkkojen valvontaa ja sen avulla voidaan yhtenäisesti tarkastella, sekä tallentaa ympäristön kaikkien verkkolaitteiden tuottamat lokitiedot keskitettyyn paikkaan. Lokienhallintajärjestelmä tarjoaa näkyvyyden verkon vallitsevasta tilasta, sekä mahdollisista virheistä.

Tavoitteena oli suunnitella ja toteuttaa lokienhallintajärjestelmä Puolustusvoimien Johtamisjärjestelmäkeskuksen testiverkkoon, joka on tarkoitettu uusien työkalujen sekä järjestelmien testaus- ja koulutuskäyttöön. Lokienhallintajärjestelmä mahdollistaa lokitietojen keräämisen, arkistoinnin, ja analysoinnin eri lokilähteiltä yhtenäisesti, sekä järjestelmien tapahtumien reaaliaikaisen seurannan ja mahdollisten poikkeamien tunnistamisen. Lisäksi tavoitteena oli oppia ymmärtämään tarkemmin lokienhallinnan käytäntöjä, sekä lokien elinkaaren eri vaiheita.

Lokitieto termiä käytetään tietotekniikassa erilaisten verkkolaitteiden, järjestelmien tai sovellusten tuottamaa aikaleimalle merkittyä tapahtumakohtaista tietoa. Lokitiedoista selviää yleensä mitä tietojärjestelmissä on tapahtunut ja milloin, sekä mitkä asiat ovat kyseiseen tapahtumaan johtaneet. Lokitietoja on tyypillisesti käytetty järjestelmien testaukseen, virheiden paikantamiseen, sekä lisäinformaation tuottamiseen järjestelmien toiminnasta. Lokitietoihin kerättävä tieto on kuitenkin sittemmin laajentunut esimerkiksi käyttäjien kirjautumisiin ja toimiin, metriikkaan, sekä forensiikkaan.

Lokienhallintajärjestelmä toteutettiin Elastic Stack -ryhmän työkaluilla. Elastic Stack tarjoaa työkalut keskitettyyn lokienhallintaan, joilla voidaan suorittaa erilaisia datan hallintaan käytettäviä toimenpiteitä. Se mahdollistaa datan vastaanottamisen eri formaateissa, datan hakujen tekemisen, datan analysoinnin sekä visualisoinnin. Elastic Stack tarjoaa kaikki lokienhallintaan tarvittavat askeleet lokien vastaanottamisen lokilähteeltä elinkaarensa päähän.

Lopputuloksena saatiin toteutettua toimiva lokienhallintajärjestelmä, jolla voidaan kerätä testiympäristön verkkolaitteiden lokitiedot, suorittaa niille tarvittava prosessointi ja tallentaa sen jälkeen monitorointia ja myöhempää käsittelyä varten.

Avainsanat (asiasanat)

Syslog, Lokiviesti, LOG, Lokienhallinta, Elasticsearch, Logstash, Kibana

Muut tiedot (salassa pidettävät liitteet)

jamk

Description

Pesonen Jere

Network devices log management implementation

Jyväskylä: JAMK University of Applied Sciences, December 2022, 61 pages

Bachelor's Degree Programme in Information and Communications Technology. Bachelor´s thesis.

Permission for open access publication: Yes

Language of publication: Finnish

Abstract

Log management of network devices is essential act for comprehensive network monitoring, and it is used for viewing, archiving, and analyzing log messages generated by whole network. Log management system provides visibility to monitor networks state and possible occurring errors.

Objective was to plan and setup working log management system for the Finnish Defense Forces C5 Agency's test network, which is produced for education and development actions only. Log management system enables a possibility for comprehensive log collecting, archiving, and analyzing platform for multiple log sources. System also presents real time monitoring of target network and vision for possible deviations that occurred or are about to occur in systems. Also, the intention was to enhance knowledge of log management practices, and log life cycle events.

Log as a term is used in information technology to represent different systems and applications providing event data their status and conditions. Log data informs usually about what has occurred in the system, when it happened, and which events led to current situation. Typically log messages were used for testing, locating errors, and generating additional information from systems behavior. Since then, log messages information has been expanded to user behavior, metrics, and forensics.

Log management system was implemented using platform named Elastic Stack. It provides tools for centralized log monitoring, which can be used to perform different kind of acts for processing data. It enables all data processing activities, like receiving data from different sources, performing searches, data-analysis, and visualization. Elastic Stack provides all log management steps from receiving logs to the end of their life cycles.

Result provided complete log management system, which can be used to collect all needful data from whole network, apply needed filters, and save it for monitoring and later handling.

Keywords/tags (subjects)

Syslog, Log messages, LOG, Log management, Logstash, Elasticsearch, Kibana

Miscellaneous (Confidential information)

Sisältö

Lyhenteet ... **3**
1 Lähtökohdat ... **4**
1.1 Lokienhallinta ... 4
1.2 Toimeksiantaja ... 5
1.3 Tavoitteet ... 5
1.4 Tutkimusmenetelmä ... 6
2 Lokitieto ... **7**
2.1 Lokin määritelmä ... 7
2.2 Lokiviestit ... 8
2.3 Lokien tietoturva ... 8
2.4 Lokiputki ... 9
2.5 Lokien keräys ... 10
2.6 Syslog-protokolla ... 11
2.7 Lokien normalisointi ... 13
2.8 Lokien analysointi ... 13
2.9 Lokien säilytys ... 14
2.10 Lokien visualisointi ... 14
2.11 Lokienhallintajärjestelmän hyödyntäminen verkkolaitteiden valvonnassa ... 15
3 Työkalut ... **16**
3.1 Elastic Stack ... 16
3.2 Beats ... 16
3.3 Logstash ... 17
3.4 Elasticsearch ... 20
3.5 Kibana ... 20
4 Toteutus ... **21**
4.1 Toteutusympäristö ... 21
4.2 Lokienhallinnan topologia ... 21
4.3 Lokien kerääminen verkkolaitteilta ... 23
4.4 Palvelimet ... 25
4.5 Elasticsearchin konfigurointi ... 26
4.6 Kibanan konfigurointi ... 28
4.7 Logstashin konfigurointi ... 29
4.7.1 Input ... 30

4.7.2 Filter ... 30
4.7.3 Output ... 33
4.7.4 Lumberjack ... 33
4.8 Lokiviestien reitti verkkolaitteelta Elasticsearchille ... 34
4.9 Visualisointi ... 36
4.9.1 Discover ... 36
4.9.2 Dashboard ... 37
4.9.3 Alerts ... 39
5 Tulokset ... 43
6 Pohdinta ... 44
Lähteet ... 46
Liitteet ... 49
Liite 1. Cisco reitittimen konfiguraatiot ... 49
Liite 2. Ubuntu 22.0.04 LTS asennus ... 50
Liite 3. Elasticsearchin asennus ... 51
Liite 4. Kibanan asennus ... 53
Liite 5. Logstashin asennus ... 55
Liite 6. Logstash masterin pipeline ... 56
Liite 7. Logstash collectorin pipeline ... 60

Kuviot

Kuvio 1. Lokiputki ... 10
Kuvio 2. Esimerkki grok-suodattimesta ... 19
Kuvio 3. Lokienhallintajärjestelmän verkkokuva ... 22
Kuvio 4. Elasticsearchille paikallisesti tehty curl-kysely ... 28
Kuvio 5. Kibana enrollment token -lomake ... 29
Kuvio 6. Vuokaavio lokiviestien reitistä ... 35
Kuvio 7. SSH-kirjautumiset esitettynä discovery-työkalulla. ... 37
Kuvio 8. Esimerkki visualisoinnin luomisesta ... 38
Kuvio 9. Valmis dashboard-näkymä ... 39
Kuvio 10. Port flapping -säännön määritykset ... 40
Kuvio 11. Brute force -säännön määritykset ... 42
Kuvio 12. Alerts-näkymä ... 43

Taulukot

Taulukko 1. Syslog-viestien vakavuusasteet 12
Taulukko 2. Beats-agentit ja niiden käyttötarkoitus 17
Taulukko 3. Lokirivin käsittely grok-suodattimella. 32

Lyhenteet

API	Application Programming Interface
APT	Advanced Package Tool
CA	Certificate Authority
CURL	Client Uniform Resource Locator
DNS	Domain Name System
DSL	Domain Specific Language
IP	Internet Protocol
JSON	JavaScript Object Notation
KQL	Kibana Query Language
NTP	Network Time Protocol
REGEX	Regular Expression
RFC	Request For Comment
SIEM	Security Incident and Event Management
SNMP	Simple Network Management Protocol
SSH	Secure Shell
SYSLOG	System Logging Protocol
TCP	Transmission Control Protocol
TLS	Transports Layer Security
UDP	User Datagram Protocol
VRF	Virtual Routing and Forwarding
YAML	Yet Another Markup Language

1 Lähtökohdat

1.1 Lokienhallinta

Ajankohtaisen lokitiedon hyödyntäminen on tärkeää tietoverkkojen tehokkaan valvonnan ja ylläpidon kannalta. Verkkojen ja palveluiden jatkuvat muutokset vaativat pikaisia ylläpitotoimenpiteitä verkon ja sen palveluiden saatavuuden näkökulmasta. Verkkolaitteiden lokitiedon tehokas valvonta vaatii keskitetyn lokienhallintapalvelimen, joka mahdollistaa verkon kaikkien laitteiden lokien tarkastelemisen ja analysoinnin yhdessä paikassa.

Lokienhallintajärjestelmä mahdollistaa sen, että kaikki verkko-operointiin liittyvä hyödyllinen data voidaan kerätä kohde verkon kaikilta eri laitteilta, ja tuoda tarkasteluun ja analysoitavaksi keskitettyyn paikkaan. Tämä helpottaa ja mahdollistaa käyttäjälle verkon vikatilojen ymmärtämisen ja ennakoinnin. Lokienhallintapalvelun ja lokien keräyksen automatisoinnin mahdollistaminen vaatii palvelun, jolla kerätään, arkistoidaan ja visualisoidaan verkon tapahtumia. Sen lisäksi, että lokitapahtumien visualisointi tarjoaa kattavan näkemyksen kaikkien valvottavien laitteiden nykytilasta, se tarjoaa kriittistä dataa myös aikaisemmista tapahtumista, jotka ovat jääneet huomiotta ja ovat mahdollisesti voineet vaikuttaa verkkolaitteiden toimintaan. (Ljubojević, Bajić & Mijić 2018.)

Lokitieto tarjoaa arvokasta tietoa tietoteknisten järjestelmien, verkkolaitteiden, sekä tietoverkon tilasta. Lokitiedon käyttö verkon ylläpidon kannalta on tärkeää, ja se mahdollistaa verkkolaitteiden tapahtumien tarkastelun ja poikkeamien juurisyyn selvittämisen verkon toimivuuden takaamiseksi. Verkkolaitteiden tapahtumat, vikatiedot, häiriötilat, tietoturvapoikkeamat, sekä muut virheilmoitukset tallentuvat laitteen lokitietoihin, jolloin tärkeää informaatiota pystyy tarkastelemaan jälkikäteen verkkolaitteen käyttöliittymästä. Lokitieto itsessään on ainoastaan tekstiä, jonka aktiiviseen valvontaan tarvitaan kyseiseen ympäristöön suunniteltu ja implementoitu lokienhallintajärjestelmän, joka automatisoi lokitiedon analysoinnin ja nostaa hälytyksiä ennaltaehkäisevänä toimenpiteenä verkon ylläpitäjän nähtäväksi. Ilman edellä mainittua järjestelmää lokien tarjoamat mahdollisuudet valvonnan kannalta jäävät täysin hyödyntämättä, kun verkon virheet ovat ehtineet jo vaikuttaneet sen palveluihin, eikä tulevaa palvelupoikkeamaa ole osattu ennakoida lainkaan. Lokienhallintajärjestelmälle voidaan määritellä käyttötapauksia, jolloin tiettyjen tapahtumien esiintymät nostavat hälytyksen, ja antavat ylläpitäjälle mahdollisuuden reagoida verkon poikkeamiin ennaltaehkäisevästi.

Toimiva verkon lokinhallintajärjestelmä koostuu useista komponenteista, jotka luovat lokeille elinkaaren valvottavalta laitteelta julkaisujärjestelmään, säilytykseen ja päättyen lopulta lokin arkistointiin tai hävittämiseen. Julkaisujärjestelmä on käyttöliittymä, jossa verkon lokitietoja pääsee tarkastelemaan keskitetysti yhdessä paikassa. Usein se tarjoaa myös mahdollisuuden luoda hälytyssääntöjä, joihin automaatio tarttuu ennalta määrättyjen konfiguraatioiden perusteella ja nostaa tapaukset esille verkon ylläpitäjälle. Lokienhallintajärjestelmään sisältyy lokien keräys- ja käsittelypalvelu, indeksointi- ja analysointipalvelu, sekä julkaisujärjestelmä. (Zhang ym. 2017.)

1.2 Toimeksiantaja

Työn toimeksiantaja on Puolustusvoimien johtamisjärjestelmäkeskus (PVJJK). Puolustusvoimien johtamisjärjestelmäkeskus on pääesikunnan alainen laitos, joka perustettiin vuoden 2007 alussa kun Puolustusvoimien tietotekniikkalaitos, maanpuolustusalueiden johtamisjärjestelmäosastot sekä maanpuolustusalueiden tietotekniikkakeskukset yhdistyivät. PVJJK toimii tällä hetkellä ympäri Suomea 17 eri paikkakunnalla ja se työllistää yhteensä noin. 450 työntekijää, joista merkittävä osa on siviilityöntekijöitä. PVJJK:n organisaatio koostuu eri osastoista, joihin kuuluu esikunta, kyberosasto, palveluosasto, neljä kappaletta verkko-osastoja sekä johtamisjärjestelmäkoulu. (Puolustusvoimien johtamisjärjestelmäkeskus 2022.)

PVJJK:n tehtäviin kuuluu Puolustusvoimien yhteisten tietoteknisten palveluiden järjestäminen sekä kyberpuolustuksen hoitaminen. PVJJK:n yksi tärkeä tehtävä on valvoa Puolustusvoimien tietoliikennettä ja tietoverkkoja. PVJJK:n yhteistyökumppaneita ovat Erillisverkot Oy sekä Valtion tieto- ja viestintätekniikkakeskus Valtori. (Puolustusvoimien johtamisjärjestelmäkeskus n.d.)

1.3 Tavoitteet

Työn tavoitteena on tuottaa toimiva lokienhallintajärjestelmä, joka mahdollistaa verkon laitteiden lokien keräyksen, arkistoinnin ja analysoinnin keskitetysti yhdellä palvelulla. Lokien tarkastelu pyritään tekemään mahdollisimman yksinkertaisesti, jotta verkon tapahtumia on helppo seurata lokienhallintajärjestelmän avulla. Lisäksi tavoitteena on luoda muutama automaattinen hälytyssääntö, jotka hälyttävät ennalta määritellyistä kiinnostavista tapahtumista, ja vaativat mahdollisesti käyttäjän toimenpiteitä.

Tämän työn tarkoituksena on kartoittaa uusia mahdollisuuksia, joita voidaan hyödyntää ja ottaa käyttöön verkon valvonnan tukena. Työn implementointi vaihe suoritetaan Puolustusvoimien Johtamisjärjestelmäkeskuksen testiympäristössä, eikä tarkoituksena ole käsitellä lainkaan tuotantoverkkoja, joka mahdollistaa sen, että työ tehdään täysin julkisena. Testiympäristö koostuu eri verkkolaitteista ja palveluista. Tällä yritetään simuloida aitoa verkkoympäristöä. Työn tuloksena saadusta lokienhallintajärjestelmästä opittuja käytäntöjä ja tekniikoita tutkitaan mahdollisuutena käyttöönottaa tulevaisuudessa myös tuotantoympäristössä.

Vastaavanlaista verkkolaitteiden lokienhallintaan käytettävää järjestelmää ei kohdeympäristöstä löydy vielä lainkaan. Verkkolaitteiden lokitieto kerätään ja sitä on mahdollista tarkastella, mutta sen suodattamiseen, indeksointiin ja analysointiin ei ole vielä implementoitu minkäänlaista ratkaisua. Tämän kehittämistyön tarkoituksena on suunnitella ja luoda pohja, joka mahdollistaa lokien hallinnan, analysoinnin, sekä hälytyssääntöjen luonnin. Järjestelmän tarkoitus on saada reaaliaikainen kuva verkon tilasta ja sen häiriöistä. Lopputuloksena saadaan valmis lokiputki verkkolaitteilta lokienkeräyspalvelimen kautta keskitettyyn tietokantaan ja julkaisujärjestelmään.

1.4 Tutkimusmenetelmä

Opinnäytetyö toteutetaan tutkimuksellisena kehittämistyönä. Tutkimuksellinen kehittämistyö tarkoittaa yhdistelmää teoriasta sekä käytännön kehittämistyöstä, jolla voidaan uudistaa työelämän käytäntöjä. Menetelmä on yhdistelmä sekä teoriaa, että konkretiaa ja tutkimus etenee tutkimuksellisesta osuudesta loogisesti kohti konkreettista kehittämistyötä. Lähtökohtana tutkimuksellisessa kehittämistoiminnassa on työelämästä löytyneet tutkimuskysymykset ja mahdolliset ongelmat. Pääpaino on käytännön työvaiheessa, jonka pohjana ovat teoriaosuudessa läpikäydyt tutkimusasetelmat ja periaatteet. (Tutkimuksellinen kehittämistyö n.d.)

2 Lokitieto

2.1 Lokin määritelmä

Lähes kaikki tietotekniikkaan liittyvät järjestelmät tuottavat tapahtumalokia, joka toimii tallenteena ja listaa kaikki tapahtumat ja muutokset, joita järjestelmässä tapahtuu. Lokitiedon määritelmänä pidetään aikaleiman ja jonkun tiedon yhdistelmää, eli se vastaa kysymyksiin, mitä on tapahtunut ja milloin. (Lidster 2022.)

Lokien käsite on määritelty valtiovarainministeriön lokiohjeessa seuraavasti:

> *Loki dokumentoi tapahtumia, jotka ovat tapahtuneet organisaation järjestelmissä, verkoissa tai muussa ympäristössä ja toiminnassa. Lokitiedot voivat olla automaattisten järjestelmien keräämiä merkintöjä tai manuaalisesti kerättäviä lokitietoja, kuten vierailijaloki. (Lokiohje 2009.)*

Järjestelmät muodostavat lokitietoa tapahtumina aina tietyssä formaatissa. Yksi tapahtuma sisältää aina ympäristössä tapahtuneen yksittäisen tapauksen. Tapahtuma sisältää dataa tietyssä jossain formaatissa, joka sisältää yleensä mm. ajankohdan ja muuta informaatiota jostakin järjestelmässä tapahtuneesta muutoksesta (Peng, Tao & Leng 2005). Lokitapahtumalle olennaista on, että se sisältää kriittiset tiedot, jotta lokista on sen tarkastelijalle hyötyä. Näitä ovat muun muassa päivämäärä, kellonaika, tapahtuman lähde, tapahtuman vakavuusaste, sekä tapahtuman tiedot. Näillä edellä mainituilla kentillä saadaan jo hyvät perustiedot lokitiedon tarjoamasta tapahtumasta.

Chuvakin, Schmidt & Phillipsin (2013) mukaan lokitietojen reaaliaikainen keräys ja seuranta on verkkoympäristössä erittäin tärkeää, mutta se jätetään liian niin usein täysin huomiotta. Lokit tarjoavat arvokasta tietoa laitteen häiriöistä jälkikäteen, mutta onko lokien tarkastelu tarpeellista, mikäli järjestelmät toimivat moitteettomasti? Vastaus kysymykseen kuuluu: kyllä on. Verkkolaitteen tuottama loki on arvokas informaationlähde sekä erilaisten tapahtumien jälkitarkasteluun, että laitteen nykytilan monitorointiin ja mahdollisten häiriöiden ennaltaehkäisemiseen (Zhang ym. 2017). Ideaalitilanteessa lokit tarjoavat myös muita vastauksia, kuin sen, että mikä on jo mennyt pieleen, kuten esimerkiksi: ”Kuinka hyvin meillä menee?”, ”Onko tulevaisuudessa tulossa ongelmia?” ja ”Onko tapahtunut poikkeamia, jotka saattavat vaikuttaa verkon toimintaan?”. Näiden lisäksi lokit tarjoavat tietoa myös käyttäjän toiminnasta, ja näin ollen vastaavat kysymykseen: ”kuka

teki ja mitä?”. Näiden kysymysten vastaukset ovat verkon toiminnan kannalta kriittisiä, joten lokien yhtenäisen keräyksen ja niiden analysoinnin tulisi olla osa yrityksen jatkuvaa toimintaa, jotta verkon toiminnasta pystytään ylläpitämään ajankohtaista tilannetietoa ja tuottamaan ennaltaehkäisevät toimet ennen kuin tapahtumat vaikuttavat palveluiden saatavuuteen. Tästä syystä lokienvalvontatyökalu tulisi olla verkon valvonnan tukena jatkuvassa käytössä, eikä ainoastaan silloin, kun jokin jo mennytpieleen. (Chuvakin, Schmidt & Phillips 2013.)

2.2 Lokiviestit

Suuri osa lokiviesteistä jättää noudattamatta mitään spesifisoitua tai ennalta määriteltyä lokiformaattia. Eri lokiprotokollat ja -tekniikat käyttävät omanlaisiaan lokiformaatteja, joten lokien käsittelyvaiheessa ne tulevat vaatimaan yksilöllisiä käsittely- ja suodatusmäärityksiä. Lokiformaatit vaihtelevat yleensä eri järjestelmien välillä, mutta myös saman pohjakäyttöjärjestelmän ja lokien keräys tekniikan sisällä saattaa olla eroavaisuuksia, sillä tekniikoita käsittelevät standardit saattavat olla vanhentuneita, mikä johtaa käytettävän formaatin epäjohdonmukaisuuteen. Ainoat lokitietoja yhdistävät tekijät ovat yleensä ainoastaan aikaleima, lokin lähdejärjestelmän IP-osoite tai laitenimi, sekä vapaamuotoinen viesti, joka sisältää informaation lokiviestingeneroineen tapahtuman sisällöstä. Lokiviestin aikaleima kertoo tapahtuman ajankohdan ja lokilähde kertoo, että mistä järjestelmästä lokiviesti on peräisin. (Chuvakin, Schmidt & Phillips 2013.)

Järjestelmien tuottama loki voi olla joko teksti- tai binääriformaatissa, jonka kääntämiseen tarvitaan työkalu, joka muuntaa viestin ihmisen ymmärtämään tekstimuotoon. Binäärimuodon hyötyinä voitaisiin mainita, että ne ovat pienemmässä koossa ja näin ollen vaativat vähemmän prosessointitehoa käsittelyyn ja tallentamiseen. (Chuvakin, Schmidt & Phillips 2013.)

2.3 Lokien tietoturva

Tärkeä osa lokienhallintajärjestelmää on myös lokienkeräyksen tietoturvan varmistaminen. Lokit saattavat joko tarkoituksellisesti tai sattumalta kerätä arkaluonteista dataan kuten esimerkiksi käyttäjien salasanoja, joka aiheuttaa tietoturvaongelmia. Esimerkiksi Cisco-verkkolaitteissa salasanat kirjoitetaan selkokielisenä tekstinä lokeihin, jos tätä ei erikseen ole muutettu konfiguraatiolla. (Kent & Souppaya 2006.)

Myös lokien kuljettaminen vaatii tietoturvan kannalta huomiota. Esimerkiksi Syslog-protokolla ei sisällä minkäänlaista salausta, vaan loki kuljetetaan täysin selväkielisinä, jolloin kuka tahansa oikealla työkalulla pystyy sitä tarkastelemaan. Varsinkin runkoverkon yli siirrettäessä dataa pelkän Syslogin käyttöä tulisi välttää. Esimerkkiratkaisu tähän olisi se, että lokit kerättäisiin yhtenäiselle palvelimelle lähiverkon sisällä, joka suorittaisi salauksen ja lähettäisi lokit eteenpäin keskitettyyn paikkaan.

Lokienhallinnan osalta tulee varmistaa niiden luotettavuus, eheys ja saatavuus. Luotettavuudella tarkoitetaan, että data on suojattua eikä siihen pääse käsiksi sellaiset henkilöt, joilla ei ole vaadittuja oikeuksia tai tarvetta käsitellä kyseistä dataa. Eheydellä vahvistetaan, että loki säilyy muuttumattomana ja paikkansapitävänä koko elinkaarensa ajan. Saatavuus huolehtii siitä, että lokit ovat järjestelmän ylläpitäjän saatavilla tarpeen tullen helposti ja vaivattomasti. (Kent & Souppaya 2006.)

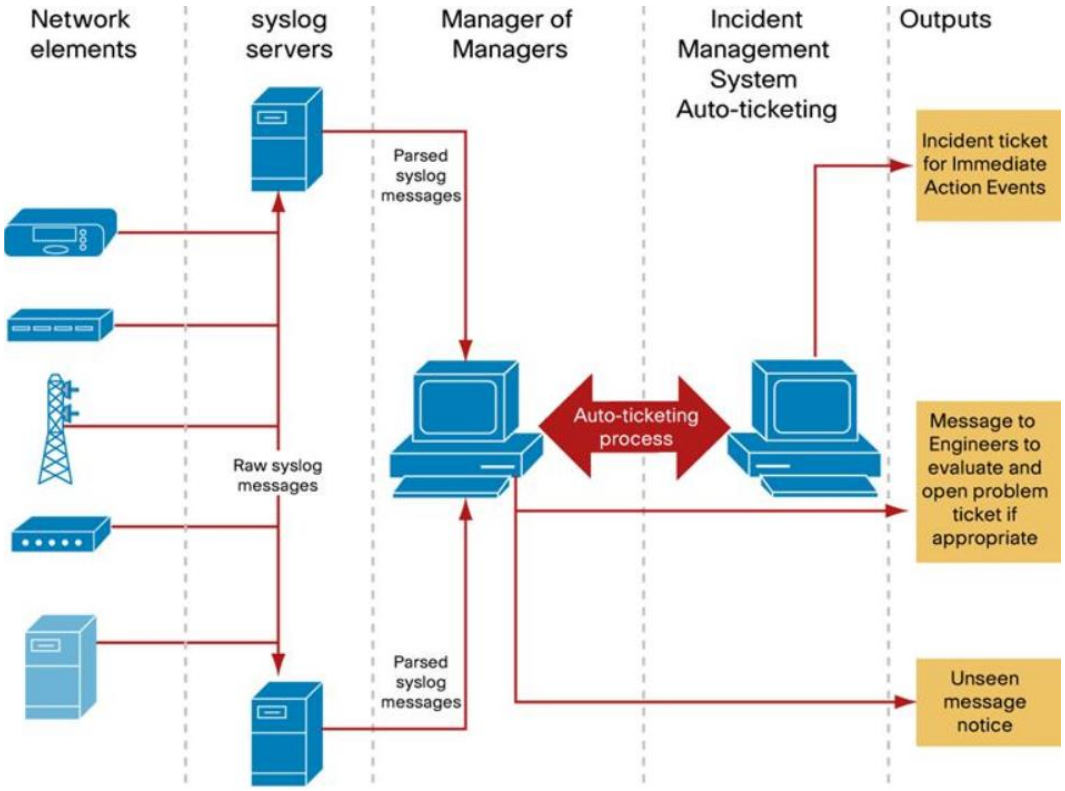
Kentin ja Souppayan (2006) listaamia lokien tietoturvakäytäntöjä:

- Rajaa pääsy lokien hallintaan ja tarkasteluun vain valtuutetulle käyttäjille
- Varmista, että lokitieto pysyy muuttumattomana koko elinkaarensa ajan
- Vältä keräämästä lokiin tarpeetonta sensitiivistä dataa, joka aiheuttaa ylimääräisiä tietosuoja käytäntöjä
- Järjestä lokien arkistointi niin, että kriittiset lokit pysyvät turvallisesti säilössä retention mukaisen ajan
- Turvaa lokienhallintaprosessi, jotta lokien keräys, käsittely ja säilytys pysyy muuttumattomana ja lokienhallintajärjestelmä eheänä
- Suojaa lokien siirto lokilähteeltä lokienhallintajärjestelmään, varsinkin jos siirto tapahtuu lähiverkon ulkopuolella.

2.4 Lokiputki

Lokiputkella tarkoitetaan yksittäisen lokitiedon elinkaareen liittyviä toimenpiteitä ja tapahtumaketjua, jonka läpi jokainen lokiviesti kulkeutuu lokienhallintajärjestelmässä. Lokitiedon elinkaari kattaa koko lokiputken tapahtumat aina lokilähteeltä lokienhallintajärjestelmään. Tämä prosessi sisältää lokien lähettämisen, keräyksen, normalisoinnin, analysoinnin, säilömisen, arkistoinnin, sekä mahdollisen hävittämisen. (Kent & Souppaya 2006.)

Kuvio 1. on esitetty havainnollistava kuva lokiputkesta lokilähteeltä lokienhallintajärjestelmään.



Kuvio 1. Lokiputki (Lähde: https://www.cisco.com/c/dam/en/us/products/collateral/services/high-availability/white_paper_c11-557812.doc/_jcr_content/renditions/white_paper_c11-557812_7.jpg)

2.5 Lokien keräys

Lokien keräyksellä tarkoitetaan lokitiedon varastointia verkkolaitteelta keskitetylle lokienkeräys-palvelimelle. Lokien keräystapoja on useita, mutta monet järjestelmät tukevat natiivisti jotain tiet-tyä lokienkeräys metodia. Esimerkiksi Linux-järjestelmissä yleisimmin käytetty on Syslog (Todd 2017). Tämä tapahtuu yksinkertaisesti määrittelemällä laitteelle palvelimen osoite tai laitenimi, jonne lokit lähetetään käyttämällä UDP- tai TCP-tiedonsiirtoprotokollaa. UDP-protokolla on nope-ampi ja kevyempi vaihtoehto, kun taas TCP on luotettavampi ja se takaa, että tieto pysyy siirron aikana eheänä. (Todd 2017.)

Lokien keräys tavat voidaan jakaa kahteen eri kategoriaan, joko push- tai pull-metodi. push-meto-dilla lokilähteet itse lähettävät generoimansa lokiviestit verkon yli määritellylle lokienkeräyspalveli-melle, kun taas pull-pohjaisella metodilla lokienhallintatyökalu hakee halutun data suorittamalla erillisen kyselyn verkkolaitteelle. Kolme yleisimmin käytettyä push-pohjaista lokien keräys tapaa

ovat Syslog, SNMP, ja Windows Event log, joista SNMP tukee myös pull-pohjaista tapaa. (Chuvakin, Schmidt & Phillips 2013.)

Lähestulkoon kaikkia verkkoympäristöön liittyviä komponentteja voidaan käyttää lokilähteenä. Toddin (2017) mukaan lokilähteenä pidetään yleensä jotain järjestelmää, laitetta tai sovellusta, joka tuottaa tapahtumalokia (Todd 2017). Toisaalta Chuvakin, Schmidt & Phillips (2017) käsittelevät kirjassaan lokilähteinä kuitenkin lokien generointiin ja keräämiseen käytettäviä protokollia (Syslog, SNMP ja Windows Event Log). Näin ollen termiä lokilähde on siis käytetty useammassa eri yhteydessä, mutta tässä työssä lokilähteistä puhuttaessa käsitellään nimenomaan lokien keräämiseen käytettäviä protokollia. Tässä työssä käsitellään ainoastaan reitittimien ja kytkimien Syslogilla tuottamaa laitteistolokia, joka tarjoaa dataa laitteen tilan monitorointiin ja vianselvitykseen, sekä käyttäjien toimenpiteiden seuraamiseen. Työssä ei siis käsitellä SNMP:tä tai Event logia lainkaan.

2.6 Syslog-protokolla

Linux-pohjaisten järjestelmien käyttämä ja tietoliikenne protokollaa hyödyntävä yleisimmin lokitukseen käytetty tekniikka. Syslog koostuu Syslog-palveluprosessista (syslogd, rsyslog tai syslog-ng), joka vastaanottaa lokiviestit sovelluksilta tai käyttöjärjestelmiltä ja konfiguraatiosta riippuen tallentaa ne tiedostoon tai lähettää ne käsiteltäväksi lokienhallintapalvelimelle. (Chuvakin, Schmidt & Phillips 2013.)

Syslog-protokolla on alun perin kehitetty Kalifornian yliopistossa, jotta heidän itse kehittämäänsä UNIX-jakeluun saataisiin prosessi, joka kerää laitteen tapahtumat ja tallentaa ne lokaaliin tiedostojärjestelmään lokitiedostoiksi. Verkkolaitteen lokitiedostot ovat selkokielisiä tekstitiedostoja, joista ylläpitäjä voi tarkastella järjestelmän lokittamia tapahtumia laitteen käyttöliittymästä. Nykyään on hyvin yleistä, että kytkimet, reitittimet, palomuurit ja muut verkkolaitteet lokittavat dataa käyttämällä Syslog-protokollaa. (Huang, Zhang & Jiang 2012.)

Lokiviesti koostuu kolmesta osasta:

- PRI (protokollan laskema lokin prioriteetti arvo)
- HEADER (sisältää lähdelaitteen identifiointi tiedot)
- MSG (sisältää itse viestin tapahtumasta).

Syslog viestiin sisältyy myös erillinen severity-arvo, joka sisältää protokollan määrittelemän kyseisen lokiviestin vakavuusarvon. Lokien keräystä voidaan suodattaa jo lähdelaitteella severity-arvon perusteella, esimerkiksi ainoastaan vakavuusluokan 3 ja sitä vakavammat viestit lähetetään käsiteltäväksi lokipalvelimelle. Mitä pienempi numero, sitä vakavampi tapahtuma on kyseessä, kuten .

Taulukko 1, on esitetty (Syslog n.d. muokattu).

Taulukko 1. Syslog-viestien vakavuusasteet (Syslog n.d.)

Arvo	**Vakavuus**	**Kuvaus**
0	Emergency	Järjestelmä on epävakaa
1	Alert	Vaatii välittömiä toimenpiteitä
2	Critical	Kriittinen tila
3	Error	Virhe
4	Warning	Varoitus
5	Notice	Merkittävä tapaus
6	Informational	Tiedottava viesti
7	Debug	Debuggaus viesti

Syslog ei itsessään sisällä mitään tietoturvaa tukevia mekanismeja. Se ei sisällä minkäänlaista autentikointi tapaa, jolla voitaisiin varmistua siitä, että lokipalvelimen vastaanottamat lokit ovat juuri sen laitteen lokeja, jonka se väittää olevan. Lisäksi Syslogista ei löydy minkäänlaista salausta, vaan viestit lähetetään palvelimelle selkokielisenä tekstinä. (Syslog n.d.)

Protokolla on ollut laajasti käytetty protokolla, jolla on kerätty lokiviestejä jo internetin alkuajoista lähtien. Laajasta käytöstä huolimatta Syslog yhteen toimivuuden kanssa on havaittu haasteita esimerkiksi epäjohdonmukaisten lokiviestien käytössä. Syslog-formaatteja on useampia, esim. RFC3164 ja RFC5424, joista viimeisin julkaistu standardi on vuodelta 2009 (RFC 5427), jota ei ole sittemmin päivitetty. (Tsunoda & Keeni 2014)

2.7 Lokien normalisointi

Lokien noudattaessa jotain tiettyä rakennetta, helpottaa se lokitiedon käsittelyä sekä prosessointia, ja antaa lokeille merkityksen (Wilkins 2022.). Kun lokitiedot on kerätty yhtenäiselle palvelimelle, on vuorossa lokien normalisointi. Normalisointi on datan hallintatekniikka, joka tarkoittaa lokiviestin tietojen erottelemista eri kenttiin, jonka avulla pyritään yhtenäistämään lokiviestien tietyt kentät (esim. aikaleima ja lähde laitteen IP-osoite) samanlaiseen formaattiin. Tämä mahdollistaa lokien indeksoinnin ja säilömisen tietokantaan. Lokitieto ennen normalisointia on ainoastaan merkkijono, jonka tarkastelu ja analysointi on erittäin haastavaa.

2.8 Lokien analysointi

Lokien analysointi tarkoittaa prosessia, jossa järjestelmien tuottamaa lokia tarkastellaan joko verkon poikkeaminen ennaltaehkäisevästä näkökulmasta, tai jo menneiden tapahtumien syitä selvittäen. Lokien analysointia voidaan tehdä usealla eri tekniikalla, joko ihmisvoimin käyttäen lokienhallintajärjestelmän visualisointipalvelua, tai vaihtoehtoisesti automatiikkaa ja koneoppimista hyödyntäen tunnistamisella lokiviestien tuottamia erilaisia kaavoja. Automatiikkaa hyödyntäen voidaan tunnistaa lokirivien tuottamien tapahtumaketjuissa useamman normaalin lokirivin tuottama kaava, joka saattaa indikoida jostain järjestelmässä tapahtuvasta poikkeamasta. Kaavojen tunnistamiseen käytetään työkaluja, jotka tunnistavat ennalta määritellyt tapahtumaketjut lokimassassa, ja nostavat havainnoista hälytykset.

Lokien tehokkaan analysoinnin mahdollistaa lokienhallintajärjestelmän visualisointipalvelu, jolla voidaan tarkastella ja suorittaa hakuja kaikista verkosta kerätyistä lokeista kootusti yhdessä paikassa. Tehokas lokien analysointi edellyttää sitä, että lokien keräys, suodattaminen, normalisointi ja indeksointi toimii moitteettomasti. (What is Log Analysis? 2022.)

2.9 Lokien säilytys

Lokit tarjoavat monenlaista tietoa, ne ovat väliaikaista dataa menneistä tapahtuvista ja ne voidaan luokitella myös tärkeyden perusteella. Lokien keräyksen ja hallinnan seurauksena tärkeänä osana tulee myös lokien säilytysaika, jolla määritellään elinajan pituus säilytystarpeellisuuden mukaan. Vähemmän kriittiset lokit voidaan määritellä poistettavaksi aikaisemmin, jolloin ne eivät vie turhaan lokienhallintajärjestelmän resursseja. Lokitiedon säilytysaika voidaan määritellä minkä tahansa tiedon perusteella, joka lokirivistä saadaan irti, esimerkiksi eri lähdejärjestelmän lokit voivat olla enemmän tärkeitä kuin toisen, ja näin ollen niiden lokien säilytysaika määritellään pidemmäksi. Lokiensäilytyspolitiikkaan voi omilla perusteilla luoda useita eri kategorioita, joiden perusteella lokien elinaika määritellään. (What is Log Retention? n.d.)

2.10 Lokien visualisointi

Visualisoinnilla tarkoitetaan jonkin datan muuntamista graafiseen muotoon, kuten esimerkiksi kaavioiksi, kuvaajiksi, kartoiksi ja taulukoiksi. Lokiviestien kannalta niiden esittäminen näköaistille havainnollisempaan muotoon auttaa ihmistä hahmottamaan selkeästi datavirrassa tapahtuvat muutokset ja poikkeavat tapahtumat, jotka halutaan nostaa selkeästi esille verkon ylläpitäjälle. Visualisoinnilla summataan datan tarjoama tieto tavalla, joka mahdollistaa tarkastelijan keskittymisen kriittisempiin muutoksiin lokivirrassa. Visualisointi helpottaa ihmistä ymmärtämään erilaisia trendejä, kuvioita ja malleja, joita datavirta pitää sisällään.

Visualisointiin käytettävästä työkalusta käytetään yleisesti nimitystä SIEM, joka on palvelu, joka interaktiivisesti monitoroi kohdeympäristön laitteita lokien perusteella, ja auttaa havainnoimaan poikkeamia seuraamalla tapahtumien kulkua (Elastic Security for SIEM & security analytics n.d.).

2.11 Lokienhallintajärjestelmän hyödyntäminen verkkolaitteiden valvonnassa

Ilman lokienhallintajärjestelmää IT-ympäristössä tapahtuvat muutokset jäisivät analysoimatta asiaan kuuluvalla tavalla. Verkossa tapahtuvat poikkeamat tulisi kyllä huomattua muuta kautta, mutta niiden juurisyiden selvittäminen olisi erittäin haastavaa, ja veisi todella paljon ylimääräistä aikaa. Lokienhallintajärjestelmä tarjoaa ennakoivan ratkaisun verkon ylläpitäjälle, joka auttaa pysymään ajan tasalla verkossa tapahtuvista poikkeamista ja potentiaalisista ongelmista. Lokienhallintajärjestelmä tarjoaa reaaliaikaisen näkyvyyden verkkoympäristön käyttäytymiseen normaalitilassa ja hälytykset, jotka hälyttävät verkon toiminnan muutoksista määriteltyjen ehtojen mukaan. Se on kriittinen osa verkon ylläpitoa, joka mahdollistaa verkon ylläpitäjälle lokitietojen reaaliaikaisen haun ja käsittelyn keskitetyssä ympäristössä, hälytyssääntöjen tekemisen sekä havainnointien raportoinnin mahdollistavan työkalun. Se helpottaa myös häiriöiden juurisyyn selvittämisessä, kun kaikki lokiviestit ja mahdollisuus lokitietojen hakujen tekemiseen on yhdistetty keskitettyyn paikkaan. Lokitiedot ovat ensimmäinen lähde verkon epäsäännöllisyyksien havainnoimiseen. (Why is log management important? 2022.)

Lokienhallinnan hyötyjä (Why is log management important? 2022.):

- Datan keräys ja yhtenäistäminen keskitettyyn sijaintiin
- Lokien säilöminen kunnollisessa ja yhtenäisessä formaatissa sekä varastointi tarpeelliseksi määritellyksi ajaksi
- Omien hälytyssääntöjen luominen ja hälytysten generoiminen omien määritysten perusteella
- Mahdollistaa vikojen, häiriöiden sekä järjestelmän muutoksen analysoinnin lokiviestien perusteella.

3 Työkalut

3.1 Elastic Stack

Elastic Stack (aikaisemmin ELK Stack) on avoimen lähdekoodin projekti, joka yhdistää usealla palvelulla datan keräyksen, prosessoinnin, analytiikan ja visualisoinnin. Se kykenee vastaanottamaan dataa eri lähteistä erilaisissa formaateissa ja käsittelemään datana reaaliaikaisesti. Pino sisältää Elasticsearchin, joka on tiedon indeksointi, haku ja analytiikkatyökalu, sekä Logstashin, joka vastaanottaa dataa usealta eri lähteeltä, tarjoaa palvelinpuolen datan käsittely putken, käsittelee ne samanaikaisesti ja toimittaa eteenpäin Elasticsearchille. Kibana mahdollistaa datan tarkastelun ja visualisoinnin selainpohjaisella käyttöliittymällä. Lisäksi uusimpana lisäyksenä pinoon löytyy Beats, jolla on lisätty mahdollisuuksia kerätä erilaista monitoroitavaa dataa kohdelaitteilta esimerkiksi Logstashille käsiteltäväksi. Elastic Stack voidaan ottaa käyttöön paikallisesti kohdeympäristön sisällä, tai se voidaan ulkoistaa ”software as a service” palveluntarjoajalle. (Yasar 2022.)

3.2 Beats

Beats on uusin lisäys Elastic Stack -ryhmään ja se mahdollistaa datan keruun työasemilta sekä palvelimilta, ja helpottaa esimerkiksi sellaisten laitteiden monitorointia, jotka eivät tue Syslogia. Beats-tuoteperheeseen kuuluu useita data-shippereitä, jotka toimivat samalla periaatteella, mutta keräävät eri tietoa kohdejärjestelmästä, kuten esimerkiksi mitä tahansa tiedostoja, metriikkaa tai verkkoliikennettä. Beats-agentilla voidaan kerätä ja lähettää data joko suoraan Elasticsearchille tai ensin Logstashille käsiteltäväksi. Tällä hetkellä Beats-agentteja on seitsemän kappaletta, joilla voi kerätä erilaista dataa lokaalisti ja lähettää ne sitten haluttuun keskitettyyn kohteeseen. (What are Beats? n.d.)

Taulukko 2. Beats-agentit ja niiden käyttötarkoitus (What are Beats? n.d. Muokattu)

Agentin nimi	**Kerätty data**
Filebeat	Tiedostot
Metricbeat	Laitteiston metriikka
Packetbeat	Verkkoliikenne
Winlogbeat	Windowsin tapahtumaloki
Auditbeat	Audit loki
Heartbeat	Saatavuusloki
Functionbeat	Pilvidata

3.3 Logstash

Logstash on datan keräys-, suodatus- ja normalisointi palvelu, jolla voidaan vastaanottaa tietoa eri lähteistä, käsitellä se ennalta määrätyillä suodattimilla ja lähettää haluttuun kohteeseen tallennettavaksi ja analysoitavaksi. Logstash mahdollistaa merkkijonon suodattamisen ja tärkeän datan jaottelemisen erillisiin kenttiin ja luo yhtenäisen formaatin ja rakenteen, jotta tieto voidaan tallentaa tietokantaan jatko toimenpiteitä varten. Logstash on koodattu Jruby-kielellä, ja se vaatii toimiakseen Javan (Hallberg 2013).

Vaikka Logstash on alun perin esitelty uutena innovaationa lokien keräykseen, tarjoaa se nykyään myös paljon muita mahdollisuuksia tuon lisäksi. Se sisältää laajan valikoiman erinäisiä lisäosia, joita

voidaan hyödyntää vastaanotetun datan käsittelyssä. (Logstash Introduction N.d.) Logstash koostuu neljästä avain konseptista: pipeline, input, filter ja output.

Logstashin datan käsittely prosessista käytetään nimitystä pipeline, joka tarkoittaa Logstashin suorittamia datan käsittelyn vaiheita aina datan vastaanottamisesta sen suodattamiseen ja lähettämiseen eteenpäin. Pipelineja voi olla yhdellä Logstashilla useampi, ja jokainen niistä sisältää erillisen konfiguraation datan käsittelyyn. Eli mistä käsiteltävä data tulee, mitä sille tehdään ja minne sen matka jatkuu seuraavaksi. Input, filter ja output ovat Logstashin ydinkonfiguraatioita, joilla määritellään datan käsittelyn pipelinen sisällä.

Inputilla määritellään data, jota kyseisessä pipelinessa halutaan käsitellä. Datan vastaanottamiseen on useita eri vaihtoehtoja, mutta yleisimpinä voidaan pitää seuraavia esimerkkejä:

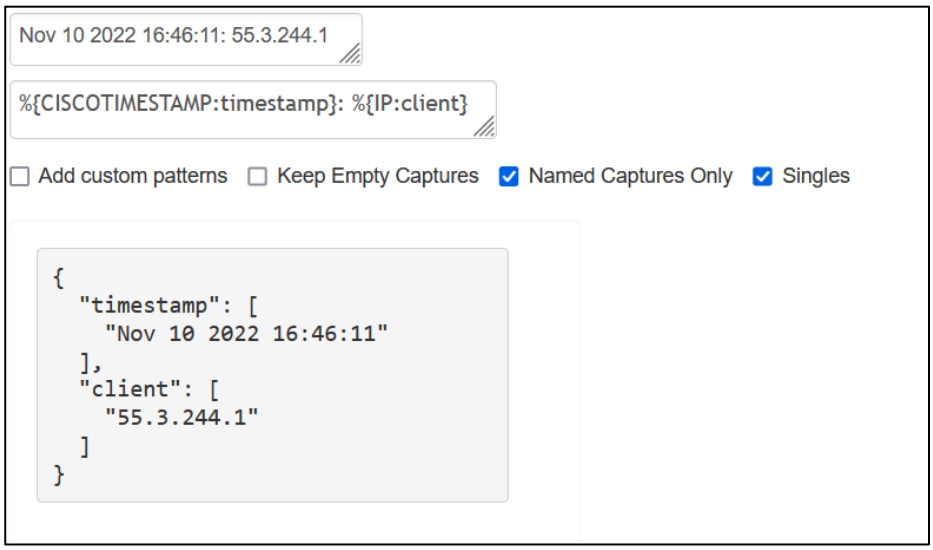
- Tiedosto, luetaan data lokaalista tiedostojärjestelmästä
- UDP, data vastaanotetaan määriteltyyn porttiin, johon toinen järjestelmä lähettää tiedon, tämä on yleinen tapa vastaanottaa lokitietoa verkkolaitteilta
- Beats, data vastaanotetaan määriteltyyn porttiin, johon lokinkeräys agentti ”Beats” lähettää halutut tiedot.

Filter-määrityksiä käytetään käsittelemään ja suodattamaan dataa, joka inputilla on vastaanotettu. Suodatusvaiheessa suoritetaan niin sanottu datan normalisointi, jolloin merkkijonosta saadaan parsittua haluttu tieto erillisiin kenttiin, jotta se voidaan tallentaa Elasticsearch indeksiin. Filter-työkalun määrityksiä ovat esimerkiksi seuraavat:

- Grok: Tärkeä työkalu, jolla suoritetaan varsinainen datan parsiminen, jolloin vastaanotettu data muutetaan jäsenneltyyn muotoon
- Mutate: Työkalulla voidaan muuttaa käsiteltävää dataa, esim. uudelleen nimeämällä, poistamalla ja siirtämällä
- Drop: Tapahtuma hylätään ilman prosessointia ja poistetaan pipelinesta.

Grok-suodattimilla tehdään suurin työ datan käsittelyssä. Se on erinomainen työkalu parsimaan jäsentelemätön data ja muuntamaan se jäsenneltyyn ja haettavaan muotoon. Grok toimii hakemalla tekstistä tiettyjä malleja, jotka vastaavat käsiteltäviä lokeja, ja sen syntaksi on mallia ”%{SYNTAKSI:TUNNISTE}”. Grok hakee siis tekstistä syntaksia vastaavaa tekstikuviota (pattern), erottaa sen omaan kenttäänsä ja nimeää kentän määritellyllä tunnisteella.

Internetistä löytyy useampi erilainen grok debugger -applikaatio, joilla voi testata erilaisia suodattimia omiin lokiriveihin ja ne näyttävät välittömästi lopputuloksen. Kuvio 2. näkyy yksinkertainen esimerkki, jossa ylimmälle riville on syötetty kuvitteellinen lokiviesti ja toiselle rivillä kyseiseen viestiin sovellettava grok-suodattimen. Applikaatio näyttää lopputuloksen tulosteena kenttien alapuolella, jossa näkyy suodatuksen jälkeinen lopputulos, kun lokiviesti on muunnettuna jäsenneltyyn muotoon, jossa sekä aikaleima, että IP-osoite on jaoteltu omiin erillisiin kenttiin (Kuvio 2). Tämä on helppo tapa todentaa grok-suodattimen toimivuus ja esittää, kuinka pelkän merkkijono data muuntuu käsitellyn mahdollistavaan muotoon.



Kuvio 2. Esimerkki grok-suodattimesta (Lähde: grokdebug.herokuapp.com)

Outputilla määritetään datan seuraava määränpää, kun se poistuu kyseisestä pipelinesta. Outputmäärityksen jälkeen Logstashin työ on ns. valmis. Yleisiä ulostulomäärityksiä ovat esimerkiksi:

- Elasticsearch, indeksoi ja tallentaa datan Elasticsearch tietokantaan
- File, tallentaa datan määriteltyyn tiedostoon lokaalisti
- UDP/TCP, data siirretään tiedonsiirtoprotokollalla toiselle palvelimelle.

3.4 Elasticsearch

Elasticsearch on Elastic Stackin keskeinen komponentti, jonka kehitti alun perin Shay Banon vuonna 2010. Se on Apache Lucenen päälle rakennettu avoimen lähdekoodin haku ja analytiikkatyökalu, joka kykenee vastaanottamaan ja käsittelemään eri muodoissa olevaa dataa. Lokien elinkaaren kannalta Elasticin tehtävä on suorittaa datan tallentaminen indekseihin ja vaativienkin hakulauseiden tehokas prosessointi. Indeksi on kokoelma dokumentteja ja sen voisi kuvailla olevan vastine relaatiotietokannan taululle, kun dokumentti puolestaan vastaa tietokannan riviä. Elasticsearchiin tallennettava data on tapana jakaa eri indekseihin jonkin yhdistävän tekijän perusteella. Esimerkiksi verkkolaitteiden lokitiedot voitaisiin jakaa eri indekseihin laitevalmistajan perusteella. (Srivastava 2019, 6–9.)

Elasticsearchin hakutoiminnot ja tiedonhallinta perustuu dokumenttimuotoiseen tietokantaan. Elasticsearch säilöö datan indekseihin JSON-dokumentteina, joissa tieto esitetään niin sanotusti key-value pareina, eli jokaiselle arvolle (value) on olemassa oma otsikkonsa (key). Indeksointiin, hakuihin ja päivityksiin käytetään DSL-kieltä ja REST API -rajapintaa. (Data in: documents and incides n.d.)

Yksittäistä Elasticsearch instanssia kutsutaan nimellä solmu (node). Instansseja voidaan pystyttää useampia, jolloin ne yhdistettynä muodostavat klusterin. Yksi Elasticsearch-instanssi toimii aina niin sanottuna master-instanssina, johon muut solmut yhdistetään. Klusteroinnin avulla tuotetaan redundanssia, sekä lisätään datan käsittelyn skaalautuvuutta. Mikäli haku suoritetaan klusterilla, siihen osallistuvat kaikki kyseiseen klusteriin liitetyt solmut.

3.5 Kibana

Kibana on viimeinen palanen Elastic Stack -pinosta ja se tarjoaa Elasticsearchille visualisointi- ja hallintakäyttöliittymän. Kibana on selaimella käytettävä työkalu, jolla voidaan tarkastella ja esittää Elasticsearchin indeksoimaa dataa useilla eri tavoilla, kuten esimerkiksi tuottamalla visuaalisia kaavioita, taulukoita ja dashboardeja. Visualisointi mahdollistaa datan esittämisen ihmiselle mieleisessä muodossa, esimerkiksi viiva tai pylväsdiagrammi on hyvin selkeä tapa kuvata, vaikka yleisesti valvonnassa olevien verkkolaitteiden lokimääriä. Kibana sisältää visualisointityökalujen lisäksi pääsyn myös Elasticsearchin kehitys- ja hallintatyökaluihin.

Kibana sisältää useita eri työkaluja ja lisäosia, joita voidaan hyödyntää lokien käsittelyssä ja visualisoinnissa. Tässä työssä keskitytään kuitenkin ainoastaan kolmeen olennaiseen työkaluun, joita ovat: discover, dashboards ja alerts. Discover on työkalu, jolla voi tarkastella dokumentteja ja suorittaa hakuja eri indekseistä. Dashboardilla voi tehdä erilaisia näkymiä, johon yhdistetään erilaisia datan esitystapoja, kuten taulukoita ja kaavioita. Alerts mahdollistaa hälytyssääntöjen tekemisen, hälytysten generoimisen ja niiden valvonnan.

4 Toteutus

4.1 Toteutusympäristö

Työn käytännön toteutuksena lokienhallintajärjestelmä implementoidaan Puolustusvoimien johtamisjärjestelmäkeskuksen testiympäristöön, joka on verkkojen kehitystä ja koulutusta varten pystytetty tuotantoverkosta täysin erillinen ympäristö. Kohdeympäristö koostuu useista sekä fyysisistä, että virtuaalisista verkkolaitteista. Verkon rakentaminen on vielä kesken, mutta sieltä on tarkoitus tulevaisuudessa löytyä myös palveluita sekä työasemia, jotta verkkoon saataisiin simuloitua todenmukaista verkkoliikennettä. Ympäristön lokienhallinta implementoidaan tässä vaiheessa ainoastaan verkon virtuaalisille verkkolaitteille, jotka ovat Cisco-merkkisiä reitittimiä.

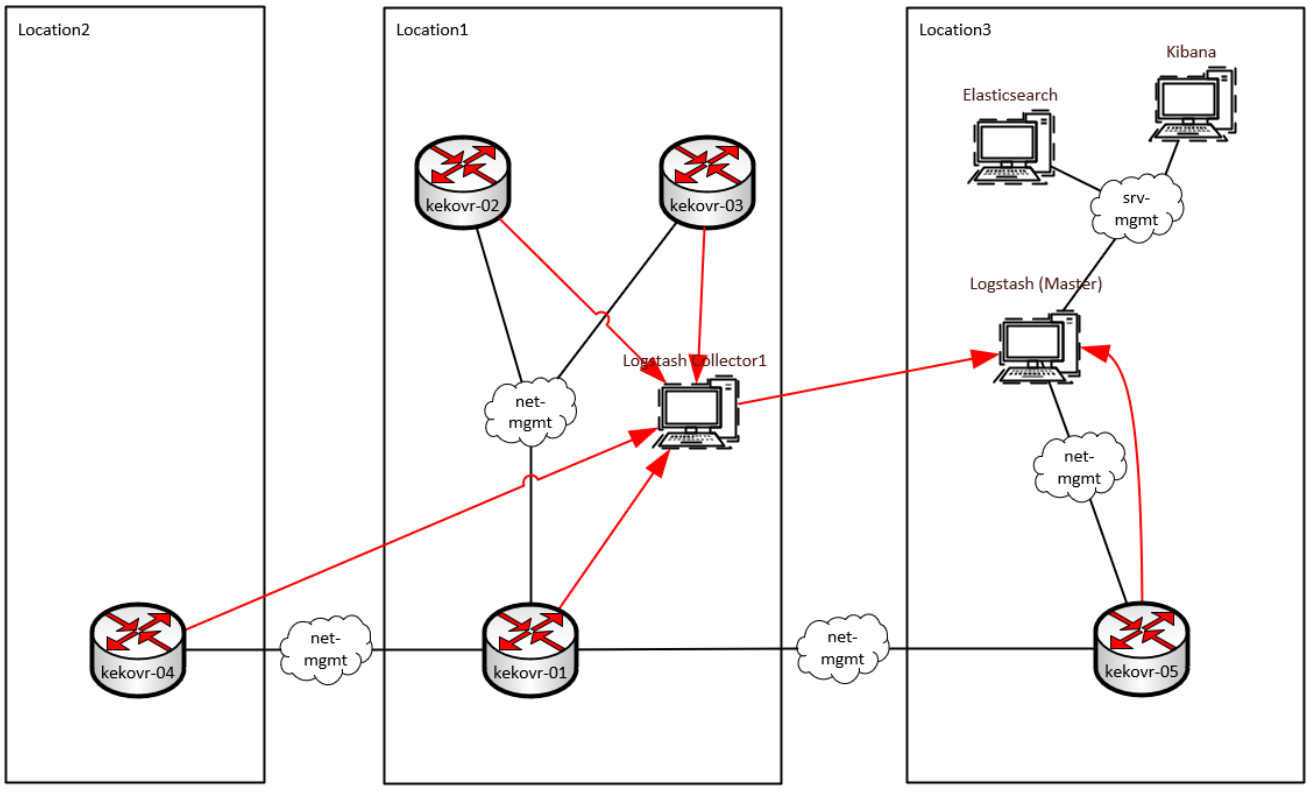
4.2 Lokienhallinnan topologia

Kuvio 3. esitetty toteutetun lokienhallintajärjestelmän looginen kuvaus, jossa näkyy lokien kulku verkkoympäristössä lähdelaitteilta keräykseen ja analyysijärjestelmään. Punaisilla nuolilla on merkitty looginen reitti, mitä kautta lokit viedään verkkolaitteilta isäntä Logstashille. Kyseisellä toteutuksella on tarkoitus kuvata lokien keräyksen kolmessa erilaisessa lokaatiossa. Ensimmäisenä on lokaatio 1, jossa sijaitsee itse lokienhallintajärjestelmä ja verkkolaite lähettää lokinsa suoraan master Logstashille. Toisessa lokaatiossa verkkolaitteiden lokit kerätään ensin keskitetysti fyysisen sijainnin perusteella yhdelle palvelimelle, joka suorittaa raakakäsittelyn ja toimituksen yhtenäisesti lokaatioon 1. Viimeisessä lokaatiossa oman Logstashin toteuttaminen ei ole mahdollista, vaan lokit kerätään käyttämällä joko jonkun muun lokaation keräintä, tai mahdollisesti suoraan master-keräintä. Master Logstash suorittaa ympäristön kaikkien verkkolaitteiden lokitiedon lopullisen käsittelyn, jonka jälkeen lokit siirretään Elasticsearchille indeksoitavaksi.

Lokien siirtely tapahtuu ainoastaan hallintaan käytettävässä verkossa nimeltä net-mgmt, joka on täysin eriytetty verkkoympäristön muusta liikenteestä. Verkkolaitteet ja Logstashit käyttävät lokien siirtoon hallintaverkkoa, kun taas Elasticsearch, master Logstash ja Kibana keskustelevat keskenään käyttämällä palvelinhallintaverkkoa srv-mgmt. Logstash master-instanssi toimii yhdyskäytävänä eri verkkojen välillä ja toimittaa net-mgmt puolelta vastaanottamansa ja käsittelemänsä lokitiedot Elasticsearchille srv-mgmt-verkon puolella.

Lokien keräys ja käsittely koostuu neljästä vaiheesta:

- Verkkolaitteet lähettävät lokiviestit käyttämällä Syslog-protokollaa määritysten mukaisesti joko omalle lokienkeräyspalvelimelle, tai suoraan master Logstashille.
- Keräinpalvelin suorittaa alustavan lokien suodattamisen ja lähettää viestit eteenpäin master-palvelimelle käyttämällä Lumberjack-protokollaa.
- Logstash Master palvelin suorittaa lokien lopullisen suodattamisen ja normalisoinnin, jonka jälkeen se lähettää lokit indeksoitavaksi ja tallennettavaksi Elasticsearchille.
- Kibanaa käytetään visualisoimaan ja analysoimaan Elasticsearchiin tallennettuja lokeja.



Kuvio 3. Lokienhallintajärjestelmän verkkokuva

Lokienhallintaan käytettävät työkalut asennetaan palvelimille, jotka ovat nimenomaan palvelinylläpitoon tarkoitettuja ainoastaan komentolinjan sisältämiä Ubuntu Server 20.04 LTS -pohjaisia virtuaalikoneita. Ubuntu Server on avoimeen lähdekoodiin perustuva maksuton Linux-jakelu.

4.3 Lokien kerääminen verkkolaitteilta

Kuten aikaisemmin mainittu, tässä toteutuksessa on tarkoitus ainoastaan kerätä, ja tutkia verkkolaitteiden tuottamaa lokitietoa. Oletuksena verkkolaitteet yleensä tallentavat jotain lokitietoa laitteen puskuriin, jota voidaan tarkastella laitteen komentoriviltä (System Message Logging 2008). Erillisellä konfiguraatiolla saadaan määriteltyä sekä lokitietojen keräykseen käytettävä lokienkeräyspalvelin, että asetukset, jotka määrittelevät mitä kaikkea lokitietoa kerätään. Kokonaisuudessaan verkkolaitteelle tehdyt määritykset löytyvät liitteestä 1.

Ensin konfiguraatioon määritellään lokien keräyspalvelin, johon lokiviestit halutaan lähettää. Laitteelle tulee kertoa lokipalvelimen IP-osoite ja mistä vrf-alueesta kyseinen verkko-osoite löytyy. Vrf on virtuaalireititystekniikka, joka mahdollistaa useiden eri reititystaulujen ylläpitämisen yhdellä laitteella. Vrf-alueen määrittäminen tässä tapauksessa on välttämätöntä, kun niitä on useampi ja reititin on yhdistetty useampaan eri verkkoon. Samalle riville voidaan määritellä myös siirtoon käytettävä tiedonsiirtoprotokolla sekä portti, jota lokienhallintapalvelin kuuntelee. Lokien keräyspalvelin määritellään verkkolaitteelle seuraavalla komennolla.

```
logging host x.x.x.x vrf net-mgmt transport udp port 515
```

Laitteelle tulee määritellä tietty rajapinta, jota käytetään lokiviestien lähettämiseen. Loopback-paluuosoite on tähän erittäin käytännöllinen, koska se on laitteen sisäinen verkkoliityntä, ja sillä voidaan yksilöidä ja tunnistaa laitteita riippumatta siitä, että mitä verkkoa pitkin lokiviestit lähetetään. Rajapinnaksi määritelty IP-osoite näkyy myös lokipalvelimella lokiviestin lähteenä. Rajapinnan määrittelyn käytettävästä komennosta esimerkki alla.

```
logging source-interface Loopback 51 vrf net-mgmt
```

Näillä kahdella edellä mainitulla komennolla saadaan jo lokiviestien lähettäminen toimimaan, mutta lokitietojen sisältöä ja tarkkuutta voidaan vielä määritellä erilaisilla parametreillä. Verkkolaitteelle voidaan rajata lokiviestien generoimista vakavuusasteen perusteella, kuten 2.6 osiossa esitetään. Verkkolaitteet määritellään kirjoittamaan lokiviestit kaikista informational-tason ja sitä vakavammista tapauksista seuraavalla komennolla.

```
logging trap informational
```

Kirjautumiset, käyttöoikeuksien lisäykset, sekä esimerkiksi laitteelle muodostetut SSH-yhteydet ovat asioita, joista ehdottomasti halutaan generoida lokirivi, mutta niiden päätyminen lokiviesteihin vaatii myös lisäkonfiguraatioita. Nämä ovat kriittisiä tietoja, jotka mahdollistavat tapahtumaketjun seuraamisen, kun joku käyttäjä on tehnyt muutoksia verkkolaitteella. Seuraavilla määrityksillä lisätään kaikki onnistuneet ja epäonnistuneet kirjautumiset, käyttäjän oikeuksien korottamiset, sekä kaikki SSH-kirjautumistapahtumat.

```
login on-success log
login on-failure log
logging userinfo
ip ssh logging events
```

Lisäksi valvontaan haluttiin lisätä myös kaikki verkkolaitteilla tehdyt konfiguraatiomuutokset. Verkkolaitteella otetaan konfiguraatiomuutoksista generoituvat lokimerkinnät ja määritellään ne kirjattavaksi Syslogiin selkotekstinä. Hidekeys-parametrillä konfiguraatioriveistä piilotetaan mahdollisesti siellä esiintyvät salasanat. Määritykset otetaan käyttöön seuraavilla komennoilla.

```
archive
log config
logging enable
notify syslog contenttype plaintext
hidekeys
```

4.4 Palvelimet

Ennen ohjelmistojen asennuksia tulee valmistella palvelin, jonka komennot löytyvät kokonaisuudessaan liitteestä 2. Palvelimen valmistelussa suoritetaan käyttöjärjestelmän asentaminen sekä viimeisimpien päivitysten lataaminen. Tämän jälkeen suoritetaan verkkoasetusten määrittely, jotta palvelin saadaan yhdistettyä haluttuun verkkoon halutulla IP-osoitteella. Lisäksi tulee määritellä myös kellonaikaan liittyvät asetukset sekä aikavyöhyke. Ympäristössä on käytössä NTP-palvelin, jolta verkkolaitteet ja palvelimet hakevat kellonajan, jotta ympäristön kaikkien verkkolaitteiden ja palvelimien kello pysyy synkronoituina. Kellonaikojen synkronointi on erittäin tärkeää lokienhallinnan kannalta, jotta voidaan olla varmoja, että lokiviestien aikaleimat pitävät paikkansa. Kun käyttöjärjestelmä on valmisteltu, voidaan aloittaa työkalujen asennus.

Työkalujen riippuvuudet

Elastic tarjoaa useamman tavan ladata paketteja heidän arkistoistaan (repository). Yksinkertainen ja helppo tapa on käyttää käyttöjärjestelmän omaa paketinhallintatyökalua, joka tässä tapauksessa on APT. Elasticin kaikki paketit on allekirjoitettu julkisella avaimella, joka tulee lisätä palvelimelle ennen asennuspakettien latausten aloittamista. Ubuntun latauksien hallinta käyttää julkista avainta varmistamaan ladattujen pakettien luotettavuuden. Julkinen avain ladataan Elasticin verkkosivuilta wget-työkalua käyttäen ja tallennetaan tiedostoon ”/usr/share/keyrings/elasticsearch-keyring.gpg” seuraavalla tavalla.

```
wget -qO - /
https://artifacts.elastic.co/GPG-KEY-elasticsearch /
| sudo gpg --dearmor -o /
/usr/share/keyrings/elasticsearch-keyring.gpg
```

Ubuntu ei vakiona tiedä, että mistä Elasticin asennuspaketit löytyvät, vaan käyttöjärjestelmälle tulee syöttää Elasticsearchin arkisto, jotta se osaa hakea paketit Elasticsearchin verkkosivuilta. Arkiston syöttäminen onnistuu luomalla uusi tiedosto hakemistoon ”/etc/apt/sources.list.d” ja kirjoittamalla sinne Elasticin aikaisemmin tallennetun julkisen avaimen tiedosto sijainti, sekä Elasticsearch

web-sivusto, josta asennuspaketit löytyvät. Elasticin uusi version on 8, mutta tässä vaiheessa voitaisiin tarpeen mukaan määritellä ladattavaksi myös joku vanhemmista julkaisuista. Komento Elasticsearch repositorion lisäämiseen löytyy alta.

```
echo "deb /
[signed-by=/usr/share/keyrings/elasticsearch- keyring.gpg]
https://artifacts.elastic.co/packages/8.x/apt stable /
main" | sudo tee /etc/apt/sources.list.d/elastic-8.x.list
```

Elastic Stackin työkalut vaativat toimiakseen openjdk ja apt-transport-https-paketit, jotka ladataan käyttämällä APT-työkalua. Lopuksi ajetaan vielä kerran järjestelmän päivityskomento, jonka jälkeen suoritetaan itse asennuspaketin lataus. Tähän asti kaikkien eri työkalujen asennustoimenpiteet ovat samat, kunnes install-kohdassa määritellään vain se työkalu, joka halutaan kyseiselle palvelimelle ladata. Kaikki työkalut voi toki asentaa myös samalle palvelimelle, mutta ainakin Logstashin suositellaan eriytettävän resurssien jakamisen helpottamiseksi. Asennuskomentojen esimerkit löytyvät alta.

```
apt install openjdk-8-jdk
apt install apt-transport-https
apt-get update
apt-get install(elasticsearch/logstash/kibana)
```

4.5 Elasticsearchin konfigurointi

Kun lataus on suoritettu käyttämällä APT-työkalua, niin oletuksena Elasticin asetustiedostot löytyvät hakemistosta /etc/elasticsearch, tärkeimpänä elasticsearch.yml, joka löytyy liitteestä 3 ja se sisältää työkalun ajossa käytettävät peruskonfiguraatiot. Konfiguraatiotiedostot ovat kirjoitettu YAML-merkintäkielellä. Suurimmaksi osaksi konfiguraatiot on asetettu automaattisesti asennuksen yhteydessä tehtyjen valintojen perusteella, mutta ainakin HTTP API yhteydet tule hyväksyä muualtakin kuin paikalliselta koneelta, ainakin siinä tapauksessa, jos Kibana pyörii eri palvelimella, kuin Elasticsearch.

Salaukset

Elasticsearchin ensimmäisen käynnistyskerran yhteydessä luodaan automaattisesti sekä ilmoittautumistunnus (enrollment token), että pääkäyttäjätunnus. Komentolinjalle tulostuu pääkäyttäjän salasana sekä tuo uniikki tunnus, jolla pystyy luomaan Kibanan ja Elasticiin välisen yhteyden automaattisesti ilman manuaalisten konfiguraatioiden tekemistä. Nämä merkkijonot näytetään ainoastaan Elastichin ensimmäisen käynnistyskerran yhteydessä, joten ne kannattaa laittaa hyvään talteen (Start the Elastic Stack with security enabled automatically n.d.). Pääkäyttäjällä on kaikki oikeudet Elasticsearchin hallinnassa, sillä pystyy muun muassa määrittelemään ohjelmiston tietoturva-asetuksia, luomaan ja muokkaamaan erilaisia rooleja, käyttäjiä, sekä niiden käyttöoikeuksia. (Set up basic security for the Elastic Stack n.d.)

Salasanasuojauksen lisäksi työkalujen välinen todentaminen, sekä tiedon salaus kannattaa suorittaa kuljetuskerroksella käyttämällä TLS-protokollaa. TLS on tiedon suojaukseen käytettävä standardi protokolla, jolla suojataan Elastic Stack -työkalujen keskinäinen yhteys ja solmujen välinen viestintä sekä kuljetus-, että http-tasoilla. Salausprotokollan käyttäminen varmistaa, ettei haitallinen tai ei haluttu instanssi pysty liittymään Elasticsearch klusteriin. Elastic generoi asennuksen yhteydessä CA-varmenteen ja tallentaa sen hakemistoon ”/etc/elasticsearch/certs/”. Varmenne takaa työkalujen välisen autentikoinnin, sekä salauksen. (Set up basic security for the Elastic Stack n.d.)

Elasticsearchin asennuksen yhteydessä turvallisuusominaisuudet ovat vakiona käytössä ja niiden konfiguraatiot luodaan automaattisesti. HTTP-rajapinnan ja sertifikaatin toimivuuden voi varmistaa yhdistämällä curl-komennolla localhostiin ja syöttämällä turvallisuuden osalta määritellyt parametrit, tässä tapauksessa varmenne, ja pääkäyttäjätunnus. Tämän jälkeen komentolinja kysyy vielä käyttäjätunnukselle salasanaa, jonka syötettyä tulostuu HTTP-kyselyn vastaus. Toimivan Elasticsearchin vastaus tulisi näyttää kutakuinkin samalta, kuin Kuvio 4.

```
root@elasticmaster:/etc/elasticsearch/certs# curl --cacert http_ca.crt -u elastic https://localhost:9200
Enter host password for user 'elastic':
{
  "name" : "elasticmaster",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "f572SpunSd-IEX4_njME0Q",
  "version" : {
    "number" : "8.5.0",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash" : "c94b4700cda13820dad5aa74fae6db185ca5c304",
    "build_date" : "2022-10-24T16:54:16.433628434Z",
    "build_snapshot" : false,
    "lucene_version" : "9.4.1",
    "minimum_wire_compatibility_version" : "7.17.0",
    "minimum_index_compatibility_version" : "7.0.0"
  },
  "tagline" : "You Know, for Search"
}
```

Kuvio 4. Elasticsearchille paikallisesti tehty curl-kysely

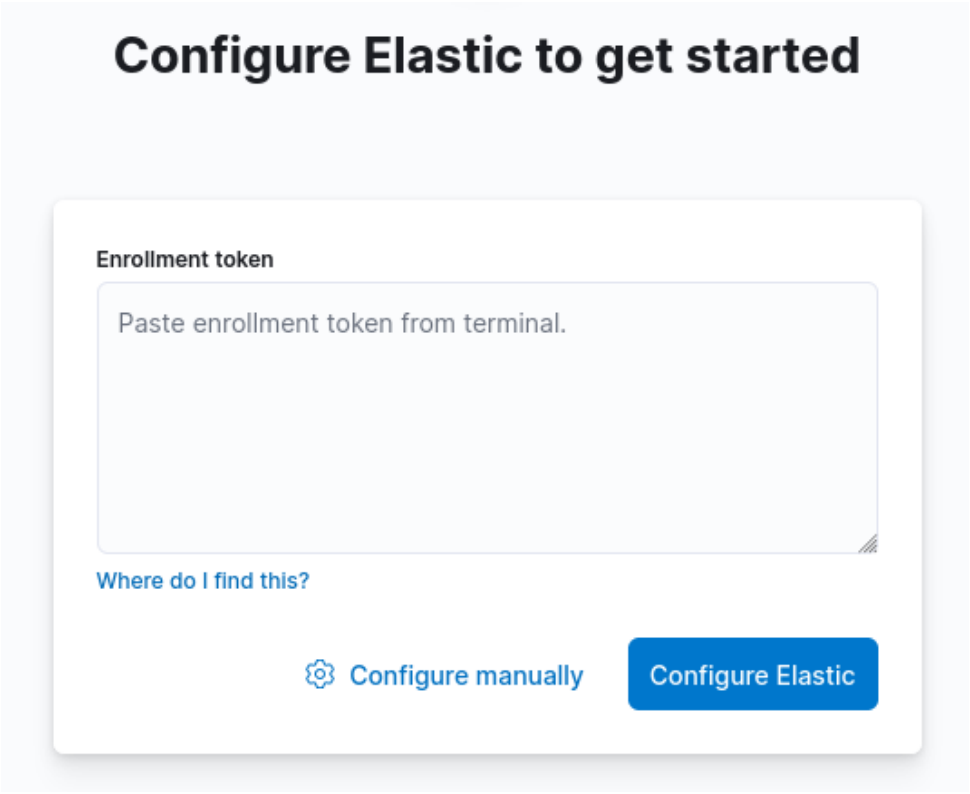
4.6 Kibanan konfigurointi

Kibanan konfiguraatiotiedosto löytyy liitteestä 4. Kuten muillakin työkaluilla, Kibanan tärkeimmät konfiguraatiot ja määritykset löytyvät tiedostosta ”/etc/kibana/kibana.yml”. Ainoa muutos, joka tässä vaiheessa tulee tehdä, on asettaa palvelimen IP-osoite, johon Kibanan verkkokäyttöliittymä sidotaan, jolloin siihen pystyy yhdistämään myös muilta koneilta.

Jotta Kibanan ja Elasticsearch-klusterin välinen kommunikointi voidaan sallia, onnistuu se automaattisesti käyttämällä Elasticsearchilla generoitua ilmoittautumistunnusta. Elasticsesarchin mukana tulee työkalu nimeltä create-enrollment-token, jolla voidaan generoida tunnukset Kibanalle sekä muille Elasticsearch-solmuille seuraavalla tavalla.

```
./bin/elasticsearch-create-enrollment-token -s kibana
```

Kibana kysyy mahdollista ilmoittautumistunnusta ensimmäisellä keralla, kun web-käyttöliittymä avataan (Kuvio 5). Generoitu tunnus tulee syöttää annettuun kenttään, jonka jälkeen se suorittaa Kibanalle automaattisesti kyseiseen Elasticsearchiin yhdistämiseen vaaditut turvallisuus määritykset ja konfiguraatiot, sekä luo instanssien välisen yhteyden. kibana.yml-tiedoston lopussa näkyy automaattisesti luodut konfiguraatiorivit (Liite 4).



Kuvio 5. Kibana enrollment token -lomake

4.7 Logstashin konfigurointi

Kun Logstash on asennettu, voidaan seuraavaksi alkaa konfiguroimaan sen määrityksiä. Logstashin asennukseen määrityksiin käytettävät komennot löytyvät liiteestä 5. Tärkeät konfiguraatiot löytyvät lähinnä kolmesta tiedostosta: logstash.yml, pipeline.yml ja vähintään yksi conf-päätteinen tiedosto, jossa on itse lokien vastaanottamista, käsittelyä ja siirtoa koskevat määritykset. Sekä master, että collector Logstashien pipeline-konfiguraatiot löytyvät liitteistä 6–7, josta näkyy, että pipelineen määritellyt konfiguraatiot ovat pilkottu useampaan tiedostoon, jotta niiden hallitseminen ja ymmärtäminen olisi hieman yksinkertaisempaa. Lokiviestien reitti konfiguraatiotiedostojen välillä on esitetty Kuvio 6.

Logstash.yml sisältää käytännössä kaikki sen ajoon liittyvät asetukset. Tiedostossa määritellään esimerkiksi tietyt pipelinen määritykset, konfiguraatiotiedostojen sijainnit, itse Logstash työkalun tuottamien lokien asetukset sekä mahdolliset sertifikaatit, mikäli Elasticsearchilla on määritetty käyttämään sellaisia.

Pipeline.yml tiedostossa määritellään konfiguraatiotiedostojen sijainnit, jotka sisältävät input-, filter- ja outputmääritykset. Logstashin konfiguraatio tiedostot löytyvät vakiona hakemistosta ”/etc/logstash/conf.d”. Tiedostoja voi olla useampia, jolloin pipeline.yml tiedostossa määritellään, että mitä kyseisistä tiedostoista halutaan Logstashin käyttävän. Yksinkertainen, ja vakionakin käytössä oleva tapa on määritellä tiedostoon jokerimerkillä, että käytetään kaikkia conf-päätteisiä tiedostoja.

4.7.1 Input

Input-plugineilla määritellään lähteet, joista Logstash vastaanottaa dataa, jota pipelinessa halutaan käsitellä. Master Logstash palvelin vastaanottaa lokeja suoraan samassa lokaatiossa olevilta verkkolaitteilta, mutta myös sen lisäksi toiselta loki keräimeltä, jolloin inputkonfiguraatio vaatii erilaisen määrityksen. Verkkolaitteet lähettävät Syslogin käyttämällä UDP-protokollaa, joten määrityksiin asetetaan protokolla sekä portti, jota logstash kuuntelee, kun taas toiselta Logstashilta tulevat viestit vastaanotetaan beats-lisäosalla. Esimerkit molemmista inputkonfiguraatioista on liitteessä 7. ”1-input-udp.conf” ja ”1-input-lumberjack.conf”. Lisäksi tässä vaiheessa voidaan vapaaehtoisesti lisätä tietoa lokiriveihin, kuten lokin tyyppi ja tageja, jotka helpottavat lokien jatkokäsittelyä myöhemmin.

4.7.2 Filter

Filter-määrityksissä puretaan Logstashin vastaanottama viesti eri kenttiin indeksointia varten. Tässä vaiheessa käytettään ehdollisia lauseita, joilla saadaan määriteltyä tietyt lokiviestit käsiteltäväksi oikealla grok-suodattimella. Liitteessä 7. näkyy eri filter-konfiguraatiotiedostoja, jotka sisältävät erilaisia suodattimia, joiden soveltamista voidaan rajata ehtolauseilla riippuen siitä missä muodossa saapuvien lokiviestien oletetaan olevan. Kyseisessä tapauksessa, eri lähteiltä saapuvat lokit on merkattu tägeillä kuten ”Cisco” tai ”Lumberjack”, jonka perusteella osataan soveltaa oikeita suodattimia.

Lokiviestien saapuessa Logstashille, suoritetaan ensimmäisenä Syslogin oletuskenttien suodatus. Ensimmäinen suodatus on tarkoitus tehdä aina sillä Logstashillä, joka ensimmäisenä vastaanottaa kyseisen lokiviestin. Tällä pystytään jakamaan kuormaa Logstashien välillä.

Taulukko 3 on esitetty esimerkki seuraavan lokirivin suodattamisesta ja kenttiin jaottelusta grok-suodattimilla. Alta löytyy esimerkki lokirivi, jonka Ciscon virtuaalireititin on generoinut portin muutoksesta.

```
<189>9655: Feb 14 09:40:09.325: %LINK-3-UPDOWN: Interface
GigabitEthernet1, changed state to up
```

Taulukon 3 ensimmäisessä sarakkeessa on sovellettavan grok-suodattimen syntaksi, toisessa sarakkeessa on syntaksia vastaava REGEX-malli, kolmannessa on otsikkokentän nimi ja viimeisessä suodatuksen jälkeinen lopputulos. Grok etsii tekstistä REGEX-malleja ja normalisoi tiedon sitä mukaa JSON-muotoon. Grok-suodattimia voi käyttää myös sisäkkäin, kuten ”CISCOTIMESTAMP”-suodattimen riviltä näkyy, että REGEX-kaava sisältää ”MONTH”, ”MONTHDAY”, ”YEAR” ja ”TIME” -suodattimet, jotka käsittelevät osiltaan eri ajanilmauksia. Kyseiseen lokiriviin sovellettava grok-suodatin kokonaisuudessaan näyttää seuraavalta.

```
%{SYSLOG5424PRI}%{NONNEGINT:version}: /
%{CISCOTIMESTAMP:syslog_timestamp}: /
%%{CISCO_REASON:facility}-%{INT:severity_level}- /
%{CISCO_REASON:facility_mnemonic}: /
%{GREEDYDATA:syslog_message}
```

Taulukko 3. Lokirivin käsittely grok-suodattimella.

Grok-suodatin	**REGEX-kaava**	**Otsikko**	**Tulos**
%{SYSLOG5424PRI: syslog5424_pri}	<\b(?:[0-9]+)\b>	syslog5424_pri	189
%{NONNEGINT: version}	\b(?:[0-9]+)\b	version	9655
% {CISCOTIMESTAMP: syslog_timestamp}	%{MONTH} +%{MONTHDAY} (?: %{YEAR})? %{TIME}	syslog_timestamp	Feb 14 09:40:09.325
%{CISCO_REASON: facility}	(?:\b\w+\b \s*)*	facility	LINK
%{INT:severity_level}	(?:[+-]?(?:[0-9]+))	severity_level	3
%{CISCO_REASON: facility_mnemonic}	(?:\b\w+\b \s*)*	facility_mnemonic	UPDOWN
%{GREEDYDATA: syslog_message}	.*	syslog_message	Interface GigabitEther-net1, changed state to up

Lokirivin lopussa löytyvän viestiosuus tallennetaan aluksi yhteen kenttään nimeltä “syslog_message”, mutta master Logstashilla siitä voidaan suodattaa vielä tarpeellista tietoa erillisiin kenttiin. Esimerkiksi Taulukon 3 viimeisellä rivillä oleva tulossarakkeen viesti sisältää hyödyllistä tietoa portin tilan muutoksesta. Liitteessä 7. “21-filter-message.conf” on nähtävillä, kuinka portin tunnus sekä muutoksen tiedot on parsittu omiin kenttiinsä ehtolauseen jälkeen.

4.7.3 Output

Lokien vastaanoton ja suodattamisen päätteeksi määritellään vielä, että minne lokien matka jatkuu seuraavaksi. Liitteessä ”3-output-elasticsearch.conf” näkyy master Logstashin outputkonfiguraatio, jossa on määritelty Elasticsearchin IP-osoite ja portti. Output-konfiguraatioihin tulee myös määritellä mahdollisen sertifikaatin sijainti, sekä Elasticsearchiin kirjoitusoikeudet omaava käyttäjä ja tälle salasana. Lisäksi voidaan määritellä indeksin nimi, johon kyseisen pipelinen lokiviestit halutaan Elasticsearchissa tallentaa.

4.7.4 Lumberjack

Kahden Logstash-instanssin väliseen tiedonsiirtoon käytetään Lumberjack-protokollaa, joka omat erilliset määritykset. Lumberjack käyttää Logstashin inputkonfiguraatiossa beats-lisäosaa, jonka alle määritellään samoin kuin aikaisemmin vähintään portti, johon verkkolaitteet lähettävät lokiviestinsä. Lähetykseen tarvitaan erikseen lumberjack-output-lisäosa, joka ladataan käyttämällä logstash-plugin-työkalua seuraavalla tavalla.

```
.bin/logstash-plugin install logstash-output-lumberjack
```

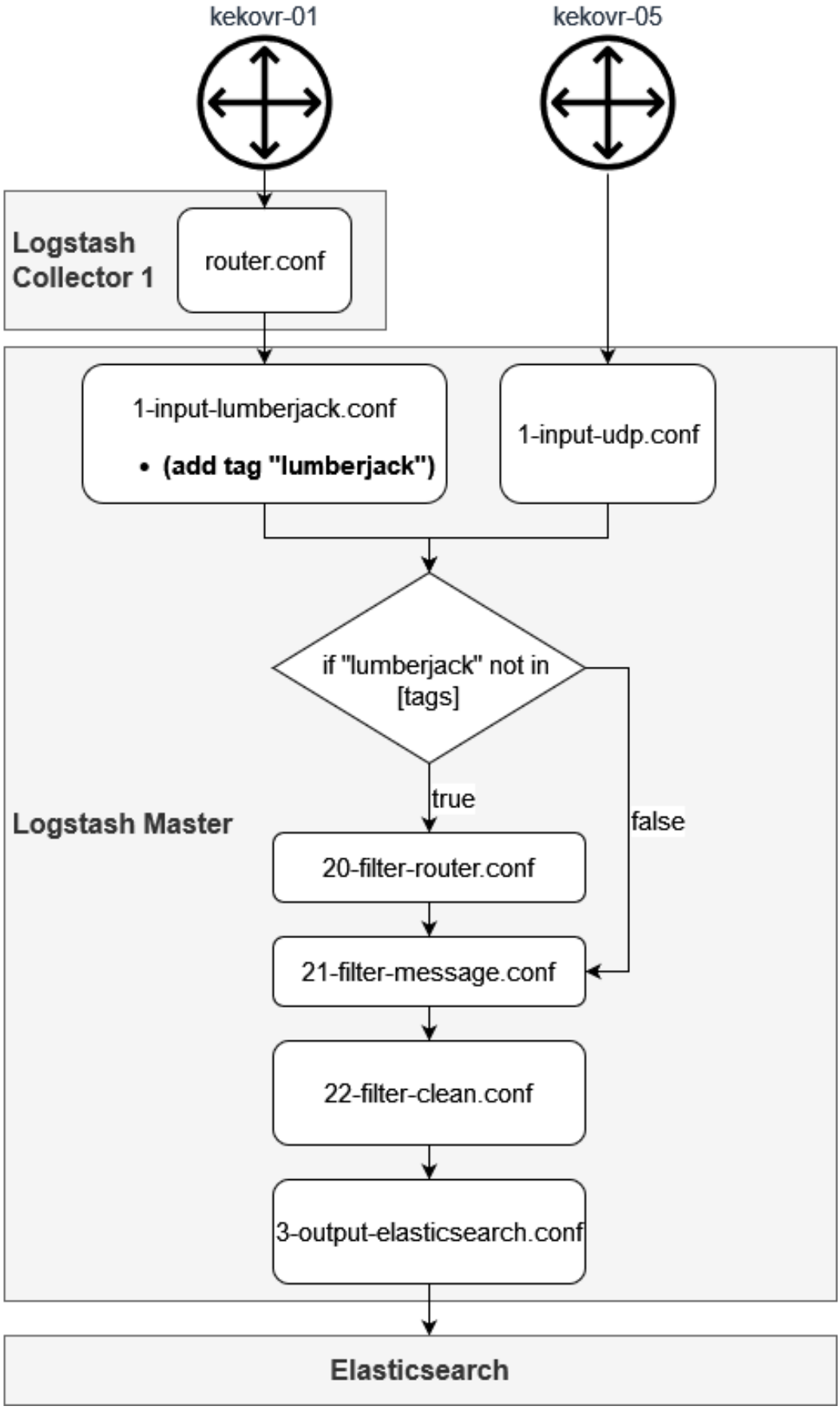
Lumberjack-yhteyden ollessa salattu, tulee vastaanottavan palvelimen generoida ja listata SSL-sertifikaatti sekä avain, joista sertifikaatti tulee siirtää myös lähettävälle Logstash-instanssille. Sertifikaatti generoidaan käyttämällä Openssl-työkalua. Komento generoi sekä key, että cert tiedostot, joista cert sisältää salaukseen käytettävän datan, ja key purkamiseen. Sertifikaatin generoinnista on kuvattu esimerkki alla.

```
openssl req -x509 -batch -nodes -newkey rsa:2048 /
-keyout lumberjack.key -out lumberjack.cert -subj /
CN=logstashmaster
```

Sertifikaattien määritteleminen Logstashin input- ja outputkonfiguraatioille on nähtävissä liitteissä 7. ”1-input-lumberjack.conf” ja 8. ”router.conf”.

4.8 Lokiviestien reitti verkkolaitteelta Elasticsearchille

Valmis toteutus on esitetty vuokaaviossa (Kuvio 6). Kaaviossa esitetään lokiviestien kulku verkkolaitteiden, Elasticsearchin, sekä Logstash instanssien ja konfiguraatiotiedostoiden välillä. Kuten kaaviosta voi havaita, Logstash Collector 1 -palvelimen keräämät ja suodattamat lokiviestit kulkevat hieman eri reittiä, koska lokiviestien suodattaminen on jo osin tehty ennen kuin lokiviestit saapuvat master Logstashille. Logstashien välisellä Lumberjack-yhteydellä saapuvat lokit merkitään tagilla “lumberjack”, jonka perusteella voidaan määrittää, että minkä suodattimien läpi kyseiset lokiviestit ajetaan master Logstashilla. Vuokaaviossa kuvataan ehtolause, jossa tagin perusteella jätetään soveltamatta tiedoston “20-filter-router.conf” suodattimia verkkolaitteilla, joilla vastaavat asiat on jo tehty Logstash Collector1 -palvelimella.



Kuvio 6. Vuokaavio lokiviestien reitistä

4.9 Visualisointi

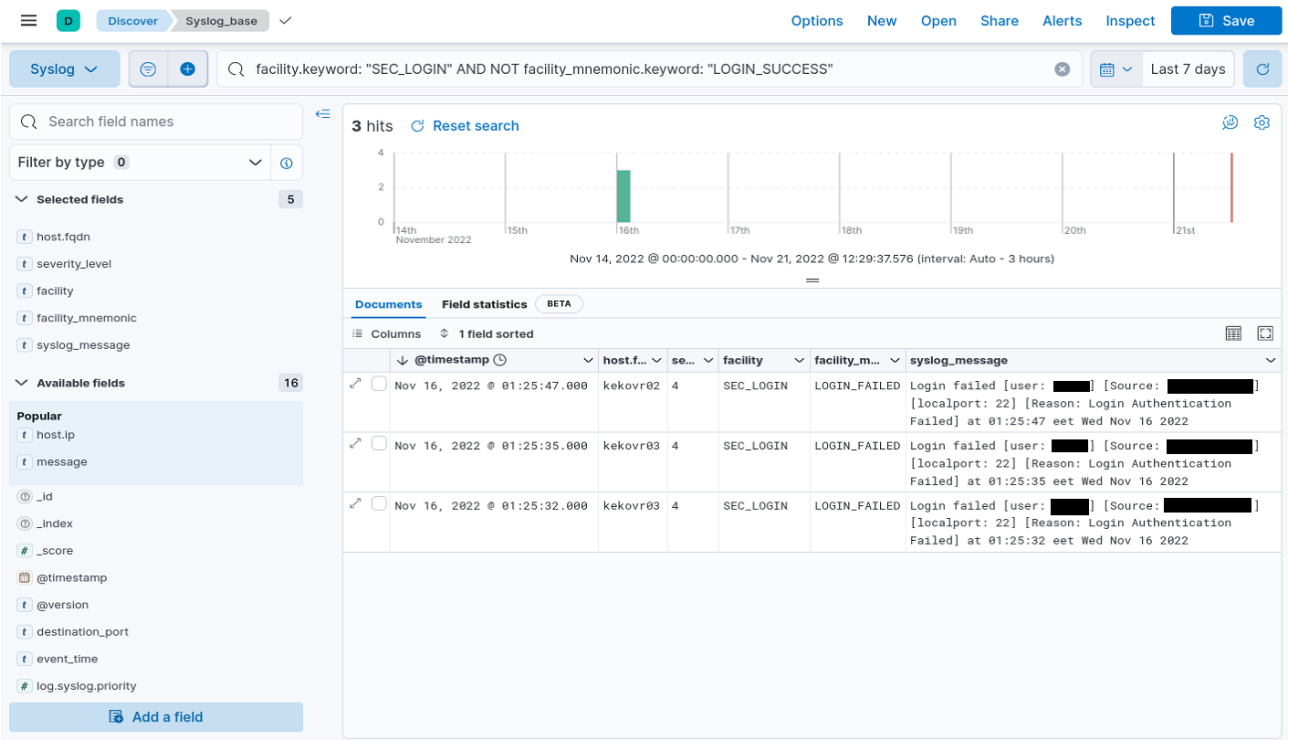
Työn viimeisenä osuutena on lokitietojen visualisointi. Kun lokitiedot on saatu kerättyä, suodatettua sekä indeksoitua Elasticsearchiin, voidaan aloittaa lokitietojen analysointi Kibanan web-käyttöliittymällä. Aikaisemmin työssä esiteltiin kolme Kibanan tarjoamaa työkalua, joiden toimintaa käydään läpi tarkemmin.

4.9.1 Discover

Discover-työkalua käytetään siis lokitietojen hakuihin indekseistä, sekä tarkasteluun taulukkomallisessa näkymässä. Työkalulla voidaan hakea indeksin perusteella lokirivejä määritellyltä aikaväliltä. Lisäksi hakutuloksia voi suodattaa erilaisilla hakulauseilla. Kibana käyttää hakujen tekemiseen KQL-kieltä, jolla voi esimerkiksi valita lokirivit näytettäväksi tai piilotettavaksi jonkun tietyn kentän arvon perusteella. Esimerkki hakulauseesta on nähtävissä alla.

```
facility.keyword: "SEC_LOGIN" AND NOT facility_mne    mo
nic.keyword: "LOGIN_SUCCESS"
```

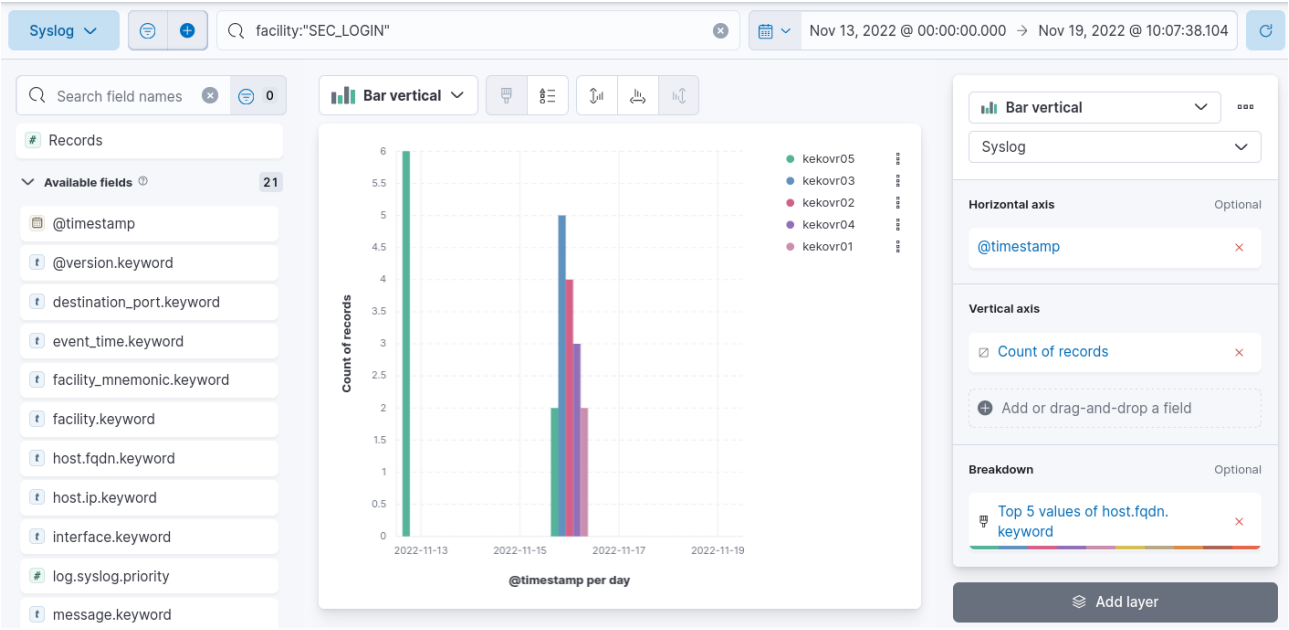
Kyselyllä suodatetaan näkyviin siis ainoastaan ne rivit, joiden "facility" kentän arvo on "SEC_LOGIN", ja noista arvoista suodatetaan pois vielä ne, joiden kentästä "facility_mnemonic" löytyy arvo: "LOGIN_SUCCESS". Kuvio 7. näkyy, kuinka syslog_message kentästä löytyy lisäksi tärkeää dataa, kuten käyttäjätunnus, lähde IP-osoite, sekä kohdeportti, jotka olisivat hyödyllistä erotella omiin kenttiinsä. Kyseisen datan erotteluun käytetty grok-suodatin löytyy liitteestä 6. "21-filter-message.conf" otsikon alta, jossa käyttäjätunnus, lähde IP-osoite ja kohdeportti on eroteltu omiin erillisiin kenttiinsä. Tämä helpottaa kirjautumisyrityksiin liittyvien hakujen rajausta.



Kuvio 7. SSH-kirjautumiset esitettynä discovery-työkalulla.

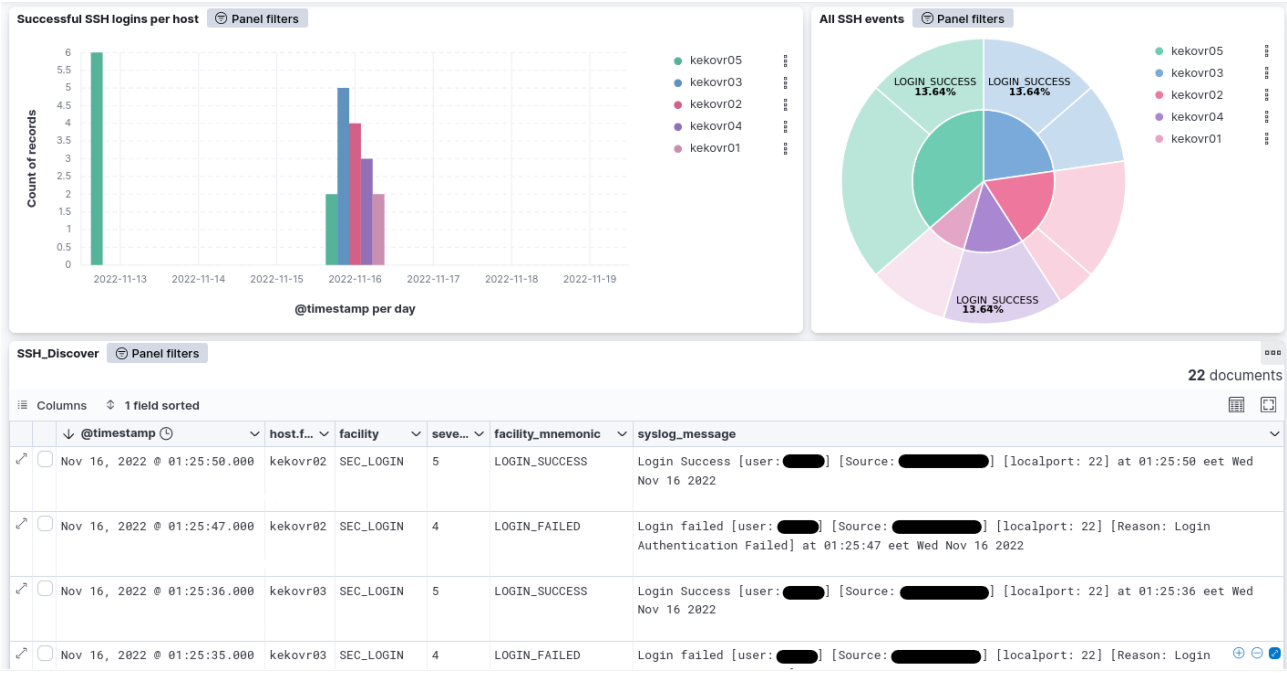
4.9.2 Dashboard

Dashboard-työkalua käytetään datan esittämiseen graafisessa muodossa. Sillä voidaan esittää dataa ymmärrettävässä muodossa, jolloin erilaiset poikkeamat ja muutokset trendissä ovat helposti havaittavissa. Dashboard Create -työkalulla suodatetaan haluttu otanta datasta KQL-hakulauseilla samoin kuin aikaisemmin tehtiin discover-osiossa, jonka jälkeen määritellään visualisoinnin tyyppi. Vaihtoehtoina on monenlaisia pylväs-, viiva- ja ympyräkaavioita, maantieteellisiä karttoja, lämpökarttoja ja taulukoita. Visualisoinnin valinnan jälkeen tulee määritellä data, jonka perusteella visualisointi halutaan luoda. Kuvio 8. on esimerkki yksinkertaisen pylväskaavion luonnista, joka kuvaa kaikki ympäristön SSH-tapahtumat. Ensimmäisenä halutaan suodattaa lokirivit taas facility-arvon perusteella ja sisällytetään visualisointiin ainoastaan ”SEC_LOGIN”-arvon sisältävät lokirivit. Pylväskaavioon määritellään x- ja y-akselien data, jonka perustella kaavio piirretään. X-akseli kuvaa aikajanaa ja esittää tiedon lokirivin aikaleima sarakkeen perusteella, kun taas y-akseli kuvaa kyseisellä haulla löytyvien tapahtumien määrää. Lisäksi saatu data rajataan vielä esitettäväksi ainoastaan viiden eniten tapahtumia sisältävän laitteen osalta.



Kuvio 8. Esimerkki visualisoinnin luomisesta

Kuvio 9. on esimerkki valmiista dashboardista, johon on koottu useampi visualisointi SSH-kirjautuimisiin liittyen. Aiemmin mainitun pylväsdiagrammin viereen on tehty myös ympyrädiagrammi, josta näkee laitekohtaisesti, että kuinka paljon ja minkä tyyppisisiä SSH-tapahtumia on verkossa esiintynyt. Ympyrä näyttää myös prosenttiosuuden onnistuneista ja epäonnistuneista kirjautumisista. Mikäli teksti ei ole mahtunut näytettäväksi diagrammissa, se tulee esiin viemällä hiiren osoitin kyseiseen kohtaan diagrammia. Lisäksi graafien alapuolelle on lisätty taulukko, jossa näkyy tarkemmat tiedot kaikista SSH-tapahtumista.



Kuvio 9. Valmis dashboard-näkymä

4.9.3 Alerts

Alerts-työkalu mahdollistaa hälytyssääntöjen tekemisen lokiviestien perusteella ja hälytysten nostamisen määritellyistä tapahtumista verkossa. Hälytyssääntö voi koostua joko yksittäisestä lokirivistä, joka itsessään on valvonnan kannalta mielenkiintoinen tapahtuma, tai sitten useamman lokirivin yhdistelmästä, jonka muodostama kaava indikoi jostain poikkeavasta. Hälytyssääntöä tehtäessä, tulee määritellä indeksi, johon kyseinen sääntö tekee haut, sekä KQL-kysely, jolla sisällytetään lokiriveistä ainoastaan halutut tulokset. Mikäli hälytyssääntö halutaan muodostaa useamman lokirivin yhdistelemästä tapahtumaketjusta, tulee tulokset ryhmitellä halutun kentän sisältämän arvon perusteella, ja määritellä tapahtumien minimi lukumäärä, jonka toteutuessa hälytyssääntö nostaa hälytyksen. Seuraavissa osioissa on esitelty esimerkit muutaman hälytyssäännön teosta.

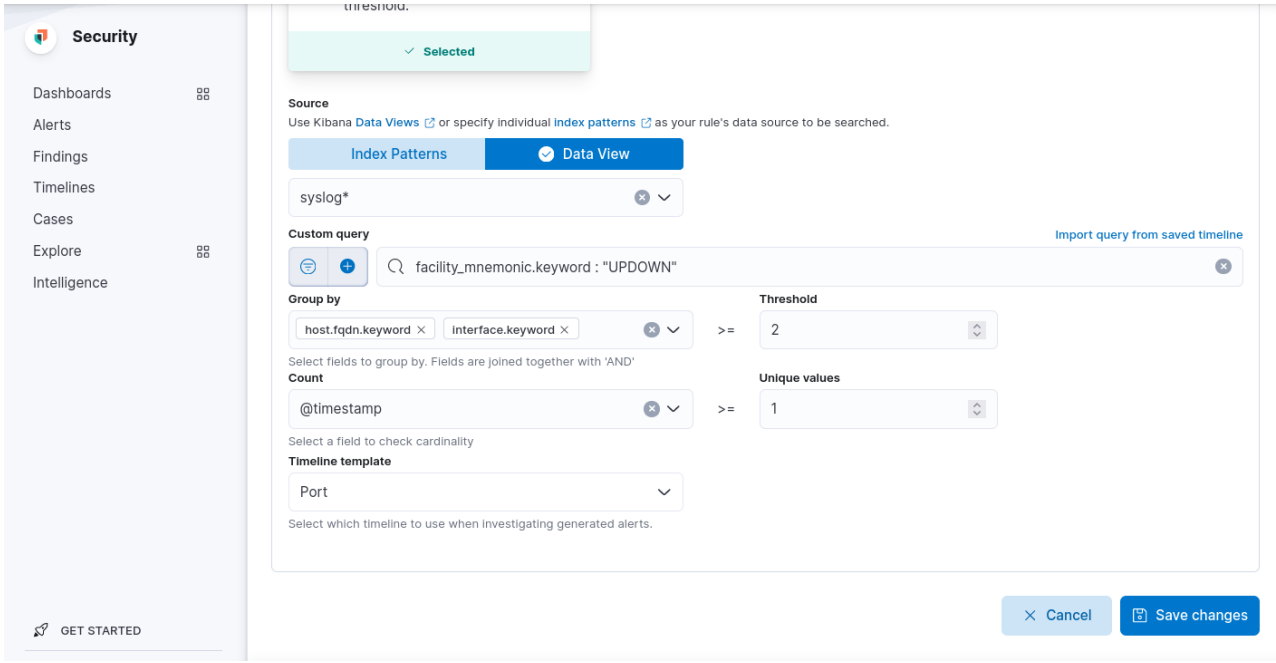
Port flapping -hälytyssääntö

Port flapping -säännöllä seurataan verkkolaitteen yksittäisen portin tilan muutoksia. Sääntö yksinkertaisesti tarkkailee porttien tilan vaihtelua tietyn aikavälin sisällä ja nostaa hälytyksen, mikäli vaihteluiden määrä ylittää määritellyn tason. Jos jonkun portin havaitaan räpyttelevän ylös alas

esimerkiksi 10 kertaa viidessä minuutissa, voidaan nostaa hälytys, jolloin ylläpitäjä osaa alkaa selvittämään tapahtumien juurisyytä.

Hälytyssääntö määritellään koskemaan ainoastaan lokirivejä, jotka koskevat ainoastaan porttien tilanmuutoksia. Sääntöön lisätään hakulause, joka sisällyttää sääntöön ainoastaan ”facility_mnemonic” kentän sisältämän arvon ”LINK”. Tämän jälkeen hälytyssääntö ryhmittelee tuloksena saadut lokirivit sekä laitenimen, että portin numeron perusteella. Näin hälytyssääntö voidaan määritellä hälyttämään silloin, kun hakua vastaavista lokiriveistä löytyy tietty määrä tapauksia yhden laitteen yhdestä portista. Lopuksi lisätään vielä lukumäärä, josta hälytys halutaan nostaa, sekä hälytyssäännön ajon intervalli. Kuvio 10. on esitetty port flapping -hälytyssäännön luonti.

Hälytyssäännön toimivuuden voi todentaa manuaalisesti verkkolaitteen konfiguraatioissa sulkemalla ja avaamalla yhtä porttia, jolloin jokaisesta muutoksesta generoituu uusi rivi. Porttia joutuu avaamaan ja sulkemaan niin monta kertaa, että se täyttää hälytyssääntöön määritellyt ehdot. Esimerkissä on asetettu rajaksi kolme lokiriviä, joten se on helposti toistettavissa ja hälytyssäännön toimivuus saadaan todennettua.



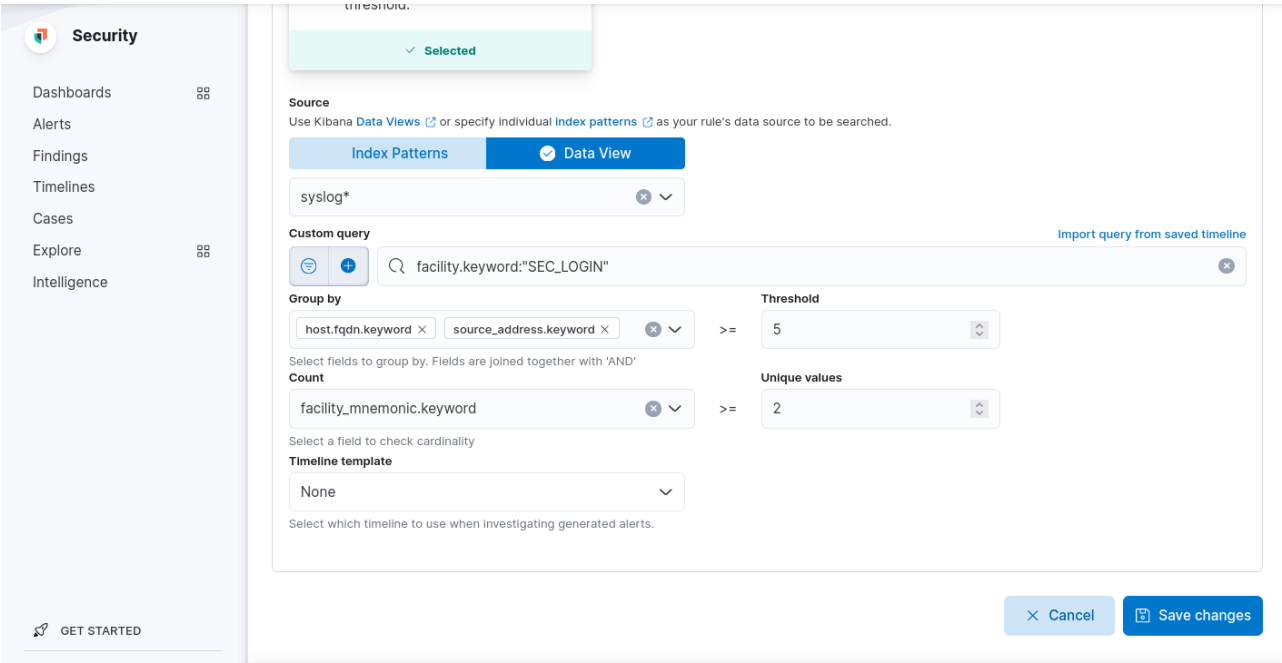
Kuvio 10. Port flapping -säännön määritykset

Brute force -hälytyssääntö

Brute force -säännöllä seurataan kirjautumisyrityksiä, ja yritetään saada kiinni luvattomat kirjautumisyritykset, jossa käyttäjä yrittää salasanan arvaamalla päästä kirjautumaan järjestelmään. Sääntö tarkkailee sekä epäonnistuneita kirjautumisyrityksiä, että onnistuneita kirjautumisia tietyssä aikaikkunassa ja nostaa hälytyksen epäilyttävästä kirjautumiskäytöksestä yhdestä IP-osoitteesta.

Hälytyssääntö määritellään koskemaan ainoastaan lokirivejä, jotka sisältävät tapahtumia, jotka liittyvät SSH-kirjautumisiin. Sääntöön lisätään hakulause, joka sisällyttää sääntöön ainoastaan ”facility” kentän sisältämän arvon ”SEC_LOGIN”. Tämän jälkeen hälytyssääntö ryhmittelee tuloksena saadut lokirivit laitenimen sekä kirjautumisen lähde IP-osoitteen perusteella. Hälytyssääntöön määritellään myös uniikkien arvojen määrä, jolloin ”facility_mnemonic”-kentästä tulee löytyä vähintään kahta eri arvoa. Näin saadaan havaittua tapaukset, jolloin yhdestä IP-osoitteesta on tullut sekä epäonnistuneita, että onnistuneita kirjautumisia. Lopuksi lisätään vielä lukumäärä, josta hälytys halutaan nostaa, sekä hälytyssäännön ajon intervalli.

Hälytyssäännön toimivuus todennetaan yksinkertaisesti ottamalla verkkolaitteeseen SSH-yhteys, syöttämällä paljon epäonnistuneita kirjautumistunnuksia ja lopuksi yksi onnistunut. Jokaisesta yrityksestä generoituessa uusi lokirivi, tulee kirjautumisia tehdä niin useita, että hälytyssäännön määritykset täyttyvät. Esimerkissä on asetettu rajaksi viisi lokiriviä, joista yhden on oltava onnistunut kirjautuminen. Kuvio 11. on esimerkki, jolla havainnollistetaan brute force -hälytyssäännön luominen.

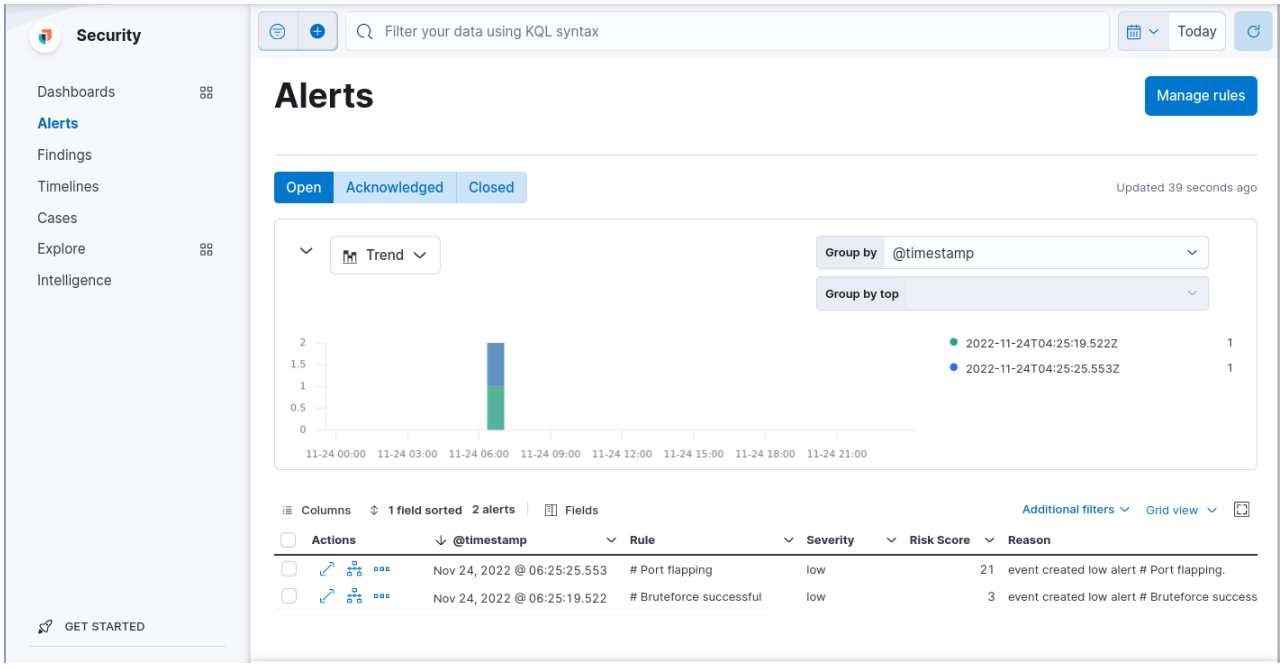


Kuvio 11. Brute force -säännön määritykset

Hälytysnäkymä

Hälytyssääntöjen tunnistamat tapaukset nostavat hälytyksen Kibanan alerts-välilehdelle, josta voi tarkastella ja hallita hälytyssääntöjen nostamia hälytyksiä. Sivuston voi määrittää päivittymään tietyllä aikavälillä, jolloin uudet hälytykset nousevat välittömästi ylläpitäjän nähtäville ja niihin pääs-tään reagoimaan heti. Alerts-näkymä on esimerkiksi kätevä verkonvalvontakeskukseen valvontanäkymäksi, jolloin kyky reagoida verkon poikkeamiin on välitön. Kuvio 12. on esimerkki hälytysnäkymästä, jossa on näkyvillä myös molempien edellä mainittujen hälytyssääntöjen nostamat hälytykset.

Alerts-näkymässä voi siis tarkastella reaaliajassa hälytyssääntöjen tuottamia hälytyksiä, sekä niiden aiheuttamia lokirivejä, jotka ovat avattavissa yhdellä klikkauksella. Sivulla voi myös hallinnoida hälytyksen käsittelyprosessia sekä toimenpiteitä, joita siitä aiheutuu. Hälytyksiä voi myös suodattaa erilaisilla KQL-hakulauseilla, sekä ryhmitellä jonkun kentän perusteella. Esimerkissä hälytykset on ryhmitelty aikaleima-kentän perusteella. Hälytysten trendiä voi tarkastella automaattisesti tuotetun kaavion perusteella, jonka muotoa voi myös vapaasti muokata. Esimerkissä hälytysten trendi on esitetty pylväskaaviona.



Kuvio 12. Alerts-näkymä

5 Tulokset

Työn tavoitteena oli suunnitella ja toteuttaa Puolustusvoimien Johtamisjärjestelmäkeskuksen testiympäristöön toimiva lokienhallintajärjestelmä, joka mahdollistaa verkkolaitteiden lokien keräyksen, säilömisen, sekä tarkastelun ja analysoinnin keskitetyllä palvelulla. Tarkoitus oli keskittää lokien analysointi yhteen työkaluun, joka tarjoaa tehokkaan työkalun verkon tapahtumien seuraamiseen ja verkon vallitsevan tilan seurantaan. Tarkoituksena oli myös luoda muutama automaattinen hälytyssääntö, jotka nostavat hälytyksen valvonnan kannalta tärkeistä, ennalta määritellyistä tapahtumaketjuista.

Lopputuloksena saatiin toteutettua toimiva lokienhallintajärjestelmä, joka vastaa hyvin pitkälti tavoitteita, jotka sille toimeksiannossa asetettiin. Verkkoympäristön verkkolaitteiden lokitieto saadaan kerättyä ja suodatettua lokienkeräyspalvelimella, jonka jälkeen lokitiedot tallennetaan keskitettyyn paikkaan, jossa yhtenäinen lokiviestien tarkastelu on helppo suorittaa lokienanalysointityökalulla. Myös hälytyssääntöjä saatiin toteutettua muutama kappale ja niiden toiminta pystyttiin todentamaan generoimalla tarkoituksellisesti hälytyksen laukaisemiseen tarvittavia lokiviestejä verkkolaitteella. Toteutus luo hyvän pohjan vastaavanlaisen verkkolaitteille koh-

dennetun lokijärjestelmän käyttöönottoon, sekä sen yhteensovittamisen kokonaislokiarkkitehtuuriin myös tuotantoympäristöissä. Vaikka konsepti ja järjestelmä ovat täysin valmiita, sen määritykset vaativat vielä jatkokehitystä.

6 Pohdinta

Suhteessa odotuksiin työ onnistui hyvin ja lopputulokseen voidaan olla erittäin tyytyväisiä. Se mahdollistaa nyt kohdeverkon lokienhallinnan juuri sellaisessa keskitetyssä lokienhallintaympäristössä, kun tavoitteisiin oli asetettu. Työssä selvitettiin lokienhallintajärjestelmän tuottamiseen vaadittavat riippuvuudet ja tekniset määritykset aina valvottavasta lähdejärjestelmästä saakka eri työkalujen konfigurointiin. Se tarjoaa perinpohjaisen läpikäynnin siitä, minkälaisia konfiguraatioita työkalut ja verkkolaitteet vaativat vastaavanlaiseen lopputulokseen pääsemiseksi. Työn perusteella voidaan implementoida toimiva lokienhallintajärjestelmä Cisco-verkkolaitteiden lokien keräykseen Elastic stack -työkalulla.

Järjestelmän toteuttaminen testiympäristöön tarjoaa paljon uusia mahdollisuuksia lokidatan hyödyntämiseen sekä verkonvalvontaan, kun uusia asioita ja tekniikoita voidaan testata turvallisesti tuotantoympäristön ulkopuolella. Ennen suunnittelun aloittamista, toimeksiantajan kanssa oli puhetta eritoten useiden erilaisten hälytyssääntöjen ja käyttötapausten luomisesta, jotka kohdeympäristön mahdollisuuksien takia jäi pienemmälle huomiolle. Hälytyssääntöjä saatiin toki luotua, ja niiden toimivuus todennettua yksinkertaisilla käyttötapauksilla, mutta mitään työssä läpikäytyjä käyttötapauksia ei vielä sellaisenaan voida ottaa käyttöön tuotantoympäristössä. Työtä tehdessä todettiinkin, että tuotantoympäristön kaltaisia tapahtumia ja lokiviestejä on vaikea toistaa ja simuloida testiympäristössä, joten todellisten käyttötapausten tunnistaminen ja testaaminen vaatii myös tuotantoverkon osallisuuden. Työn painopiste siirtyi pelkästään täydellisen lokienhallintajärjestelmän toteuttamisesta, enemmän sen käytäntöjen ja toiminnallisuuksien mahdollisimman kattavaan läpikäyntiin.

Järjestelmän pystyttäminen on kuvattu tässä työssä mahdollisimman yksinkertaisesti, jonka perusteella voisi näyttää, että sen käyttöönotto olisi lyhyt ja helppo prosessi. Todellisuudessa työkalujen ja järjestelmäkokonaisuuden pystyttäminen edellyttää paljon vianselvitystyötä sekä erilaisten virheilmoitusten juurisyiden selvittämistä, joista suurin osa johtui loppujen lopuksi pienistä yksittäi-

sistä konfiguraatiomuutoksista. Esimerkiksi Elastic stack -instanssien välisten yhteyksien pystyttäminen vaati paljon selvittelyä ja vianmääritystä varsinkin salausten suorittavien sertifikaattien osalta. Alun perin työkaluja pystytettäessä otettiin linja, että tietoturvan on koko ajan kehitystyössä mukana, joka lisäsi paljon osaltaan paljon haasteita työkalujen välisten yhteyksien pystyttämisessä. Esimerkiksi todennuksien ja salausten varmistavien sertifikaattien generoiminen ja käyttöönotto työllisti todella paljon.

Lokienhallintajärjestelmä saatiin toteutettua kohdeympäristöön ja toimii juuri niin kuin sen kuuluukin, mutta kohdeympäristön rajoitusten vuoksi itse lokien analysointi, kokonaisvaltainen määrittely, sekä mitä lokitietoja kerätään, jäivät pienemmälle huomiolle. Seuraavassa vaiheessa voitaisiinkin siirtyä jo tutkimaan tuotantoympäristön tuottamia lokiviestejä ja kehittämään hälytyssääntöjä ja visualisointeja niiden perusteella. Lokienhallintajärjestelmä vaatii jatkuvaa ylläpitoa ja kehitystyötä juuri siinä ympäristössä, jossa sitä on konkreettisesti tarkoitus käyttää. Näin ollen järjestelmää ei voi ikinä todeta valmiiksi, eikä sitä voi tietyn pisteen jälkeen kehittää erillisessä testiympäristössä. Järjestelmän implementointi avasi paljon lokienhallintajärjestelmien toimintoja ja lisäsi osaamista, joka auttoi ymmärtämään lokien hallintaan liittyvää prosessia. Tämä on erityisen hyvä tulos tuotantopuolen lokienhallinnan kehitystyötä ajatellen.

Lähteet

Chuvakin, A., Schmidt, K. & Crishtopher, P. 2012. Logging and Log Management: The Authoritative Guide to Understanding the Concepts Surrounding Logging and Log Management. Syngress Publishing. Viitattu 7.9.2022.

Data in: documents and incides. N.d. Elasticsearchin dokumentaatio. Viitattu 2.11.2022. https://www.elastic.co/guide/en/elasticsearch/reference/current/documents-indices.html

Elastic Security for SIEM & security analytics. N.d. Artikkeli elastic.co verkkosivustolla. Viitattu 4.11.2022. https://www.elastic.co/security/siem

Hallberg, J. 2013. Keskitetty lokienhallinta Labranet-ympäristössä. Opinnäytetyö. Jyväskylän ammattikorkeakoulu. Viitattu 11.10.2022. http://urn.fi/URN:NBN:fi:amk-2013121621306

Huang, J. Zhang, M. & Jiang, Y. 2012. The design and implement of the centralized log gathering and analysis system. IEEE sähköinen kirjasto. Viitattu 29.9.2022. https://doi.org/10.1109/CSAE.2012.6272772

Kent, K & Souppaya, M. 2006. Guide to Computer Security Log Management. Recommendations of the National Institute of Standards and Technology, Special Publication 800-92. Viitattu 29.9.2022. https://csrc.nist.gov/publications/detail/sp/800-92/final

Lidster, W. 2022. Network Logging: Definition & Tools. Verkkoluennon materiaali sivustolla study.com. Viitattu 10.9.2022. https://study.com/academy/lesson/network-logging-definition-tools.html

Ljubojević, M., Bajić, A. & Mijić, D. 2018. Centralized monitoring of computer networks using Zenoss open source platform. IEEE sähköinen kirjasto. Viitattu 10.9.2022. https://doi.org/10.1109/INFOTEH.2018.8345528

Logstash Introduction. N.d. Dokumentaatio elastic.co verkkosivuilla. Viitattu 11.10.2022. https://www.elastic.co/guide/en/logstash/current/introduction.html

Lokiohje. 2009. Vahtiohje suomidigi.fi sivustolla. Viitattu 10.9.2022. https://www.suomidigi.fi/ohjeet-ja-tuki/vahti-ohjeet/vahti-32009-lokiohje

Peng, W., Tao, L. & Sheng, M. 2005. Mining Logs Files for Computing System Management. Konferenssijulkaisu. IEEE sähköinen kirjasto. Viitattu 10.9.2022. https://doi.org/10.1109/ICAC.2005.40

Puolustusvoimien johtamisjärjestelmäkeskus. 2022. PVJJK esittelymateriaali. Viitattu 23.10.2022.

Puolustusvoimien johtamisjärjestelmäkeskus. N.d. Keskuksen esittely puolustusvoimat.fi sivustolla. Viitattu 23.10.2022. https://puolustusvoimat.fi/-/puolustusvoimien-johtamisjarjestelmakeskus-viettaa-10-vuotisjuhlaansa

Set up basic security for the Elastic Stack. N.d. Elasticsearchin dokumentaatio elastic.co verkkosivuilla. Viitattu 9.11.2022. https://www.elastic.co/guide/en/elasticsearch/reference/master/security-basic-setup.html

Srivastava, A. 2019. Elasticsearch 7 quick start guide : get up and running with the distributed search and analytics capabilities of Elasticsearch. Packt Publishing. Viitattu 1.11.2022.

Start the Elastic Stack with security enabled automatically. N.d. Elasticsearchin dokumentaatio elastic.co verkkosivuilla. Viitattu 10.11.2022. https://www.elastic.co/guide/en/elasticsearch/reference/current/configuring-Stack-security.html

System Message Logging. 2008. Lokitusohje Ciscon verkkosivuilla. Viitattu 17.10.2022. https://www.cisco.com/c/en/us/td/docs/routers/access/wireless/software/guide/SysMsgLogging.html

Syslog. N.d. Artikkeli Syslogista paessler.com sivustolla. Viitattu 11.3.2022. https://www.paessler.com/it-explained/Syslog

Todd, B. 2017. Creating a logging infrastructure. SANS julkaisu. Viitattu 23.9.2022. https://www.sans.org/white-papers/38130/

Tsunoda, H & Keeni, G. 2014. Managing Syslog. Verkkoartikkeli. IEEE sähköinen kirjasto. https://doi.org/10.1109/APNOMS.2014.6996575

Tutkimuksellinen kehittämistyö. N.d. JAMK Opinnäytetyö ohje. Viitattu 23.10.2022. https://oppimateriaalit.jamk.fi/opinnaytetyo/toteutustavat-ja-rakenne/tutkimuksellinen-kehittamistyo/

What are Beats?. N.d. Elastic dokumentaatio. Viitattu 23.10.2022. https://www.objectrocket.com/resource/what-are-elasticsearch-beats/

What is Log Analysis?. 2022. Artikkeli crowdstrike.com sivustolla. Viitattu 25.10.2022. https://www.crowdstrike.com/cybersecurity-101/observability/log-analysis/

What is Log Retention?. N.d. Artikkeli logicmonitor.com sivustolla. Viitattu 25.10.2022. https://www.logicmonitor.com/blog/what-is-log-retention

Why is log management important?. 2022. Artikkeli greylog.org sivulla. Viitattu 25.10.2022. https://www.graylog.org/post/why-is-log-management-important/

Wilkins, P. 2022. Logging in Action: With Fluentd, Kubernetes and More. New York : Manning Publications Co. Viitattu 29.9.2022.

Yasar, K. 2022. Elastic Stack (ELK Stack). Verkkoartikkeli techtarget.org nettisivulla. Viitattu 11.10.2022. https://www.techtarget.com/searchitoperations/definition/Elastic-Stack

Zhang, S., Meng, W., Bu, J., Yang, S., Liu, Y., Pei, D., Xu, J., Chen, Y., Dong, H., Qu, X. & Song, L. 2017. Syslog processing for switch failure diagnosis and prediction in datacenter networks. Verkkoartikkeli. IEEE sähköinen kirjasto. https://doi.org/10.1109/IWQoS.2017.7969130

Liitteet

Liite 1. Cisco reitittimen konfiguraatiot

```
logging host {logstash IP} vrf net-mgmt transport udp port 515

logging source-interface Loopback vrf net-mgmt

logging trap informational

logging userinfo

login on-success log

login on-failure log

ip ssh logging events

service timestamps log datetime localtime

         archive

                  log config

                  logging enable

                  logging size 1000

                  notify syslog contenttype plaintext

                  hidekeys
```

Liite 2. Ubuntu 22.0.04 LTS asennus

```
apt-get update -y && apt-get upgrade -y

nano /etc/netplan/XX-installer-config.yaml

         network = xx.xx.xx.xx

netplan apply

nano /etc/systemd/timesyncd.conf

         NTP=xx.xx.xx.xx

timedatectl set-timezone Europe/Helsinki

timedatectl set-ntp on

systemctl restart systemd-timesyncd.service

apt-get install openjdk-8-jdk

apt install apt-transport-https
```

Liite 3. Elasticsearchin asennus

```
wget -qO - https://artifacts.Elastic.co/GPG-KEY-Elasticsearch |/
sudo gpg --dearmor -o /usr/share/keyrings/Elasticsearch-
keyring.gpg

echo "deb [signed-by=/usr/share/keyrings/Elasticsearch-/
keyring.gpg] https://artifacts.Elastic.co/packages/8.x/apt stable main" /
| sudo tee /etc/apt/sources.list.d/elastic-8.x.list

apt-get update && apt-get install elasticsearch

systemctl daemon-reload

systemctl enable elasticsearch

systemctl start elasticsearch
```

elasticsearch.yml

```
# Path to directory where to store the data (separate multiple lo-
cations by comma):

path.data: /var/lib/Elasticsearch

# Path to log files:

path.logs: /var/log/Elasticsearch

# Enable security features

xpack.security.enabled: true

xpack.security.enrollment.enabled: true

# Enable encryption for HTTP API client connections, such as
Kibana, Logstash, and Agents

xpack.security.http.ssl:
```

```
enabled: true

keystore.path: certs/http.p12

# Enable encryption and mutual authentication between cluster
nodes

xpack.security.transport.ssl:

enabled: true

verification_mode: certificate

keystore.path: certs/transport.p12

truststore.path: certs/transport.p12

# Create a new cluster with the current node only

cluster.initial_master_nodes: ["elasticmaster"]

# Allow HTTP API connections from anywhere

# Connections are encrypted and require user authentication

http.host: 0.0.0.0
```

Liite 4. Kibanan asennus

```
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch |
sudo gpg --dearmor -o /usr/share/keyrings/elasticsearch-
keyring.gpg

echo "deb [signed-by=/usr/share/keyrings/elasticsearch-
keyring.gpg] https://artifacts.elastic.co/packages/8.x/apt stable
main" | sudo tee /etc/apt/sources.list.d/elastic-8.x.list

apt-get update && apt-get install kibana

systemctl daemon-reload

apt-get start kibana

apt-get enable kibana
```

kibana.yml

```
server.host: "xx.xx.xx.xx"

server.port: 5601



# Enables you to specify a file where Kibana stores log output.

logging:

  appenders:

    file:

      type: file

      fileName: /var/log/kibana/kibana.log

      layout:
```

```
        type: json

  root:

    appenders:

      - default

      - file

# Specifies the path where Kibana creates the process ID file.

pid.file: /run/kibana/kibana.pid



# This section was automatically generated during setup.

elasticsearch.hosts: ['https://xx.xx.xx.xx:9200']

elasticsearch.serviceAccountToken: x

elasticsearch.ssl.certificateAuthorities:
[/var/lib/kibana/ca_1668249584061.crt]
```

Liite 5. Logstashin asennus

```
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch |
sudo apt-key add -

echo "deb https://artifacts.elastic.co/packages/8.x/apt stable
main" | sudo tee -a /etc/apt/sources.list.d/elastic-8.x.list

apt-get update && apt-get install logstash

systemctl daemon-reload

apt-get start logstash

apt-get enable logstash

./usr/share/logstash/bin/logstash-plugin install logstash-output-
lumberjack
```

Liite 6. Logstash masterin pipeline

pipelines.yml

```
- pipeline.id: all

  path.config: "/etc/logstash/conf.d/*.conf"
```

1-input-udp.conf

```
input {

  udp {

    port => 515

    type => “syslog”

    tags => [”cisco”]

  }

}
```

1-input-lumberjack.conf

```
input {

  beats {

    codec => json

    port => 514

    type => ”syslog”

    tags => [”lumberjack”, ”cisco”]

    ssl => true
```

```
    ssl_certificate => "/etc/logstash/lumberjack.crt"

    ssl_key => "/etc/logstash/lumberjack.key"

  }

}
```

20-filter-router.conf

```
filter {

  if "lumberjack" not in [tags] {

    grok {

      match => { "message" =>
"%{SYSLOG5424PRI}%{NONNEGINT:version}:%{CISCOTIMESTAMP:syslog_time
stamp}:   %%{CISCO_REASON:facility}-%{INT:severity_level}-
%{CISCO_REASON:facility_mnemonic}: %{GREEDYDATA:syslog_message}"

      }

    }

  }

}
```

21-filter-message.conf

```
filter {

  if [facility] == "SEC_LOGIN" {

    grok {
```

```
      match => {

         "syslog_message" => "%{CISCO_REASON}
%{SYSLOG5424SD:syslog5424sd} at     %{TIME} %{WORD:timezone}
%{GREEDYDATA:event_time}"

      }

    }

    grok {

      match => {

        "syslog5424sd" => "\[user: %{USERNAME:username}] \[Source:
%{IP:source_address}] \[localport: %{INT:destination_port}]"

      }

    }

  }

  if [facility] == "LINK" {

    grok {

      match => {

        "syslog_message" => "Interface %{WORD:interface},
%{GREEDYDATA:event}"

      }

    }

  }

}
```

22-filter-clean.conf

```
filter {

  mutate {

    add_field => { “[host][fqdn]” =>  “%{[host][ip]}” }

    remove_field => { “[event][original]” }

  }

  dns {

    reverse => “[host][fqdn]”

    action => “replace”

    nameserver => [ “xx.xx.xx.xx” ]

  }

  date {

    match => [ "syslog_timestamp", "MMM  d HH:mm:ss", "MMM dd
HH:mm:ss" ]

  }

}
```

3-output-elasticsearch.conf

```
output {

  elasticsearch {

    hosts => ["https://xx.xx.xx.xx:9200"]

    index => “syslog"

    ssl => true

    cacert => "certs/http_ca.crt"

    user => "elastic"

    password => "xxx"

  }
```

Liite 7. Logstash collectorin pipeline

router.conf

```
input {

  udp {

    port => 515

    type => “syslog”

  }

}
```

```
filter {

  grok {

    match => {

         "message" => "%{SYSLOG5424PRI}%{NONNEGINT:version}:
%{CISCOTIMESTAMP:syslog_timestamp}: %%{CISCO_REASON:facility}-
%{INT:severity_level}-%{CISCO_REASON:facility_mnemonic}:
%{GREEDYDATA:syslog_message}"

    }

  }

}

output {

  lumberjack {

    codec => json

    hosts => [ “xx.xx.xx.xx” ]

    ssl_certificate [ “/etc/logstash/certs/lumberjack.cert” ]

    port 514

  }

}
```