Jouni Hiltunen

LOKIEN LAJITTELU KONEOPPIVIEN JÄRJESTELMIEN AVULLA

JYVÄSKYLÄN YLIOPISTO
INFORMAATIOTEKNOLOGIAN TIEDEKUNTA
2022

TIIVISTELMÄ

Hiltunen, Jouni
Lokien lajittelu koneoppivien järjestelmien avulla.
Jyväskylä: Jyväskylän yliopisto, 2022, 41 s.
Tietojenkäsittelytiede/kyberturvallisuus, Pro Gradu tutkielma
Ohjaaja: Lehto, Martti

Poikkeamantunnistus ja tietoturvapoikkeamien hallinta perustuu järjestelmistä kerättävään tapahtuma- ja lokitietoon. Tietojärjestelmien kasvava käyttö ja monimutkaisuus kasvattaa samalla kertyvää lokia ja sen keräämiseen, järjestelyyn ja analysointiin tarvitaan uusia menetelmiä. Tutkimuksessa analysoitiin lokien käyttökohteita ja pyrittiin löytämään keinoja hyödyntää koneoppivia järjestelmiä lokien järjestämiseksi suunnittelututkimuksen menetelmin. Pyrkimyksenä oli löytää menetelmät ja työkalut, joiden avulla monimuotoisista lokilähteistä kertyvät erimuotoiset lokimerkinnät voidaan ryhmitellä samaan tapahtumaan liittyviksi joukoiksi ennen poikkeamantunnistusta ja sen hallintaa. Tutkimuksessa havaittiin, että tietosuojasäännösten vaatima loki, tietojärjestelmien ylläpidossa käytetty loki ja poikkeamantunnistuksessa seurattu loki asettavat kukin omat vaatimuksensa lokien käsittelyssä käytetylle järjestelmälle. Tietoturvan seurannassa lokimerkintöjä voidaan käsitellä alkiojoukkoina, joiden analysoinnissa tiedonlouhintamenetelmät erityisesti Frequent Pattern Mining ja Frequent Pattern Tree ovat käyttökelpoisia lokimassan toistuvien rakenteiden tunnistamisessa, jotka voidaan syöttää koneoppiville järjestelmille tietoturvapoikkeamien tunnistamiseksi. Lokien keräykseen, esikäsittelyyn ja varastointiin voidaan hyödyntää samoja laskentaresursseja hajautetun tietoaltaan muodossa, joka tarjoaa skaalautuvan ja kustannustehokkaan ratkaisun suurien datamassojen käsittelyyn.

Asiasanat: lokienhallinta, tiedonlouhinta, poikkeamantunnistus

ABSTRACT

Hiltunen, Jouni
Sorting Log Data with Machine Learning
Jyväskylä: University of Jyväskylä, 2022, 41 p.
Cyber Security Masters Thesis
Supervisor(s): Lehto, Martti

Anomaly detection and Securoty Incident Management is done with event and lod data gathered from systems. Constantly growing size, use and xcomplexity in information systems grows the amount of logs formed and new methods are needed to gather, index and analyze them. This study analyzes the log use cases and atempts to find ways to utilize machine learning in log indexing using the design study method. Target was to find methods and tools to group hetereogenerous log entries from different systems by event for anomaly detection and incident management purposes. It was found that audit logs required by regulations, security logs used for Information Security Management and event logs used for system maintenance set different demands for log management system. It was found that log entries can be considered as group of items and therefore analyzed using data mining methods. Frequent Pattern Mining and Frequent Pattern Tree were found useful in identifying recurring patterns in logs. Frequent patterns can be subsequently used as an input for machine learning systems to identify security incindents. Distributed datalake was found practical in gathering, preprocessing and mining logs in large masses.

Keywords: data mining, anomaly detection, log management

KUVIOT

SISÄLLYS

1 Johdanto

Nykyisin organisaatioiden IT-infra on hajautunut paikalliseen, on-premise ympäristöön ja pilvessä oleviin eri tasoisiin palveluihin alkaen virtuaalisista palvelimista aina ohjelmistoon palveluna[1]. Nämä järjestelmät tuottavat palomuuri-, tapahtuma-, virhe- ja seuranta/audit lokeja. Lisäksi lokilähteinä on erinäisiä tietoturva tuotteita kuten virustorjunta, tunkeutumisenhavainta ja hunajapurkkijärjestelmiä, jotka omalta osaltaan tuottavat tila ja tapahtumatietoja.

Pelkästään yksittäinen palomuuri tuottaa lokia 1 Gigatavun päivävauhtia, palomuureja saattaa olla keskisuuressa yrityksessä 10-15 kpl. Eri loki lähteiden tuottama data on muodoltaan erimuotoista, päiväys- ja aikavyöhykkeistä lähtien. Jotta Tietoturvaa valvova ja poikkeamiin reagoivan tahon saama tilannekuva olisi ajantasainen ja todellinen, on sen sekä kerättävä ja analysoitava kaikkien näiden lähteiden tuottama relevantti lokitieto. tiedonsiirron tarvitsema kaistanleveys ja analysointiin tarvittavat ihmisresurssit ovat kuitenkin molemmat rajallisia, lisäksi kaupalliset Security Information and Event Management, SIEM tuotteet hinnoitellaan keräämiensä lokien koon perusteella.

Perinteisessä SIEM-järjestelmässä lokitiedon analysointi ja tunkeutumisenhavainta perustuu sääntöpohjaisiin suodattimiin. Lokidatan määrän kasvaessa ja tietoturvaloukkausten monimutkaistuessa, SIEM toimittajat ovat lisänneet ohjelmiin koneoppivia algoritmeja. Keskitetyn järjestelmän ongelma kuitenkin on se, että se voi analysoida vain saamansa tiedon.

1.1 Motivaatio

Pääosa loki lähteiden tuottamasta datasta on kuitenkin normaaliin toimintaan liittyvää kohinaa, mikäli se voidaan tunnistaa, voidaan sen edelleen lähetys jättää tekemättä. Lokidatan lajittelun mahdollistamiseksi tulisi samaan tapahtumaan, kuten yksittäisen sähköpostin lähetykseen tai vastaanottamiseen liittyvät palvelin, palomuuri ja virustarkistus merkinnät tunnistaa, merkitä ja arvioida. Mikäli

[1] eng. Software as a Service, SaaS

tapahtuma ei liity tietoturvapoikkeamaan voidaan kaikki lokimerkinnät jättää huomiotta SIEM:ssä.

1.2 Keskeiset käsitteet

Tapahtuma on mikä tahansa havaittavissa oleva ilmiö järjestelmässä tai verkossa (Kent & Souppaya, 2006) tapahtumia ovat sekä järjestelmien ja verkon asianmukainen käyttö, että poikkeamat.

Poikkeama on tapahtuma, jonka seurauksena organisaation vastuulla olevien tietojen ja palvelujen eheys, luottamuksellisuus tai tarkoituksenmukainen käytettävyys taso on tai saattaa olla vaarantunut.

Loki on diaari tapahtumista organisaation tietojärjestelmissä tai verkossa (Kent & Souppaya, 2006) Alun perin lokeja käytettiin ongelmanratkaisussa, mutta ne ovat laajentuneet ja palvelevat organisaatioissa useita eri käyttötarkoituksia. Loki koostuu lokitiedoista, jotka ovat dokumentteja jonkin tietyn asian tapahtumisesta jonakin tiettynä ajanhetkenä. Lokitiedot voivat olla automaattisen järjestelmän luomia tai käsin kerättyjä. Valtioneuvoston tietoturvallisuuden johtoryhmän mukaan lokeja kerätään ennalta määriteltyä tarkoitusta varten ennalta määritellyn ajan.(Traficom, 2022)

Käyttöjärjestelmä- ja sovelluslokiin merkitään ohjelmien ja palvelujen suorittamat toimenpiteet, kuten käynnistäminen ja sammuttaminen, virhetilanteet ja merkittävimmät onnistuneet toimenpiteet. Pääkäyttäjät voivat useimmissa käyttöjärjestelmissä ja sovelluksissa määrittää minkä tasoisia tapahtumia ja millä tarkkuudella tapahtumat merkitään lokiin. (Kent & Souppaya, 2006)

Ylläpitoloki kertoo tehdyistä muutoksista, virhetilanteiden hallintaan liittyvistä toimenpiteistä, käyttöoikeuksien lisäämisestä/poistamisesta ja normaaliin ylläpitotoimintaan liittyvistä tarkistuksista ja havainnoista. (Traficom, 2022)

Haltijaloki kertoo kenen hallussa verkkotunniste, kuten IP tai MAC osoite on ollut kullakin ajanhetkellä. Haltijaloki on osa järjestelmien seurantaa ja sitoo käyttäjäidentiteetin ja laiteidentiteetin toisiinsa.

Muutosloki sisältää tiedot järjestelmän tieto sisällön muutoksista, poistoista ja lisäyksistä. Lisäksi muutoslokiin merkitään järjestelmäparametrien ja asetusten muutokset. (Traficom, 2022)

Pääsynvalvontaloki on verkko tason laitteen keräämää tietoa niiden läpi kulkevista yhteyksistä tai verkossa toimivien sovellusten keräämä tieto, mistä on otettu yhteys johonkin suojattuun kohteeseen. Yleensä kirjaa pidetään myös epäonnistuneista yrityksistä ottaa yhteys tai yrityksistä ylittää omat käyttövaltuudet.

Tekoälylle ei löydy yleisesti pätevää määritelmää. Historiallisesti tekoälyn on ajateltu koneellisesti jäljittelevän ihmisen osoittamaa kykyä ajatella, järkeillä, oppia, suunnitella ja kommunikoida. Nykyisessä viestinnässä tekoälyllä viitataan lähinnä joukkoon tiedonlouhinta ja koneoppimismenetelmiä ja erityisesti syväoppiviin neuroverkkoihin.

Koneoppiminen on tekoälyn osa-alue, joka pyrkii syötetyn datan perusteella muuttamaan ohjelmaa siten että se pystyy paremmin suorittamaan jonkin

tehtävä. Koneoppiminen jakaantuu ohjattuun, jossa algoritmi alustetaan koulutus datan avulla, vahvistusoppimiseen, jossa tehtävän ohjelman tuloksia käytetään oppimisalgoritmin säätämiseen ja ohjaamattomaan, jossa datasta ei tiedetä mitään ennalta. (Shai & Shai, 2014)

Neuroverkko on koneoppimisen osa-alue, joka jäljittelee biologista hermo verkkoa, joka oppii tunnistamaan hahmoja sille esitellyn opetusaineiston perusteella. Neuroverkossa on joukko painotettuja syötteitä, jotka annetaan keinotekoisille neuroneille, jotka laskevat funktionsa mukaan syötteelle tuloksen. Tulokset käytetään joko sellaisenaan tai syötetään uusille neuroneille. Useampia neuronikerroksia sisältävää verkkoa kutsutaan syväksi neuroverkoksi ja sen tekemään oppimista kutsutaan syväoppimiseksi.

1.3 Tutkimuskysymys

Voiko koneoppiva järjestelmä tunnistaa ja merkitä samaan tapahtumaan liittyvät, eri lokilähteistä saapuvat merkinnät?

1.3.1 Hypoteesit

Hypoteesi 1: Eri lähteistä saapuvista lokimerkinnöistä irrotetut piirteet voidaan koneoppivan algoritmin perusteella lajitella samaan tapahtumaan liittyviksi ryhmiksi.

2 Tutkimusmenetelmä

Tutkimusmenetelmäksi valittiin suunnittelututkimus (Peffers;Tuunanen;Rothenberger;& Chatterjee, 2007), jonka tavoitteena on muodostaa malli, jonka avulla strukturoimatonta lokidataa voidaan analysoida koneoppivin järjestelmin. Tutkimus tehdään osana Fujitsu Finland Enterprise and Cyber Security osaston tarjoamankehitystä

2.1 Ongelman kuvaus ja perustelu.

Tavoitteena on löytää tehokkaampia keinoja käsitellä SIEM palvelulle saapuvaa lokidataa. Keskisuurissa ja suurissa organisaatioissa syntyvän lokidatan määrä on arviolta useita teratavuja vuorokaudessa. Tietoturvapolitiikasta, ylläpidon vaatimuksista ja lainsäädännöllisistä vaatimuksista, lokitietoa joudutaan säilyttämään jopa vuosia. Lokitiedon käsittelyn tehostaminen tarjoaa merkittäviä kustannussäästöjä ja mahdollistaa nopeamman ja tarkemman reagoinnin tietoturvauhkiin.

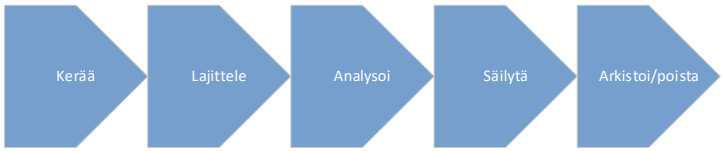
2.1.1 Lokien hallinnan merkitys kybertuvallisuudelle

Fujitsu Finland Enteprise and Cyber Security tarjoaa asiakkailleen tietoturvatapahtumien ja -loukkausten havainta ja hallintapalvelua (SIEM Service), jonka tarkoitus on parantaa tilannekuvaa ja parantaa asiakkaan mahdollisuuksia ymmärtää kohtaamiaan tietoturvariskejä ja tekemään tietoon perustuvia päätöksiä asian vaatimista toimista. SIEM Serviceä käytetään tarkkailemaan lokeja ja muita datalähteitä ja hallinnoimaan tietoturvapoikkeaman elinkaarta palvelun puitteissa. Palvelu tarjoaa tietoturvapoikkeaman luokittelun väärien hälytysten vähentämiseksi, jotta voidaan keskittyä tärkeysjärjestyksessä korkealla oleviin tapahtumiin. Tapahtumat tiketöidään poikkeamanhallintaprosessissa määritellyn muutoksenhallintajärjestelmän kautta Fujitsun tietoturva-analyytikolle, joka arvioi poikkeaman ja reagoi siihen poikkeaman vakavuuden vaatimalla tavalla. Matalan tason poikkeamissa ohjeistetaan käyttäjätukea tarvittaviin toimiin poikkeamasta palautumiseksi esim. havaittaessa tunnettuja haittaohjelmia tai käyttäjän klikatessa pahantahtoista linkkiä. Vakavammissa poikkeamissa tapahtuman tutkimus ohjataan analyytikolle, joka diagnosoi sen käyttäen MITRE ATT&C (MITRE, 2021) mallin mukaisesti, rajaa poikkeaman vaikutuksen ja aloittaa toimet siitä toipumiseksi. Poikkeamia ja niihin liittyvää hallintaa pyritään automatisoimaan SOAR[2] tuotteiden avulla erityisosaamista ja kokemusta vaativan ihmistyön vähentämiseksi. Laajoissa ja kriittisiin järjestelmiin kohdistuvissa

[2] eng. Security Orchestration and Automation

poikkeamissa käynnistetään MIM [3] prosessi, jossa kerätään erillinen tiimi, jolle alistetaan tarvittavat resurssit poikkeamasta palautumista varten. Esimerkkinä MIM prosessista on Lahden Kaupungin tietoverkkoon vuonna 2019 kohdistunut kyberhyökkäys, jolla oli vakavia vaikutuksia sosiaali- ja terveyspalveluihin. (YLE, 2019) Edistyneiden ja teknisesti taitavien APT-hyökkääjien tapauksessa voi olla syytä epäillä tietomurtoa, vaikka käytettävissä ei ole selvää tunnusmerkistöä, joka viittaa poikkeamaan. Tällöin asiakas antaa analyytikolle toimeksiannon uhkien metsästykseen[4], jonka prosessissa muodostetaan kaksi hypoteesia, poikkeamahypoteesi ja siihen liittyvät indikaattorit sekä oletushypoteesi ts. poikkeamaa ei ole tapahtunut. SOC palvelun analyytikko tutkii poikkeaman indikaattorien esiintymisen kohdejärjestelmässä ja mikäli indikaattorit vahvistavat poikkeaman käynnistetään havainnoista poikkeamanhallinta prosessi. Muussa tapauksessa analyytikko päätyy vahvistamaan oletushypoteesin ja raportoituaan arvioimansa haavoittuvuus tason uhkaa vasten ja annettuaan mahdolliset toimenpidesuositukset sulkee toimeksiannon. Esimerkkinä uhkien metsästyksestä on Solarwinds toimitusketjuhyökkäyksen (Center for Internet Security, 2021) jälkeen suoritetut kartoitukset asiakkaiden ympäristöissä mahdollisen tietomurron varalta. Uhkien metsästyksessä analysoidaan suuria määriä lokitietoa pitkältä aikaväliltä tukeutuen usein muihin kuin tietoturvalokeihin.

Lokit kerätään ohjelmistoista, verkkolaitteista ja tietoturvan hallintaa varten asennetuista järjestelmistä, kuten virustorjunta, palomuuri, hunajapurkki tai tunkeutumisen havaintajärjestelmät. Lokit kerätään kohteen verkkoympäristöön asennetulle SIEM palvelimelle, jolle asennettu ohjelmisto kykenee vastaanottamaan tiedosto ja binäärilokeja sekä hakemaan tiedostomuotoisia lokeja verkkopalveluista. Vastaanotetut lokit indeksoidaan halua varten ja ennalta määriteltyjen sääntöjen mukaan havaituista virhetilanteista ja poikkeamista nostetaan ilmoitus, jonka päivystävä analyytikko arvioi, luokittelee ja antaa ohjeet poikkeaman hallinnasta. Matalan tärkeys tason poikkeamista ilmoitetaan sähköpostitse ja tapahtumasta avataan tiketti palvelua tilaavan organisaation tiedonhallintajärjestelmään ja jatko toimenpiteet jätetään IT-lähituelle. Korkean tärkeys tason vakavissa tietoturvapoikkeamissa tapahtumista käynnistetään erikseen sovittujen käytäntöjen mukaisesti prosessi tietoturvaloukkausten hallitsemiseksi, jossa tietoturvaloukkausten hallinta- ja johtotiimeille ilmoitetaan ja tilanne eskaloidaan tarpeen mukaisesti.



Kuva 1. Lokitiedon käyttö SIEM-järjestelmässä

[3] eng. Major Incident Management
[4] eng. Threat Hunting

Tietoturvapoikkeamien lisäksi lokitietoja käytetään kuormituksen, suorituskyvyn, saatavuuden, toimintahäiriöiden ja ohjelmistolisenssien käyttöasteen seuraamiseksi. Näihin tarkoituksiin käytettäviä lokeja ei tyypillisesti seurata reaaliaikaisesti eikä SIEM-järjestelmien kautta. Lokien hallintajärjestelmä ja SIEM-järjestelmä ovat usein päällekkäisiä tai rinnakkaisia järjestelmiä. Toista tai molempia voidaan käyttää tietosuojaloukkausten todisteiden keräämisessä tai käyttäjän oikeusturvan turvaamisessa silloin, kun käsitellään yksityisyyden suojan tai turvaluokitetun tiedon käsittelyä.

Tiedonlouhintaa hajautetuista tietoturvalokeista on tutkittu poikkeamantunnistuksessa (Shu;Smiy;Yao;& Lin, 2013) (Jakrarin & Primsopa, Applying Hadoop for log analysis toward distributed IDS, 2013). Mikäli valvonnasta ja vasteesta huolehtiva taho joutuu toimimaan vain SIEM-palveluun toimitettavan lokitiedon varassa, riippuu vasteen nopeus toimitettavan lokitiedon tuoreudesta ja kattavuudesta. Hajautetun lokitiedon louhinnalla saadaan todennäköisesti suurin hyöty etsittäessä tietoturvaloukkauksia, joiden havaitsemiseen reaaliaikaiset havainta- ja hälytyspalvelut eivät sellaisenaan sovellu. Esim. Fireeye analyysin perusteella valtiollisten toimijoiden suorittamat, kohdistetut kyberturvaloukkaukset [5], jatkuvat 56 päivästä 141 päivään. (Mandiant, 2021)

IT-järjestelmän jokainen komponentti muodostaa lokia jokaisesta tapahtumasta asetustensa mukaisesti. Poissulkien säädösten vaatima lokien säilytys, kaikkien lokien siirtäminen SIEM järjestelmään ei kuitenkaan ole tarkoituksenmukaista, sillä ne eivät ole käyttökelpoisia tietoturvapoikkeaman analysoinnissa ja siitä toipumisessa. Laajojen IT-ympäristöjen tuottamien lokien määrä, niiden siirtoon vaadittava kaistanleveys ja SIEM järjestelmien kapasiteetti ottaa vastaan ja indeksoida lokimerkintöjä asettaa käytännön rajoituksia. Asiakkaan käytössä olevien kaupallisten lokijärjestelmien kustannukset koostuvat lisenssikustannuksista, lokien käsittelyyn ja säilytykseen vaadituista fyysisistä resursseista ja järjestelmän itsensä ylläpito- ja päivitystyöstä.

2.1.2 Lokien ominaisuudet

Tiedon turvaominaisuuksiksi määritellään tavallisimmin luottamuksellisuus, eheys ja kokonaisuus. Määritelmät eivät sinällään ole käyttökelpoisia lokien käsittelyssä. Pääosin muodostettu lokitieto ei ole luottamuksellisuutensa suhteen luokiteltua ja lokitietojen saatavuusvaatimukset riippuvat niiden käyttötarkoituksesta ja lokitietojen ajallisesta tuoreudesta. Tutkimuksessa lokitiedon ominaisuuksista käytetään määritelmiä *ajantasaisuus, eheys ja kokonaisuus*. Ominaisuudet painottuvat eri tavoin eri tarkoituksiin kerättävien lokien mukaan.

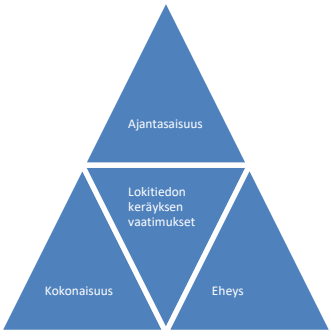
Ajantasaisuus mittaa viivettä lokimerkinnän muodostamisen ja sen lajittelun välillä. Ajantasaisuuteen vaikuttavat lokien siirron tiheys, siirrossa esiintyvä viive ja lokitiedon lajitteluun eli indeksointiin kuluva aika. Tiheämmin ja pieninä

[5] eng. Advanced Persistent Threat APT

osina tehtävä siirto ja lajittelu vaatii enemmän verkon kaistanleveyttä ja laskenta tehoa, kuin harvemmin ja suurempina erinä käsitelty.

Eheys mittaa eroa lokimerkinnässä sen luomisen ja sen säilytyksen välillä. Lokitiedon tietosisällöstä ja sen käyttötarkoituksesta riippuen voidaan lokimerkintää käyttää kokonaisuudessaan tai osittain. Esim. verkkopalvelimen liikennelokeja analysoitaessa voidaan hyödyntää vain lähde IP-osoitetta, tällöin voidaan lokimerkintä ottaa vastaan sellaisenaan kuin se saapui, mutta indeksoida ja säilyttää vain haluttu tieto. Tietoturvahyökkäyksissä hyökkääjä pyrkii usein joko tuhoamaan tai muuttamaan lokitietoja niiden säilytyspaikassa tai ääritapauksessa siirron aikana. Riskejä lokitiedon eheydelle säilytyksessä voidaan pienentää pääsynhallintaa tehostamalla ja parantamalla ajantasaisuutta, siirrossa taas voidaan pienentää protokollalaajennuksilla tai IPSEC toteutuksella, mutta niissä tapauksissa, joissa lokia kerätään käyttäjän oikeusturvan takaamiseksi, on syytä käyttää lokien siirtoon keinoja, joissa todisteketjun koskemattomuus ja eheys voidaan varmentaa.

Kokonaisuus mittaa kaikkien syntyneiden lokimerkintöjen ja kaikkien järjesteltyjen lokimerkintöjen välistä suhdetta. Lokitietojen suurempi kokonaisuus mahdollistaa kattavamman ja tarkemman analyysituloksen, mutta kuten ajantasaisuus on myös kokonaisuus kompromissi käytettävissä olevien resurssien suhteen. Hyökkääjän suorittaman tahallisen lokitiedon tuhoamisen lisäksi riskin lokitiedon kokonaisuudelle muodostaa tiedon korruptoituminen säilytyspaikassa ja verkko-ongelmat. Yleisimmin käytössä olevat lokien siirtoprotokollat kuten Simple Network Management Protocol ja BSD syslog, siirtävät yksittäisiä tapahtuma merkintöjä sitä mukaa kuin ne syntyvät UDP:n yli. Siirron onnistumista tai lokitiedon eheyttä ei voida taata protokolla tasolla. Oletuksena ylläpitoloki säilötään paikallisesti tiedostoon järjestelmässä, joka sen on luonut ja joissakin tapauksissa se siirretään edelleen käsiteltäväksi tiedostojen siirtoon tarkoitetuilla protokollilla kuten FTP tai SFTP Siirron onnistumisen varmistamiseksi verkko tason keräimet kahdennetaan ja niiden eteen asennetaan kuormantasaajat.



Kuva 2. Lokitiedon ominaisuudet

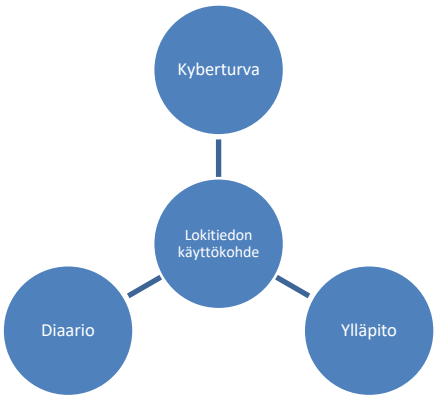
2.1.3 Lokien keräystarkoitus

Tarjous- ja tietopyynnöissä kuvattujen vaatimusten perusteella lokien käsittelyn tarve keskitetyssä lokienhallintajärjestelmässä jaetaan tässä tutkimuksessa kolmeen.

Diaarioloki kerätään joko lakisääteisesti, asetusten niin vaatiessa, organisaation varautuessa toimiensa rikostekniseen tutkintaan tai muuten muodostaakseen sellaisen seuranta jäljen, josta määrätty toiminta voidaan myöhemmin luotettavasti arvioida ja varmentaa. Diaariolokia kerättäessä on otettava huomioon, että se todennäköisesti sisältää yksilöiviä henkilötietoja, jolloin on noudatettava erityistä huolellisuutta sen luottamuksellisuuden ja eheyden säilyttämisessä sekä siirron, että varastoinnin aikana. Diaariolokia ei voida lähtökohtaisesti anonymisoida, koska tapahtuma ja sen suorittajan tulee olla myöhemmin tunnistettavissa. Organisaation loki- ja tietoturvapolitiikan niin salliessa loki voidaan mahdollisesti pseudonymisoida, eli henkilötietojen käsittelemistä siten, että henkilötietoja ei voida enää yhdistää tiettyyn rekisteröityyn käyttämättä lisätietoja, edellyttäen että tällaiset lisätiedot säilytetään erillään ja niihin sovelletaan teknisiä ja organisatorisia toimenpiteitä, joilla varmistetaan, ettei henkilötietojen yhdistämistä tunnistettuun tai tunnistettavissa olevaan luonnolliseen henkilöön tapahdu (Traficom, 2021). Diaariolokin tärkeimmät ominaisuudet ovat lokitiedon kokonaisuus ja eheys, diaariolokin ajantasaisuudesta voidaan jossain määrin tinkiä, koska diaariolokia ei käytetä ajantasaiseen toimintaan.

Seurantaloki kerätään järjestelmien, sovellusten ja verkkolaitteiden suorituskyvyn arviointiin, virhetilanteiden havaitsemiseen ja niistä toipumiseen sekä lähteeksi raakadatalle tiedonlouhintaan toiminnan kehittämistä varten. Seurantalokia muodostettaessa ja käsiteltäessä on huolehdittava siitä, ettei niistä muodostu henkilörekisteriä ts. seurantalokissa ei käsitellä yksilöiviä henkilötietoja. Seurantalokin tärkein ominaisuus on sen kokonaisuus ja toissijaisesti ajantasaisuus. Seurantalokin eheydestä voidaan tinkiä sikäli, että sen käyttötarkoituksiin voidaan käyttää myös lokitiedosta muodostettua analysoitua ja rikastettua tietoa.

Tietoturvaloki kerätään haittaohjelmien torjuntaohjelmista, tunkeutumisenhavaintajärjestelmistä, pääsynhallinnasta, käyttöjärjestelmien ja sovellusten tietoturvailmoituksista sekä paketti- ja sovelluspalomuurien liikennetiedoista. Tietoturvalokia käytetään tietoturvapoikkeamien havainnointiin ja niiden selvitykseen ja niistä toipumiseen. Tietoturvapoikkeamien hallinnassa aikainen havainnointi, kattava poikkeaman laajuuden selvitys ja toipumisen varmentaminen vaatii, että lokitieto on mahdollisimman ajantasaista, kattavaa ja kokonaista sekä eheää. Keskitetyn lokienhallinnan vaatimukset määrittävät kaikissa tapauksissa ensisijaiseksi tavoitteeksi tietoturvalokin tunnistamisen ja sen ohjaamisen SIEM:lle.



Kuva 3. Lokitiedon käyttökohteet

2.2 Tavoitteiden asettelu

Lokienkeräysjärjestelmän tavoitteet toteutettiin osallistumalla osana Fujitsu Finland Enterprise & Cyber Securityn tarjoustiimiä kahteen julkiseen tarjouskilpailuun (Helsingin ja Uudenmaan Sairaanhoitopiiri, 2021) (Helsingin kaupunki, 2021), yhteen yksityiseen tarjouspyyntöön ja yhteen yksityisyyteen tietopyyntöön lokien käsittelystä ja säilytyksestä. Tarjouksissa analysoitiin jatkuvasti kasvavan lokimassan vaikutus nykyisin käytössä olevan teknologian kustannuksiin ja kerättiin tekniset suorituskykyvaatimukset koneoppivien menetelmien hyödyntämiseen tulevissa kilpailuissa. Lisäksi suunnittelun pohjana käytettiin olemassa olevista asiakassuhteista kertynyttä tilastotietoa ja niissä käsiteltyjä palvelupyyntöjä.

Tarjouspyynnöissä havaittiin yhteisinä piirteinä IT järjestelmien jatkuva kasvu ja monimutkaistuminen, kasvava pyrkimys hyödyntää julkisia pilvipalveluita, kuten Microsoft Office 365 ja pyrkimys mitoittaa lisenssit ja käytettävät resurssit todellisen käytön mukaisesti. Kaikissa tapauksissa asiakkailla on kuitenkin käytössään laajat konesali ja työasemaympäristöt menneisyydessä tehtyjen hankintojen vuoksi. Kahdessa tarjouspyynnössä asiakkaalla oli IT-ympäristöstä erillinen prosessi- ja automaatioinfrastruktuuri, joka asetti rajoituksia lokitietojen keräykselle. Lähtötietojen pohjalta järjestelmälle määrättiin seuraavat toiminnalliset ja suorituskykyvaatimukset:

1. Järjestelmän tulee tukea Fujitsu Finland Enterprise & Cyber Security SOC palvelun käyttämiä SIEM työkaluja.
2. Järjestelmän tulee kyetä keräämään ja käsittelemään minimissään 2 Teratavua tekstimuotoista lokia vuorokaudessa.
3. Järjestelmän tulee kyetä keräämään ja käsittelemään lokia konesaleista, jotka on maantieteellisesti hajautettu.
4. Järjestelmän tulee kyetä keräämään lokia virtuaalisista ympäristöistä ja pilvipalvelualustoilta.

5. Järjestelmä tulee olla varmennettavissa siten, että yksittäisen komponenttien rikkoutuminen vaaranna lokitietojen keräystä tai siihen kerättyjä lokitietoja.

2.3 Suunnittelu ja kehitys.

Kaistanleveyden säästämiseksi, käytännöllisin vaihtoehto lokitietojen keräämiseksi on paikallisverkossa sijaitseva lokipalvelin, joka vastaanottaa raakalokia joko vastaanottamalla lokien siirtoon tarkoitettuja protokollia, kuten SNMP tai syslog ja hakemalla tai vastaanottamalla tiedostoja.

Kun lokimerkintä on vastaanotettu, voidaan suorittaa sen esilajittelu, jossa tunnetut, ajantasaisuutta vaativat tietoturvamerkinnät lähetetään välittömästi edelleen SIEM palvelimelle. Eheyttä vaativat lokimerkinnät on myös syytä toimittaa SIEM:lle jossa ne voidaan arkistoida hallitusti. Käytännössä esilajittelussa tietoturva- ja diaarioloki toimitetaan välittömästi eteenpäin ja jäljelle jääneet ylläpitolokimerkinnät kirjoitetaan paikalliseen tiedostojärjestelmään jatko analyysia varten, esilajitteluun voidaan käyttää säännöllisiin lausekkeisiin perustuvaa sääntöjoukkoa, jonka viimeisenä sääntönä on käsky säilöä lokimerkintä paikallisesti. Järjestelmistä kerätty lokidata on puolirakenteista, koska niillä on ennalta määrätty rakenne kenttineen, käytännössä lokien keräyspalvelimista muodostuu hajautettu tietoallas, jolloin kertyneeseen lokimassaan voidaan soveltaa hajautetun tietoaltaan tiedonlouhinnassa käytettyjä teknologioita ja menetelmiä. Näin ollen, kaikkia toisiinsa liittyviä lokimerkintöjä ei edes tarvitse siirtää SIEM järjestelmään, SIEM palvelimella olevat ja tietoaltaassa olevat lokimerkinnät voidaan yhdistää toisiinsa hajautetulla laskennalla ja tietoturvapoikkeamien tutkinta suorittaa tekemällä hakuja.

2.3.1 Aiempi tutkimus

Boulat Chainourovin mukaan Splunk kykenee havaitsemaan aikaleimaan ja tapahtumatunnukseen perustuen tilastolliset anomaliat satojen kiintolevyjen Hadoop toteutuksesta ja tarjoaa kustannustehokkaan vaihtoehdon suurien tietomäärien siirtämiseen SIEM palvelimen ja Hadoop-järjestelmän välillä. (Chainourov, 2017)

Teknisesti tiedonkäsittelyn prosessi SOC/SIEM:ssä noudattelee tiedonlouhinnan periaatteita siten kuin Jiawei et. al. ne esittävät. SIEM poistaa saadusta lokimassasta epäolennaisuudet ja epäjohdonmukaisuudet, se muuntaa yhdistää ja muuntaa datan louhittavaan muotoon ja esittelee louhinnan tulokset analyytikolle arviointia varten. (Kamber;Han;& Pei, 2011, s. 7). Kirjassa erotellaan louhinnan tietovarastot yhdistämättömiin, löyhästi, puolitiukasti ja tiukasti yhdistettyihin, riippuen siitä kuinka paljon tiedonlouhintajärjestelmä käyttää tietovaraston toiminnallisuuksia louhintatyössä. (Kamber;Han;& Pei, 2011, ss. 34-37)

Jakrarin ja Primsopa tutkivat anomalian tunnistusta K-Means algoritmilla, joka oli toteutettu Hadoop MapReduce puitejärjestelmän avulla tunkeutumisen havainnan kannalta. Tunnusmerkistöön perustuvasta tunkeutumisen havainnasta poiketen, anomalian tunnistamisella kyetään havaitsemaan myös sellainen toiminta, jota ei ole ennen kyetty erottelemaan pahantahtoiseksi. Tutkimuksessa vertailtiin K-Means klusterointi-algoritmin suorituskykyä verrattuna yleisesti käytössä oleviin, tunnusmerkistöön perustuviin lokianalysaattoreihin ja havaittiin sen olevan tehokkaampi suurten lokimassojen analysoinnissa. (Jakrarin & Primsopa, 2013)

Verkkopalvelinten liikennelokeista louhitaan assosiaatiosääntöjä käyttäjien toimintakaavojen löytämiseksi. Sipola et.al. analysoi Apache-verkkopalvelimen liikennelokeja muuttamalla polkujen dynaamiset osat 2-grammeiksi ja muodostamalla lokeista matriisin, jonka rivit olivat yksittäisiä lokimerkintöjä ja sarakkeet löydettyjen 2-grammien frekvenssi. Ulottuvuuksien pienentämisen jälkeistä matriisia analysoitiin klusterointi-algoritmilla anomalioiden löytämiseksi. Tutkimuksessa käytettiin spektriklusterointia, mutta kirjoittajat teorisoivat myös K-Means- ja tiheysalgoritmit ovat käyttökelpoisia. Tutkimuksessa klusterointi tunnisti kaikki yritykset manipuloida verkkosovelluksista http-haittapyynnöillä anomalioiksi. (Juvonen;Sipola;& Lehtonen, 2011). Iváncsy & Vajk tutkivat tiedonlouhintaa toistuvia kaavoja tunnistamalla verkkopalvelinten liikennelokeista. Liikennelokien tiedonlouhinnan tarkoituksena on löytää selaajien käyttötottumuksia markkina-, käytettävyystutkimukseen. Verkon selauskäyttöä analysoidessa etsitään käyttäjien usein selaamia sivuja, käyttäjien selaamaa sivujärjestystä ja puurakenteita sivujen selausjärjestyksessä. Kaikille kolmelle on oma algoritminsa. Tutkijoiden mukaan usein selattujen sivujen analysointiin kehitetty algoritmi löytää tehokkaasti pienet usein toistuvat joukot tehostaen myös suurempien joukkojen löytymistä. (Iváncsy & Vajk, 2006)

2.3.2 Jäsentely ja lajittelu

Lokimerkintöjen jäsentely ja merkintä samaan tapahtumaan liittyviksi ryhmiksi. Analyytikon suorittamassa jäsentelyssä lokit ryhmitellään aikaleiman, lähde/kohde IP-osoitteen, protokollan, lokin luoneen sovelluksen ja lokisisällön perusteella tapahtumiin liittyviksi ryppäiksi. Kehitys- ja testausvaiheen tarkoituksena on löytää koneoppiva järjestelmä, joka kykenee tekemään saman joko ohjatusti tai täysin itsenäisesti. Tavoitteena on valmistella rypäs seuraavalle tekoälylle, joka tekee päätöksen siitä, poikkeaako lokirypäs järjestelmän normaalista toiminnasta siinä määrin, että se on syytä lähettää SIEM järjestelmälle joko poikkeamantunnistusta tai ongelmanselvitystä varten.

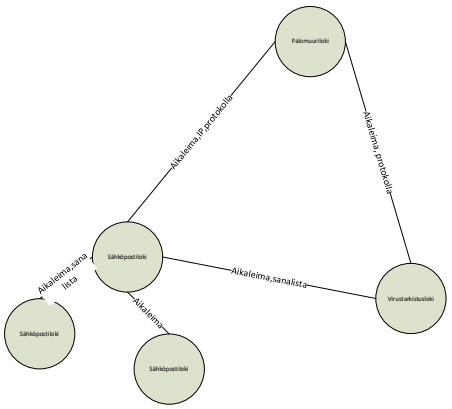
Lokimerkintä koostuu tapahtumatiedoista, joten jokainen tapahtumatieto on alkio, joka kuuluu joukkoon A. Kaksi vuorovaikutuksessa olevaa prosessia tuottaa lokimerkinnät A ja B, jotka sisältävät saman tapahtumatiedon, käytännössä ainakin aikamerkinnän, mahdollisesti joukon muita tunnistettuja piirteitä. Toisiinsa liittyvät lokimerkinnät voidaan ilmaista joukkona C, joka on joukkojen A ja B unioni.

$$A \neq B, A \cap B \neq \emptyset \text{ ja } A \cup B = C$$

Kertynyttä lokimassaa voidaan siis louhia päämääränä tunnistaa kaikki assosiaatio säännöt:

$$A \rightarrow B$$

Assosiaatiosääntöjen avulla lokimerkintöjä voidaan tutkia graafeina, jonka solmuina on yksittäiset lokimerkinnät ja kaarina tietorakenteesta löytyvät vastaavuudet.



2.3.3 Teknologiavalinnat

Fujitsu Finland Oy:n Enterprise & Cyber Security yksikön tarjoama SIEM/SOC palvelu rajoittaa tarjoamansa SIEM alustat kaupallisesti tuettuihin ohjelmistoihin. Käytännössä uusia järjestelmiä tarjotaan joko konesaleihin rakennettavilla Splunk ohjelmistoilla tai Microsoftin Azure ympäristöihin integroituun Azure Sentinel pilvipalvelulla. SIEM palvelut integroidaan tietohallintojärjestelmään [6], jonka avulla havaitut poikkeamat ja niihin vastaaminen on osa asiakkaan tietohallintaprosessia. Chainourovin tutkimuksen jälkeen kyseiseen integraatioon on kehitetty Splunk Hunk joka mahdollistaa Hadoop järjestelmässä säilytetyn datan indeksoinnin ja hakujen suorittamisen. Splunkin Enterprise lisenssi mahdollistaa myös siinä olevien tekoäly omaisuuksien käytön datan hyödyntämisessä.

2.3.4 Tietovarasto

Hajautetun tietovaraston totetuttamistavaksi arvioitiin NFS verkkotiedostojärjestelmää (Microsoft, 2021), Postgresql SQL relaatiotietokantaa (Postgresql, 2022), Elasticsearch-hajautettua hakukonetta ja Hadoop hajautettua tietoallasta (Achari, 2015).

NFS palvelin ei tarjoa työkaluja tiedonlouhintaan, joten se on yhdistämätön tietovarasto. Teknisesti NFS on toteutettu client-server mallilla, jolloin hajautetut lokien keräyspalvelimet varastoisivat lokeja keskitetylle NFS-palvelimelle, jota

[6] eng. IT Service Management System, ITSM

SIEM hyödyntää tiedon louhintaan. NFS hylättiin, koska se ei mallinna minimoi tiedonsiirron tarvetta hajautetun keräämisen ja keskitetyn käsittelyn välillä.

Postgresql on avoimeen lähdekoodiin perustuva relaatiotietokanta, joka mainostaa itseään maailman edistyneimmäksi. Teknisten ominaisuuksiensa puolesta Postgresql mahdollistaa maantieteellisesti hajautetun asennuksen ja korkean suorituskyvyn. Ohjelma mahdollistaa SQL-kyselyiden käytön tiedonlouhinnassa ja sen toteuttaman JSON rajapinnan vuoksi tietokannan sisältö on käytettävissä myös http(s) protokollan kautta verkon yli tapahtuvassa laskennassa. Hakujen tehostamiseksi lokimerkintöjä voidaan esikäsitellä ennen tietokantaan vientiä säilöen lokien eri piirteitä tietokannan eri sarakkeisiin. Muodostettu tietokanta voidaan indeksoida hakujen tehostamiseksi. SQL relaatiotietokannat muodostavat louhintaan löyhäsi tai puolitiukasti yhdistetyn tietovaraston. Monimuotoisen lokitiedon vienti relaatiotietokantaan vaatii kohtalaisen kattavan esikäsittelymallin. Lisäksi luotettava, suorituskykyinen ja hajautettu relaatiotietokanta on toteutuksena vaativa. Relaatiotietokanta hylättiin toteutuksen raskauden vuoksi.

Elasticsearch on yleisesti käytettyyn Lucene kirjastoon perustuva hakupalvelin. Ohjelmistoa voidaan käyttää hakujen tekemiseen lähes minkä tyyppisistä dokumenteista tahansa ja se mahdollistaa hajautetun asennuksen, korkean käytettävyyden ja saatavuuden ja sitä voidaan skaalata koneresursseja lisäämällä. Elasticsearch on käytössä useissa keskitetyissä lokien hallintajärjestelmissä hakukoneena ja lokienvarastointipaikkana. Järjestelmä kykenee käsittelemään monimuotoisia lokeja ja se huolehtii lokien varastoinnista ja uudelleenjärjestelystä automaattisesti asennuksen koon kasvaessa. Ratkaisu on myös pilviyhteensopiva. Elasticsearch indeksoi ja hakee datan reaaliaikaisesti, joka tarkoittaa, että suorituskyky rajoittaa sen mahdollisuuksia hyväksyä suuria määriä uutta dataa kerralla. Skaalautuvuudesta huolimatta, Elasticsearch järkestelmän asennusvaiheessa joudutaan tekemään joitakin perusoletuksia datan kertymisestä ja käsittelystä, joita on hankalaa muuttaa enää sen jälkeen, kun käyttö on aloitettu. Elasticsearch muodostaa SIEM:n kanssa tiukasti yhdistetyn tietovaraston, jonka käyttö on tarkoituksenmukaista silloin, kun tavoitteena on käsitellä kohtalaiset määrät jatkuvana syötteenä saapuvaa dataa reaaliaikaisesti. Laajoissa organisaatioissa kertyvän tapahtuma lokin käsittelyssä Elasticsearchin vaatimat laskenta- ja muistiresurssien tarve voivat kasvaa hallitsemattomasti.

Yleisimmin käytetty hajautettu tietoallas ja laskentajärjestelmä yritysmaailmassa on Apache Hadoop, joka on välittömästi käyttöönotettavissa avoimen lähdekoodin lisenssiensä vuoksi. Lisäksi useimmat tietoaltaan käyttöön suunnitellut järjestelmät, erityisesti tiedonlouhinta ja tekoälyratkaisut ovat suoraan yhteensopivia Hadoopin kanssa. Datan indeksointi ja hakujen tekeminen toteutetaan MapReduce rinnakkaisen puitekehyksen avulla, jossa Hadoop klusterille annetaan datan prosessointitehtävä. Elasticsearch:stä eroten, MapReduce toteutetaan eräajona, joka soveltuu suurien datamäärien käsittelemiseen kerralla. Ellei haluta hyödyntää organisaatiolla mahdollisesti jo olemassa olevaa tietoallasta esim. silloin kun tavoitellaan kustannussäästöjä ylläpito- ja päivitystyössä, on järkevää hyödyntää Hadoopia.

2.3.5 Lokien välityspalvelin

Lokien vastaanottamiseen ja esikäsittelyyn on olemassa useita teknologisesti kypsiä ratkaisuja, tarkasteluun valittiin Apache Flume (Sammer & Lai, 2021), BSD Syslog, Fluentd, Greylog (Graylog, 2022) ja Logstash.

Apache Flume on avoimeen lähdekoodiin perustuva ohjelmisto joka on lähtökohtaisesti suunniteltu keräämään, esikäsittelemään ja toimittamaan suuria määriä lokidataa Hadoop tiedostojärjestelmään. Se on pääsääntöisesti luotu toteutettavaksi Unix-tyyppisillä palvelimilla. Ohjelma sisältää vikasietoisuutta parantavia ominaisuuksia sekä mahdollisuuden hallita keräyspalvelimien asetuksia keskitetysti. Logstash ja Fluentd ohjelmistoista poiketen, Flume on toeutettu Java ohjelmointikielellä, joka vaatii toteutettavilta järjestelmiltä enemmän levy- ja muistiresursseja. Flumen reitityslogiikassa lokimerkintä saapuu lähteeseen[7] joka ohjaa sen kanavaan[8] joka asetusten mukaisesti toimittaa sen nieluun[9]. Lokimerkinnän käsittelyä lähteessä, kanavassa ja nielussa voidaan hallita erikseen toisistaan riippumatta. Lisäksi Flume kykenee käynnistämään samalla fyysisellä tai virtuaalisella palvelimella useampia prosesseja, joista jokainen pitää sisällään yhden tai useampia lähde-kanava-nielu yhdistelmiä. Flume tukee lähtökohtaisesti useimpia käytössä olevia loki- ja kohdeformaatteja, sekä mahdollistaa omien laajennusten lisäämisen.

BSD Syslog palvelin sisältyy Unix-tyylisiin käyttöjärjestelmiin ja sitä käytetään BSD syslog formaatin mukaisien lokiviestien vastaanottamiseen ja edelleen lähetykseen. Palvelin hylättiin tarkastelussa, koska se ei sisältänyt tarvittavia toiminnallisuuksia muiden lokiformaattien vastaanottamiseksi, rajatun lokien käsittely- ja reititys ominaisuuksiensa takia.

Fluentd on avoimeen lähdekoodiin perustuva datan keräin, joka on suunniteltu keräämään sisään tuleva lokiteto ja järjestämään se lähteestä riipumattomaan JSON data formaattiin. JSON muodossa olevaa dataa voidaan helpommin hyödyntää muissa sovelluksissa kuten tietokannoissa tai tiedonlouhintaalgoritmeissa. Fluentd on toteutettu alhaisen tason ohjelmointikielillä pyrkien minimoimaan rajallisten laskenta ja muistiresurssien käyttö. Ohjelmaan on sisällytetty ominaisuuksia, joiden avulla keräinten vikasietoisuutta parannetaan ja pyritään välttämää kerätyn datan katoaminen. Ohjelma tukee satoja laajennuksia mahdollistaen lokien vastaanottamisen tai noutamisen eri muodoissa ja metodeilla. Fluentd mahdollistaa myös lokimerkintöjen reitittämisen eri kohteisiin, kuten SIEM:lle tai paikalliseen Hadoop tiedostoon. Ohjelmaa tukee aktiivinen kehittäjäjoukko ja se on käytössä suurissa yrityksissä. Fluentd:ssä ei ole keskitettyä hallintaa, joka hankaloittaa sen käyttöä niissä toteutuksissa, joissa asennettavien palvelinten määrä on suuri ja hajautettu laajalle.

Graylog on keskitettyyn lokienhallintaan tarkoitettu tuote, josta on olemassa sekä kaupallinen, että avoimeen lähdekoodiin perustuva versio. Kaupalliseen tuotteeseen on saatavilla tuki- ja ylläpitopalvelut. Graylogiin on saatavilla

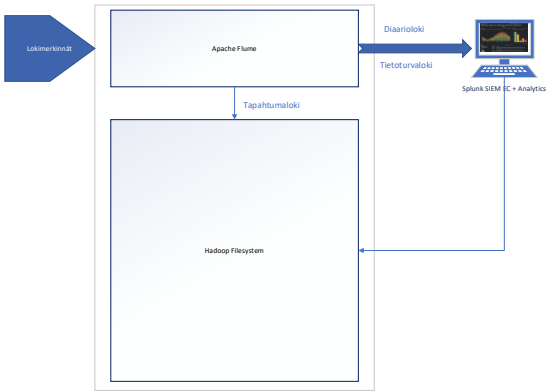
[7] eng. source
[8] eng channel
[9] eng. sink

varsin kattavat lisäosat, joiden avulla lokien vastaanotto sen integrointi muihin yritysjärjestelmiin, lokien lajittelu ja esikäsittely sekä siirto eteenpäin on toteutettavissa useissa erikäyttötapauksissa. Graylog on myös laajennettavissa täysimittaisaeksi SIEM järjestelmäksi tietoturvapoikkeamien hallintaa varten ja sisältää tarvittavat työkalut tietoaltaiden, kuten Hadoop hyödyntämiseksi. Graylogin ominaisuudet ylittävät lokien keruujärjestelmälle asetetut vaatimukset. Lisäksi järjestelmä suorittaa säilöö kerätyn datan indeksiin ts. indeksoi kaiken kerätyn datan palvelimella, joka vaatii suhteellisesti paljon muisti- ja laskentaresursseja, jotka eivät siten ole käytössä tietoaltaan hajautetun laskennan vaatimuksiin. Laajoissa asennuksissa resurssihukka kertautuisi jokaisen asennetun lokienkeräimen myötä.

Logstash on osa suosittua ELK[10] lokitietojen keräys ja analysointi pakettia se on ominaisuuksiltaan hyvin lähellä Fluentd sovellusta, joskin sen käyttää hieman erilaista logiikkaa lokitietojen reititykseen. Fluentd lisää saapuvaan lokimerkintään tunnisteen eli tagin ja reitittää lokimerkinnän tunnisteelle määritellyn sääntöjoukon perusteella, vertaan Logstash jokaista lokimerkintää if-then sääntöjoukkoon ja ohjaa sen perille sääntöön täsmäävään kohteeseen. Logstash ei sisällä vikasietoisuusominaisuuksia vaan ne täytyy toteuttaa muilla järjestelmätyökaluilla. Kuten Fluentd, myöskään Logstash ei sisällä keskitettyä palvelinten hallintaominaisuutta.

2.3.6 Kokonaisarkkitehtuuri

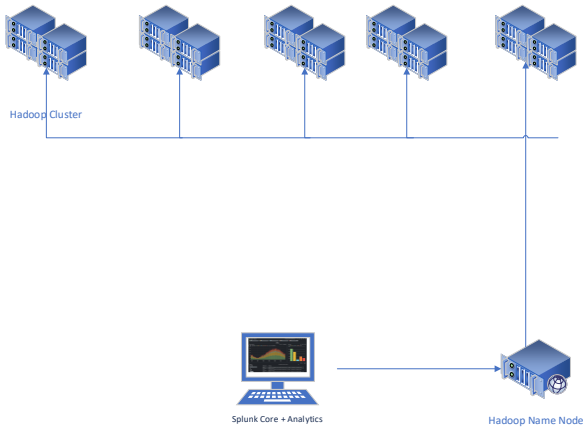
Koska tavoitteena on rakentaa järjestelmä, joka skaalautuu mahdollisesti satoihin lokien keräysnoodeihin, kymmeniin tuhansiin lokilähteisiin ja ennalta arvioimattomaan määrään erilaisia lokiformaatteja, päätettiin keräysjärjestelmäksi valita Apache Flume. Flumella voidaan tarkemmin kontrolloida lokimerkinnän käsittelyä siirron aikana ja keskitetysti ohjata keräyspalvelimia ilman erikseen vaadittavaa konfiguraationhallintapalvelua.



Kuva 4. Lokikeräin

[10] Elasticsearch, Logstash, Kibana

Keräysnoodien määrää voidaan kasvattaa vikasietoisuuden ja keräyskapasiteetin kasvattamiseksi. Lisääminen kasvattaa samalla tietoaltaan laskenta-, muisti ja levytilaresursseja ja siten hajautetun laskennan suorituskyä. Suurin julkisuudessa esitelty Hadoop tietoallas on Yahoo!-yhtiöllä, jonka 2008 suoritettujen testien mukaan yhden teratavun tekstin käsittely kesti 900 noodin Hadoop klusterilla 209 sekuntia (O'Malley, 2008). Heuristisesti arvioiden vuorokauden aikana voidaan vaatimusten mukaiset 4 teratavua käsitellä vähäisemmin resurssein.



Kuva 5. Tietoallas

Hadoop laskentatehtäviä ja -resursseja kontrolloiva Name Node voi toimia myös Flume ohjelmistojen Master palvelimena hallinnoiden koko asennuksen keräys, käsittely, varastointi ja analyysityötä.

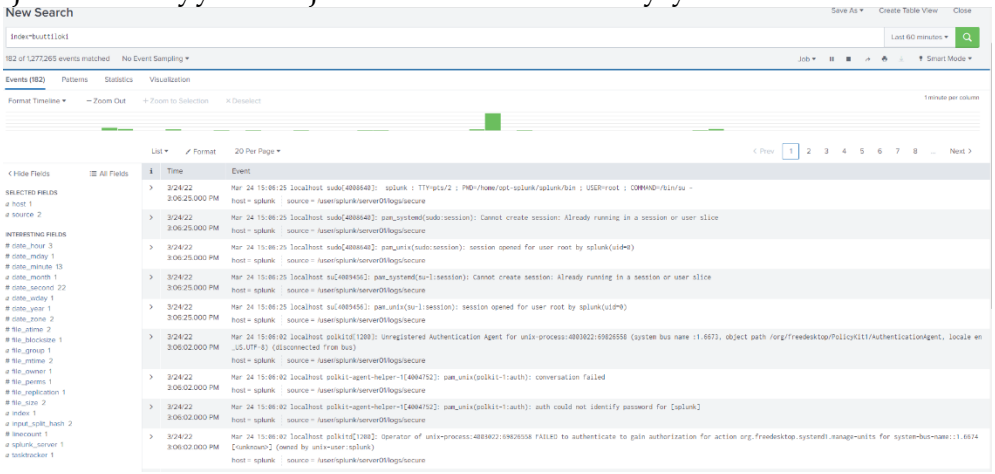
2.4 Demo

Demojärjestelmä toteutettiin rakentamalla Apache Flume lokikeräin, joka sisältää paikallisen Single node Hadoop tiedostojärjestelmän ja tietoaltaaseen liittyvän laskentaohjelman sekä lokien lajittelu ja edelleen lähetys säännöstön. SIEM palvelimena käytettiin Splunk Enterprise ohjelmistoa ja siihen liitettyä Splunk Analytics laajennusta.

Kerättäviksi lokeiksi valittiin Unix järjestelmäloki ja tietoturvaloki.Lokien keräystä varten käynnistettiin yksi Flume agentti, jolle määriteltiin kaksi lähdettä, kaksi kanavaa ja kolme nielua. Toinen lähteistä ottaa vastaan UDP Syslog viestejä ja toinen verkon yli TCP:llä lähetettyjä lokitiedostoja. Molemmissa lähteissä on säännöstö, joka valitsee tietoturva- ja diaariomerkinnät ja ohjaa ne kanavaan, joka johtaa ne edelleen nieluun, joka lähettää ne Splunk palvelimelle. Kaikki ne merkinnät, joita ei erikseen valita Splunkille meneviksi, ohjataan kanavaan, jonka nielu kirjoittaa ne paikalliseen Hadoop tiedostoon.

Siinä missä Splunkin oma indeksi näyttää päivittää hakuindeksinsä sitä mukaa kuin tietoa tulee sisään, päivittää Analytics-laajennus omansa vain silloin kun haku tehdään. Haku voidaan tehdä joko ajastetusti tai käsin. Tästä johtuen kahden tietovaraston sisältöä ei voida yhdistää samaan hakuindeksiin vaan

Splunk-järjestelmään kertyneet lokit ja Hadoop-tietoaltaaseen säilötyt lokit indeksoitiin omiksi hakuindekseiksi ja niistä muodostettiin hakutoiminnoilla raportteja SOC analyytikolle ja automatisoiduille hälytyksille.



Kuva 6. Hadoop virtuaalinen hakuindeksi

3 Materiaalit ja menetelmät

3.1 Lokien vastaanottaminen

Laajoissa ja hajaututetuissa järjestelmissä lokien siirto tehdään useilla menetelmillä, joista hallitsevin on Internet Engineering Task Force:n määrittämä BSD syslog protokolla (Lonvick, 2001). Syslog lähettää lokimerkinnät reaaliaikaisesti UDP tai TCP protokollaa käyttäen on oletuksena selväkielinen eikä sisällä mekanismeja viestien autentikointiin tai sisällön varmistamiseen. Jos syslog protokollaa ei voida käyttää tietoturvasyistä, noudetaan lokitiedostot lokipalvelimelle tai ne lähetetään sinne Apache Flumella, Fluentd, ryslogd. Protokolla. Hadoop on suunniteltu käsittelemään mieluummin harvoja suuria tiedostoja kuin useita pieniä. Tästä johtuen jokainen lokityyppi kannattaa ottaa vastaan omalla flume lähteellä, ohjata omaan kanavaan ja kirjoittaa oman nielun avulla tiedostoon. Näin jokainen lokityyppi on omassa tiedostossaan, joka helpottaa niiden esikäsittelyä.

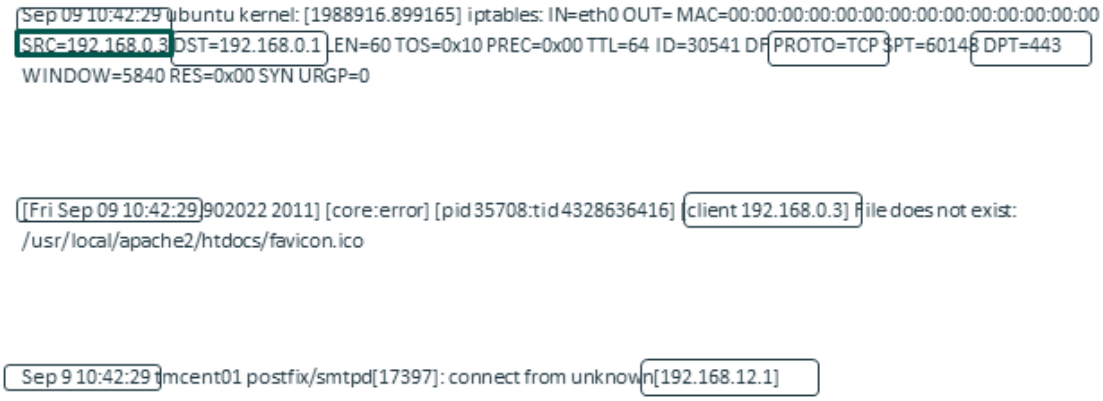
3.2 Lokien esikäsittely

Tiedonlouhinnassa datan esikäsittelyn tarkoitus on parantaa datan laatua ja sitä kautta tiedonlouhinnan tuloksia (Kamber;Han;& Pei, 2011, s. 47). Raakana kerätty lokitieto ei sellaisenaan ole käyttökelpoista tietoturvapoikkeamien selvityksessä, haasteena on monesta lähteestä kerättyjen lokien monimuotoisuus, lokit muodostetaan pääosin tekstimuotoisina, joskin esim. verkkoliikenteen analyysiin käytettävät snifferit tallentavat pakettidatan binäärimuodossa. binäärimuotoinen data tulee analyysiä varten muuntaa tekstiksi, joka osaltaan vaatii laskentaresursseja. Ainoa varmasti saatavilla oleva piirre on aikaleima, joka sekin voi olla merkitty esim. aikavyöhykkeen kellonajan tai maailman ajan[11] mukaisesti. Aikaleiman esitysmuoto ei myöskään ole aina standardi vaan Yhdysvaltain käytännön mukaisesti toimivat järjestelmät esittävät päivämäärän muotoa kuukausi-päivä-vuosi. Tekstimuotoiset viestit toistavat samoja tapahtumaa kuvaavia viestejä pl. tapahtumille mahdollisesti annettava tapahtumatunnus

Lokitiedon muodostamiselle on useita yleisiä käytäntöjä, muttei yleisesti käytössä olevaa standardia. Lokidatan louhinnan tehostamiseksi merkinnät esikäsitellään ennen niiden kirjoittamista tietoaltaaseen irrottamalla niistä tunnistettavissa olevat piirteet. Lokitiedon sisällön esitystapa vaihtelee. Esim. sähköpostin lähetyksestä jää palomuureihin merkintä porttinumerolla TCP 25 ja itse sähköpostipalvelimeen protokollalyhenteellä smtp. Lokisisältö itsessään voi olla tekstiä tai XML-koodattua. Keräysvaiheessa lokidatan piirteet pyritään normalisoimaan muuttamalla aikaleima, protokolla, postinumero, lähde- ja

[11] eng. Universal Coordinated Time, UCT

kohdeosoitteet ja mahdollinen tapahtumatunnus omiksi yhteismuotoisiksi kentikseen. Lokieräimet sisältävät toiminnallisuuden valmiina yleisimmille lokimuodoille. Niissä tapauksissa, joissa loki ei noudata tunnettuja muotoja, on tarpeen kehittää tarkoitusta varten esikäsittelylaajennus. Esikäsitelty loki voidaan varastoida erikseen raakana kerätystä lokista. Niissä tapauksissa, joissa esikäsittelyä ei ole voitu suorittaa, tulee piirteiden irrotus suorittaa tietoaltaaseen varastoidusta raakalokista louhintaa tehdessä, joka kuluttaa tietoaltaan resursseja ja tehtävään kuluvaa aika joka kerta, kun irrotus suoritetaan.



Jotta saatu lokimerkintä voidaan muuntaa valittuun tietorakenteeseen, esikäsitellään lokimerkintä seuraavasti:

1. Muunnetaan loki ASCII tekstiksi
2. Irrotetaan merkinnästä aikaleima.
3. Muunnetaan aikaleima UCT-muotoon
4. Tunnistetaan lokittava järjestelmä
5. Tunnistetaan lokittava ohjelmisto
6. Tunnistetaan lähde-IP
7. Tunnistetaan lähdeportti
8. Tunnistetaan kohde-IP
9. Tunnistetaan kohdeportti
10. Tunnistetaan protokolla
11. Muodostetaan kaikista muista merkkijonoista sanaindeksi
12. Kirjoitetaan esikäsitelty lokimerkintä JSON-formaatissa tietovarastoon

3.3 Lokien Säilöntä

Tietovarastoksi valittiin Apache Hadoop. Hadoop on avoimeen lähdekoodiin nojautuva projekti, jonka tavoitteena on luotettava ja skaalautuva ohjelmisto hajautettuun laskentaan. Asentamalla Hadoop lokien keräys- ja edelleen lähetyspalvelimille voidaan hyödyntää niiden vaatimia resursseja hajautettuun ja rinnakkaiseen käsittelyyn, erityisesti tapauksissa, joissa vaaditaan korkeaa

saatavuutta, joka vuoksi koneresursseja joudutaan monentamaan. Oletuksena on, että edut tulevat olemaan merkittäviä suurissa ja maantieteellisesti hajautetuissa organisaatioissa, joiden tuottama lokidata voidaan täten analysoida entistä paremmin ajantasaisen kyberturvakuvan tuottamiseksi.

Testiympäristössä kaikki Hadoop-palvelimen komponentit asennettiin samalle palvelimelle, raskaissa Big Data ympäristöissä Hadoop-toteutukset on rakennettu Master/Slave toteutuksina, joissa laskenta tehoa voidaan lisätä Slavenoodien määrää kasvattamalla.

3.4 Data

Eri tietojärjestelmien tuottamat lokimerkinnät voidaan ryhmitellä joukoiksi, mikäli lokimerkintään liittyvät tapahtumat vaativat tietojärjestelmiltä keskinäistä vuorovaikutusta. Esim. sähköpostin lähettäminen tai vastaanottaminen vaatii liikennettä palomuurin läpi ja mahdollisesti nimipalvelukyselyä.

SOC palvelun kokemusten perusteella, suurin lokityyppi, jota SIEM järjestelmä käsittelee, on palomuurien pääsynhallintaloki. Tietoturvan valvontaan riittävän tarkaksi määritelty loki tuottaa vuorokaudessa noin yhden gigatavun dataa. Keskisuurissa ja suurissa organisaatioissa palomuureja käytetään organisaation ulkoisen verkkoliikenteen suodattamiseen ja organisaation sisäisten verkkosegmenttien erottamiseen. Todennäköisimmin järjestelmien välinen vuorovaikutus näkyy palomuurilokeissa. Palomuurit jaotellaan paketti-, yhteys- ja sovellussuodattimiin, riippuen siitä millä protokolla tasolla suodatus tapahtuu. Nykyaikaisten kaupallisten palomuurien ominaisuudet sallivat niiden toimia kaikilla tasoilla. Perusmuotoiseen palomuurin lokimerkintään kuuluu ainakin:

Aikaleima
Lähteen IP-osoite
Lähteen portti
Kohteen IP-osoite
Kohteen portti
Protokolla, kuten ICMP,TCP,UDP
Verkkosovitin
Toimenpide, kuten Pass/Drop

Muita käytettävissä olevia tietoja voivat olla palomuurilaitteen tunnus, sovellusprotokolla tai käytetty sääntötunnus.

Sovellusten tuottaman lokin liikennemerkinnät sisältävät samaa osoitetietoa kuin palomuurin pääsynhallintaloki, lisäksi lokissa on järjestelmän tunnus, ohjelmiston tunnus, transaktion tunnus, sanallinen merkintä tai koodi transaktion onnistumisesta, mahdollinen virheviesti ja statistiikkaa.

Analyysiä varten tietorakenteen pohjaksi valittiin palomuurin tuottama lokimerkintä, on täydennetty lokista löydetyistä sanoista muodostetulla listalla.

Rivinumero	Aikaleima	Järjestelmätunnus	Ohjelmistotunnus	Lähde-IP	Lähdeportti	Kohde-IP	Kohdeportti	Protokolla	Sanalista

3.5 Koneoppimismenetelmät

Lokimerkintöjen ryhmittely on koneoppimisen kannalta luokitteluongelma, toisiinsa liittyvien lokimerkintöjen joukkoa voidaan pitää luokkana, joita on ennalta tuntematon määrä. Tästä syystä mm tukivektorikoneita ei tarkasteltu tässä tutkimuksessa.

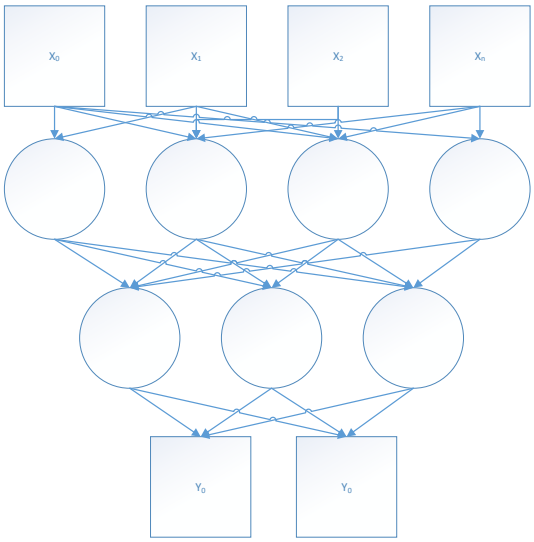
3.5.1 Neuroverkot

Koneoppimisen ja tekoälyn saralla nopein kehitys lähimenneisyydessä on tapahtunut syväoppivien neuroverkkojen alueella. Syväoppivia neuroverkkoja on käytetty kuvantunnistukseen, Go- (Chen, 2016) ja Shakkikoneina (Pitkänen, 2019) ja proteiinien taitto-ongelmaa tutkittaessa (Warren, 2020). Neuroverkkojen avulla kyetään luomaan tehokkaita lajittelukoneita, niin kauan kuin opetukseen käytettävä data on hyvänlaatuista, datan laadun vaihtelun vaikutus neuroverkon tarkkuuteen on osoitettu sekä kokeellisesti (Suprami, 2002), että käytössä olevia järjestelmien käyttökokemusten perusteella. Olettaen että lokimerkintöjen joukot tunnistetaan, voidaan olettaa, että neuroverkko voidaan kouluttaa tunnistamaan anomaliat organisaation normaalista toiminnasta.

Tutkittaessa lokimerkintöjen ryhmittelyä, törmäämme puutteelliseen ennakkotietoon. Emme tiedä moneenko ryhmään lokimerkinnät tulee luokitella. Neuroverkot kykenevät tarkkaan lajitteluun silloin kun luokkien määrä on tiedossa eli suljettujen joukkojen tehtävissä. Haettaessa toistuvia kaavoja emme etukäteen tiedä montako luokkaa on olemassa, joskin kokemuksen ja havaintojen perusteella voimme esittää arvauksia. Avoimen joukon[12] lajittelutehtäviin on kehitetty joukko algoritmeja, joissa tunnistamattomien luokkien käsittely vaihtelee. Tieto luokitellaan johonkin tunnettuun joukkoon ja kaikki muu merkitään ei kiinnostavaksi, kuten kuvantunnistusohjelmassa Cat or Not (Peden, 2019) tai se luokitellaan kuuluvaksi tunnistamattomaan luokkaan (Boult & Bendale, 2015). Kumpikaan vaihtoehto ei ole tyydyttävä haettaessa rakenteita lokidatasta, josta meillä ei ole aikaisempaa analyysitietoa.

Neuroverkkoa luotaessa vaarana on myös vinoumat opetusdatassa. Toistaiseksi meillä ole toistaiseksi muuta keinoa muodostaa opetusdataa, kuin järjestelmäasiantuntijoiden ja SOC-analyytikoiden ylläpidossa ja poikkeamanhallinnassa keräämä heuristiikka. Neuroverkko on siis altis samoille vahvistus-, saatavuus ja valintaharhoille, kuin sitä opettava asiantuntija.

[12] eng. Open Set



Kuva 7. Neuroverkon rakenne

3.5.2 Klusterointi

Klusteroinnissa rypästäminen on ohjaamaton koneoppimismenetelmä, jossa pyritään tavallisimmin jakamaan datapisteet joukkoon ryppäitä. Mikäli tutkittavasta datasta ei tiedetä mitään, tulee ensiksi selvittää, moneenko klusteriin data luokitellaan yhdellä algoritmilla. Sen jälkeen data jaotellaan luokkiin toisella. Kirjallisuustutkimuksen perusteella K-Means klusterointia on tutkittu anomaliantunnistukseen analysoiden yhdemukaisia lokeja. (Jakrarin & Primsopa, 2013)

Klusterointi ei perustu ennalta tunnettuun luokitteluun, vaan on oppimista perustuen havainnoitiin. Klusterointi on haastavaa silloin, kun käsitellään valtavia datasettejä, jolloin usein pyritään käsittelemään pienempää otosta. Tällöin on vaarana oppimisharhat tunnistamaan poikkeamat niiden ulkopuolelle jäävistä havainnoista. Klusterointi kykenee käsittelemään vain yhdenmukaista dataa, monimuotoiset lokit tulee ensin normalisoida esim. muuttamalla toistuvat merkkijonot numeerisiksi ja käsittelemällä niistä muodostettua indeksiä. Klusteroinnin haasteet nousevat myös nopeasti mitä suurempaa määrää datan ulottuvuuksia joudutaan käsittelemään. Klusterointi arviointiin metodina käyttökelpoiseksi sitten kun toistuvat tietorakenteet on monimuotoisesta lokimassasta tunnistettu ja esikäsitelty.

3.5.3 Time series data

Lokimerkinnät asettuvat sarjaan aikajanalle, toisiaan seuraavia lokimerkintöjä analysoiden pitäisi kyetä muodostamaan malli, joka ennustaa tunnettujen lokimerkintöjen sarjasta seuraavan odotettavissa olevan, arvioi sitä vastaanotettuun ja lähettää poikkeavat eteenpäin.

Ongelmana TSD analyysin käytössä tarkoitusta varten arvioitiin epätarkkuus lokimerkinnän aikaleimassa. Lokimerkinnän aikaleima merkitään tietokoneen omaan sisäiseen CPU kellonaikaan perustuen. CPU kellon tarkkuus heittää

joitakin sekunteja vuorokaudessa ja joudutaan synkronoimaan usein. Microsoftin suositus on, että tietokoneen sisäisen ajan tulisi olla enintään 5 minuuttia pielessä ja vaikka sisäinen kello synkronoitaisiin useita kertoja vuorokaudessa, ei ole takuuta, että tapahtumat asettuvat aikajanalla toisiinsa nähden oikein. Tietokoneen kello ilmoitetaan usein, miten sekunnin tarkkuudella.

3.5.4 Frequent pattern mining

Frequent Pattern Mining eli FPM tarkoittaa pyrkimystä löytää toistuvia rakenteita, jotka esiintyvät datassa usein. FPM on käytössä mm. tutkittaessa ostoskäyttäytymistä asiakastietokannoista. FPM voidaan määritellä kahdella säännöllä

1. Etsi datasta kaikki usein toistuvat joukot.
2. Muodosta vahvat assosiaatiosäännöt usein toistuville joukoille.

FPM kiinnostavuuden mittareina on tuki ja luottamus. Tuki mittaa kuinka monessa tietoalkiojoukkossa tietoalkiot toistuvat, luottamus sitä kuinka vahva kahden tietoalkion välinen riippuvuus on. Molemmat ilmaistaan prosentteina, rajaarvot minimituelle ja minimiluottamukselle asetetaan louhittavan datan ominaisuuksien perustella.

FPM malleja ovat sarjoina toistuvien kaavojen tunnistukseen pyrkivä Sequential Pattern Mining, rakenteisten kaavojen tunnistukseen pyrkivä Structured Pattern mining ja transaktionaalisten- tai relaatiotetokantojen usein toistuvien kaavojen tunistukseen pyrkivän Frequent Item Mining.

Suurten lokimäärien analysoinnissa Sequential Pattern Mining törmää samaan ongelmaan Time Series Data analyysin kanssa. Samaan tapahtumaan liittyvät lokimerkinnät eivät välttämättä ole samassa järjestyksessä jokaisen tapahtuman kanssa. Structured Pattern Mining on tiedonlouhinnan yleistetty tapaus, joka on käyttökelpoinen suurien tietorakenteiden analysoinnissa. Tutkimuksessa keskitytään rakenteiden löytämiseen hyvin lyhyen aikavälin sisällä. Olettaen, että organisaation tietojärjestelmien sisäiset kellot on synkronoitu käytännöllisellä tarkkuudella, voidaan samalla aikaleimalla löytyvistä lokimerkinnöistä etsiä usein toistuvia kaavoja.

Frequent Item Mining algoritmina käytetään Apriori joka käy dataa läpi iteratiivisesti etsien k alkiojoukosta k+1 alkiojoukkoa. Jos halutaan löytää usein toistuva n alkion suuruinen joukko, joudutaan data käymään läpi n kertaa.

Oletetaan esimerkissä, että meillä on datasetti D joka koostuu tietoalkiojoukoista I_1,I_2….I_n. Asetetaan minimitueksi 50% eli 3 esiintymää.

Datasetti D	
I_1	(a,b,c,d,e)
I_2	(a,b)
I_3	(b,c)
I_4	(a,d,e)
I_5	(a,c,d)
I_6	(a,b,d,f)

alkio	Datasetit	tuki
a	I_1, I_2, I_4, I_5, I_6	5
b	I_1, I_2, I_3, I_6	4
c	I_1, I_3, I_5	3
d	I_1, I_5, I_6	3
e	I_1	1
f	I_6	1

alkiot	Datasetit	tuki
a,b	I_1, I_2, I_6	3
a,c	I_1, I_5	2
a,d	I_1, I_4, I_5, I_6	4
b,c	I_1, I_3	2
b,d	I_1, I_6	2
c,d	I_1, I_5	2

alkiot	Datasetit	tuki
a,b,c	I_1	1
a,b,d	I_1, I_6	2
a,b,e	I_1	1
a,b,f	I_6	1
a,d,c	I_1	1
a,d,e	I_1, I_6	2
a,d,f	I_6	1

Esimerkissä ei löydetty yhtään kolmen alkion joukkoa, joka olisi täyttänyt minimituen vaatimukset, kahden alkion joukosta kiinnostavin on (a,d), joka esiintyy 4 kertaa kuuden datasetin joukossa. Assosiaatiosääntöjä

$$a \rightarrow b$$
$$b \rightarrow a$$

Lasketaan tuki säännöille:

$a \rightarrow b = P(a|b) = \frac{3}{5} = 0{,}6$ eli 60%

$b \rightarrow a = P(b|a) = \frac{3}{4} = 0{,}75$ eli 75%

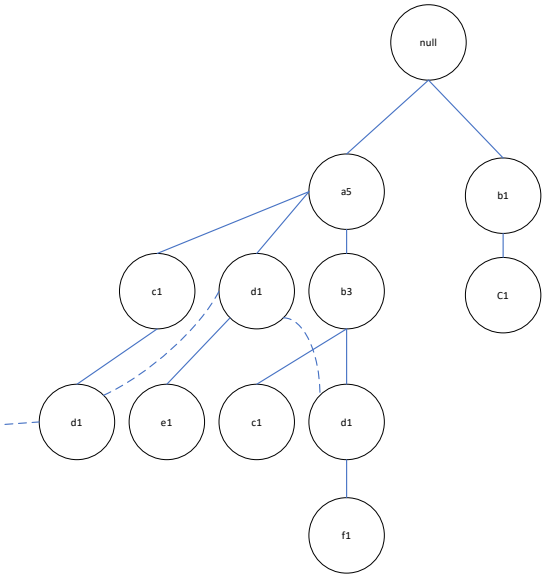
Apriori algoritmi on iteratiivisuutensa vuoksi raskas. Suuria datasettejä ei voida käsitellä muistissa ja levyjärjestelmien I/O rajoitukset hidastavat prosessia. Apriori algoritmin korvaajaksi on ehdotettu Frequent Pattern Tree algoritmia (Han;Pei;Yin;& Mao, 2004) jossa datasetti tiivistetään Pienemmäksi FP-tree nimiseksi datastruktuuriksi skannaamalla ensin tietoalkiot ja asettamalla ne frekvenssinmukaan laskevaan järjestykseen.

Datasetti D	
I_1	(a,b,c,d,e)
I_2	(a,b)
I_3	(b,c)
I_4	(a,d,e)

I_5	(a,c,d)
I_6	(a,b,d,f)

alkio	tuki
a	5
b	4
c	3
d	3
e	1
f	1

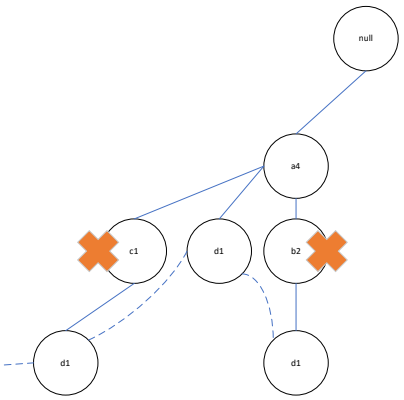
Alkiojoukot asetetaan puun juureen, aina joukon toistuessa laskuria nostetaan yhdellä.



Kuva 8. Frequent Pattern Mining Puu

Kun puu on muodostettu, voidaan etsiä rekursiivisesti kaikki toistuvat alkiojoukot esim. alkion d:n suhteen. Poistetaan kaikki alkion d jälkeiset alaoksat päivitetään laskuri osoittamaan tukea alkion d suhteen ja poistetaan kaikki oksat jotka eivät täytä minimitukivaatimusta. Havaitaan että mahdollisista alkiojoukoista

{a,b,d},{a,c,d},{a,d] jäljelle jää alkiojoukko {a,d}



Kuva 9. Typistetty Frequent Pattern Mining puu

(Kamber;Han;& Pei, 2011, ss. 227-283)

3.6 Lokien analysointi

Monista erimuotoisista lähteistä muodostuvaa lokimassaa louhittaessa valittiin seuraava lähestymistapa:

Valitaan otos aikaväliltä, jonka voidaan heuristisesti arvioiden olettaa edustavan organisaation tyypillistä käyttöä. Otoksen suuruuteen vaikuttaa lokimassa ja käytettävissä oleva laskenta teho. Esimerkiksi voimme valita kaikki lokimerkinnät yhtenä arkipäivänä klo 0800-17:00

Esikäsitellään lokimassasta valittu otos tapahtumiksi aiemmin määriteltyyn tietorakenteeseen.

Rivinumero	Aikaleima	Järjestelmätunnus	Ohjelmistotunnus	Lähde-IP	Lähdeportti	Kohde-IP	Kohdeportti	Protokolla	Sanalista

Muodostetaan samalla aikaleimalla merkityistä tapahtumista tapahtumajoukko jonka uniikit alkiot sisältävät järjestelmätunnuksen J, ohjelmistotunnuksen O ja protokollan P ja sanalista L.

hh:mm:ss	$(J,O,P,L)_1,(J,O,P,L)_2,(J,O,P,L)_3\ldots (J,O,P,L)_n$
hh:mm:ss+1	$(J,O,P,L)_1,(J,O,P,L)_{3\ldots}$

Louhitaan toistuvat kuviot Frequent Item Mining algoritmilla, kuten Apriori tai puu rakenteita etsivä Frequent Pattern Growth. Organisaatioiden normaaliin

päivittäiseen rutiiniin liittyvä toiminta toistuu pitkällä aikavälillä usein, sitä etsittäessä asetetaan minimituen raja korkealle. Deterministisinä koneina, tietojärjestelmien välisiä transaktioita tutkittaessa voimme odottaa toisistaan riippuvien tapahtumien lokimerkinnöiltä korkeaa luottamusta. Asetamme mielivaltaisesti päättäen minimituen 20% ja minimiluottamuksen 95%.

Löydettyämme kiinnostavimman kattavan joukon johdamme joukon assosiaatiosääntöjä järjestelmien, ohjelmistojen ja protokollien välillä. Laajoista ja hajautetuista järjestelmistä ei ole toistaiseksi saatavilla lokidatasta koostuvaa datasettiä. Testiä varten ladattiin lokidatasetti osoitteesta:

http://log-sharing.dreamhosters.com/hnet-hon-var-log-02282006.tgz

Josta eroteltiin kahden päivän, Marraskuu 16. ja 21. 2005 järjestelmä ja Apacche verkkopalvelimen liikenneloki. Kaikki tapahtumat ovat samalta palvelimelta ja samalta kuukaudelta ja käyttävät samaa tcp-protokollaa. Datasetti sisältää pääosin toisistaan riippumattomia järjestelmälokeja, http-palvelimen ja smtp-palvelimen liikennelokeja. Datasetti sisälsi n riviä syslog-muotoisia lokeja. Työhön käytettiin Apache Spark data-analytiikka työkalua, joka sisältää valmiiksi toteutetut funktiot tiedon esikäsittelyyn ja louhintaan (Apache Foundation, 2022). Tyypillinen syslog-viesti

Jan 1 compo myapplication 1234 ID47 [example@0 class="high"] BOMmyapplication is started

Luettiin Spark tietorakenteeseen

Jan 1	12:02:21	compo myapplication 1234 ID47 [example@0 class="high"] BOM-myapplication is started

josta itse lokiviesti luetaan sanalistaksi. Koska järjestelmätunnus ”compo” toistuu jokaisessa viestissä, pudotetaan se pois:
{<myapplication>, <1234>, <ID47>, <[example@0 class="high"]>, <BOM-myapplication>, <is>, <started>}

Listat louhittiin Frequent Pattern Growth algoritmilla toistuvien sanojen löytämiseksi. lokiviesteissä aina toistuvien sanojen yhdistelmillä luottamus eli confidence on 1. Niiden tuki eli support on yhtä suuri. Sanalista muodostetaan pisimmästä yhdistelmästä, jolla on suurin noste eli lift. Ensimmäinen lokista muodostettu sanalista:

{”Connection”, ”timed”, ”out”, ”with”, ”stat=Deferred:”, ”dsn=4.0.0”, ”mailer=esmtp”,} määrittelee kaikki lokitapahtumat jossa sähköpostin edelleenlähetys on epäonnistunut yhteysvirheen vuoksi. Sanalista jolla on seuraavaksi korkein tuki:

{”proto=SMTP”, ”class=0”, ”daemon=MTA”} viittaa onnistuneeseen postin lähetykseen.

4 TULOKSET

Security Operations Center-palveluun tutustumalla havaittiin, että samoihin palveluihin liittyvät lokimerkinnät eivät säännönmukaisesti kulje samojen keräyspalvelimien kautta. Tietoverkon, sovelluspalvelimien ja sovellusten hallinta on hajautettu sekä maantieteellisesti, verkkotopologisesti että ylläpito- ja hallintahenkilöstön osalta. Samaan tapahtumaan liittyvät lokimerkinnät voivat hajautua eri keräyspalvelimille ainakin palomuurin ja sovelluslokin suhteen. Lokimerkintöjen hajautuminen on vielä laajempaa, mikäli tapahtumaan liittyvä tietoverkkoinfran palveluiden käyttö, kuten nimipalvelukyselyt, välityspalveluiden- tai pilvipalveluiden käyttö otetaan huomioon.

Lokimerkintöjen välisiin riippuvuuksiin perustuva poikkeamien tunnistus tarkoitettu koneoppivalla järjestelmällä on sitä epätarkempi mitä vähemmän sillä on dataa käytettävissään. Jotta kerätystä lokimassasta saadaan täysi hyöty, tulisi se keskittää kokonaisuudessaan esisuodatukseen ennen SIEM järjestelmään lähetystä. Vaikka esisuodatuksella voitaisiin pienentää SIEM:n päätyvää lokimassaa ja saavuttaa säästöjä silloin kun SIEM:n kaupallinen lisenssi perustuu transaktioiden ts. sisään tulevan lokimassan määrään, on säästö kyseenalainen siirtoon vaaditun kaistanleveyden ja erillisen suodatusjärjestelmän vaatiman ylläpidon kustannusten vuoksi.

Tietojenlouhinnan metodit soveltuvat erinomaisesti lokimassan seulontaan. Frequent Pattern Mining osoittautui käyttökelpoiseksi työkaluksi lokiviestien sanalistojen tunnistuksessa. Samaa työkalua voidaan käyttää myös usein toistuvien lokiviestien yhdistelmiä tunnistuksessa lokiviestidatassa. Tunnistetut rakenteet voidaan syöttää neuroverkolle poikkeamien tunnistusta varten.

5 JOHTOPÄÄTÖKSET

Tietoturvapoikkeamien hallinnassa etsitään tasapainoa analysoitavan tilannekuvan kattavuuden ja reagointinopeuden välillä. Lokitietoa analysoitaessa tulee tehdä valinta ajantasaisuuden ja kokonaisuuden välillä. Käytännössä tämä ilmenee SIEM järjestelmään lähetettävän ja Hadoop tietoaltaaseen ohjattavan lokimassan välillä.

Säilytettävä raakaloki menettää merkitystään sitä mukaa kuin se vanhenee, samalla suurempi lokimassa kasvattaa analysointiin tarvittavia resursseja ja sitä mukaa aikaa. Jossain vaiheessa on todennäköistä, että saavutetaan piste, jossa lokimassan kasvattaminen tai laskentatehon lisääminen ei enää tuota tarkempia tuloksia poikkeaman tunnistuksessa tai muussa tiedon jalostuksessa.

5.1 Suositukset

Lokidatan käyttökelpoisuus poikkeamanhallinnassa heikkenee datan vanhetessa. Datan käyttökelpoisuus tiedonlouhinnassa kuitenkin säilyy ennallaan ja jopa kasvaa mitä enemmän dataa saadaan kerättyä ja säilöttyä. Tiedon säilytykseen ja prosessointiin käytettävien resurssien hinta on laskenut voimakkaasti, joka mahdollistaa yhä suurempien datamassojen säilömisen. Toisaalta dataa muodostetaan yhä nopeammin, tässä kilpajuoksussa muodostuva data ylittää varastointikyvyn, joten kaikessa tiedon arkistoinnissa, myös lokidatassa tulisi kiinnittää suurin huomio tiedon elinkaaren hallintaan.

Siinä missä raaka-lokitiedon varastoinnin mielekkyys on kyseenalaista, voidaan niistä louhittuja tietorakenteita päivittää sitä mukaa kuin uutta dataa kertyy.

5.2 Jatkotutkimus

Tietoaltaiden käyttö tietoturvatapahtumien arviointiin tarjoaa uusia mahdollisuuksia ja jatkotutkimusta tarvitaan parempien työkalujen löytämiseksi lokitietojen hyödyntämisessä. Tietoaltaan rinnalla dataa voidaan analysoida lennosta erinäisillä streaming- ratkaisuilla. Esim. Apache Kafka tarjoaa mahdollisuuden varastoida kertyvää lokitietoa erinäisiin pipeline-rakennelmiin. Ajantasaisuus/kokonaisuus-tasapainoa etsittäessä.

Neuroverkkojen käyttö toistuvien rakenteiden riskiarvioinnissa tarjoaa mahdollisuuksia automatisoida poikkeamantunnistusta. Neuroverkon tarkkuus riippuu kuitenkin pääosin sen opetukseen käytettävän datan laadusta. Tunnistettuja ja käsiteltyjä poikkeamia voidaan todennäköisesti käyttää opetusdatajoukon muodostamiseen täydennettynä tiedonlouhinnan metodeilla löydetyillä

rakenteilla. Jatkotutkimuksella voidaan löytää metodit opetusdatan louhimiseen poikkeamanhallintaprosessista kertyneiden tietojen perusteella.

Write Ahead Lokissa, lokimerkintä muodostetaan ennen toiminnon suorittamista. WAL käytetään relaatiotietokannoissa varmistamaan operaatioiden johdonmukaisuus ja datan eheys. Laajoissa ja hajautetuissa ympäristöissä WAL tarjoaa mahdollisuuden toiminnon tietoturvariskin arvioimiseen ja valtuuttamiseen, mikäli luotettava riskiarviointi moottori on käytettävissä.

LÄHTEET

Helsingin ja Uudenmaan Sairaanhoitopiiri. (21. May 2021). *Hankintailmoitus HUS 067-2020*. Noudettu osoitteesta Hilma Julkiset hankinnat: https://www.hankintailmoitukset.fi/fi/public/procurement/49854/notice/72568/overview

Achari, S. (2015). *Hadoop Essentials*. Packt Publishing, Limited.

Apache Foundation. (26. 04 2022). *Apache Spark - Unified engine for large-scale data analytics*. Noudettu osoitteesta Apache Spark: https://spark.apache.org/

Boult, T. E.;& Bendale, A. (2015). *Towards Open Set Deep Networks.* Colorado Springs: University of Colorado.

Center for Internet Security. (15. 03 2021). *The SolarWinds Cyber-Attack: What You Need to Know*. Noudettu osoitteesta CSIS: https://www.cisecurity.org/solarwinds

Chainourov, B. (2017). *LOG ANALYSIS USING SPLUNK HADOOP CONNECT.* Monterey: Naval Postgraduate School.

Chen, J. X. (2016). The Evolution of Computing: AlphaGo. *Computing in Science & Engineering*, 4 - 7.

Graylog. (03. 02 2022). *Graylog*. Haettu 03. 03 2022 osoitteesta https://www.graylog.org/: https://docs.graylog.org/docs

Han, J.;Pei, J.;Yin, Y.;& Mao, R. (2004). Mining Frequent Patterns without Candidate. *Data Mining and Knowledge Discovery, 8*, ss. 53–87.

Helsingin kaupunki. (28. June 2021). Kyberturvan palvelujärjestelmä asiantuntijapalveluineen. *Eu Hankintailmouts H057-21*.

Iváncsy, R.;& Vajk, I. (2006). Frequent Pattern Mining in Web Log Data. *Acta Polytechnica Hungarica Vol. 3, No. 1*, 77-90.

Jakrarin, T.;& Primsopa, K. (2013). Applying Hadoop for log analysis toward distributed IDS. *Proceedings of the 7th International Conference on Ubiquitous Information Management and Communication*, (ss. 1-6).

Juvonen, A.;Sipola, T.;& Lehtonen, J. (2011). Engineering Applications of Neural Networks. *IFIP Advances in Information and Communication* (ss. 172-181). Boston: Springer.

Kamber, M.;Han, J.;& Pei, J. (25. 07 2011). *Data Mining: Concepts and Techniques : Concepts and Techniques.* Elsevier Science & Technology.

Kent, K.;& Souppaya, M. (2006). *Guide to Computer Security, NIST Special Publication 800-92.* National Institute of Standards and Technolog.

Lonvick, C. (Elokuu 2001). *RFC 3164 The BSD syslog Protocol*. Noudettu osoitteesta The Internet Engineering Task Force (IETF): https://datatracker.ietf.org/doc/html/rfc3164

Mandiant. (2021). *M-Trends 2021.* Fireeye Inc.

Microsoft. (12. 23 2021). *Network File System overview*. Noudettu osoitteesta Microsoft docs: https://docs.microsoft.com/en-us/windows-server/storage/nfs/nfs-overview

MITRE. (01. 03 2021). *Enterprise Matrix*. Noudettu osoitteesta https://attack.mitre.org/matrices/enterprise/: https://attack.mitre.org/matrices/enterprise/

O'Malley, O. (2008). *TeraByte Sort on Apache Hadoop.* Yahoo!

Peden, R. (30. 04 2019). *Cat or Not - An image Classifier using Python and Keras*. Noudettu osoitteesta Code Project: https://www.codeproject.com/Articles/4023566/Cat-or-Not-An-Image-Classifier-using-Python-and-Ke

Peffers;Tuunanen;Rothenberger;& Chatterjee. (2007). Design science research methodology for information systems research. *Journal of management information systems*, 44-45.

Pitkänen, J. (2019). *AlphaZero shakkikoneena.* Noudettu osoitteesta JYX Digital Repository: http://urn.fi/URN:NBN:fi:jyu-201905172653

Postgresql. (10. 02 2022). *Posygresql feature Matrix*. Noudettu osoitteesta Postgresql Documentation: https://www.postgresql.org/about/featurematrix/

Sammer, E.;& Lai, M. (21. 06 2021). *Flume Wiki*. Noudettu osoitteesta Apache Flume: https://cwiki.apache.org/confluence/display/FLUME/Flume+NG

Shai, S.-S.;& Shai, B.-D. (2014). *Understanding Machine Learning: From Theory to Algorithms.* Cambridge University Press.

Shu, X.;Smiy, J.;Yao, D. (.;& Lin, H. (2013). *Massive Distributed and Parallel Log Analysis For Organizational Security.* Department of Computer Science Virginia Tech.

Suprami, G. (2002). *Impact of Data Quality on Deep Neural Network Training*. University of Michigan.

Traficom. (2021). *Tunnisteet ja tietosuoja. Anonymisointi ja sen rajat.* Noudettu osoitteesta Kyberturvallisuuskeskus: https://www.kyberturvallisuuskeskus.fi/sites/default/files/media/publication/Tunnisteet%20ja%20tietosuoja.pdf

Traficom. (28. 02 2022). *Näin keräät ja käytät lokitietoa*. Noudettu osoitteesta Kybereturvallisuuskeskus: https://www.kyberturvallisuuskeskus.fi/fi/ajankohtaista/ohjeet-ja-oppaat/nain-keraat-ja-kaytat-lokitietoja?toggle=Lokeja%20koskeva%20lains%C3%A4%C3%A4d%C3%A4nt%C3%B6&toggle=Lokitus%20ja%20SIEM

Warren, M. (16. 03 2020). *Solving the protein folding problem with artificial intelligence*. Noudettu osoitteesta Century Science: https://www.centuryscience.org/protein-folding

YLE. (09. 07 2019). *KRP vahvistaa: Lahden tietoverkkoon kesäkuussa kohdistunut hyökkäys oli tahallinen*. Noudettu osoitteesta Yle Uutiset: https://yle.fi/uutiset/3-10868609

LIITE 1 APACHE FLUME KONFIGURAATIO

```
# Agentin komponentit
a1.sources = netcat1,syslog1
a1.sinks = h1,s1
a1.channels = spl,hdp
a1.sources.syslog.type = syslogudp
a1.sources.syslog1.port = 5140
a1.sources.syslog1.host = localhost
a1.sources.syslog1.channels = spl,hdp
a1.sources.netcat1.type = netcat
a1.sources.netcat1.bind = 0.0.0.0
a1.sources.netcat1.port = 6666
a1.sources.netcat1.channels = spl,hdp
# Set channel for hadoopfs
a1.sinks.h1.type = hdfs
a1.sinks.h1_fs.channel = hdp
a1.sinks.h1.hdfs.path = /flume/events
a1.sinks.h1.hdfs.filePrefix = %y-%m-
a1.sinks.h1.hdfs.fileSuffix = log
# set channel for splunk
a1.sinks.s1.channel = spl
# Mapping for multiplexing selector
a1.sources.netcat1.selector.type = multiplexing
a1.sources.netcat1.selector.header = slunkrouting
a1.sources.netcat1.selector.mapping.<Value1> = spl
a1.sources.netcat1.selector.default = hdp
# Mapping for multiplexing selector
a1.sources.syslog1.selector.type = multiplexing
a1.sources.syslog1.selector.header = splunkrouting
a1.sources.syslog1.selector.mapping.<Value1> = spl
a1.sources.syslog1.selector.default = hdp
```

LIITE 2 SPARK OHJELMA FPM SANALISTOILLE

```
from pyspark import SparkContext
from pyspark.ml.feature import StringIndexer
from pyspark.sql.session import SparkSession
from pyspark.sql.functions import split, col

sc = SparkContext("local", "data app")
spark = SparkSession.builder.appName("ML").getOrCreate()

#logfile = "/home/jouni/logs/var/log/boot.log"
#logfile = "/home/jouni/logs/var/log/maillog*"
#logfile = "/home/jouni/logs/var/log/messages*"
#logfile = "/home/jouni/logs/var/log/secure*"
logfile = "/home/jouni/logs/var/log/*"
#logfile = "/home/jouni/gradu/Nov-16.log"

df = spark.read.text(logfile).toDF('logline')

df = df.withColumn("date", split(df['logline'], "[A-Z][a-
z]{2}\s[0-9]{1,}\s[0-9]{2}:[0-9]{2}:[0-9]{2}\d",       li-
mit=1).getItem(0))
df = df.withColumn("message", split(df['logline'], 'combo
',).getItem(1))


df = df.select(split(col("message")," ").alias("words"))


from pyspark.sql.functions import array_distinct
df   =   df.withColumn("words_without_dupes",   array_dis-
tinct("words"))


#df.show()

from pyspark.ml.fpm import FPGrowth
fpGrowth = FPGrowth(itemsCol="words_without_dupes", minSup-
port=0.02, minConfidence=1.0)
model = fpGrowth.fit(df)

# Display frequent itemsets.
#model.freqItemsets.show()

# Display generated association rules.
#model.associationRules.show()
model.associationRules.sort(col("lift").desc()).show(trun-
cate=False)
```

```
from pyspark.sql.functions import col, concat_ws
export   =   model.associationRules.withColumn("antecedent",
concat_ws(" ",col("antecedent")))
export    =    export.withColumn("consequent",    concat_ws("
",col("consequent")))
export.printSchema()
export.write.option("delimiter","|").csv("/tmp/rules.csv")


# transform examines the input items against all the asso-
ciation rules and summarize the
# consequents as prediction
model.transform(df).sort(col("prediction").desc()).show()
```