TAMPEREEN
AMMATTIKORKEAKOULU

LINUX-PALVELIMEN LOKIENSEURANTA

Vilhelm Hahl

Opinnäytetyö
Toukokuu 2019
Tietojenkäsittely
Tietoverkkopalvelut

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tietojenkäsittely
Tietoverkkopalvelut

HAHL, VILHELM:
Linux-palvelimen lokienseuranta

Opinnäytetyö 51 sivua, joista liitteitä 5 sivua
Toukokuu 2019

Lokien seuraaminen on tärkeä osa palvelimen ylläpitoa. Lokeja seurataan, jotta saataisiin tietoa palvelimen tapahtumista. Linux-järjestelmässä syntyy paljon lokeja, ja ylläpitäjän tulee saada kerättyä niistä ylläpidon kannalta olennaiset tiedot. Lokeja voi seurata joko tutkimalla manuaalisesti tekstimuotoisia lokitiedostoja tai käyttämällä jotakin lokienseurantaan tarkoitettua työkalua.

Opinnäytetyö tutki Linux-palvelimen lokienseurantaa yhden lokienseurantajärjestelmän, Logwatchin näkökulmasta. Opinnäytetyön tavoite oli tutkia, miksi lokienseurantaa tarvitaan, miten manuaalinen lokitiedostojen tutkiminen eroaa Logwatchin kaltaisen työkalun käytöstä, kuinka Logwatchia käytetään tehokkaasti, sekä kuinka lokienseurantaa automatisoidaan. Opinnäytetyön tarkoituksena oli asentaa virtuaalikoneeseen Linux-palvelin ja testata siinä Logwatchin toimintaa ja verrata sitä manuaaliseen lokien tutkimiseen.

Opinnäytetyö perehdyttää lukijan Linux-palvelimen lokien seuraamisen yleiskuvaan, antaa katsauksen manuaalisesta lokitiedostojen tutkimisesta sekä kertoo, kuinka lukija voi käyttää Logwatchia palvelimen lokienseurantaan. Opinnäytetyössä selvisi, että rajoituksistaan huolimatta Logwatch helpottaa lokienseurantaa huomattavasti manuaaliseen lokitiedostojen tutkimiseen verrattuna. Tosin Logwatchin rajoitusten vuoksi manuaalista lokitiedostojen tutkimista tarvitaan silti joissakin tilanteissa.

Asiasanat: lokienseuranta, lokitiedostot, logwatch, linux, palvelimet

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in Business Information Systems
Option of Network Services

HAHL, VILHELM:
Tracking the Logs of Linux Server

Bachelor's thesis 51 pages, appendices 5 pages
May 2019

Log tracking is an essential part of server maintenance. Server administrator has to differentiate logs, which are essential for the maintenance. The administrator may inspect log files manually or use a tool for log analysis.

This Bachelor's thesis inspects Linux servers and log tracking from the point of view of one log tracking system, Logwatch. The objective of this study was to examine the need for log tracking, what are the differences between manual inspection of log files and using a tool like Logwatch, how to use Logwatch, and how to automate log tracking. The purpose of this study was to install a Linux server, examine the use of Logwatch, and to compare Logwatch and manual log analysis.

The study gives an overview of log tracking and manual log file inspection and explains how to use Logwatch. The results suggest that Logwatch has certain limitations, but still it enhances the log tracking considerably compared to manual log inspection. Still, the manual inspection may sometimes be needed.

Key words: log tracking, log files, logwatch, linux, servers

SISÄLLYS

ERITYISSANASTO

Daemon	Taustaprosessi, joka käynnistyy tietokoneen käynnistyessä, ja toimii jatkuvasti taustalla
Manuaalisivu	Komentoriviltä käytettävän Linux-ohjelman mukana tuleva, ohjelman käyttöä opastava ohje, jota selataan antamalla komento *man ohjelman-nimi*
Optio	Komennolle annettava asetus, jolla säädetään komentoa ja joka ilmaistaan yhdellä tai kahdella väliviivalla, esimerkiksi *-i* komennossa grep *-i* tai *--ignore-case* komennossa *grep --ignore-case*
Pakettienhallinta	Tyypillinen tapa asentaa ohjelmia Linux-järjestelmään keskitetystä ohjelmalähteestä
Root	Linux-järjestelmissä oleva käyttäjätili, jolla on kaikki oikeudet järjestelmään ilman salasanan kirjoittamista
Stdout	Standardi ulostulo, eli ohjelman tuloste tulostuu päätteeseen
Su	Komento, jolla sisäänkirjautunut käyttäjä voi kirjautua sisään toisena käyttäjänä, kun tietää toisen käyttäjän salasanan
Sudo	Ohjelma, joka sallii komennon suorittamisen rootin oikeuksilla ja joka kirjoitetaan ajettavan komennon eteen

1 JOHDANTO

1.1 Tausta

Lokien seuraaminen on tärkeä osa palvelimien ylläpitoa. Ylläpitäjän on tärkeää pysyä selvillä siitä, mitä palvelimella tapahtuu. Linux-järjestelmät tuottavat kaiken aikaa loke-ja järjestelmän toiminnasta ja tästä lokiaineistosta olisi hyvä saada palvelimen ylläpidon kannalta olennainen tieto ylläpitäjän nähtäville.

Tämä opinnäytetyö keskittyy Linux-järjestelmiin ja tutkii palvelimen lokienseurantaa yhden lokienseurantajärjestelmän, nimeltään Logwatch, näkökulmasta. Opinnäytetyön tulosten on tarkoitus perehdyttää lukija Linux-palvelimen lokien seuraamisen yleiskuvaan, antaa nopea katsaus manuaalisesta lokitiedostojen tutkimisesta, sekä
kertoa kuinka lukija voi käyttää Logwatchia palvelimensa lokienseurantaan.

1.2 Tavoite ja tarkoitus

Opinnäytetyön tavoite on tutkia, mitä hyötyä on lokienseurannasta tai miksi sitä kannattaa tehdä, miten manuaalinen lokitiedostojen tutkiminen eroaa lokien tutkimiseen tarkoitetun työkalun, kuten Logwatch käytöstä, kuinka Logwatchia käytetään tehokkaasti, sekä kuinka lokienseurantaa automatisoidaan ja lokiraportit lähetetään keskitettyyn paikkaan.

Opinnäytetyön tarkoituksena on asentaa virtuaalikoneeseen Linux-palvelin ja testata siinä Logwatchin toimintaa ja verrata sitä manuaaliseen lokien tutkimiseen.

1.3 Työn kulku

Opinnäytetyö alkaa virtuaalisen palvelimen käyttöönotolla. Tämän jälkeen lukijalle annetaan yleiskuva lokienseurannasta ja manuaalisesta lokitiedostojen tutkimisesta. Tämän jälkeen opinnäytetyö kuvaa Logwatchin toimintaa ja toiminnan automatisointia.

Opinnäytetyössä pyritään selittämään asiat niin yksinkertaisesti, että lukija pystyisi ymmärtämään aihetta, vaikka hallitsisi vain hyvät perustiedot tietotekniikasta. Opinnäytetyön raportoinnissa pyritään ottamaan huomioon myös se, että tätä opinnäytetyötä tehdessä lokeja tutkitaan virtuaalisessa laboratorioympäristössä, eikä tuotantoympäristössä. Käytettävä järjestelmä ei siis ole todellisessa palvelinkäytössä palvelemassa todellisia käyttäjiä. Tässä työssä lokit siis suurelta osin syntyvät toimenpiteistä, joilla pyritään saamaan lokeja aikaan. Tuotantoympäristössä syntyvät lokit olisivat todennäköisesti hiukan erilaisia. Tämä pyritään opinnäytetyötä tehdessä ottamaan huomioon siten, että tulokset olisivat yleistettävissä todelliseen käyttöön.

Komentoriville kirjoitettavat komennot on opinnäytetyössä merkitty *kursiivilla*. Selvyyden vuoksi opinnäytetyössä käytetään hakemistojen ja tiedostojen absoluuttisia polkuja. Normaalisti ensin mennään haluttuun hakemistoon, vaikka hakemistoon /var/log/ komennolla *cd /var/log/* ja sitten tulostetaan haluttu tiedosto auth.log komennolla *less auth.log*. Sitten voidaan tulostaa toinen hakemistossa oleva tiedosto. Opinnäytetyön lukeminen on kuitenkin helpompaa, kun ilmoitetaan tiedoston absoluuttinen polku, esimerkiksi *less /var/log/auth.log*. Näin lukijan ei tarvitse muistella, missä hakemistossa kulloinkin oltiin.

2 YMPÄRISTÖN PYSTYTYS

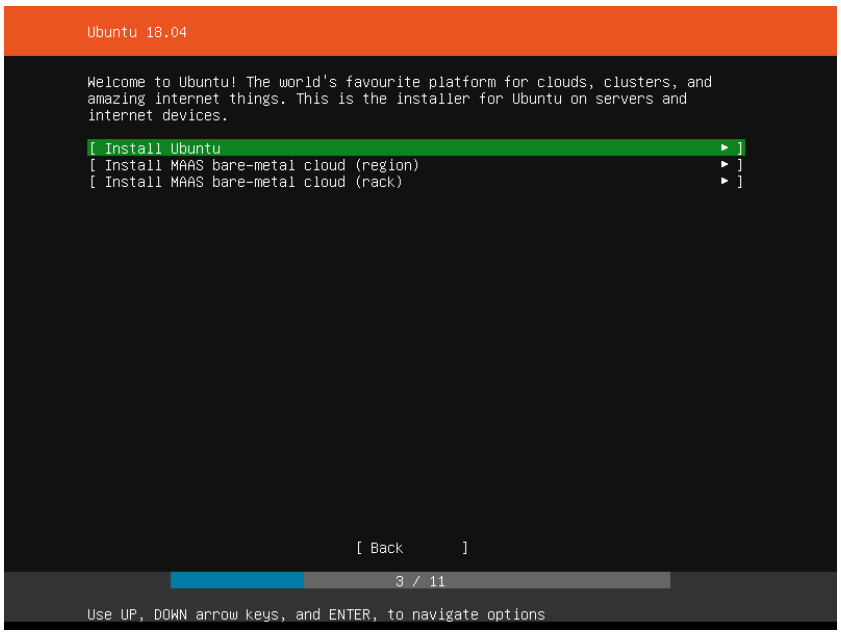
2.1 Käytettävä ympäristö

Opinnäytetyön ympäristönä käytetään virtuaalikoneeseen asennettua Linux-palvelinta. Virtualisointiin käytetään ohjelman VirtualBox versiota 5.2.18. Virtuaalikoneelle annetaan 4096 MB keskusmuistia ja 15 GB levytilaa. Käyttöjärjestelmänä on Ubuntu Server 18.04.1.0, 64-bittinen versio. Koska kyseessä on Linux-palvelin, niin käytössä ei ole graafista käyttöliittymää. Logwatchista on käytössä versio 7.4.3. Lisäksi käytössä on sähköpostinvälitysohjelma Postfixin versio 3.3.0.

2.2 Käyttöjärjestelmän asennus

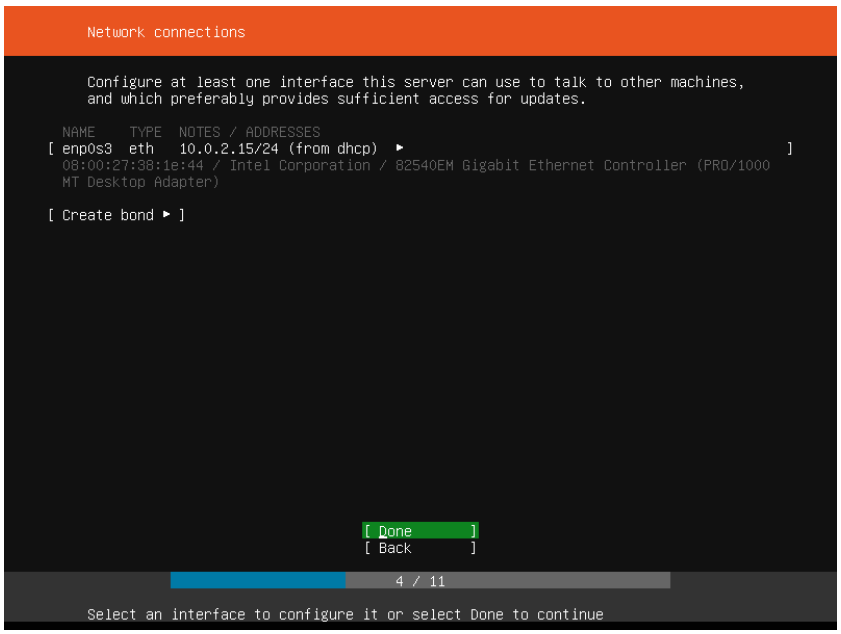
Aluksi luodaan VirtualBoxilla uusi virtuaalikone. Virtuaalikoneen luomisen prosessia ei tässä esitellä. Virtuaalikoneen asetuksista sille annetaan Ubuntu Serverin asennuslevykuva, josta käyttöjärjestelmä asennetaan. Sitten virtuaalikone käynnistetään, jolloin asennus alkaa. Seuraavassa on kuvattu käyttöjärjestelmän asennukseen kuuluvat vaiheet.

1. Ensimmäisessä ruudussa valitaan kieli. On yllättävää, että vaikka asennus on saatavilla vain 16 eri kielellä, niin suomi on silti mukana. Tässä asennus tehdään kuitenkin englanniksi. Näin internetistä löytyviä englanninkielisiä käyttöön liittyviä ohjeita on helpompi hyödyntää.
2. Seuraavaksi valitaan näppäimistöasettelu ja sen variantti, tässä tapauksessa Finnish ja Finnish
3. Valittavana on kolme asennustyyppiä: normaali asennus sekä kaksi MAAS-vaihtoehtoa. MAAS (Metal As A Service) on järjestelmä, joka sallii käyttää fyysisiä palvelimia virtuaalikoneiden tavoin ja mahdollistaa pilvessä olevan suuren palvelinjoukon helpomman hallinnan (Canonical Ltd. n.d.). Valitaan normaali asennus kuten kuvassa 1.



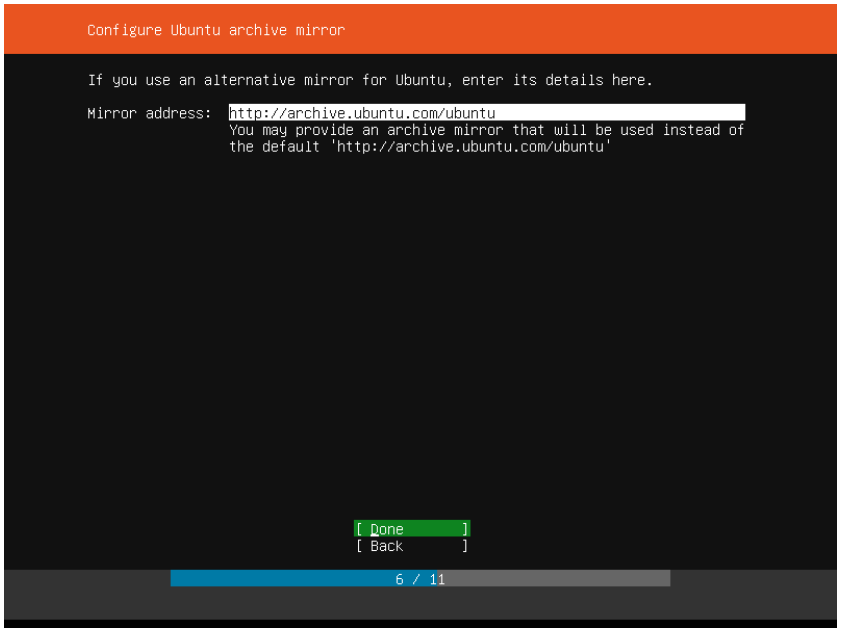
KUVA 1. Käyttöjärjestelmän asennus, asennustyypin valinta

4. Valitaan ja konfiguroidaan käytettävä verkkokortti. Tässä yhteydessä palvelimelle voidaan myös asettaa staattinen IP-osoite. Yleensä palvelimille halutaan IP-osoite, joka ei vaihdu, jotta pystytään aina tietämään millä IP-osoitteella palvelin on tavoitettavissa. Se ei kuitenkaan ole tämän opinnäytetyön kannalta oleellista, joten valitaan automaattiset IP-asetukset dhcp:n kautta kuten kuvassa 2.



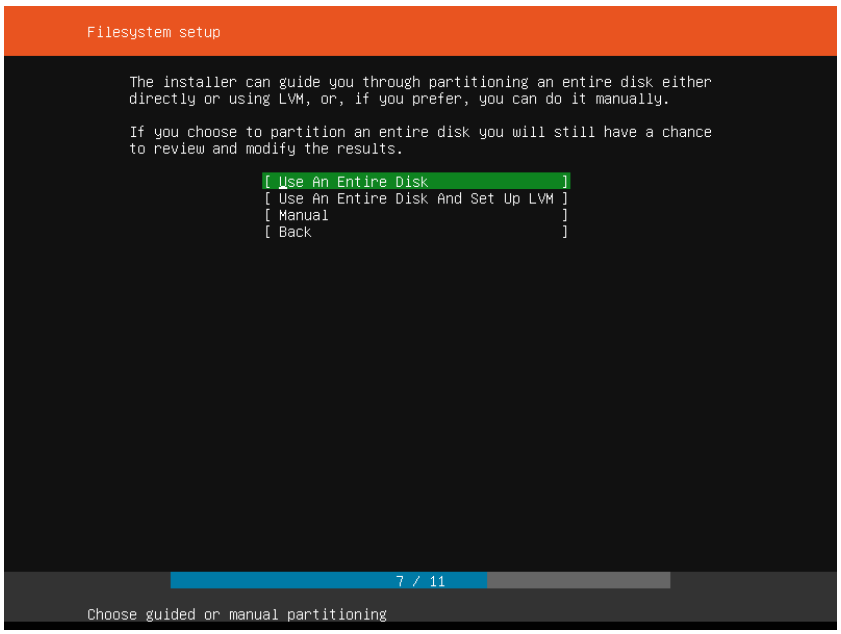
KUVA 2. Käyttöjärjestelmän asennus, verkkokortin asetukset

5. Verkkoyhteyttä varten voidaan määritellä välityspalvelin, jos tarvitaan. Tässä sitä ei tarvita, joten jätetään kenttä tyhjäksi.
6. Tarvittaessa voidaan määritellä jokin muu peilipalvelin ohjelmapakettien lataamiseen, virallisen palvelimen sijasta. Säilytetään virallinen palvelin, kuten kuvassa 3.



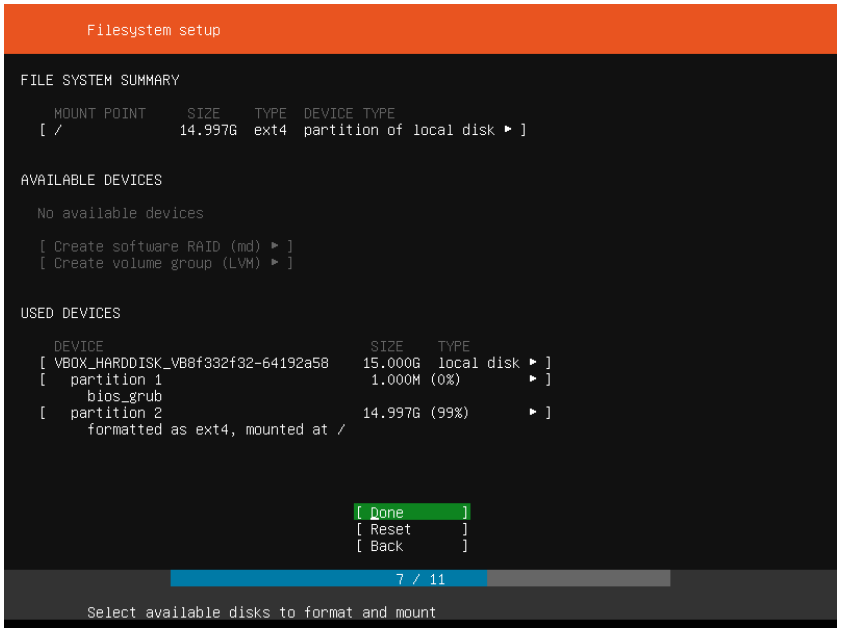
KUVA 3. Käyttöjärjestelmän asennus, käytettävä peilipalvelin

7. Levyn osioinnin suhteen vaihtoehtoina on käyttää koko levyä, tai sama kuin edellinen mutta loogisella taltionhallinnalla (LVM), tai tehdä osiot itse eli manuaalinen osiointi. LVM mahdollistaa osioiden helpomman koon muuttamisen jälkikäteen. Manuaalista osiointia taas käytetään jos halutaan, etteivät kaikki järjestelmän osat ole samalla levyosiolla, vaan eri osat eri osioilla. On monia syitä, miksi näin ehkä haluttaisiin tehdä, mutta ne eivät ole tämän opinnäytetyön kannalta olennaisia, joten käytetään yhtä osiota, kuten kuvassa 4.



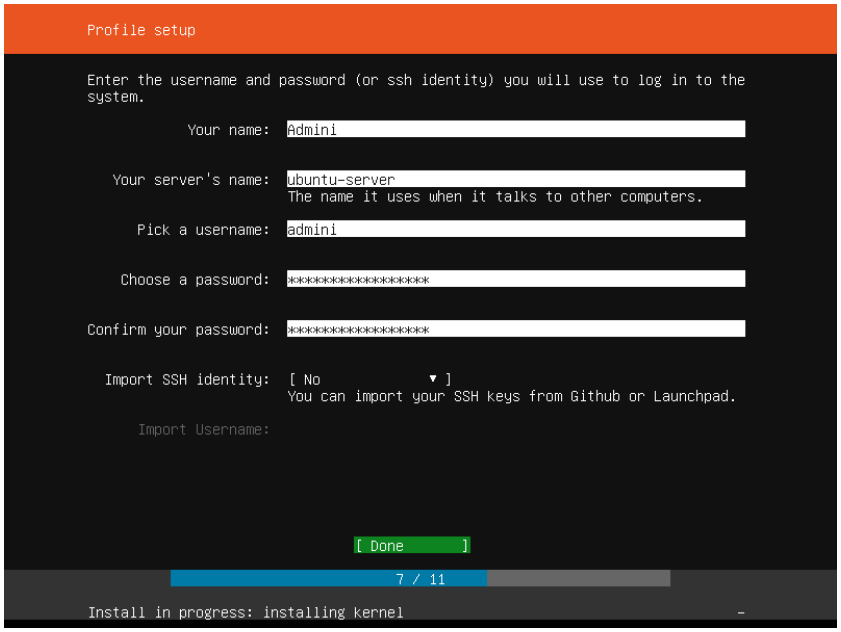
KUVA 4. Käyttöjärjestelmän asennus, osiointityyppi

8. Valitaan levy, jolle järjestelmä asennetaan. Tässä ympäristössä on käytössä vain yksi levy.
9. Asennusohjelma näyttää millainen osiointi ollaan luomassa ja sitä pystyy vielä halutessaan muuttamaan. Hyväksytään osiointi ja aloitetaan asennus kuten kuvassa 5.



KUVA 5. Käyttöjärjestelmän asennus, osiointi

10. Annetaan käyttäjän koko nimeksi Admini, palvelimen isäntänimeksi ubuntu-server, käyttäjänimeksi admini sekä salasana. Tässä olisi myös mahdollisuus tuoda SSH-avaimet palvelusta Github tai Launchpad (kuva 6).



KUVA 6. Käyttöjärjestelmän asennus, profiili

11. Seuraavaksi asennusohjelma antaa mahdollisuuden asentaa palvelinkäyttöön tarkoitettuja snappaketteja, eli tietynlaisia ohjelmapaketteja. Jatketaan valitsematta niitä, koska niitä ei tarvita opinnäytetyön ympäristössä.
12. Käynnistetään järjestelmä uudelleen asennuksen valmistuttua.

2.3 Asennuksen jälkeiset toimet

Asennuksen jälkeen tarkistetaan ja asennetaan päivitykset komennoilla *sudo apt-get update* (päivitä pakettilähteet) ja *sudo apt-get upgrade* (asenna päivitykset).

Komennolla *date* (tulosta päivämäärä ja aika) nähdään, että päivämäärä on oikein, mutta kellonaika 2 tuntia jäljessä, eli aikavyöhyke on väärä. Se korjataan poistamalla tiedosto localtime ja kopioimalla oikea aikavyöhyketiedosto sen paikalle komennoilla *sudo rm /etc/localtime* (poista tiedosto) ja *sudo cp /usr/share/zoneinfo/Europe/Helsinki /etc/localtime* (kopioi tiedosto paikasta toiseen).

Opinnäytetyön myöhempiä vaiheita varten on tarpeen luoda uusi perustason käyttäjä, jolla ei ole oikeutta käyttää sudoa. Annetaan komento *sudo adduser perustyyppi* (luo uusi käyttäjä nimeltään perustyyppi) ja sen jälkeen uuden käyttäjän tiedot. Testataan vielä, ettei Perustyyppi pysty käyttämään sudoa. Kirjaudutaan ulos ja takaisin sisään uudella käyttäjällä, annetaan komento *sudo apt-get update* (päivitä pakettilähteet), ja saadaan ilmoitus ”perustyyppi is not in the sudoers file. This incident will be reported.” Eli perustyyppi ei ole sudoers-tiedostossa. Tästä tapahtumasta raportoidaan. Sudoers on tiedosto, jossa on määritelty käyttäjien oikeudet sudon käyttöön.

Kirjaudutaan takaisin sisään käyttäjänä Admini. Koska Logwatch löytyy suoraan Ubuntun pakettienhallinnan kautta, se voidaan asentaa helposti komennolla *sudo apt-get install logwatch* (asenna paketti). Logwatchin mukana asennetaan automaattisesti myös Postfix-sähköpostinvälitysohjelma. Sitä asennettaessa avautuu ruutu, jossa sen asetukset voidaan tehdä (kuva 9 luvussa 5.2). Valitaan toistaiseksi ”No configuration”, eli ei tehdä mitään asetuksia. Postfixin asetukset tehdään opinnäytetyön myöhemmässä vaiheessa, luvussa 5.2, ”Lokiraporttien lähettäminen sähköpostilla”.

3 LOKIEN SEURAAMINEN YLEISESTI

3.1 Miksi seurata lokeja?

Lokien seuraaminen kuuluu palvelimen ylläpidon perusrutiineihin. Lokeja seurataan siksi, että saataisiin tietoa palvelimen tapahtumista. Palvelimella on jatkuvasti jonkinlaisia tapahtumia, jotka johtuvat niin normaalista käytöstä kuin myös ylläpidollisista toimista. Lokien seuraaminen auttaa palvelimen ylläpitäjää pysymään selvillä siitä, millaisessa tilassa palvelin on. Linux-järjestelmissä on sisäänrakennettuina hyvät lokitusjärjestelmät, ja käyttöjärjestelmässä syntyy käyttäjän näkymättömissä jatkuvasti suuret määrät lokeja. Suurin osa lokeista on sellaisia, jotka eivät ylläpitäjää kiinnosta, ja lokien seasta tulisikin pystyä löytämään ylläpidon kannalta olennaiset lokit.

Lokeja voi seurata tai tutkia kahdella tavalla. Vähemmän ylläpidollista työtä aiheuttava tapa on tutkia lokeja jälkikäteen, kun selvitetään jotakin ongelmaa tai tapahtumaa. Tällöin siis jotakin ongelmallista on jo tapahtunut, ennen kuin lokeja tutkitaan. Tämä lähestymistapa lokien seurantaan ei aiheutakaan välttämättä yhtään ylläpidollista työtä silloin, kun kaikki on kunnossa, tai pikemminkin kun asioiden uskotaan olevan kunnossa. Säännöllisen lokienseurannan laiminlyöminen voi kuitenkin kostautua silloin, kun ongelma on jo päässyt tapahtumaan, ja ongelman selvitystyö on käynnissä samalla, kun verkossa on ongelmia.

Parempi, joskin työläämpi lähestymistapa on ennakoiva lokienseuranta, joka tarkoittaa säännöllistä lokien tutkimista. Tällä pyritään pitämään yllä hyvää tilannekuvaa palvelimen tilasta ja havaitsemaan ongelmat nopeasti tai mahdollisesti jopa estämään ongelmien syntyminen tai paheneminen. Ellei ylläpitäjä ole erikseen konfiguroinut palvelimelle mitään hälytyksiä, niin palvelin ei juuri ilmoittele, jos jokin ei toimi normaalisti. Se vain kirjaa asian lokitiedostoon. Niin kauan, kun palvelimen tarjoamat palvelut näyttävät ulkoisesti toimivan, ylläpitäjä ei osaa epäillä, että jokin olisi vialla, ellei aktiivisesti seuraa lokeja.

Tähän asti on kerrottu lähinnä erilaisista vikatilanteista, ja niiden aiheuttamasta tarpeesta seurata lokeja. Todennäköisesti vielä suurempi syy lokien seuraamiseen on kuitenkin tietoturva. Taitava tietomurtaja pyrkii murtautumaan järjestelmiin niin, että sitä ei huo-

mattaisi. Tällöin palvelimen tarjoamiin palveluihin ei pitäisi tulla häiriöitä, jotka herättäisivät ylläpitäjän huomion. Ellei ylläpitäjä seuraa, mitä palvelimella tapahtuu, hyökkääjä saa toimia palvelimella vapaasti, kunhan ei sotke palvelimen normaalia toimintaa. Ellei käytössä ole tunkeutumisenhavaitsemisjärjestelmää tai muita turvajärjestelmiä, niin lokien seuranta saattaa jäädä ainoaksi tavaksi havaita tunkeutuminen.

3.2 Lokien poistaminen automaattisesti - Logrotate

Vaikka lokeja tutkittaisiin vain jälkeenpäin, tarvitaan silti jokin tapa varmistaa, että lokit ovat varmasti saatavilla silloin kun niitä tarvitaan. Ylläpitäjähän saattaa poistaa vanhoja lokeja levytilan vapauttamiseksi. Esimerkiksi tietomurron tutkinnassa saatetaan tarvita vanhoja lokeja, jotka on jo poistettu. Tämän vuoksi lokienseurantaa varten kannattaa joka tapauksessa olla jonkinlainen systeemi, jolla olennaiset lokit saadaan talteen.

Ubuntussa onkin valmiina ohjelma, joka automaattisesti poistaa vanhoja lokeja. Vanhat lokit eivät siis välttämättä ole enää saatavilla, vaikka ylläpitäjä ei olisikaan manuaalisesti poistanut niitä. Kyseinen ohjelma, Logrotate, ”kierrättää” (englanniksi rotate) lokeja. Lokien kierrättäminen tarkoittaa, että vanha lokitiedosto arkistoidaan ja asetuksista riippuen pakataan, ja lokitus aloitetaan uudestaan tyhjästä tiedostosta. Logrotate säilyttää asetuksissa määritellyn määrän kierrätettyjä versioita lokitiedostoista. Kun tuo määrä ylittyy, vanhat lokitiedostot poistetaan automaattisesti. Linux ja eri ohjelmat tuottavat jatkuvasti lokitietoja, ja ellei niitä ajoittain poistettaisi, ne saattaisivat lopulta täyttää koko levyn.

Logrotaten oletusasetukset sijaitsevat tiedostossa /etc/logrotate.conf, ja hakemistossa /etc/logrotate.d/ sijaitsevat yksittäisten ohjelmien lokien asetukset, joilla voidaan korvata oletusasetukset tietyn ohjelman lokien kohdalla. Kyseiseen hakemistoon ylläpitäjä voi myös tehdä omia asetuksiaan ohjelmille, joiden lokeille ei valmiina ole omia asetuksia. Logrotaten manuaalisivun mukaan Logrotaten asetuksissa voi määrittää muun muassa, pakataanko kierrätetyt lokit vai ei, kuinka usein lokit kierrätetään (tunneittain, päivittäin, viikoittain kuukausittain, vuosittain), kuinka monta päivää vanhat lokit kierrätetään, mitä kokoa suuremmat lokitiedostot kierrätetään, kuinka monta versiota lokitiedostosta säilytetään ja paljon muuta. (Troan, Brown & Kaluza 2002.) Manuaalisivun mukaan normaalisti käyttöjärjestelmä ajaa Logrotaten kerran päivässä, jolloin sitä ti-

heämmin (esimerkiksi tunneittain) toistuva lokien kierrättäminen vaatii, että ylläpitäjä itse konfiguroi käyttöjärjestelmän ajamaan Logrotaten tiheämmin. Ohjelmien tietyin väliajoin tapahtuvaan automaattiseen ajoon käytetään cron-nimistä työkalua, jota käsitellään tässä opinnäytetyössä myöhemmin luvussa 5.1.

Tiedostossa /etc/logrotate.conf olevista oletusasetuksista nähdään esimerkiksi, että niiden ohjelmien lokit, joille ei ole omia asetuksia, kierrätetään viikoittain ja lokitiedostojen versioita säilytetään neljä. Näitäkin asetuksia ylläpitäjä tietenkin pystyy muuttamaan. Joskus voi olla tilanne, että jokin ohjelma tuottaa tavattoman paljon lokia, niin että levytilaa kuluu liikaa. Tällöin ylläpitäjän kannattaa määrittää tämän ohjelman lokit kierrätettäväksi tavallista tiheämmin ja/tai pienentää säilytettävien vanhojen lokitiedostojen määrää. Vaihtoehtoisesti ylläpitäjä voi yrittää selvittää syytä suurelle lokitiedoston koolle ja vähentää lokitusta.

3.3 Manuaalinen lokitiedostojen tutkiminen

Kun suunnitellaan lokienseurannan aloittamista, niin ensimmäinen vaihtoehto, joka monelle varmaankin tulee mieleen, on manuaalinen lokitiedostojen tutkiminen tekstieditorilla tai tulostamalla niitä näytölle. Valtaosa Linuxin ja sen ohjelmien tuottamista lokeista on perustekstitiedostoja, jolloin niitä pystyy tutkimaan yksinkertaisesti avaamalla ne millä tahansa perustekstitiedostoja käsittelevällä tekstieditorilla. Tämä on yksikertaisin lokien tutkimisen muoto siinä mielessä, että se ei vaadi mitään erikseen asennettavaa järjestelmää tai ohjelmaa. Yksinkertaisimmillaan lokeja ei tarvitse edes lähettää minnekään; tarvitsee vain ottaa yhteys palvelimeen ja avata tiedosto tekstieditoriin tai tulostaa se näytölle. Nopeasti ajateltuna voisi luulla, että tällä tavalla lokien seuraaminen on kaikkein helpointa, mutta todellisuudessa asia ei ole niin yksinkertainen.

Ensiksi katsotaan, missä lokitiedostot sijaitsevat. Filesystem Hierarchy Standard määrittelee, että hakemistoa /var/log/ ja sen alihakemistoja käytetään lokien säilyttämiseen (LSB Workgroup & The Linux Foundation 2015). Valtaosa Linux-jakeluista, myös Ubuntu, toimii näin. Hakemiston /var/log/ alta löytyy opinnäytetyössä käytetystä virtuaalikoneesta sekalaisia lokitiedostoja sekä muutama alihakemisto. Tämä ilmenee komennoilla *cd /var/log/* (siirry hakemistoon) ja *ls* (listaa hakemiston tiedostot).

Hakemistossa näkyvät tietenkin lokit vain niiltä daemoneilta ja ohjelmilta, jotka ovat jo lokittaneet tiedostoihin jotakin. Jos siis palvelimelle asennetaan vaikka Apache www-palvelin, sen lokia alkaa ilmestyä hakemistoon vasta sen jälkeen, kun se on toiminnassa. Lokitiedostojen määrä siis lisääntyy sitä mukaa, kun uusia ohjelmia asennetaan. Useimmilla daemoneilla on oma lokitiedosto, joka on yleensä samanniminen kuin daemon itse. Jotkin käyttävät useampaa lokitiedostoa, jotka sijaitsevat omassa alihakemistossaan. Kaikilla ei kuitenkaan ole omaa lokitiedostoa, ja tällöin ne yleensä kirjaavat lokiaan syslogiin, eli system logiin.

Tärkeimpiä Ubuntun lokitiedostoja on lueteltu esimerkiksi Ubuntun Community Help Wikissä (Community Help Wiki 2018). Tässä niistä joitakin tärkeimpiä:

- Authorization log, eli valtuutusloki sijaitsee tiedostossa /var/log/auth.log. Se pitää kirjaa käyttäjien valtuuttamisesta, esimerkiksi sudon käyttämisestä ja sisäänkirjautumisesta paikallisesti ja etänä SSH:lla. Aiemmin opinnäytetyössä, kun käyttäjällä Perustyyppi yritettiin käyttää sudoa, saatiin virheilmoitus: ”perustyyppi is not in the sudoers file. This incident will be reported.” Tämä tapahtuma raportoidaan juuri valtuutuslokiin, josta näkee, että kyseinen käyttäjä on yrittänyt käyttää sudoa. Mukana on päivämäärä ja kellonaika, sekä mitä komentoa käyttäjä yritti ajaa.
- Debug log sijainnissa /var/log/debug kirjaa virheviestejä järjestelmältä ja sovelluksilta, jotka lähettävät lokeja syslogd-järjestelmän kautta debug-tasolla.
- Kernel Log sijainnissa /var/log/kern.log pitää lokia Linux-ytimen viesteistä.
- System Log sijainnissa /var/log/syslog pitää lokia järjestelmän osista ja ohjelmista, joille ei ole omaa lokitiedostoa.
- Joitakin lokeja ei ole tarkoitettu suoraan ihmisen luettavaksi, vaan jonkin ohjelman käytettäväksi. Näitä ovat esimerkiksi seuraavat:
    - Login failures sijainnissa /var/log/faillog on tarkoitettu käytettäväksi komennolla *faillog*, ja se näyttää epäonnistuneet kirjautumisyritykset. Kokeillessani komentoa se ei tosin toiminut suoraan niin kuin voisi olettaa. Se ei tulostanut mitään, ja edes optiolla *--all* (komento *faillog --all*) se ei ilmoittanut epäonnistuneita kirjautumisyrityksiä tapahtuneen, vaikka olin sellaisia tehnyt, sekä paikallisesti että SSH:n kautta. Selvityksistä huolimatta en saanut selvitettyä, miten komennon saisi toimimaan, mutta tämä yksi komento ei välttämättä ole opinnäytetyön kannalta välttämätön. Onnistuneet ja epäonnistuneet kirjautumisyritykset kirjautuvat myös valtuutuslokiin /var/log/auth.log.

- Last logins sijainnissa /var/log/lastlog on tarkoitettu käytettäväksi komennolla *lastlog*, ja se näyttää kunkin käyttäjän viimeisimmän kirjautumisen. Sen tulosteesta näkee myös IP-osoitteen, josta viimeksi otettiin SSH-yhteys, jos viimeisin kirjautuminen tapahtui etänä.
- Login records sijainnissa /var/log/wtmp on tarkoitettu käytettäväksi komennolla *who* näyttämään tällä hetkellä sisäänkirjautuneet käyttäjät.

Paljon muitakin tärkeitä lokitiedostoja on, mutta tässä ei ole tarpeen luetella niitä kaikkia. Edellä kuvattuja poikkeuksia lukuun ottamatta lokit ovat yleensä perustekstitiedostoja ja niitä voidaan lukea tekstieditorilla. Palvelimella, jolla ei ole graafista käyttöliittymää, ei tietenkään ole graafisia tekstieditoreita vaan tekstipohjaisia editoreita. Lisäksi on muita ohjelmia, joilla pystyy tulostamaan tekstiä tekstitiedostoista. Tässä opinnäytetyössä käytetään tekstieditoria nimeltä Nano. Lokitiedoston, vaikkapa valtuutuslokin pystyy avaamaan tekstieditoriin komennolla *nano /var/log/auth.log*. Lokitiedostoja ei kuitenkaan ole tarkoitus muokata, vaan ainoastaan katsella, joten tekstieditoria ei tarvita. Komennolla *cat tiedosto* voidaan tulostaa tekstitiedoston sisältö näytölle. Se ei kuitenkaan kelpaa kovin pitkien tiedostojen tulostamiseen, koska koko tiedoston sisältö tulostuu näytölle yhtenä pötkönä, eikä päätteessä pysty rullaamaan ylöspäin nähdäkseen, mitä ruudulla näkyvää tekstiä ennen tulostui. Jos siis tekstiä on enemmän, kuin mitä ruudulle kerralla mahtuu, täytyy käyttää sivutusohjelmaa *less*, joka mahdollistaa tekstissä ylös- ja alaspäin siirtymisen. Sivutusohjelmasta poistutaan painamalla näppäintä q. Hyödyllisiä komentoja ovat myös *zcat* ja *zless*, joilla pystyy tulostamaan pakatun tiedoston sisällön. Näitä komentoja voi käyttää silloin, kun haluaa tutkia Logrotaten kierrättämiä pakattuja lokitiedostoja. Näin niitä ei tarvitse manuaalisesti ensin purkaa. Komentoja *zcat* ja *zless* käytetään muuten samalla tavalla kuin komentoja *cat* ja *less*, mutta pakattuihin tiedostoihin.

Usein koko tiedoston tutkiminen ei ole tarkoituksenmukaista, joten siitä voidaan tulostaa vain osa. Lokitiedostoissa uusimmat tiedot näkyvät tiedoston alaosassa ja ne voidaan tulostaa komennolla *tail* (tulosta tiedoston loppu). Oletuksena *tail* tulostaa tiedoston 10 viimeistä riviä ja lukumäärää voidaan muuttaa optiolla *-n*. Jos siis halutaan vaikka tietää, mitä APT-pakettienhallintaohjelma on viime aikoina tehnyt, voidaan käyttää komentoa *tail -n 20 /var/log/apt/history.log* (tulosta tiedoston 20 viimeistä riviä). Hyödyllinen komento on myös *tail -f*, joka seuraa tiedoston kasvua. Komento jää seuraamaan tiedostoa, ja kun tiedostoon tulee uusia rivejä, ne näytetään heti ruudulla. Komennosta

on hyötyä lähinnä vianselvityksessä, kun yritetään toistaa ongelmia aiheuttava toimenpide. Tiedoston seuraaminen lopetetaan painamalla *control+c*.

Tiedostoista voi myös etsiä hakusanoilla haluttua asiaa käyttämällä komentoa *grep*. Kuvitellaan, että ylläpitäjä haluaa nähdä epäonnistuneet kirjautumisyritykset ja hakee hakusanoilla ”authentication failure”. Näitä merkkijonoja löytyy valtuutuslokista sekä isolla että pienellä alkukirjaimella, joten kannattaa käyttää optiota *-i* tai *--ignore-case*, jolloin molemmat tapaukset löytyvät. Jos tulostetta tulee enemmän kuin mitä ruudulle kerralla mahtuu, voi tulosteen vielä putkittaa sivutusohjelmalle *less*. Putkitus tarkoittaa, että tuloste ohjataan syötteeksi toiselle ohjelmalle, ja se tehdään merkillä |. Ylläpitäjä käyttää siis komentoa *grep -i ”authentication failure” /var/log/auth.log | less*. Hakusanojen etsimiseen lokitiedostoista voi käyttää myös optioita *-A* ja *-B*, joilla *grep* näyttää täsmäävän rivin lisäksi tietyn monta riviä sen jälkeen ja sitä ennen. Esimerkiksi kernelilokista voidaan etsiä virheitä komennolla *grep -i -A 3 -B 3 ”error” /var/log/kern.log | less*. Näin saadaan nähdä, mitä muuta on tapahtunut samaan aikaan etsittävän tapahtuman kanssa.

Vaikka manuaalinen lokienseuranta saattaa aluksi vaikuttaa yksinkertaiselta, se käy kuitenkin työlääksi, jos sitä alkaa tekemään laajemmassa mittakaavassa. Jos lokeista on tarkoitus säännöllisesti tarkistaa vain joitakin yksittäisiä asioita, kuten epäonnistuneet kirjautumisyritykset kuten edellä, niin sen toki voi näinkin tehdä. Jos taas tarkoituksena on seurata lokeja laajemminkin ja ylläpitää hyvää tilannekuvaa siitä, mitä palvelimella tapahtuu, niin tutkittava lokiaineisto kasvaa melko suureksi. Lokitiedostoista hakeminen hakusanojen avulla toimii sekin vain, jos tietää, mitä on etsimässä. Kun palvelimella on lukuisia seurattavia palveluita, niin tutkittavia lokitiedostojakin on paljon. Eikä monilla organisaatioilla ole pelkästään yhtä palvelinta, vaan monia. Ylläpitäjän pitäisi siis ottaa yhteys erikseen jokaiselle palvelimelle ja tutkia kaikki olennaiset lokitiedostot. Tämä alkaa olla jo niin työlästä, että sellainen jää helposti tekemättä, muulloin kuin silloin kun selvitetään jotakin ongelmaa. Ja kuten aiemmin todettiin, vanhat lokit eivät välttämättä ole enää saatavilla silloin, kun selvitetään, mitä palvelimella on joskus aiemmin tapahtunut.

Sanoisin siis, että jos lokeja aikoo seurata vähänkään laajemmin ja ennakoivasti, eikä vasta kun jokin ongelma on jo tapahtunut, niin siinä kannattaa käyttää apuna jotakin

työkalua, joka helpottaa toimintaa ja auttaa tekemään siitä rutiinia. Seuraavaksi opinnäytetyössä tutkitaan yhtä tällaista työkalua.

4 LOGWATCH

4.1 Yleistä

Koska manuaalisessa lokienseurannassa on omat haasteensa, on Linuxille saatavilla lukuisia ohjelmia, joiden tarkoituksena on helpottaa lokienseurantaa. Tässä opinnäytetyössä tutkitaan seuraavaksi lokien analysointiin tarkoitettua järjestelmää nimeltä Logwatch. Se on avoimen lähdekoodin vapaa ohjelmisto, ja julkaistu MIT-lisenssillä (Bjorn, Bauer, Tremaine & Poplawski 2019). Logwatchin perusidea on skannata lokit ja koostaa niistä raportti perustuen ylläpitäjän määrittelemään listaan seurattavista palveluista. Yleensä halutaan, että Logwatch skannaa lokit päivittäin ja lähettää lokiraportin ylläpitäjälle.

Linuxille on saatavilla valtavasti erilaisia lokienseurantajärjestelmiä, joten miksi opinnäytetyöhön valittiin juuri Logwatch? En väitä, että Logwatch olisi välttämättä paras saatavilla oleva järjestelmä, varsinkaan kaikkiin mahdollisiin käyttöympäristöihin tai tarkoituksiin, mutta Logwatch hoitaa homman ilman pitkää perehtymistä tai monimutkaisten tietojärjestelmien käyttöönottoa. Logwatch on perustasolla melko helppo käyttää ja nopea ottaa käyttöön, jos tietää vähänkään Linux-ylläpidosta ja komentorivin käytöstä. Se on myös ilmainen ja vapaasti käytettävissä, eikä ylläpitäjän tarvitse pähkäillä lisenssikysymyksiä, kuten vaikka sitä kuinka monelle palvelimelle sen saa asentaa. Se on saatavilla Ubuntulle ja todennäköisesti monille muillekin Linux-jakeluille suoraan pakettienhallinnan kautta, joten sen asentaminen on hyvin nopeaa ja helppoa. Se ei myöskään vaadi graafista käyttöliittymää, joten kaikki sen käyttöön liittyvät osa-alueet pystyy tekemään palvelimella komentorivin kautta.

4.2 Asetuksien tekeminen

Logwatchin asetukset sijaitsevat tiedostossa /usr/share/logwatch/default.conf/logwatch.conf. Asetustiedostoa voisi alkaa suoraan muokkaamaan, mutta se ei ole suositeltavaa. Jos myöhemmin haluttaisiin palauttaa oletusasetukset, niitä ei olisikaan enää missään. Ainoa tapa saada oletusasetukset takaisin, lukuun ottamatta sitä, että hankkisi ne varmuuskopiosta tai jostain ulkopuolisesta lähteestä, olisi poistaa Logwatch asetustie-

dostoineen ja asentaa se uudelleen. Oikeaoppinen tapa kuitenkin on tehdä asetukset jonnekin muualle, jolloin asennushakemistoon jää oletusasetukset. Filesystem Hierarchy Standard määrittelee, että hakemistoa /etc ja sen alihakemistoja käytetään isäntäkohtaisten konfiguraatiotiedostojen säilyttämiseen (LSB Workgroup & The Linux Foundation 2015). Logwatchia asennettaessa onkin jo automaattisesti luotu hakemisto /etc/logwatch/conf/, jonne asetustiedosto voidaan kopioida komennolla *sudo cp /usr/share/logwatch/default.conf/logwatch.conf /etc/logwatch/conf/*. Ellei /etc-hakemistossa ole asetustiedostoa, Logwatch käyttää asetuksia tiedostossa /usr/share/logwatch/default.conf/logwatch.conf, muuten se käyttää automaattisesti /etc-hakemistossa olevaa asetustiedostoa. Näin siis periaatteessa, siitä lisää tämän alaluvun lopussa. Asetustiedostossakin olevat asetukset voidaan yli ajaa komentoriville annettavilla optioilla Logwatchia ajettaessa.

Seuraavaksi tehdään Logwatchin asetukset. Avataan asetustiedosto rootin oikeuksilla tekstieditoriin Nano komennolla *sudo nano /etc/logwatch/conf/logwatch.conf*. Asetukset ovat muodossa asetus = arvo. Asetuksien tekemistä auttaa huomattavasti se, että asetustiedostossa on paljon kommentteja, jotka selittävät asetuksia. Kommentit ovat merkittynä risuaidalla ja ylläpitäjäkin voi lisätä omia kommenttejaan. Seuraavassa on listattuna kaikki eri asetukset:

- LogDir: oletus lokihakemisto, jonka alta lokeja etsitään. Oletuksena /var/log/. Normaalisti tätä ei tarvitse muuttaa.
- TmpDir: väliaikaishakemisto, oletuksena /var/cache/logwatch/. Kyseistä hakemistoa ei ole valmiiksi luotu, joten sen voi joko vaihtaa yleiseen väliaikaishakemistoon /tmp/ tai luoda kyseisen hakemiston komennolla *sudo mkdir /var/cache/logwatch* (luo hakemisto), niin kuin tässä tehdään. Tästä tulee eräs ongelma: kun Logwatchia yrittää ajaa ilman sudoa, saadaan virheilmoitus ”You do not have permission to create temporary directory under /var/cache/logwatch. You are not running as superuser.” Eli käyttäjällä ei ole oikeuksia väliaikaishakemiston luomiseen hakemiston /var/cache/logwatch alle. Ongelmaan on neljä eri ratkaisua.
    - Ensiksikin Logwatchin väliaikaishakemistoksi voidaan vaihtaa /tmp/.
    - Toiseksi Logwatchia voidaan ajaa aina sudon kanssa. Näin sitä ajetaan rootin oikeuksilla, eli suurimmilla mahdollisilla oikeuksilla. Tästä seuraa se, että ylläpitäjä joutuu aina kirjoittamaan salasanansa ajaessaan Logwatchia manuaalisesti. Lisäksi ei ole parhaiden käytäntöjen mukaista ajaa rootin oikeuksilla ohjelmaa, joka ei niitä välttämättä tarvitse.

- Kolmas vaihtoehto on antaa kaikille käyttäjille kyseinen oikeus, josta virheilmoitus valittaa. Komennolla *ls -l /var/cache/* (tulosta hakemiston sisältö lisätietoineen) nähdään, että hakemiston /var/cache/logwatch oikeudet ovat drwxr-xr-x root root. D tarkoittaa hakemistoa. Ensimmäiset rwx tarkoittavat, että hakemiston omistajalla root on luku-, kirjoitus- ja suoritusoikeus. Hakemiston kohdalla suoritusoikeus tarkoittaa oikeutta avata hakemisto. Seuraavat r-x tarkoittavat, että ryhmään root kuuluvilla on luku- ja suoritusoikeus. Viimeiset r-x tarkoittavat samaa kaikkien muiden käyttäjien kohdalla. Komennolla *sudo chmod o+w /var/cache/logwatch/* (muuta tiedoston tai hakemiston oikeuksia) voidaan antaa kaikille käyttäjille kirjoitusoikeus hakemistoon. Ei kuitenkaan ole parhaiden käytäntöjen mukaista antaa käyttäjille enempää oikeuksia, kuin he välttämättä tarvitsevat.
- Neljäs vaihtoehto on muuttaa hakemiston omistajuus tai ryhmä. Hakemistolle voidaan muuttaa ryhmäksi admini komennolla *sudo chown root:admini /var/cache/logwatch/* (muuta omistajuus tai ryhmä). Sitten vielä annetaan ryhmälle kirjoitusoikeus komennolla *sudo chmod g+w /var/cache/logwatch/* (muuta oikeuksia). Näin ainoastaan ryhmään admini kuuluvat pystyvät ajamaan Logwatchia, muut sudon avulla. Jos palvelimella on useita ylläpitäjiä, joiden pitää pystyä ajamaan Logwatchia, mielellään ilman sudoa, tämä ratkaisu ei ole ihanteellinen, ellei sitten hakemiston ryhmäksi laiteta jotakin sellaista ryhmää, johon kaikki, tai vain valitut ylläpitäjät kuuluvat. Opinnäytetyöhön tämä ratkaisu kelpaa hyvin.

- Output: ulostulo. Vaihtoehtoina ovat mail (lähetä sähköpostilla), file (tallenna tiedostoon) ja stdout (standardi ulostulo, tulosta päätteeseen). Pidetään toistaiseksi oletus stdout.
- Format: muotovaihtoehtoina text ja html. Pidetään oletus eli teksti.
- Encode: koodaus, vaihtoehtoina base64 ja none. Pidetään oletus, eli ei koodausta.
- MailTo: minne sähköposti lähetetään, mikäli Logwatchin ulostuloksi on asetettu sähköposti. Se voi olla paikallinen tili tai sähköpostiosoite. Pidetään toistaiseksi oletus Root.
- Mailto_host1: lähetä tietyn isännän lokiraportit tiettyyn osoitteeseen. Tätä voidaan käyttää silloin, kun Logwatchia ajetaan järjestelmässä, johon kerätään lokeja useilta isänniltä. Tällöin sanan host1 paikalle laitetaan halutun koneen isäntänimi ja arvoksi sähköpostiosoite, johon kyseisen isännän raportit lähetetään.

Tämä asetus on oletuksena kommentissa, ja sen käyttöön ottamiseksi siitä pitää ottaa #-merkki pois.

- MailFrom: mitä merkitään sähköpostin lähettäjäkenttään. Pidetään oletus eli Logwatch.
- Filename: mihin tiedostoon raportti tallennetaan, mikäli ulostulona on tiedosto. Huomaa, että raportti tallennetaan aina samaan tiedostoon, korvaten vanha tiedosto. Oletuksena asetus on kommentissa ja arvona on /tmp/logwatch. Mikäli ulostulona halutaan käyttää tiedostoa, niin tiedoston sijainnin kannattaa toki olla jokin muu kuin väliaikaishakemisto, ettei tiedostoa poisteta itsestään. Arvoksi voi laittaa vaikka /home/admini/logwatch.
- Archives: skannataanko myös vanhat kierrätetyt lokit. Oletuksena asetus on kommentoitu, ja ilman asetuksen määrittämistä kierrätetyt lokit skannataan. Omissa testeissäni ilmeni, että tästä asetuksesta riippumatta Logwatch skannaa aina viimeisimmän arkistoidun lokitiedoston. Esimerkiksi valtuutuslokin nimi on auth.log. Kun se kierrätetään, siitä tulee auth.log.1 ja toisella kierrätyksellä auth.log.2.gz. Pääte gz tarkoittaa, että tiedosto on pakattu. Logwatch skannasi aina myös tiedoston auth.log.1 vaikka arkistojen skannaus oli pois päältä. Samoin kävi myös muiden lokien suhteen. Pidetään arkistojen skannaus päällä.
- Range: miltä ajalta lokit skannataan. Vaihtoehtoina ovat All (kaikki), Today (tänään) ja Yesterday (eilen). Jos halutaan skannata kaikki päivän aikana syntyvät lokit, niin vaihtoehdossa Today on järkeä vain, mikäli skannaus tehdään aivan vuorokauden lopulla. Valintaa voidaan käyttää myös, jos tiedetään, että tämän päivän aikana on tapahtunut jotakin, josta halutaan tietoa. Kaikkien lokien skannaus taas tuottaa raportin, josta ei pysty näkemään, minä päivänä mikäkin asia on tapahtunut, vaan eri päivien lokit ovat sekaisin. Yleensä halutaan, että lokit skannataan kerran päivässä, ja kunkin päivän lokit ovat omassa raportissaan. Tällöin valinta Yesterday on järkevin, ja se onkin oletus.
- Detail: kuinka yksityiskohtainen raportti on. Vaihtoehtoina ovat Low (matala), Med (keskitaso) ja High (korkea). Arvon voi antaa myös numerona, jolloin 0 vastaa arvoa Low, 5 arvoa Med ja 10 arvoa High. Pidetään toistaiseksi oletus, eli Low.
- Service: minkä palveluiden lokit skannataan. Tähän on kaksi lähestymistapaa. Ensimmäinen tapa on antaa arvoksi All (eli kaikki, tämä on oletus), ja uusilla Service-riveillä ilmoittaa palveluita, jotka jätetään pois. Näiden rivien arvojen eteen merkitään miinusmerkki (esimerkiksi Service = -dhcpd). Toinen tapa on

poistaa oletus, eli All, ja erikseen merkitä jokainen seurattava palvelu. Lista palveluista, joita Logwatch pystyy seuraamaan, on hakemistossa /usr/share/logwatch/scripts/services/. Kaikkiaan seurattavia palveluita on 117 kappaletta. Pidetään toistaiseksi All.

- LogFile: määritä tietty lokitiedosto, tai tarkemmin ottaen lokitiedostoryhmä, joka tutkitaan. Tällöin lokeja etsitään kaikilta palveluilta, jotka lokittavat lokitiedostoryhmään kuuluviin lokitiedostoihin. Kun asetuksen laittaa konfiguraatiotiedostoon, kyseisten lokitiedostojen lokit tulevat raporttiin muiden lokien lisäksi. Komentorivioptiona annettuna asetus taas sulkee pois muut seurattavat palvelut, eli näyttää vain kyseiset lokit. Tämä asetus ei hyväksy mitä tahansa lokitiedostoa. Kovalla etsimisellä selvisi, että lista hyväksytyistä arvoista löytyy hakemistosta /usr/share/logwatch/default.conf/logfiles. Pidetään asetus kommentoituna.
- Mailer: sähköpostinvälitysohjelma, jota Logwatch käyttää merkitäkseen viestiin otsakkeet To-, From- ja Subject-kenttiin. Oletuksena ”/usr/sbin/sendmail -t”. Tämän voi pitää oletuksena.
- HostLimit: analysoi vain tiettyjen isäntien lokit. Voidaan käyttää silloin, kun Logwatchia ajetaan järjestelmässä, johon kerätään lokeja useilta isänniltä. Oletuksena asetus on kommentissa.

Aikaisemmin oletusasetukset kopioitiin asennushakemistosta /etc-hakemiston alle. Siitä aiheutui eräs ongelma. Selvisi nimittäin, ettei Logwatch korvaakaan oletusasetuksia /etc-hakemiston alla olevilla asetuksilla, vaan käyttää muokattuja asetuksia oletusasetuksien päällä. Pääosin tällä ei ole mitään vaikutusta. Kun muokatuissa asetuksissa on eri arvo, kuin oletusasetuksissa, tämä korvaa oletusasetuksen. Ainoa poikkeus on Service-asetus, joita voi olla asetustiedostossa useita. Kun oletusasetuksissa on määritelty, että Logwatch skannaa kaikki palvelut, ja muokatusta asetustiedostosta poistaa rivin Service = All, ja lisää erikseen palvelut joita halutaan seurata, tuloksena on virheilmoitus: ”Wrong configuration entry for ”Service”, if ”All” selected, only ”-” items are allowed”. Eli kun oletusasetuksissa on Service = All, niin Logwatch odottaa /etc-hakemistonkin alla olevissa asetuksissa olevan vain miinusmerkillä alkavia Service-arvoja, ja/tai sana ”All”. Ongelma on helppo korjata joko muuttamalla oletusasetuksien tiedostonimeä, tai muuttamalla siitä kommentiksi rivin Service = All. Kuitenkin olisin toivonut, että Logwatchin konfiguraatiotiedostossa tai manuaalisivulla olisi kerrottu Logwatchin käyttäytyvän tällä tavalla, niin ei olisi tarvinnut ihmetellä miksi se ei toimikaan niin kuin odottaisi.

4.3 Käyttö

Kun asetukset ovat kunnossa, Logwatch voidaan ajaa komentoriviltä yksinkertaisesti komennolla *logwatch*, ilman sudoa. Kuten todettua, konfiguraatiotiedostossa olevat asetukset voidaan yli ajaa komentoriville annettavilla optioilla. Poikkeuksia ovat TmpDir, Output, Format, Encode, Mailto_host1, MailFrom ja Mailer, joita ei voi antaa optioina. Valinnan Output sijasta optiolla voidaan pakottaa ulostulo tiedostona, mutta ei tulosteena tai sähköpostina. Muita asioita, joita optioilla voidaan saada aikaan, ovat virheenjäljitys (englanniksi debug), koneen isäntänimen muuttaminen raporttiin, HTML-raportin leveys merkkeinä, IP-osoitteiden näyttäminen sellaisenaan verkkotunnuksien sijasta, sekä komennon käyttöohjeen tulostaminen. (Bauer 2012.)

Huomaa, ettei asetusta TmpDir voi antaa optiona. Tapa, jolla väliaikaishakemisto aiemmin saatiin toimimaan estää tehokkaasti muita kuin ylläpitäjäkäyttäjiä ajamasta Logwatchia. Peruskäyttäjillä ei ole oikeuksia väliaikaishakemiston luomiseen, eivätkä he pysty käyttämään sudoa. Näin ollen Logwatchin ajaminen ei onnistu. Tämä voi olla ihan hyvä asia, sillä peruskäyttäjillä ei pitäisi olla mitään tarvetta seurata palvelimen lokeja ja lokit saattavat sisältää tietoja, joiden on hyvä olla vain ylläpitäjien tiedossa. Manuaalista lokien tutkimista peruskäyttäjätkin pystyvät tosin tekemään.

Koska Logwatchin ulostuloksi on tällä hetkellä määritetty stdout, komennon *logwatch* ajaminen tulostaa lokiraportin komentoriville. Ja kuten todettua, komentorivillä ei pysty rullaamaan ylöspäin, joten lokiraporttia voi tutkia paremmin komennolla *logwatch | less* (putkita tuloste sivutusohjelmaan). Raportti ei tällöin tietenkään tallennu mihinkään, vaan katoaa kun sivutusohjelman sulkee. Jos Logwatchin ulostulona haluaa pitää tulostuksen, mutta joskus satunnaisesti tallentaa raportin, se onnistuu optiolla --*filename.* Saman asian tekee tulostuksen ohjaaminen tiedostoon komennolla *logwatch > /home/admini/logwatch*. Kahdella ohjausmerkillä (>>) tulostus voidaan lisätä olemassa olevan tiedoston loppuun. Jos ensin selaa raporttia sivutusohjelmassa ja sitten päättää tallentaa sen, niin sen ei pitäisi olla ongelma, että Logwatchin joutuu ajamaan uudestaan. Testikoneessa Logwatchin skannaukseen ja lokiraportin luomiseen meni aikaa noin yksi sekunti, kun skannattiin yhden päivän lokit. Kaikkien, mukaan lukien pakatut lokit, skannaukseen meni aikaa noin puolitoista sekuntia. Tuotantokäytössä olevalla palvelimella tietenkin lokien määrä on paljon suurempi, mutta todennäköisesti on suoritintehon mää-

räkin, joten Logwatchin ajon ei pitäisi viedä kauaa, joskin luultavasti kauemmin kuin testikoneessa.

Yleensä toki halutaan säilyttää lokiraportit siltä varalta, että niitä tarvitsisi myöhemmin. Tähän asti käsitellyillä menetelmillä ainoat järkevät tavat lokiraporttien säilyttämiseen ovat se, että Logwatchia ajaa manuaalisesti päivittäin joko optiolla *--filename* antaen aina eri tiedostonimen, tai ohjaten tulostuksen aina saman tiedoston loppuun komennolla l*ogwatch >> /home/admini/logwatch*. Opinnäytetyön automatisointia käsittelevässä luvussa 5 käsitellään ylläpitäjän rutiinien kannalta vaivattomampia tapoja lokien seuraamiseen, samalla säilyttäen lokiraportit.

Lokiraportin muotona voi tekstin lisäksi olla HTML. HTML-muotoisesta raportista on kuvakaappaus kuvassa 7. HTML-raporttia ei tietenkään pysty järkevästi tutkimaan ilman graafista käyttöliittymää, joten se soveltuu vain raportteihin, jotka lähetetään palvelimelta muualle tutkittavaksi, esimerkiksi sähköpostilla. Tässä tapauksessa tiedosto siirrettiin virtuaalikoneesta isäntäkoneelle sftp-protokollalla. HTML-muotoinen lokiraportti sisältää samat tiedot kuin tekstimuotoinenkin, ainoastaan järjestettynä taulukoihin sekä linkit kunkin osa-alueen alkuun. Mitään erikoista lisää se ei siis tarjoa.

KUVA 7. Ote Logwatchin HTML-raportista

Se mitä palveluita ja millä yksityiskohtaisuustasolla kannatta seurata, riippuu täysin kunkin organisaation käyttöympäristöstä ja tarpeista. Logwatchin konfiguraatiotiedostossa suositellaan, että kaikkien palveluiden seuraaminen on sopiva asetus useimmille käyttäjille. Näin on ainakin siinä tapauksessa, että yksityiskohtaisuus- eli Detail-asetuksena on low eli matala, koska raportin pituus ei tällöin ole kovin suuri. Suuremmalla yksityiskohtaisuusasetuksella raportin pituus kuitenkin kasvaa huomattavasti, jolloin seu-

rattavien palveluiden määrää voi olla syytä rajata. Testikoneessa Logwatch tuotti matalalla yksityiskohtaisuusasetuksella raportin, jonka pituus oli 121 riviä, kun seurattavina palveluina oli kaikki ja aikavälinä yksi päivä. Keskitason yksityiskohtaisuudella raportin pituus oli 824 riviä ja korkealla yksityiskohtaisuudella yllättäen vain 831 riviä, eli vain hivenen alempaa tasoa enemmän. Kun aikavälinä oli kaikki, niin vastaavat luvut olivat 963, 2974 ja 3046 riviä. Tuotantokäytössä lokiraportit ovat luonnollisesti pidempiä kuin opinnäytetyön ympäristössä. Erityisen paljon lokia tuottaa dpkg-loki, joka kertoo mitä pakettienhallinta on tehnyt, sekä Linux-ytimen tapahtumia lokittava kerneliloki. Kun nämä kaksi lokia jätti pois seurattavista palveluista, niin raportin pituus putosi 3046 rivistä vain 222 riviin. Joten itse suosittelisin, että mikäli yksityiskohtaisuusasetuksena haluaa käyttää matalinta asetusta suurempaa arvoa, niin seurattavien palveluiden määrää kannattaa vähentää. Ellei sitten tarkoituksena ole, että kaikkien palveluiden lokit kerätään talteen vain säilyttämistä varten varmuuden vuoksi, ja ylläpitäjä aktiivisesti lukee lokiraporteista vain tietyt osat.

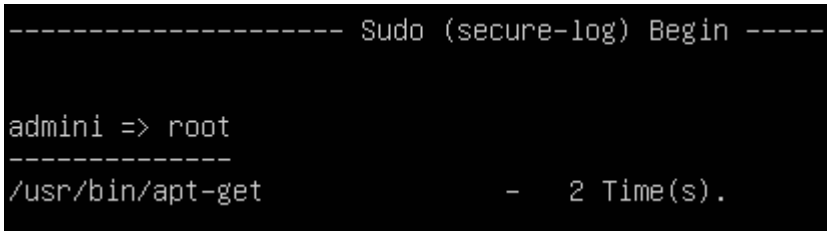
Lokiraportissa on alussa Logwatchin versio ja Logwatchin ajamiseen liittyviä tietoja. Sen jälkeen on lokitiedot jaettuna osa-alueisiin. Liitteessä 1 on täysi esimerkki Logwatchin lokiraportista.

4.4 Tietoturvan seuraaminen

Kuten aikaisemmin jo todettiin, tietoturva on merkittävä syy lokien seuraamiselle. Siksi tässä alaluvussa käsitellään lokienseurantaa Logwatchin avulla erityisesti tietoturvan varmistamisen näkökulmasta. Kannattaa huomata, että lokeja seuraamalla pystyy vain harvoissa tilanteissa estämään tietomurron tapahtumisen ennalta. Jos tietomurto kuitenkin havaitaan nopeasti sen tapahduttua, on se huomattavasti parempi tilanne kuin se, että sitä ei huomattaisi pitkään aikaan. Juuri tässä Logwatch voi olla suureksi avuksi.

Ensimmäinen kysymys, joka tietoturvan seuraamiseen liittyy, on että mitä asioita lokeista pitäisi seurata. Tärkein seurattava asia lienee se, mitä komentoja järjestelmässä ajetaan rootin oikeuksilla. Tätä tosin hankaloittaa se, että Logwatch näyttää ajetun ohjelman, mutta ei tarkkaa komentoa, kuten ilmenee kuvasta 8. Vasta kun raportin yksityiskohtaisuuden tasoksi laittaa korkean, näkyy myös mikä komento on ajettu. Tämä on ongelma varsinkin konfiguraatiotiedostoja muokatessa, koska tällöin lokiraportissa näkyy

vain, että tekstieditoria Nano, tai muu vastaava on käytetty, mikä ei kerro ylläpitäjälle vielä juuri mitään. Raportin yksityiskohtaisuuden tasona joutuukin tämän vuoksi pitämään korkean, tai sitten joutuu turvautumaan käyttäjien Bash-historiaan. Kaikki käyttäjän komentoriville syöttämät komennot tallentuvat käyttäjän kotihakemiston alle piilotiedostoon nimeltä .bash_history. Sieltä ylläpitäjä pystyisi tarkistamaan, mikä komento tarkalleen on annettu, mutta siihenkin liittyy muutama ongelma. Ensinnäkin tiedostossa ei ole aikaleimoja, eli siitä ei näy minä päivänä mikäkin komento on annettu. Annetut komennot myös tallentuvat tiedostoon vasta käyttäjän kirjautuessa ulos, mikä voi olla ongelma jos käyttäjä jättää istunnon auki pitkiksi ajoiksi. Lisäksi jos joku pahantahtoinen henkilö syöttelee järjestelmään ikäviä komentoja, niin hän voi saman käyttäjän oikeuksilla käydä muuttamassa myös Bash-historiaansa, mistä tosin itsessään jää merkintä Bash-historiaan. Ylläpitäjä ei tosin sitä välttämättä huomaisi. Luotettavampaa kuitenkin lienee, että mikäli sudolokissa näkyy epäilyttäviä tietoja, niin ylläpitäjä ajaa Logwatchin uudestaan skannaten vain sudolokin korkealla yksityiskohtaisuustasolla käyttäen komentoa *logwatch --service sudo --detail high*.



KUVA 8. Sudoloki

Toinen tärkeä seurattava asia tietoturvan kannalta on etäkirjautuminen SSH:lla. Jos palvelimella on paljon perustason käyttäjiä, heidän kirjautumisistaan voi olla vaikea pysyä selvillä. Ylläpitäjätason käyttäjiä sen sijaan ei yleensä ole kovin paljon, joten heidän kirjautumisiaan on syytä seurata. SSH:lla kirjautumista koskien Logwatchin raportista näkee kirjautuneen käyttäjän, IP-osoitteen, asiakkaan laitteen isäntänimen, sekä kirjautumisten määrän. Ylläpitäjällä olisi hyvä olla jokin tapa seurata sitä, että kirjautumiset tulevat sieltä, mistä pitäisikin. Epäonnistuneista kirjautumisyrityksistä näkee, että niitä on tapahtunut, mistä IP-osoitteesta, ja kuinka monta kertaa. Matalaa korkeammilla yksityiskohtaisuustasoilla näkee myös, millä käyttäjällä on yritetty kirjautua. Erityisesti jos lokeissa näkyy, että samasta IP-osoitteesta on yritetty kirjautua kymmenentuhatta kertaa, niin ylläpitäjän kannattaa varmistua siitä, että SSH-kirjautuminen on tehty turvalliseksi.

Tämä on niitä tapauksia, joissa lokien seuraaminen voi kertoa tietomurtoyrityksistä ennen niiden onnistumista.

Myös su-komennon käyttöä on syytä seurata. Hyökkääjähän saattaisi kirjautua SSH:lla peruskäyttäjänä, mutta sitten su-komennolla vaihtaa ylläpitäjäkäyttäjäksi olettaen, että hyökkääjä tietää ylläpitäjän salasanan. Tällöin SSH-lokissa ei näkyisi ylläpitäjän vaan peruskäyttäjän kirjautuneen. Ubuntussa oletuksena su-komennon käyttö on sallittua kaikille, mutta se voidaan estää peruskäyttäjiltä, mikä saattaa olla palvelinkäytössä ihan järkevää. Sitä ei kuitenkaan tässä esitellä.

On tietenkin paljon muitakin lokeja, jotka voivat olla tietoturvan kannalta merkityksellisiä, eikä tässä voida kaikkia niitä esitellä. Lokienseurannan tarpeet luonnollisesti vaihtelevat ympäristöstä toiseen ja ylläpitäjän pitää harkita mikä on tarpeellista omassa ympäristössä.

Toinen mielenkiintoinen kysymys on, että jos poistaa lokit, jääkö siitä merkintä. Hakemistossa /var/log/ olevia lokeja ei pysty muuttamaan tai poistamaan ylläpitäjäkäyttäjänkään oikeuksilla, mutta rootin oikeuksilla se toki onnistuu. Sudon avulla voi tekstieditorilla muuttaa tietyn rivin mieleisekseen, tai poistaa koko rivin. Lokeihin kuitenkin jää merkintä, että lokitiedostoa on käsitelty tekstieditorilla rootin oikeuksilla, mikä on aina epäilyttävää. Toinen vaihtoehto hyökkääjälle olisi tehdä Logrotatella pakotettu lokien kierrätys tarpeeksi monta kertaa, jotta vanhat lokit poistuisivat. Tällä toki saa lokit poistettua, mutta siitäkin jää merkintä uusiin lokeihin. Pakotetun lokien kierrätyksen tekeminen monta kertaa on sekin aika epäilyttävää.

Yleissääntönä voidaan kuitenkin pitää, että jos hyökkääjä saa palvelimelle rootin oikeudet, niin palvelimen koko tietoturva on menetetty. Rootin oikeuksilla hyökkääjä pystyisi tekemään järjestelmälle mitä tahansa, kuten muokkaaman järjestelmän toimintaa lokitukseen liittyen. Siten myöskään Logwatch ei pysty täysin aukottomasti havaitsemaan rootin oikeudet saanutta hyökkääjää, joka on kyllin taitava.

5 AUTOMATISOINTI

5.1 Lokiraporttien generoiminen - cron

Mikäli lokeja aikoo seurata säännöllisesti, niin ylläpitäjä halunnee tehdä siitä mahdollisimman vaivatonta. Muuten homma jää helposti tekemättä. Lokienseurannan rutinoitumista auttaa se, että lokiraportti on valmiina luettavaksi ylläpitäjän tullessa aamulla töihin. Tähän käytetään työkalua nimeltä cron, jolla voidaan ajastaa tehtävä tehtäväksi aina tiettyyn aikaan. Käyttäjä voi käyttää komentoa *crontab -e* lisätäkseen croniin ajastetun tehtävän. Tällöin tehtävä myös suoritetaan kyseisen käyttäjän oikeuksilla. Huomaa, että cron ei suorita tehtävää mikäli järjestelmä on sammutettuna ajastetun ajan hetkellä. Tässä oletetaankin, että lokienseurantaa tehdään palvelimella, joka on jatkuvasti päällä.

Logwatchia asennettaessa on automaattisesti lisätty croniin tiedosto /etc/cron.daily/00-logwatch, joka ajaa Logwatchin kerran päivässä optiolla *--output mail*. Ylläpitäjä saattaa haluta poistaa kyseisen tiedoston (komennolla *sudo rm /etc/cron.daily/00logwatch*), koska ylläpitäjä halunnee itse määrittää, mihin kellonaikaan, millä optioilla ja kenen käyttäjän oikeuksilla Logwatch ajetaan.

Kun käyttäjä ajaa komennon *crontab -e,* avautuu tiedosto, johon merkitään yhdelle rivillle ajastettu aika, sekä komento joka ajetaan. Rivin muoto on minuutti, tunti, kuukaudenpäivä, kuukausi ja viikonpäivä, sekä ajettava komento. Minkä tahansa ajan yksikön voi myös korvata asteriskilla (*), joka tarkoittaa mitä aikaa vain. Tässä tapauksessa Logwatch kannattaa ajaa johonkin aikaan aamuyöstä, kun järjestelmä on vähällä käytöllä. Tällöin Logwatch skannaa edellisen päivän lokit ja tekee niistä raportin tiedostoon. Logwatchin ajaminen cronin kautta ei onnistunut ilman, että cronin tiedostoon määritteli Logwatchin täydellisen polun /usr/sbin/logwath. Komennon polun saa selville komennolla *which komento*. Jotta Logwatch ajettaisiin joka aamuyö kello 3, merkitään cronin tiedostoon rivi *0 3 * * * /usr/sbin/logwath*. Logwatchin asetustiedostossa täytyy tietenkin olla sopivat asetukset, kuten että ulostulona on tiedosto.

Kuten Logwatchin asetuksia tehdessä todettiin, Logwatch ylikirjoittaa vanhan lokiraportin, kun ulostulona on tiedosto. Siksi edellä oleva yksinkertainen ratkaisu ei ole kovin hyvä, koska vanhat lokiraportit eivät säästy. Parempi ratkaisu on tehdä skripti, jolla kun-

kin päivän raportti tallennetaan omaan tiedostoonsa. Croniin pystyy komennolla *crontab -e* määrittelemään vain yksittäisiä rivejä, jotka kukin suoritetaan erikseen. Kun yhteen asiaan vaaditaan useampia komentoja, tarvitaan skriptiä. Ensin tehdään kotihakemiston alle hakemisto lokiraporteille komennolla *mkdir /home/admini/lokiraportit* (luo hakemisto). Sitten muutetaan Logwatchin asetuksista ulostuloksi stdout. Sitten luodaan kotihakemiston alle skripti komennolla *nano /home/admini/logwatch-skripti*. Skriptin sisältö on seuraava:

```
#!/bin/bash
# tallennetaan päiväys muuttujaan pvm muodossa vuosi-kuukausi-päivä
pvm=$(date +%Y-%m-%d)
# ajetaan Logwatch, ohjataan tuloste tiedostoon logwatch-vuosi-kuukausi-päivä
/usr/sbin/logwatch > /home/admini/lokiraportit/logwatch-"$pvm"
```

Sitten korvataan aiempi rivi cronin tiedostossa rivillä *0 3 * * * sh /home/admini/logwatch-skripti*. Ja näin Logwatch generoi joka aamuyö uuden lokiraportin, joka tallennetaan päiväyksellä nimettyyn tiedostoon ja tiedostot järjestyvät hakemistossa päiväyksen mukaiseen järjestykseen.

5.2 Lokiraporttien lähettäminen sähköpostilla

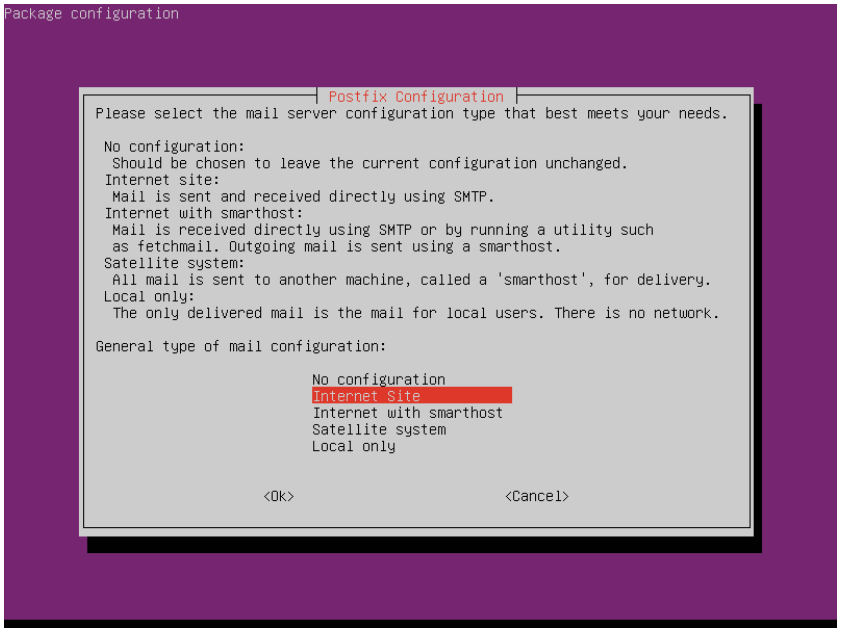
Viimeinen askel lokienseurannan käyttöönotossa on raporttien lähettäminen seurattavalta laitteelta muualle. On kaksi syytä, miksi näin haluttaisiin tehdä. Ensimmäinen on lokienseurannan vaivan vähentäminen. Erityisesti jos ylläpitäjällä on useita palvelimia seurattavanaan, niin hän ei välttämättä haluaisi kirjautua jokaiselle palvelimelle erikseen lukeakseen niiden lokiraportit. On paljon helpompaa, kun kaikkien palvelimien lokit voi lukea yhdestä paikasta. Toiseksi tietoturvan kannalta on parempi, että lokit lähetetään laitteelta pois. Näin lokiraporttia ei voi seurattavalta laitteelta käsin enää muokata sen jälkeen, kun se generoidaan ja lähetetään muualle.

Kannattaa kuitenkin pitää mielessä, että lokien lähettäminen sähköpostilla saattaa mahdollistaa niiden vakoilemisen. Riski on olemassa ainakin mikäli niitä lähetetään julkisen internetin yli, sen sijaan että niitä siirrettäisiin vain sisäverkossa laitteelta toiselle. Sähköpostijärjestelmää ei varsinaisesti tunneta hyvästä tietoturvastaan. Lokien ei pitäisi koskaan sisältää esimerkiksi salasanoja, mutta niistä pystyy näkemään esimerkiksi käyt-

täjätunnuksia sekä sen, mitkä niistä ovat ylläpitäjäkäyttäjiä. Lisäksi niissä voi olla monia muitakin hyökkääjiä kiinnostavia asioita. Tämä riski kannattaa vähintäänkin tiedostaa.

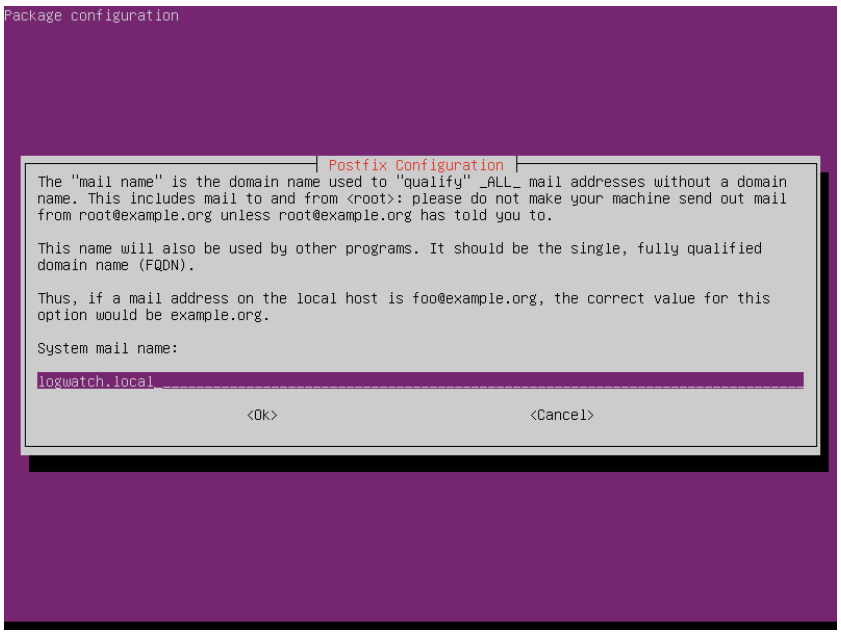
Opinnäytetyössä ei ole käytössä julkista domainia, joka vaadittaisiin sähköpostin lähettämiseksi kolmannen osapuolen sähköpostipalveluun. Jos taas tarkoituksena olisi vain lähettää lokiraportit sisäverkossa laitteelta toiselle, niin siihen esitetään parempi tapa seuraavassa alaluvussa. Sähköpostijärjestelmän toiminta ei myöskään ole opinnäytetyön keskeisintä sisältöä, joten tässä esitetään vain yleiskuva siitä, miten Postfixin asetukset tehtäisiin.

Logwatchin mukana asennettiin sähköpostinvälitysohjelma Postfix. Postfixia asennettaessa valittiin, ettei tehdä asetuksia ja siinä tilassa Postfix on edelleen. Saadakseen jälkikäteen Postfixin asetusruudun esille ylläpitäjä voi käyttää komentoa *sudo dpkg-reconfigure postfix* (uudelleen konfiguroi paketti). Siitä aukeaa ruutu, joka näkyy kuvassa 9. Kyse ei ole graafisesta käyttöliittymästä, vaikka siltä saattaa näyttää. Näppäimistöä käyttäen valitaan asetustyyppi, yleensä ”Internet Site”.



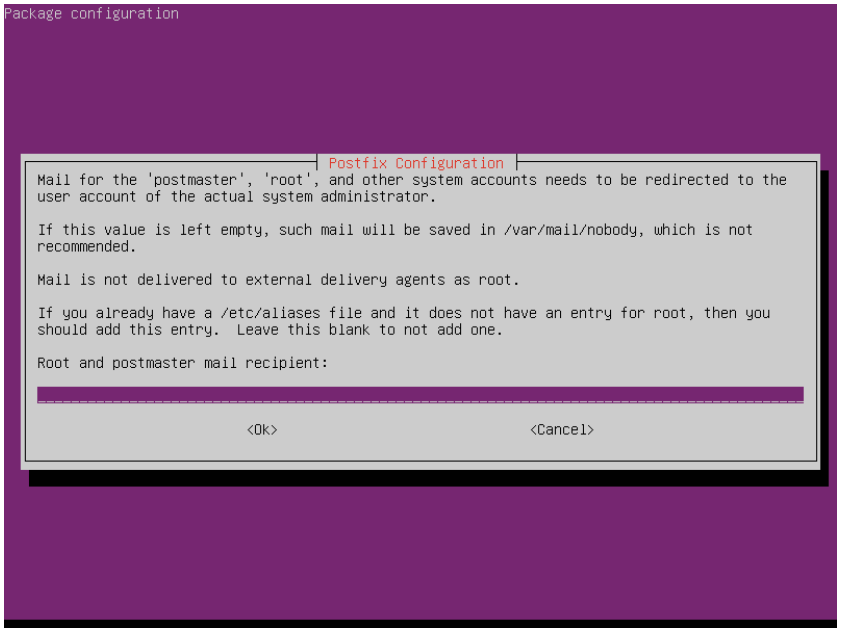
KUVA 9. Postfixin asetukset 1, konfiguraatiotyyppi

Seuraavaksi kuvassa 10 annetaan domaini, jonka tuotantokäytössä pitäisi olla julkinen domaini, joka on saavutettavissa vastaanottavalta sähköpostipalvelimelta käsin.



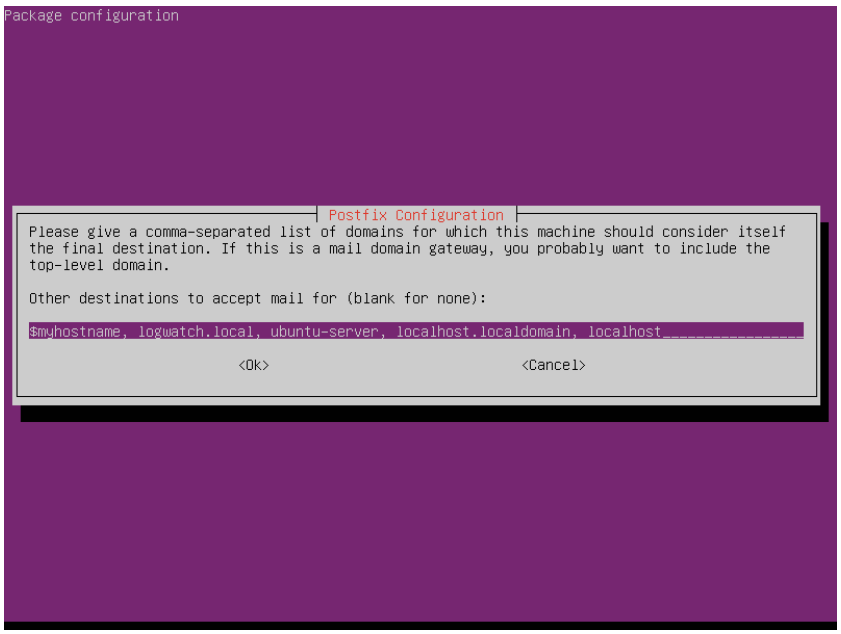
Kuva 10. Postfixin asetukset 2, domainnimi

Lokiraporttien lähetystä varten ei ole tarvetta pystyä vastaanottamaan sähköpostia, jolloin rootin vastaanottajakenttä voidaan jättää tyhjäksi kuvassa 11.



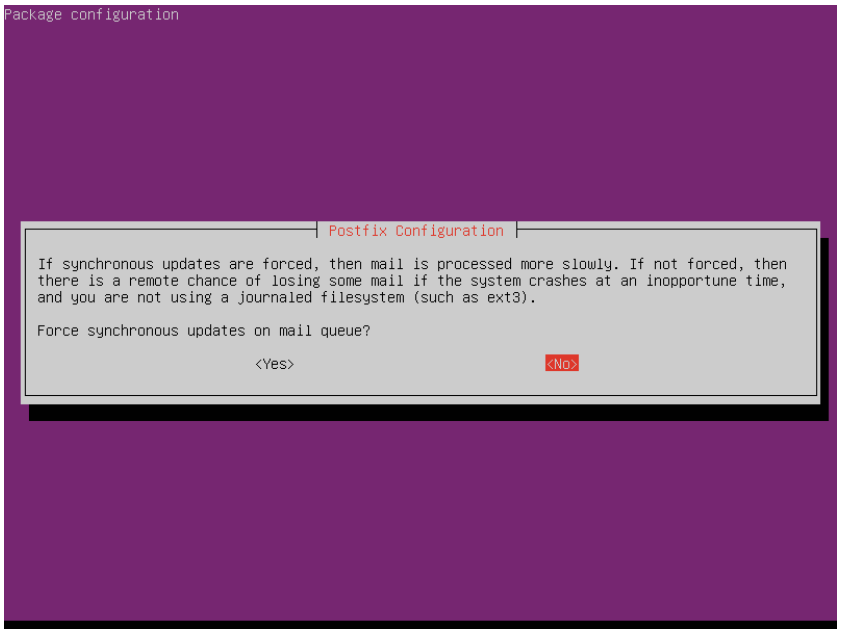
KUVA 11. Postfixin asetukset 3, root-vastaanottaja

Annetaan lista domaineista, joihin tuleva posti on tarkoitettu tälle laitteelle. Oletuksena rivillä on kaikki kuvassa 12 näkyvät arvot.



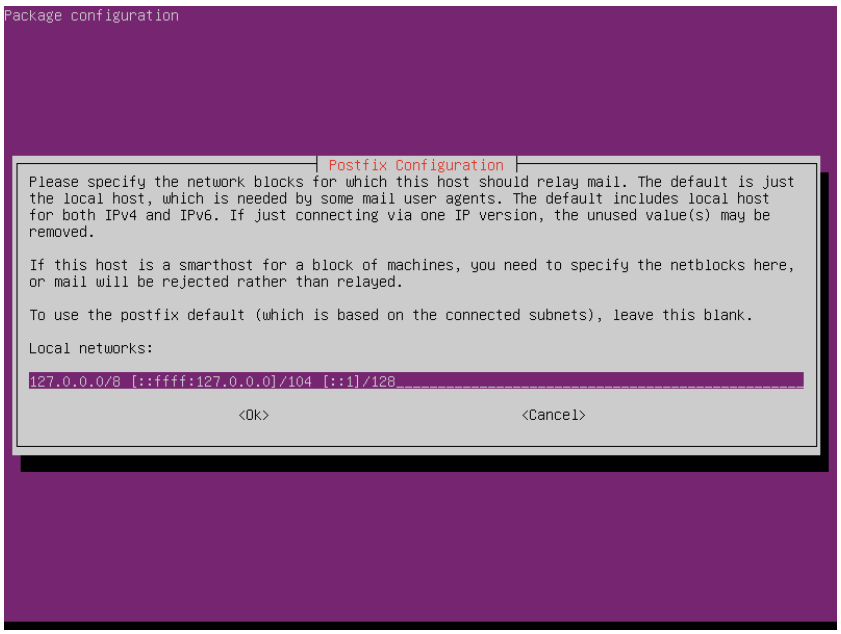
KUVA 12. Postfixin asetukset 4, muut domainit

Käytössä on journaloiva tiedostojärjestelmä ext4, joten kuvassa 13 voidaan valita ”No”. Journaloiva tiedostojärjestelmä parantaa tiedostojen säilyvyyttä jos järjestelmä kaatuu huonoon aikaan.



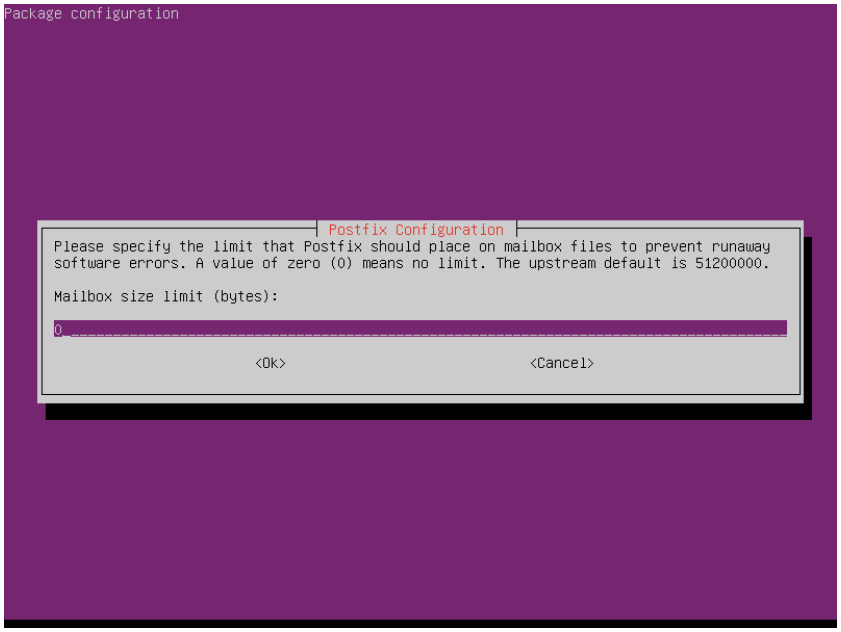
KUVA 13. Postfixin asetukset 5, synkroniset päivitykset

Jos Postfixin pitää voida edelleen lähettää postia muihin verkkoihin, se voidaan määritellä kuvassa 14.



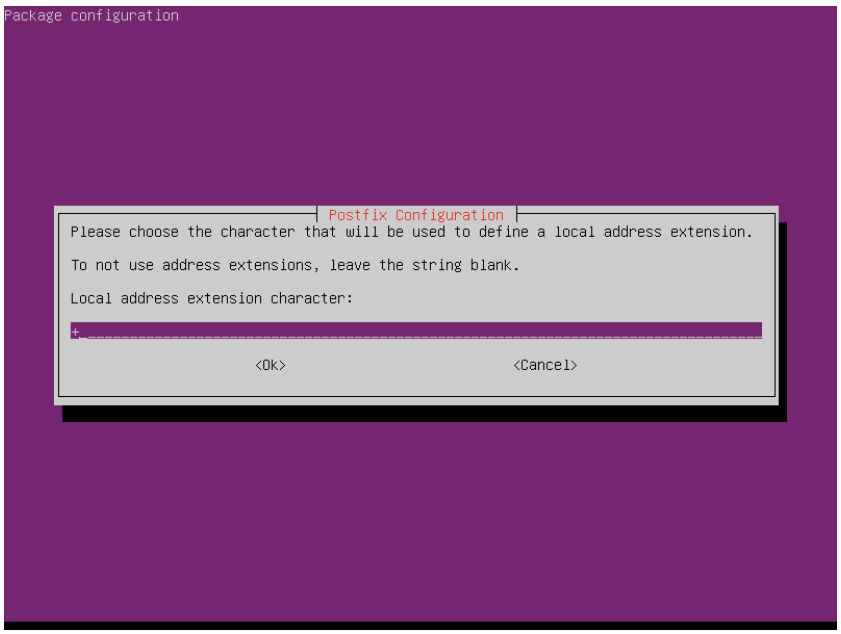
KUVA 14. Postfixin asetukset 6, muut verkkolohkot

Kuvassa 15 voidaan määrittää postilaatikon tiedostoille kokoraja tavuissa. 0 tarkoittaa ei rajoitusta.



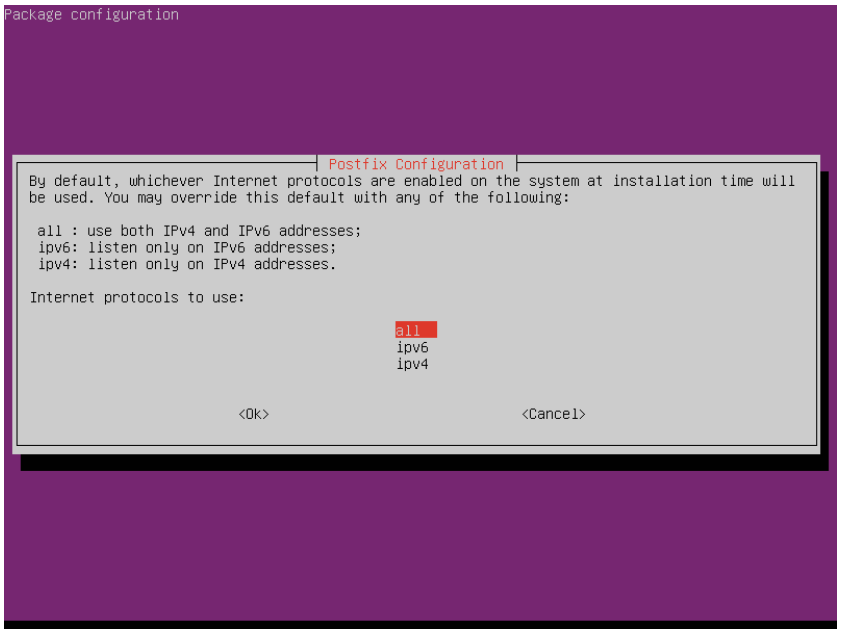
KUVA 15. Postfixin asetukset 7, postilaatikon kokoraja

Kuvassa 16 määritettävä erikoismerkki voidaan pitää oletuksena.



KUVA 16. Postfixin asetukset 8, käytettävä erikoismerkki

Kuvassa 17 valitaan, käytetäänkö IP-protokollasta versiota 4, 6 vai molempia.



KUVA 17. Postfixin asetukset 9, käytettävä IP-protokolla

Postfixin asetuksien teon jälkeen Logwatchin asetuksista ulostuloksi muutetaan mail. Asetukseen MailTo annetaan arvoksi sähköpostiosoite, johon lokiraportti lähetetään. Asetuksen MailFrom arvona voidaan pitää Logwatch tai antaa jokin sähköpostiosoite, josta viesti lähetetään. Muotona voitaisiin tässä tilanteessa tietenkin käyttää HTML-muotoa, jos ylläpitäjä haluaa tarkastella lokiraportteja selaimella.

Jos sähköpostin välitys on toimivassa tilassa, komentoriviltä voidaan tämän jälkeen ajaa komento *logwatch*, jolloin lokiraportti generoidaan ja lähetetään. Tai mieluummin tietenkin cronin avulla Logwatch ajastetaan ajettavaksi automaattisesti.

5.3 Lokiraporttien lähettäminen SSH:lla

Sähköpostilla lähettämistä yksinkertaisempi tapa on lähettää raportit SSH:lla, tarkemmin sanottuna protokollalla ja ohjelmalla scp, joka hyödyntää SSH:ta. Tämä toimii ongelmitta, vaikka raportin lähettävä palvelin olisi NATin (Network Address Translation) takana, ja vastaanottava palvelin internetissä, eikä käytössä olisi julkista domainia. Tietenkin vastaanottavalla palvelimella pitää olla joko domaini tai staattinen ip-osoite, jolla se voidaan tavoittaa. Koska scp käyttää SSH:ta, tiedostojen lähettäminen on sillä turvallisempaa salakuuntelua vastaan kuin sähköpostilla lähettäminen.

Jotta tiedostojen lähettäminen SSH:lla voidaan automatisoida, kannattaa etäpalvelimelle kirjautuminen mahdollistaa ilman salasanan kirjoittamista. Tähän voidaan käyttää SSH-avaimia. Lähettävällä palvelimella luodaan käyttäjälle uusi avainpari komennolla *ssh-keygen -t rsa* (luo uusi avainpari tyyppiä rsa). Tässä tapauksessa luotavaa avainta ei suojata salasanalla, jottei salasanaa tarvitsisi kirjoittaa avainta käytettäessä. Avain luodaan kuvassa 18.

KUVA 18. SSH-avaimen luominen

Opinnäytetyössä käytetään vastaanottavana palvelimena toista virtuaalikonetta, joka on kloonattu alkuperäisestä virtuaalikoneesta ubuntu-server. Siihen viitataan tässä nimellä vastaanotin. Sen IP-osoite on 10.0.0.20. Vastaanottimella on tietenkin samannimiset käyttäjät kuin alkuperäisellä koneellakin. Koneella ubuntu-server luodun avainparin julkinen avain kopioidaan sitten vastaanottimen Perustyyppikäyttäjälle komennolla *ssh-copy-id perustyyppi@10.0.0.20* (kopioi avain käyttäjälle tiettyyn IP-osoitteeseen), kuten näkyy kuvassa 19.

KUVA 19. SSH-avaimen kopiointi etäpalvelimelle

Tämän jälkeen vastaanottimelle pystyy kirjautumaan käyttäjänä Perustyyppi SSH-avaimella ilman salasanaa. Tämä helpottaa sitä, että skripti voi automaattisesti käydä lähettämässä tiedoston etäpalvelimelle.

Koneella ubuntu-server pidetään käyttäjän Admini cron-tiedostossa sama rivi kuin aiemmin, eli *0 3 * * * sh /home/admini/logwatch-skripti*. Logwatchin ulostulona on taas stdout. Logwatch-skriptin sisältö on nyt seuraava:

```
#!/bin/bash
# tallennetaan päiväys muuttujaan pvm muodossa vuosi-kuukausi-päivä
pvm=$(date +%Y-%m-%d)
# ajetaan Logwatch, ohjataan tuloste tiedostoon logwatch-vuosi-kuukausi-päivä
/usr/sbin/logwatch > /home/admini/lokiraportit/logwatch-"$pvm"
# lähetetään tiedosto etäpalvelimelle
/usr/bin/scp /home/admini/lokiraportit/logwatch-"$pvm" perustyyppi@10.0.0.20:/
home/perustyyppi/lokiraportit/logwatch-"$pvm"
```

Tämän jälkeen vastaanottimella käytetään komentoa *sudo crontab -e* (lisää cron työ käyttäjälle root). Työ ajoitetaan heti lokiraportin vastaanottamisen jälkeen, eli minuutin yli 3. Croniin lisätään rivi *1 3 * * * mv /home/perustyyppi/lokiraportit/* /home/admini/lokiraportit/*

Eli käyttäjällä root siirretään kaikki Perustyypin lokiraporttihakemistossa olevat tiedostot käyttäjän Admini vastaavaan hakemistoon. Kaikki nämä hakemistot pitää tietenkin olla valmiiksi luotuna. Miksi näin tehdään? Kuten aiemmin todettiin, lokeja lähetetään palvelimelta muualle paitsi ylläpitäjän työn helpottamiseksi, myös tietoturvan takia. Nyt jos hyökkääjä saisi pääsyn koneelle ubuntu-server Adminikäyttäjällä, hän saisi pääsyn myös vastaanottimelle, koska sinne pääsee ubuntu-serveriltä kirjautumaan ilman salasanaa. Siten hän voisi myös muuttaa jo lähetettyjä lokiraportteja ja näin lokien lähettämisen tietoturvahyöty menetettäisiin. Tulee kuitenkin huomata, että ubuntu-serveriltä pääsee kirjautumaan vastaanottimelle ilman salasanaa vain vastaanottimen Perustyyppikäyttäjänä, jolle SSH-avain kopioitiin. Kun lokiraportit siirretään vastaanottimella käyttäjän Perustyyppi ulottumattomiin, ubuntu-serveriltä käsin ei enää pysty muokkaamaan jo lähetettyjä raportteja ilman, että saa selville vastaanottimen Adminikäyttäjän salasanan. Vastaanottimen Perustyyppikäyttäjän paikalla kannattaa tietenkin tuotantokäytössä käyttää käyttäjätiliä, jota ei käytetä mihinkään muuhun kuin Logwatchin lokiraporttien vastaanottamiseen.

Jos seurattavana on useita palvelimia, niin ylläpitäjä voi tehdä samanlaisen järjestelyn kaikille niille, ja näin hän pystyy seuraamaan kaikkien palvelimien lokiraportteja yhdestä paikasta.

6 POHDINTA

Jos Logwatchin käyttöä vertaa manuaaliseen lokien tutkimiseen, niin pystyy huomaamaan, että Logwatch helpottaa lokienseurantaa huomattavasti. Manuaalisella tavalla ylläpitäjä joutuu selaamaan pitkiä lokitiedostoja tai etsimään niistä haulla, olettaen että tietää mitä etsii. Logwatchilla taas ylläpitäjä saa päivittäin raportin niistä osa-alueista joista haluaa, sillä yksityiskohtaisuustasolla kuin haluaa. Joten sanoisin, että jos lokienseurannasta haluaa tehdä rutiinia, niin siinä kannattaa ehdottomasti käyttää apuna jotakin työkalua ja Logwatch on siihen kelvollinen väline.

Logwatchillakin on kuitenkin rajoituksensa. Näkyvin rajoitus lienee se, ettei lokiraporteissa ole aikaleimoja. Logwatchilla voi toki jaotella eri päivien lokit eri tiedostoihin ja yleensä se riittää. Joskus kuitenkin on tarvetta tietää myös, mihin kellonaikaan jokin asia on tapahtunut ja silloin joutuu katsomaan alkuperäistä lokitiedostoa. Toisaalta Logwatchin toimintatavalla on myös hyvä puolensa. Lokiraportista pystyy helposti näkemään, kuinka monta kertaa sama tapahtuma on tapahtunut, siinä missä alkuperäisessä lokitiedostossa saman tapahtuman eri esiintymät ovat hajallaan. Tällöin manuaalisesti lokitiedostoja tutkimalla ylläpitäjä joutuisi manuaalisesti laskemaan, kuinka monta kertaa jokin tapahtuma on tapahtunut, tai käyttämään skriptiä, joka laskisi kuinka monta kertaa tietty merkkijono esiintyy tiedostossa. Tällä on merkitystä esimerkiksi silloin, kun halutaan tietää, paljonko epäonnistuneita kirjautumisyrityksiä on tapahtunut kullekin käyttäjälle tai kuinka monta kertaa samasta IP-osoitteesta on yritetty SSH-yhteyttä.

Toinen Logwatchin rajoitus on se, ettei eri palveluille voi laittaa eri yksityiskohtaisuustasoja. Ylläpitäjä saattaisi esimerkiksi haluta käyttää yksityiskohtaisuusasetuksena arvoa korkea, mutta hyvin paljon lokia tuottaville palveluille arvoa matala. Se ei kuitenkaan onnistu ilman, että Logwatchilla tekee eri ajot eri palveluille ja eri yksityiskohtaisuustasoilla. Tätä rajoitusta tosin pystyisi automatisoinnilla kiertämään ajamalla Logwatchin useampaan kertaan eri asetuksilla ja sitten yhdistämällä tiedostot yhdeksi.

Tietyissä tilanteissa sekin on rajoite, ettei lokiraportista näe työhakemistoa, eli hakemistoa jonka alla komento on annettu. Jos käyttäjä ensin siirtyy tiettyyn hakemistoon ja sitten muokkaa sudon avulla tiedostoa tekstieditorilla, Logwatch kyllä näyttää käsitellyn tiedoston nimen, kun yksityiskohtaisuuden taso on korkea. Logwatch ei kuitenkaan näy-

tä hakemistoa, jossa tiedosto on, ellei se ollut osa komentoa. Jos siis ajaa vaikka komennon *sudo nano /var/log/auth.log* (muokkaa editorilla tiedostoa), lokiraportissa näkyy hakemisto ja tiedosto. Jos taas siirtyy ensin hakemistoon komennolla *cd /var/log/* ja ajaa sitten komennon *sudo nano auth.log*, lokiraportissa näkyy vain tiedoston nimi, ei hakemistoa. Tämäkin on tilanne, jossa joutuu turvautumaan alkuperäiseen lokitiedostoon, josta löytyy tämäkin tieto.

Logwatchissa on mukana valtava määrä suodattimia, joista kukin mahdollistaa tietyn palvelun lokien seuraamisen. Kaikkien mahdollisten ohjelmien lokeja varten ei tietenkään ole suodattimia, joten kaikkia mahdollisia lokeja ei Logwatchilla pysty suoraan seuraamaan. Tähän Logwatchilla on hyvin edistyneille käyttäjille tarkoitettu ratkaisu. Hakemistosta /usr/share/doc/logwatch/ löytyy tiedosto ”HOWTO-Customize-LogWatch”, joka muun muassa kertoo, kuinka käyttäjä voi tehdä omia suodattimiaan. Näin Logwatchin voi saada seuraamaan minkä tahansa ohjelman lokeja. Sama tiedosto löytyy myös Logwatch-projektin nettisivulta (https://sourceforge.net/p/logwatch/git/ci/master/tree/HOWTO-Customize-LogWatch). Tässä opinnäytetyössä ei kuitenkaan ole selvitetty omien suodattimien tekemistä, koska kyseinen asia yksinään vaatisi hyvin suurta perehtymistä. Jos opinnäytetyön tuloksia haluaisi jalostaa pidemmälle, niin tämä olisi ehdottomasti yksi jatkokehitysideoista.

Yhteenvetona Logwatchin rajoituksista voidaan sanoa, että ne ovat hyväksyttävissä ja että Logwatchia voi niistä huolimatta käyttää lokienseurantaan. Manuaalisesta lokien tutkimisesta ei kuitenkaan täysin päästä eroon, vaan joissakin tilanteissa ylläpitäjä joutuu manuaalisesti etsimään tarkennusta lokiraportissa esiin tulleeseen tapahtumaan.

Mikäli organisaatiolla on joka tapauksessa omat sähköpostipalvelimet, niin niitä kannattaa tietenkin käyttää lokiraporttien lähettämiseen. Sen sijaan pelkästään Logwatchia varten sähköpostipalvelimien käyttöönotto tuntuu turhan monimutkaiselta.

Vaikka palvelimen lokienseurannan voi sanoa olevan hyvä käytäntö, ylläpitäjä joutuu kuitenkin ottamaan realiteetit huomioon. Ylläpitäjällä on todennäköisesti paljon muitakin tehtäviä, eikä lokienseurannalle välttämättä jää aikaa. Tämä korostuu jos palvelimia on suuri määrä. Kymmenenkään palvelimen lokien seuraaminen tuskin on vielä ylivoimainen tehtävä, mutta jos palvelimia on useampia kymmeniä niin lokien seuraaminen

käy liian työlääksi. Logwatch ei välttämättä ole kovin skaalautuva ratkaisu palvelinmäärän kasvaessa.

LÄHTEET

Bauer. 2012. Ohjelman Logwatch version 7.4.3 manuaalisivu käyttöjärjestelmässä Ubuntu Server 18.04.1.0

Bjorn, Bauer, Tremaine & Poplawski. 2019. Logwatch-projektin nettisivu. Luettu 21.01.2019. https://sourceforge.net/projects/logwatch/

Canonical Ltd. What is MAAS? | MAAS documentation. N.d. Luettu 10.12.2018 https://docs.maas.io/2.4/en/

Community Help Wiki. 2018. LinuxLogFiles. Luettu 15.01.2019. https://help.ubuntu.com/community/LinuxLogFiles

LSB Workgroup & The Linux Foundation. 2015. Filesystem Hierarchy Standard. Luettu 22.01.2019. http://refspecs.linuxfoundation.org/FHS_3.0/fhs/index.html

Troan, Brown & Kaluza 2002. Ohjelman Logrotate version 3.11.0 manuaalisivu käyttöjärjestelmässä Ubuntu Server 18.04.1.0

LIITTEET

Liite 1. Esimerkki Logwatchin lokiraportista

Logwatchin raportti, kun skannauksen aikavälinä on yksi päivä, seurattavina palveluina on kaikki ja yksityiskohtaisuuden taso on matala. Punaisella merkityt risuaidalla alkavat suomenkieliset rivit ovat itse lisäämiäni kommentteja. Muutoin raportti on muokkaamaton.

```
# Logwatchin versio, skannausaika, skannauksen kohteen aikaväli, yksityiskohtaisuus,
# ulostulo ja muoto sekä kohteen isäntänimi.
 ################### Logwatch 7.4.3 (12/07/16) ####################
        Processing Initiated: Wed Feb  6 14:03:25 2019
        Date Range Processed: yesterday
                              ( 2019-Feb-05 )
                              Period is day.
        Detail Level of Output: 0
        Type of Output/Format: file / text
        Logfiles for Host: ubuntu-server
 ##################################################################

# Linux-ytimen virheitä on tapahtunut.
 --------------------- Kernel Begin ------------------------

 WARNING:  Kernel Errors Present
    RAS: Correctable Errors collector initi ...:  3 Time(s)

 ---------------------- Kernel End -------------------------


# Käyttäjien kirjautumiseen ja valtuuttamiseen liittyviä lokeja.
 --------------------- pam_unix Begin ------------------------

# Adminilla on epäonnistunut kirjautumisyritys. Kun käyttäjänimen on kirjoittanut
```

```
# väärin, kirjoitettu käyttäjänimi ei tallennu lokeihin ilmeisesti siltä varalta, että käyttäjä
# olisi kirjoittanut salasanansa käyttäjänimikenttään.
 login:
    Authentication Failures:
      admini (): 1 Time(s)
      unknown (): 1 Time(s)
    Invalid Users:
      Unknown Account: 1 Time(s)

# Perustyyppikäyttäjällä on käytetty su-komentoa.
 su:
    Sessions Opened:
      perustyyppi -> admini: 1 Time(s)

# Admini on 6 kertaa ajanut komennon sudon kanssa.
 sudo:
    Sessions Opened:
      admini -> root: 6 Time(s)

# Käyttäjien sisäänkirjautumiset.
 systemd-user:
    Unknown Entries:
      session opened for user admini by (uid=0): 4 Time(s)
      session closed for user admini: 1 Time(s)
      session opened for user perustyyppi by (uid=0): 1 Time(s)


 ---------- pam_unix End ----------


# Koska Postfixin asetuksia ei oltu vielä tehty, Postfix ei pysty käynnistymään ja kirjaa
# lokiin virheilmoituksen.
 ---------- Postfix Begin ----------

      1   *Fatal:   General fatal                    1
```

---------- Postfix End ----------

---------- rsyslogd Begin ----------

**** Unmatched entries ****

rsyslogd's groupid changed to 106 : 3 Times

rsyslogd's userid changed to 102 : 3 Times

imuxsock: Acquired UNIX socket '/run/systemd/journal/syslog' (fd 3) from systemd. [v8.32.0] : 3 Times

---------- rsyslogd End ----------

---------- Connections (secure-log) Begin ----------

Unmatched Entries

Epäonnistuneet kirjautumisyritykset näkyvät täälläkin.

login: FAILED LOGIN (1) on '/dev/tty1' FOR 'UNKNOWN', Authentication failure: 1 Time(s)

login: FAILED LOGIN (2) on '/dev/tty1' FOR 'admini', Authentication failure: 1 Time(s)

Perustyyppikäyttäjällä yritettiin käyttää su-komentoa. Käyttäjänimen admini sijasta

tuli vahingossa kirjoitettua yllapitaja.

su: No passwd entry for user 'yllapitaja': 1 Time(s)

su: pam_systemd(su:session): Cannot create session: Already running in a session: 1 Time(s)

systemd-logind: New seat seat0.: 3 Time(s)

Järjestelmän sammuttaminen näkyy myös lokeissa.

systemd-logind: System is powering down.: 3 Time(s)

systemd-logind: Watching system buttons on /dev/input/event0 (Power Button): 3 Time(s)

systemd-logind: Watching system buttons on /dev/input/event1 (Sleep Button): 3 Time(s)

systemd-logind: Watching system buttons on /dev/input/event2 (AT Translated Set 2 keyboard): 3 Time(s)

---------- Connections (secure-log) End ----------

SSH-loki. Tässä näkyy etäyhteydenottojen tiedot, jos niitä on tapahtunut.

---------- SSHD Begin ----------

SSHD Started: 6 Time(s)

---------- SSHD End ----------

---------- Sudo (secure-log) Begin ----------

Mitä ohjelmia Admini on ajanut sudon kanssa.

admini => root

/bin/nano - 4 Time(s).

/usr/bin/apt-get - 2 Time(s).

Perustyyppikin on yrittänyt käyttää sudoa. Se ei onnistunut, mikä ei näy tässä.

perustyyppi => root

/bin/nano - 1 Time(s).

---------- Sudo (secure-log) End ----------

---------- Disk Space Begin ----------

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/sda2	15G	5.9G	8.1G	42%	/
/dev/loop0	91M	91M	0	100%	/snap/core/6350
/dev/loop1	92M	92M	0	100%	/snap/core/6259
/dev/loop2	90M	90M	0	100%	/snap/core/6130

---------- Disk Space End ----------

########## Logwatch End ##########