jamk.fi

AVOIMEN LÄHDEKOODIN PALOMUURIRATKAISUT

Jarmo Pietiläinen

Opinnäytetyö
Marraskuu 2013

Tietotekniikan koulutusohjelma
Teknologiayksikkö (TEKN+ICT)

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

KUVAILULEHTI

Tekijä(t) Pietiläinen, Jarmo	Julkaisun laji Opinnäytetyö	Päivämäärä 06.11.2013
	Sivumäärä 149	Julkaisun kieli Suomi
		Verkkojulkaisulupa myönnetty (X)

Työn nimi
AVOIMEN LÄHDEKOODIN PALOMUURIRATKAISUT

Koulutusohjelma
Tietotekniikan koulutusohjelma

Työn ohjaaja(t)
RANTONEN, Mika
HÄKKINEN, Antti

Toimeksiantaja(t)
Jyväskylän Ammattikorkeakoulu/JYVSECTEC
VATANEN, Marko

Tiivistelmä

Tämän JYVSECTECille tehdyn opinnäytetyön tarkoituksena oli tutustua avoimen lähdekoodin palomuureihin. Niiden ominaisuuksia tuli vertailla, ja niitä tuli opetella konfiguroimaan eri skenaarioita varten. Palomuureina käytettiin pfSenseä, ShoreWallia ja Vyatta Corea. Puhdasta iptables-komentoa käytettiin kerran.

Skenaariot toteutettiin virtualisoiduissa IPv4-verkoissa, jotka luotiin VirtualBox-ohjelman avulla. Työtä varten suunniteltiin eri verkkoja, joille annettiin useita toteutettavia tavoitteita. Tavoitteita olivat mm. DMZ-alueen luonti ja yrityksen sisäverkon suojaus, harjoitusverkossa olleen hallinta-aliverkon suojaus, sekä runkoreitittimien suojaaminen. Tavoitteiden toteutuksessa tarvitut komennot ja työvaiheet dokumentoitiin tarkasti.

Varsinaisten skenaarioiden lisäksi mitattiin palomuurien nopeutta. Ohjelmapohjainen palomuuri ei ole koskaan yhtä nopea kuin laitteistopohjainen. Siksi haluttiin tietää, paljonko ne hidastivat verkkoa. Kaikki nopeustestit tehtiin sadan megabitin nopeudella toimineissa verkoissa. Nopeustestit eivät olleet täysin moitteettomasti tehtyjä, joten tulokset olivat vain hieman suuntaa antavia.

Skenaarioiden toteuttamisen jälkeen voitiin todeta, että ohjelmapohjaiset palomuurit ovat hieman ristiriitaisia. Ne eivät ole lähimainkaan yhtä nopeita kuin laitteistopohjaiset palomuurit: verkon nopeus leikkaantui noin puoleen alkuperäisestä. Mutta kotikäyttäjille, pienyrityksille ja harjoitus/simulaatioverkkoihin ne ovat mainioita valintoja. Koska peruskäsitteet eivät muutu, on työssä läpikäytyjä asioita ja vaiheita mahdollista soveltaa muihinkin tuotteisiin.

Jatkokehitystä varten olisi hyvä toteuttaa nopeustesti paremmin, sekä suunnitella laajempia skenaarioita. Myös IPv6 ja VPN-tekniikat tulisi käydä läpi.

Avainsanat (asiasanat)
avoin lähdekoodi, palomuuri, VirtualBox, virtualisointi, Linux, BSD, Vyatta Core, pfSense, ShoreWall, iptables

Muut tiedot

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

DESCRIPTION

Author(s)	Type of publication	Date
Pietiläinen, Jarmo	Bachelor´s Thesis	06.11.2013
	Pages 149	Language Finnish
		Permission for web publication (X)

Title
OPEN SOURCE FIREWALL SOLUTIONS

Degree Programme
Information Technology

Tutor(s)
RANTONEN, Mika
HÄKKINEN, Antti

Assigned by
JAMK University of Applied Sciences/JYVSECTEC
VATANEN, Marko

Abstract

The purpose of this JYVSECTEC-assigned bachelor's thesis was to become familiar with open source firewall products. Their features were to be compared and configured for various scenarios. The firewalls used were pfSense, ShoreWall and Vyatta Core. The raw iptables command was used once.

The scenarios were realized in virtualized IPv4-based networks that had been built using the VirtualBox virtualization software. Various networks were designed and for each network several objectives were set. For example, in one scenario a small company required a DMZ while keeping their internal network protected; and in another scenario, the management subnet and the core routers had to be protected. The commands and tasks required to realize these objectives were meticulously documented.

Software firewalls are never as fast as hardware firewalls. Thus, it was necessary to measure exactly how much they slowed down the network. The tests were performed in networks running at the speed of 100 Mbit/s. Because the testing methods used were somewhat insufficient, the results should not be considered as complete truths, but merely as guidelines.

The results showed that software firewalls are a somewhat mixed bag of products. They are nowhere as fast as hardware firewalls: the network performance was only about a half of the original. However, they are perfectly fine choices for home users, small enterprises and for practicing/simulation networks. Because the basic concepts do not change, the ideas and scenarios covered in this work can be adapted to work with other scenarios and products.

For further development, the speed tests should be thoroughly revised and the test scenarios should be expanded. In addition, IPv6 addressing and VPN technologies should be covered.

Keywords
open source, firewall, VirtualBox, virtualization, Linux, BSD, Vyatta Core, pfSense, ShoreWall, iptables

Miscellaneous

Sisältö

Kuviot

Taulukot

Lyhenteet

ADSL	Asymmetric Digital Subscriber Line
BGP	Border Gateway Protocol
BSD	Berkeley Software Distribution
DHCP	Dynamic Host Configuration Protocol
DMZ	Demilitarized Zone
DNAT	Destination Network Address Translation
DNS	Domain Name Service
FTP	File Transfer Protocol
GNU GPL	GNU (”GNU's Not Unix”) General Public License
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
LAN	Local Area Network
MAC	Media Access Control
NAT	Network Address Translation
NTP	Network Time Protocol
OSI	Open Systems Interconnection
OSPF	Open Shortest Path First
PAT	Port Address Translation
POP3	Post Office Protocol version 3
QoS	Quality of Service
SMTP	Simple Mail Transfer Protocol
SNAT	Source Network Address Translation
SSH	Secure Shell
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UUID	Universally Unique Identifier
VLAN	Virtual Local Area Network
WAN	Wide Area Network
ZBF	Zone-Based Firewall

1 Johdanto

1.1 Tavoite

Opinnäytetyön tavoitteena oli tutustua avoimen lähdekoodin ohjelmapalomuureihin, niiden toimintaan ja konfigurointiin. Niiden ominaisuuksia tuli vertailla keskenään. Palomuureilla tuli toteuttaa eri testiskenaarioita, joissa niitä päästiin oikeasti kokeilemaan.

Opinnäytetyössä on keskitytty vain avoimen lähdekoodin palomuureihin, ei maksullisiin. Ohjelmapohjaisia palomuureja on saatavilla vaikka kuinka paljon, mutta pääasiassa keskityttiin kolmeen eri tuotteeseen. Käytetyt palomuurit (lueteltu tarkemmin luvussa 3.1, sivulla 16) valittiin niiden käyttötarkoituksen, levinneisyyden ja erilaisten toteutustapojensa perusteella. Eri asiat tehdään eri tavalla eri tuotteitta. Tällä haettiin hieman vertailtavuutta.

Opinnäytetyö tehtiin Jyväskylän Ammattikorkeakoulun koordinoiman JYVSECTEC (Jyväskylä Security Technology) -kyberturvallisuusteknologian kehittämishankkeen alaisuudessa. JYVSECTEC ylläpitää ja kehittää kyberturvallisuuden kehitysympäristöä, jossa tuotetaan kehitys-, testaus- ja koulutuspalveluita yhteistyöverkoston käyttöön.

1.2 Työn taustaa

Tietoverkkojen turvallisuus on yksi nyky-yhteiskunnan tukipilareista. Alkuaikojen täysin avoin ja suojaamaton internet on jo kauan sitten taakse jäänyttä aikaa. Verkkoja on suojattava useilla eri kerroksilla, alkaen aina infrastruktuurin fyysisestä suojauksesta ja päättyen varsinaisen käsiteltävän tiedon suojaamiseen. Suojauksen kerrosten ja suojausmenetelmien on mentävä päällekkäin.

Palomuurit sijoittuvat tässä monikerroksisessa suojamallissa usealle tasolle. Ne sekä eristävät verkkoja toisistaan ja rajoittavat niiden liikennöintiä, mutta ne myös valvovat yksittäisten ohjelmien ja palveluiden pääsyä eri verkkoihin ja kohteisiin.

Eri palomuurituotteita on saatavilla vaikka kuinka paljon, ja joskus voi olla hyvinkin vaikea valita niistä sopivin. Tämä työ ei luonnollisestikaan ole tyhjentävä katsaus kaikkiin palomuureihin, vaan lähinnä kevyt pintaraapaisu. Silti monet siinä esitetyt asiat ja käsitteet ovat yleisiä ja ne toimivat minkä tahansa palomuurin kanssa. Jotkut toteutustavatkin voivat olla tarpeeksi yleisiä, jotta niitä voi soveltaa sellaisenaan muualla.

2 Palomuurit

2.1 Mitä palomuurit ovat ja mihin niitä tarvitaan?

Palomuuri on laite tai ohjelma, joka jollain ennalta määrätyllä tavalla estää ja sallii verkkoliikennettä. Palomuurille annetaan joukko sääntöjä, jotka kuvaavat, millaista liikennettä halutaan pääsevän läpi ja millaista ei. Säännöt sisältävät asioita kuten lähde- ja kohdeosoite, protokolla, porttinumerot ja mahdollisesti eri protokolliin liittyviä tilatietoja. Palomuurit ovat siis samankaltaisia, kuin fyysiset, talojen rakenteissa esiintyvät tavallista järeämmät seinät, joiden tarkoitus on nimensä mukaan estää tulen leviämistä tulipalon aikana. (Cheswick, Bellowin & Rubin 2003, 175-176.)

Palomuureja käytetään usein erottamaan verkko vähintään kahteen eri osaan: sisä- ja ulkopuoli. Usein sisäpuolelta ulkopuolelle tapahtuva liikennöinti sallitaan, mutta toisinpäin sitä rajoitetaan niin paljon kuin mahdollista.

Palomuureja tarvitaan, koska ei ole viisasta päästää ketä tahansa käsiksi mihin tahansa verkkoon. Saapuvia ja lähteviä yhteyksiä on rajoitettava ei pelkästään tietoturvan vuoksi, vaan myös luotettavuuden vuoksi: yrityssalaisuuksia sisältäviä papereita ei säilötä eikä käsitellä julkisilla paikoilla, eikä niitä jätetä levälleen julkisille paikoille; miksi siis yrityksen sisäisen verkon sisältö pitäisi olla vastaavasti julkisesti selattavissa?

Palomuurit toimivat yleensä verkkotekniikassa laajalti käytetyn OSI-mallin kerroksilla 2-4. On olemassa myös puhtaita sovelluskerroksen, eli tason 7, palomuureja. Niitä käytetään usein rajoittamaan yksittäisten ohjelmien ja palveluiden pääsyä verkkoon. (Mts. 185-186.)

Palomuureista puhuttaessa on aina muistettava yksi tärkeä asia: palomuureja ei yleensä ole tarkoitettu haittaohjelmien torjuntaan. Niillä voidaan joissain tilanteissa estää joidenkin tunnettujen haittaohjelmien aiheuttamaa verkkoliikennettä,

mutta ne eivät korvaa varsinaisia haittaohjelmien torjuntaan tarkoitettuja ohjelmia. Torjuntaohjelmat sisältävät usein tason 7 palomuurin sisäänrakennettuna. (Mts. 194-195.)

Tietoturvan kannalta verkossa pitäisi aina olla useampi kuin yksi palomuuri. Ei pidä luottaa pelkästään yhteen verkon rajalla olevaan yhteen muuriin. Jokaisessa verkon laitteessa tulisi käyttää edes jotain palomuuritoimintoja. Jokaisen verkon laitteen tulisi pyrkiä suojaamaan itseään ja rajoittamaan omia yhteyksiään. Tämä ei ole ikävä kyllä ole aina mahdollista. Palomuurien lisäksi työasemissa tulisi aina käyttää ajantasaista haittaohjelmien torjuntaa. (Mts. 193-194.)

2.2 Palomuurien yleinen toiminta

Palomuurit ovat usein jollain tavalla reitittäviä laitteita (myös siltaavia laitteita on olemassa, katso liite 3 sivulla 145). Ne lukevat muurin läpi kulkevista paketeista pakettien osoitteet, protokollat, portit ja muut tarvittavat tiedot. Näitä tietoja verrataan palomuurisääntöihin ja jos täsmäävä sääntö löydetään, katsotaan sen tiedoista mitä paketille tulisi tehdä. Läpi päästettävät paketit ohjataan eteenpäin.

Eri palomuurit toimivat OSI-mallin eri kerroksilla. Yleisesti ottaen kaikki palomuurit pystyvät suodattamaan liikennettä vähintään IP-osoitteiden perusteella, mutta MAC-osoitteilla toimivia löytyy myös. Osoitteiden lisäksi useimmat tukevat protokollan ja porttinumeroiden mukaan tehtävää suodatusta. (Wikipedia: Firewall n.d.)

Palomuureista on olemassa kahta päätyyppiä: **tilaton** (engl. *stateless*) ja **tilallinen** (engl. *stateful*). Tilattomat palomuurit ovat puhtaita **pakettisuodattimia** (engl. *packet filter*). Ne eivät tarkkaile yhteyksiä, vaan näkevät pelkästään yksittäisiä paketteja. Ne estävät tai sallivat paketteja puhtaasti niiden osoitteiden, protokollien ja porttinumeroiden perusteella. Tilalliset palomuurit puolestaan pitävät kirjaa eri protokollien (kuten TCP) yhteyksien tilasta ja sallivat tai estävät tietyt tilat ja niiden yhdistelmät, ja niihin liittyvät paketit. Ne ”muistavat”, ainakin hetkellisesti,

verkon tilan ja tekevät päätöksiä sen perusteella. Tilalliset palomuurit eivät ole täysin aukottomia ratkaisuja, sillä on olemassa useita tekniikoita, joilla ne saadaan houkuteltua avaamaan joku portti ulkoapäin tulevalle liikenteelle. Mutta yleisesti ottaen ne ovat erittäin turvallisia. (Stateful Firewall n.d.; TCP hole punching n.d.)

Klassinen esimerkki tilattoman ja tilallisen palomuurin toiminnasta on vanha, mutta laajalti käytetty FTP-tiedostojensiirtoprotokolla. FTP käyttää kahta porttia, ja ongelmaksi muodostuukin, miten tilattoman palomuurin saa ymmärtämään, että eri porteissa liikkuvat paketit kuuluvat yhteen? Tilaton palomuuri pudottaa paketteja riippumatta siitä kuuluvatko ne sallittuun yhteyteen. Tilalliselle palomuurille tämä ei ole ongelma, koska se näkee yhteydenavauspaketit ja pitää kirjaa istunnosta ja osaa eritellä sen. Tilallinen palomuuri voidaan myös helposti asettaa estämään tiettyyn suuntaan kulkevat TCP-istunnot. Tilattomalle palomuurille tällainen voi olla hyvinkin vaikea, jollei peräti mahdoton, operaatio. (Cheswick, Bellowin & Rubin 2003; 202, 229.)

Palomuurit eivät pysty suodattamaan ihan mitä tahansa liikennettä. Esimerkiksi BitTorrent-protokollaa on hyvin vaikea suodattaa, koska se kulkee tavallisen HTTP-protokollan päällä. Jos HTTP on sallittu, toimii myös BitTorrent, ainakin jollain tapaa (palomuuri voi silti estää saapuvia yhteyspyyntöjä). Eräs tapa olisi valvoa käytetyn kaistanleveyden määrää ja arvioida tilastojen avulla liikenteen tyyppiä. Toinen tapa olisi purkaa paketin sisältö kokonaan ja katsoa, mitä se oikeasti sisältää. Tästäkään ei ole apua, jos torrent-paketit salaa. Yksikään testatuista palomuureista ei kuitenkaan tukenut pakettien aukaisemista. Pakettien perusteellinen läpikäynti vaatii käytännössä laitteistotuen, koska muuten verkko hidastuu liikaa.

2.3 Ohjelmapohjaiset ja laitteistopohjaiset palomuurit

Ohjelmapohjaiset palomuurit ovat puhtaita ohjelmia, eli ne tekevät kaikki palomuuritoiminnot (ja mahdollisen reitityksen) täysin ohjelmallisesti.

Laitteistopalomuurit sisältävät fyysisiä integroituja piirejä, jotka voivat tehdä reititystä ja/tai siltausta, sekä palomuuritoimintoja. Koska laitteisto ei ole yhtä joustava kuin ohjelmisto, tekevät laitteistopalomuurit vain lähinnä tavallista pakettisuodatusta (kuten lähdeosoitteen ja kohdeosoitteen tarkistuksen, protokollatarkistukset, yms.). Laitteistopalomuureissa voi olla myös ohjelmakomponentteja mukana. Tällöin nämä ohjelmakomponentit voivat tehdä laitteisto-osasta läpi päässeille paketeille perusteellisemmat tarkistukset. Usein kotikäyttäjien ADSL-modeemit voidaan lukea laitteistopalomuureiksi (mutta niiden nopeus ei välttämättä ole kovin kehuttava, kuten luku 6 sivulla 76 osoittaa).

Koska laitteistopalomuuri hoitaa osan asioista fyysisellä laitteistolla, on se yleensä nopeampi kuin puhdas ohjelmapalomuuri. Ohjelmapalomuurin etuna on kuitenkin lähes rajaton muokattavuus ja laajennettavuus. Nämä tulevat nopeuden kustannuksella. (Firewall Debate: Hardware vs. Software 2011; The Differences and Features of Hardware and Software Firewalls 2011.)

2.4 Demilitarisoidut vyöhykkeet

Demilitarisoitu vyöhyke (DMZ) tarkoittaa palomuurien ja tietoverkkojen yhteydessä aluetta, jossa palomuurin suojelemalla alueella sijaitseviin laitteisiin ja palveluihin voidaan ottaa yhteys ulkopuolelta. Toisin kuin suojatun sisäverkon laitteet, DMZ-alueen laitteet ja palvelut siis näkyvät ulospäin jollain tavalla. Palomuurin tehtävänä on lähinnä estää sallimaton liikenne DMZ-alueelta muihin verkkoihin. Nimitys tulee suoraan reaalimaailman vastaavasta termistä, jolla tarkoitetaan aluetta, jolla ei saa harjoittaa sotilasoperaatioita. (Cheswick, Bellowin & Rubin 2003, 14-15.) Esimerkki DMZ-alueesta löytyy kuvioista 1, 2 ja 3 (sivuilla 26 ja 27).

DMZ-alueen laitteet voivat, tilanteesta riippuen, ottaa yhteyden sisäverkossa oleviin palveluihin. Esimerkiksi yrityksen sähköpostipalvelin voi sijaita DMZ-alueella. Tämä palvelin ei kuitenkaan tallenna sähköposteja. Ne on oikeasti tallennettu sisäverkossa olevalle palvelimelle ja DMZ-alueen palvelin toimii välityspalvelimena.

Sanomattakin on selvää, että DMZ-alueen laitteet on suojattava hyvin, eikä niissä pidä pitää päällä mitään muuta kuin täysin välttämättömät palvelut. (DMZ n.d.)

Liikenne DMZ-alueelle ja takaisin hoidetaan osoitteidenmuunnosten avulla (käsitelty seuraavassa luvussa). Ulkoa DMZ-alueelle tuleva liikenne tunnistetaan joko kohdeosoitteen ja/tai -porttinumeron avulla ja muutetaan DNAT-muunnoksella sisäverkon osoitteeksi. Paluuliikenne DMZ-alueelta tunnistetaan lähdeosoitteen avulla ja muutetaan SNAT-muunnoksella takaisin ulkoverkon osoitteeksi.

2.5 Osoitemuunnokset

Palomuureihin olennaisesti liittyvä asia on osoitteenmuunnos. Osoitemuunnokset ovat pääasiassa reitittimien tekemiä, mutta koska palomuurit toimivat usein reitittiminä (tai palomuuri on osa reititintä), kuuluvat ne usein yhteen. Muunnoksia kutsutaan yhteisesti nimellä **verkko-osoitemuunnos** (NAT). Muunnoksia on olemassa useita eri tyyppejä, joita käytetään tilanteesta riippuen. Seuraavassa listassa on lueteltu eri muunnokset, niiden toiminta lyhyesti ja niiden käyttökohteita.

- **Kiinteä muunnos** (engl. *static NAT*) eli yhdestä yhteen -muunnos on kaikkein yksinkertaisin. Se vaihtaa IP-osoitteen verkko-osion (tai jopa koko osoitteen) toiseksi. Esimerkiksi verkko 172.16.1.0/24 voidaan asettaa muuttumaan verkoksi 203.0.113.0/24. Tällöin osoite 172.16.0.37/24 muuttuu ulospäin mentäessä osoitteeksi 203.0.113.37/24 ja vastaavasti päinvastoin takaisin tullessa. Kiinteätä muunnosta voidaan käyttää, jos ulkoverkossa on käytettävissä samankokoinen aliverkko kuin sisällä. Voidaan myös käyttää yksittäisten osoitteiden muuttamista kokonaan toiseksi.

- **DNAT**, eli kohde-NAT, muuttaa verkkoon tultaessa osoitteen joksikin toiseksi. Tätä käytetään usein kun ulkoverkosta pitää saada yhteys sisäverkossa olevaan laitteeseen. Tällöin ulkoverkon IP-osoite muutetaan sisäverkon laitteen osoitteeksi. DNATista käytetään usein nimitystä **portin uudelleenohjaus** (engl. *port forwarding*).

- **SNAT**, eli lähde-NAT, muuttaa verkosta lähdettäessä osoitteen joksikin toiseksi. Tätä käytetään usein muuttamaan sisäverkosta tulevien pakettien osoite ulkoverkkoon soveltuviksi.

- **Kaksisuuntainen NAT** (engl. *bidirectional NAT*) saadaan aikaan, kun DNAT ja SNAT ovat käytössä yhtä aikaa. Kaksisuuntainen NAT sallii ulkomaailman ja sisäverkon laitteen keskinäisen viestinnän. Kummastakin verkosta voi ottaa yhteyden toiseen verkkoon, eikä kumpikaan verkko tiedä toisen verkon oikeista osoitteista mitään.

- **Osoitteen piilotus** (engl. *masquerade NAT*) on erikoistapaus. Tässä yhden tai useamman kokonaisen verkon kaikki osoitteet (ja porttinumerot) muunnetaan tarvittaessa yhdeksi IP-osoitteeksi ja takaisin. Verkosta ulospäin lähdettäessä reititin verkon rajalla tallentaa taulukkoon paketin alkuperäisen lähdeportin ja -osoitteen. Lähdeportti muutetaan sitten satunnaiseksi ja lähdeosoitteeksi otetaan rajapinnan IP-osoite.

  Koska palaavissa paketeissa kohdeportti on sama kuin aiemmin valittu satunnainen portti, voidaan sen avulla poimia taulukosta alkuperäinen portti ja IP-osoite. Paluupaketin osoite ja portit vaihdetaan alkuperäisiksi ja se välitetään eteenpäin sisäverkossa olleelle alkuperäiselle laitteelle. Merkintä poistetaan taulukosta hetken kuluttua ja uuden yhteyden syntyessä valitaan toinen satunnainen portti. Osoitteen piilotusta kutsutaan usein myös nimellä **porttimuunnos** (PAT), koska se muuttelee osoitteen lisäksi porttinumeroita.

Osoitteenmuutoksiin on useita syitä ja käyttötarkoituksia. Yleensä sitä käytetään yksinkertaisesti piilottamaan sisäverkon rakenne ulkoapäin. Mahdollisen hyökkääjän työ vaikeutuu, kun hän näkee vain julkisia osoitteita eikä tiedä verkon rakenteesta mitään. Sitä tarvitaan myös, jos käytössä ei ole kuin yksi julkinen IP-osoite.

Osoitteen piilotus ei ole varsinaisesti palomuuri. Se kyllä estää ulkoverkosta tulevat ei-toivotut yhteyspyynnöt, mutta se on väärä tekniikka niiden estämiseksi (tilallinen palomuuri on oikea ratkaisu tätä varten). Piilotus estää ulkoa tulevat yhteyspyynnöt, koska porttinumerotaulukossa ei ole merkintää siitä, mihin ulkoa tuleva paketti pitäisi sisäverkossa ohjata. Merkintä on olemassa vain, jos sisältä otetaan ensin yhteys ulospäin, ja silloinkin se on vain hetkellinen. Hyökkääjä näkee vain yhden julkisen IP-osoitteen eikä tiedä sisäverkosta yhtään mitään. Kun merkintää ei ole, saapuva paketti tuhotaan.

Osoitteen piilotusta tarvitaan silloin, kun julkisia IP-osoitteita on käytössä vain yksi, mutta verkkoyhteyttä tarvitsevia koneita on enemmän kuin yksi. Esimerkiksi kotikäyttäjien ADSL-reitittimet tekevät osoitteen piilotusta. Koska palveluntarjoajilta saa vain yhden julkisen IP-osoitteen, ei sillä ilman piilotusta saisi verkkoon kuin yhden laitteen kerrallaan. Osoitteen piilotuksen ansiosta kotiverkossa voi kuitenkin olla useita eri laitteita, jotka ovat yhteydessä internetiin samanaikaisesti. Osoitteen piilotus liittyykin vahvasti vanhaan IPv4-tekniikkaan, jossa julkisten osoitteiden vähyys pakottaa sen käyttöön. Uudempi IPv6 ei tarvitse osoitteiden piilotusta, mutta palomuurin kylläkin.

Osoitteen piilotusta on moitittu ankarasti, sillä se tuhoaa internetin perinteisen ”päästä päähän”-yhteydellisyyden. Jos piilotuksen läpi halutaan sallia tietyt yhteydet ulkoapäin, tarvitaan usein hyvinkin hankalasti toteutettuja porttien uudelleenohjauksia. Uudelleenohjaus kertoo palomuurille, että se voi päästää ulkoapäin tiettyyn porttiin tulevat yhteyspyynnöt läpi sellaisenaan. (IP Network Address Translation Protocol 2005.)

2.6 Vyöhykepohjainen palomuuri

Vyöhykepohjainen palomuuri (ZBF) tarkoittaa käsitteenä palomuuria, joka jakaa verkon kahteen tai useampaan eri vyöhykkeeseen. Joka vyöhykkeeseen liitetään yksi tai useampi rajapinta, ja vyöhykkeille luodaan säännöt, jotka kuvaavat, millai-

nen liikenne saa kulkea vyöhykkeiden välillä ja mihin suuntaan. Usein saman vyöhykkeen sisällä olevien rajapintojen välillä liikenne saa kulkea vapaasti.

Vyöhykkeitä käytetään, koska ne yksinkertaistavat palomuurien konfigurointia huomattavasti. Vyöhykkeitä tukevat palomuurit usein estävät oletuksena kaiken vyöhykkeiden välisen liikenteen, ja niille on erikseen kerrottava, mikä liikenne saa kulkea ja minne. Ne siis parantavat tietoturvaa, koska oletustoiminto on aina esto. Erillisiä palomuurisääntöjä käytettäessä (eli ei-vyöhykepohjainen palomuuri) jotain ei-toivottua saattaa aina vahingossa päästä läpi. (Understanding Zone Based Firewalls 2011.)

On huomattava, ettei DMZ-tekniikalla ole nimestään huolimatta varsinaisesti mitään tekemistä vyöhykepohjaisten palomuurien kanssa. Palomuuri, joka ei ole vyöhykepohjainen, osaa silti tehdä erillisen DMZ-alueen.

Työssä käytetyistä palomuureista ShoreWall ja Vyatta Core tukevat vyöhykkeitä. ShoreWall ei itse asiassa edes tue ei-vyöhykepohjaista toimintaa, kun Vyatta Coressa se on valinnainen.

2.7 Avoin lähdekoodi, Linux ja BSD

2.7.1 Avoin lähdekoodi

”Avoin lähdekoodi” tarkoittaa, että jonkin ohjelman lähdekoodi on vapaasti kenen tahansa luettavissa. Yleensä kaupallisten ohjelmien lähdekoodeja ei jaeta valmistajan ulkopuolelle. Avoimen lähdekoodin lisenssien yksityiskohdat vaihtelevat, mutta kaikille on yhteistä se, että käyttäjä saa halutessaan muokata lähdekoodia ja kääntää ohjelmasta uuden version. Käyttäjä voi näin halutessaan korjata ohjelmassa mahdollisesti olevia vikoja ja lisätä siihen uusia ominaisuuksia. Ei-avointen ohjelmien kohdalla ei voi tehdä muuta, kuin ottaa yhteyttä valmistajaan ja toivoa, että vika korjataan seuraavassa versiossa. Lisenssiehdot yleensä myös sallivat muokattujen versioiden jatkolevityksen jollain tapaa. Usein avoimen lähdekoodin

ohjelmien käyttötarkoituksiakaan ei rajoiteta. Kaupalliset ohjelmat saattavat kieltää ohjelmien käytön joissain tilanteissa ja maissa, mutta avoimen lähdekoodin ohjelmissa tällaisia rajoituksia näkee harvoin. (The Open Source Definition n.d.)

Koska kuka tahansa voi lukea lähdekoodia, voidaan ohjelman toimintaa tutkia vapaasti. Esimerkiksi tietoturva-alalla on tärkeää, että salaus- ja suojausohjelmien lähdekoodin voi tutkia ja selvittää, toimiiko ohjelma kuten pitääkin. Ohjelmiin ei saa näin piilotettua mahdollisia takaportteja, tai tahallisia vikoja, jotka vaikuttaisivat suojaukseen.

2.7.2 Linux ja BSD

Linux on suomalaista alkuperää oleva Unix-klooni. Unix on alkujaan 1969 kehitetty käyttöjärjestelmä, joka levisi laajalle 1970- ja 1980-luvuilla. Linuxin ydintä levitetään amerikkalaisen Richard Stallmanin luoman GNU GPL -lisenssin versio 2 alla, joka sallii kenen tahansa käyttää ydintä mihin tahansa tarkoitukseen, muokata sitä vapaasti ja levittää muokattuja versioita niin kauan, kun muokattua lähdekoodia jaetaan myös saman lisenssin alla. Ytimen muokattavuuden ja pienempiin osiin jakamisen vuoksi Linux onkin nähnyt käyttöä pöytäkoneiden ja supertietokoneiden käyttöjärjestelmänä. Se myös muodostaa pohjan monille nykyisille kännyköiden käyttöjärjestelmille. (History of Linux n.d.) Kaksi tässä työssä käytetystä palomuurista rakentui Linuxin päälle.

Kalifornialaista syntyperää oleva BSD puolestaan pohjautuu alkuperäisiin Unix-järjestelmiin eikä ole siten ”klooni” Linuxin tapaan. Muilta osin erot Linuxiin ovat lähinnä lähdekooditasolla. BSD ei varsinaisesti ole itsessään Unix-jakelu, vaan kollektiivinen nimi usealle eri jakelulle. Varsinaisia jakeluita ovat mm. FreeBSD, NetBSD ja OpenBSD. FreeBSD on yleiskäyttöinen käyttöjärjestelmä ja on Linuxin tapaan erittäin muokattavissa. NetBSD puolestaan toimii melkein millä tahansa laitealustalla ja OpenBSD mainostaa itseään maailman turvallisimpana käyttöjärjestelmänä. (Berkeley Software Distribution n.d.) Tässä työssä käytetty pfSense on rakennettu FreeBSD:n päälle.

3 Testauksen taustaa

3.1 Käytetyt palomuurit

Saatavilla olevia avoimen lähdekoodin palomuureja on olemassa useita kymmeniä, mutta tähän työhön valittiin mukaan seuraavat:

- pfSense
  - Valmistajan kotisivut: pfsense.com
- ShoreWall
  - Valmistajan kotisivut: www.shorewall.net
- Vyatta Core
  - Valmistajan kotisivut: www.vyatta.org
- iptables (lyhyesti)
  - Tulee jo käyttöjärjestelmän mukana.

Jokainen näistä on esitelty lyhyesti luvuissa 3.1.1 - 3.1.4. Luku 3.2 (sivulla 20) sisältää laajemman ominaisuuksien vertailun. Tuotteista käytetyt versiot ja niiden asennuksessa käytetyt tiedostot selviävät taulukosta 1. Asennusohjeet näille löytyvät liitteestä 1 (sivulta 94 alkaen).

Taulukko 1. Käytetyt ohjelmistoversiot ja asennustiedostot

Tuote	Versio	Asennustiedoston nimi
pfSense	2.0.3	pfSense-LiveCD-2.0.3-RELEASE-i386-20130412-1022.iso
Vyatta Core	6.6-R1	vyatta-livecd_VC6.6R1_i386.iso
ShoreWall	4.5.5.3	Asennettiin `apt-get` -komennolla. Katso Debianin asennus liitteestä 1 (sivulla 103).
iptables	1.4.14	Asentui jo osana järjestelmää. Katso Debianin asennus liitteestä 1 (sivulla 103).

3.1.1 pfSense

pfSense on FreeBSD-pohjainen palomuuri. Se ei ole ainoa BSD-pohjainen ratkaisu, mutta ehkä tunnetuin niistä. BSD-pohjaiset järjestelmät eivät käytä Linux-ytimen iptables/netfilter-suodatinta (ks. luku 3.1.4 sivulla 19), vaan omaa, pf-nimistä suodatinta. pf on alkujaan OpenBSD-projektin kehittämä, mutta on sieltä levinnyt muuallekin. Tästä muodostuukin pfSensen nimi: ”*pf*” ja ”*sense*”, eli ohjelman tarkoituksena on tehdä pf-suodattimen säännöistä järkeviä ja selkeitä. Tässä työssä ei kuitenkaan käytetty pf-suodatinta raakana, toisin kuin iptablesia.

pfSense on alkujaan m0n0wall-nimisestä ohjelmasta irtaantunut projekti. m0n0wall on suunniteltu käytettäväksi lähinnä sulautettujen järjestelmien kanssa, mutta pfSense on tarkoitettu asennettavaksi täysiveriseen tietokoneeseen. Ydin on kummassakin sama, mutta pfSense tukee isoa joukkoja ominaisuuksia, joita m0n0wall ei tue. Käyttöliittymien erot m0n0wallin ja pfSensen välillä ovat lähinnä kosmeettisia.

pfSense konfiguroidaan selainpohjaisen käyttöliittymän kautta. Se sisältää myös komentorivin, mutta se on tarkoitettu lähinnä rajapintojen konfigurointiin asennusvaiheessa, ohjelman tilan tarkasteluun ja vikatilojen korjaamiseen.

pfSense on tarkoitettu käytettäväksi julkisen internetin ja sisäverkon rajalla. Sitä voi käyttää missä tahansa kohtaa verkkoa, mutta oletusasetuksillaan ohjelma ei tähän sovellu. (pfSense n.d.)

3.1.2 ShoreWall

ShoreWall (yleisesti käytetty lyhennelmä nimestä *Shoreline Firewall*) ei varsinaisesti ole itsessään palomuuri, vaan ohjelma, joka ohjaa netfilter-järjestelmän (ks. luku 3.1.4) toimintaa. ShoreWall on vyöhykepohjainen (se ei edes sisällä ei-vyöhykepohjaisia toimintoja) ja käyttää apunaan tekstitiedostoja, joissa kuvataan itse vyöhykkeet, niihin kuuluvat rajapinnat sekä säännöt sille, millainen liikenne vyöhykkeiden välillä saa kulkea ja mihin suuntaan. Tiedostoissa kerrotaan myös mahdol-

liset NAT-säännöt sekä kaikki muut palomuuriin ja suodatukseen liittyvät asiat. (ShoreWall n.d.)

ShoreWall on erillinen ohjelma, joka ajetaan järjestelmän käynnistyksen yhteydessä (tai erikseen käynnistettynä komentoriviltä). Se lukee asetustiedostonsa ja konfiguroi sitten ytimen palomuurin. Jos ShoreWallin ajaa alas sen hallintaohjelmalla, se palauttaa alkuperäiset asetukset paikoilleen. Näin sen tekemät muutokset voidaan hetkellisesti ottaa pois käytöstä, ja uusia asioita voi testata helposti. (What Is Shorewall? 2013.)

ShoreWall ei konfiguroi rajapintojen osoitteita, vaan ne on konfiguroitava itse valmiiksi käytetyn Linux-jakelun käyttämällä tavalla. ShoreWall ei myöskään sisällä esimerkiksi DHCP- eikä DNS-palvelinta, vaan ne on asennettava erikseen. ShoreWall sopiikin sellaiselle, joka haluaa koota palomuurin itse ”palasista” juuri sellaiseksi kuin haluaa, mutta ei halua opetella raa'an iptables-ohjelman komentoja. ShoreWall ei sisällä mitään graafista käyttöliittymää, mutta sitä voi ohjata suositun selainkäyttöisen Webmin-ohjelman kautta. (Webmin: Standard Modules n.d.)

ShoreWall ei välitä siitä, mikä rajapinnoista kuuluu mihinkin verkkoon. Se vain konfiguroi palomuurin ohjeiden mukaan. Siksi sitä voidaan käyttää sellaisenaan palomuurina missä tahansa kohtaa verkkoa.

3.1.3 Vyatta Core

Vyatta Core on testatuista tuotteista erilaisin: se on täysiverinen Linux-jakelu, joka on suunniteltu tekemään ohjelmapohjaista reititystä. Kaikki testatut tuotteet pystyvät tekemään reititystä, mutta vain Vyatta Core sisältää jo valmiiksi tuen RIP-, OSPF- ja BGP-reititysprotokollille. Se tukee palvelunlaatua (QoS), NAT-sääntöjä sekä palomuurina toimintaa (sekä erillisistä säännöistä koostuvaa, ja vyöhykepohjaista) sellaisenaan. Se on muokattu Debian Linux -jakelusta ja osaa täydentää komentoriviltä suoraan kaikkia komentoja. Kuten ShoreWall, Vyatta Core tu-

kee vyöhykepohjaista palomuuria, mikä helpottaa huomattavasti varsinkin DMZ-alueen luontia. (Vyatta 2012.)

Vyatta Core pystyy toimimaan sekä tilattomana että tilallisena palomuurina yhtä aikaa. Oletuksena se on tilaton ja vain halutut palomuuritoiminnot toimivat tilallisena (ohjelman voi pakottaa toimimaan koko ajan tilallisena). Tähän törmätään esimerkiksi luvuissa 4.5.1 (sivulla 46) ja 5.4.3 (sivulla 67). (Firewall reference guide 2013.)

Vyatta Coresta on olemassa myös kaupallinen versio (Vyatta Subscription Edition), joka tukee joitain lisäominaisuuksia (kuten tuki sarjaliitännöille ja valinnainen selainpohjainen konfiguraatio-ohjelma), joita Core ei tue. Tässä työssä ei kuitenkaan ole tehty mitään, mihin pelkkä Core-versio ei riittäisi. (Vyatta 2012.)

Vapaasta konfiguroitavuudestaan johtuen Vyatta Core sopii käytettäväksi missä tahansa kohtaa verkkoa. Kuten muutkin Linux-pohjaiset palomuurit, myös Vyatta käyttää palomuurin toteutukseen netfilter/iptables-järjestelmää.

3.1.4 iptables

Linuxin ydin ylläpitää netfilter-nimistä palomuuria. Se on totutettu joukkona moduuleita, joista jokainen hoitaa eri osa-aluetta ja ylläpitää omaa taulukkoaan, joka kertoo, mitä asioita eri paketeille tulee tehdä. Näitä taulukoita kutsutaan usein yhteisnimellä iptables. iptables on myös komentoriviohjelman nimi, jolla näitä taulukoita ylläpidetään. netfilter pystyy nimestään huolimatta tekemään myös NAT-muunnoksia ja erikoisempiakin pakettien muunnoksia. Se osaa toimia sekä tilattomana että tilallisena palomuurina. Työssä käytettiin pelkkää iptables-taulukkoa, joka hoitaa IPv4-liikenteen suodatusta. (iptables n.d.)

iptables/netfilter on merkittävä eräästä syystä: monet Linux-pohjaiset palomuuriratkaisut ovat pohjimmiltaan ”vain” iptablesin korkeampitasoisia käyttöliittymiä. Esimerkiksi Vyatta Core muuttaa asetuksia tallennettaessa omat palomuurisään-

tönsä iptablesille kelpaavaan muotoon ja ottaa ne sen kautta käyttöön. ShoreWall puolestaan konfiguroi käynnistyessään iptablesin omien sääntöjensä mukaan, ja palauttaa vanhat sammutuksen aikana.

iptables sopii käytettäväksi missä tahansa kohtaa verkkoa. Kaikki tässä työssä esitellyt palomuurien konfiguraatiot voisi toteuttaa ”raakana” pelkästään iptablesin avulla, mutta helppoa se ei olisi. iptablesia on käytetty sellaisenaan luvussa 5.5.2 (sivulla 74), jossa sillä rajoitetaan sisäverkon palvelimeen saapuvia yhteyksiä.

3.2 Ominaisuusvertailu

Palomuuria valittaessa on tärkeää vertailla kilpailevia tuotteita keskenään. Työssä käytettyjä palomuureja vertailtaessa törmättiin pienimuotoiseen ongelmaan: niiden ominaisuudet olivat lähes identtiset. Tuotteista kolme käytti samaa suodatusjärjestelmää (Linuxin ytimen netfilter) ja vain yksi (pfSense) oli selkeästi erilainen tässä suhteessa. Jokainen tuote pystyi tekemään kutakuinkin samoja suodatustoimintoja, joten niistä mainittiin vain muutamia (kuten osoite-, portti- ja protokollasuodatus).

Suurimpia eroja löytyi vasta, kun palomuureja tarkasteltiin laajempina kokonaisuuksina. Vain osa tuotteista toimi vyöhykepohjaisena ja jotkut toimivat vain tilallisena. Lisäksi, johtuen erilaisista konfigurointitavoista, eri tuotteet sopivat eri kohderyhmille.

Taulukoissa onkin mainittu lähinnä asioita, jotka selkeästi erottavat tuotteita toisistaan. Ne eivät ole täysin tyhjentävät; yksityiskohdista saisi vaikka kymmenen sivua taulukoita, mutta niitä ei kukaan jaksaisi lukea. Olisi esimerkiksi turha alkaa luetella mitä erityyppisiä NAT-muunnoksia eri tuotteet osaavat tehdä (kaikki luvussa 2.5 (sivulla 11) luetellut muunnokset löytyvät joka tuotteesta jossain muodossa). Siksi taulukoita tiivistettiin rankasti. Niihin pyrittiin valitsemaan asioita, jotka helpottavat oikean palomuurin valinnassa.

3.2.1 Yleiset ominaisuudet

Eri palomuurit on tarkoitettu eri tarkoituksiin. Ne sopivat eri kohtiin verkkoa. Jokaisella on omat kohdeyleisönsä: pfSensen osaa asentaa melkeinpä jokainen, kun taas ShoreWallin kanssa Linuxin komentoriviä osaamaton on täysin hukassa. Joistain löytyy kaupallista versiota, joitain voi konfiguroida selaimen kautta. Taulukko 2 listaa ja vertailee näitä ominaisuuksia. Taulukosta löytyvät myös tiivistelmät tarvittavista koneresursseista ja muista erikoisuuksista.

Taulukko 2. Yleiset ominaisuudet

	pfSense	**ShoreWall**	**Vyatta Core**	**iptables**
Järjestelmä/ ydin	BSD	Linux	Linux	Linux
Kaupallinen versio	Ei	Ei	Kyllä	Ei
Kohdeyleisö	Kotikäyttäjät	Edistyneemmät käyttäjät	Edistyneemmät käyttäjät	Edistyneimmät käyttäjät
Paras sijainti verkossa	Sisä- ja ulkoverkon rajalle	Missä tahansa kohtaa	Missä tahansa kohtaa	Yksittäiset koneet
Konfigurointitapa	Selain, osittainen komentorivituki	Komentorivi, WebMin	Komentorivi, selain kaupallisessa versiossa	Komentorivi, useita erillisiä työkaluja saatavilla
Lisäosatuki	Kyllä, kopioi ja asentaa lisäosapaketteja	Koneeseen voi asentaa normaalisti mitä haluaa	Tukee Debianin pakettien asennusta pakettihallinnan kautta	Koneeseen voi asentaa normaalisti mitä haluaa
Laitteistovaatimukset	1 GB kiintolevy 192 MB RAM	Jos Linux toimii, toimii tämäkin	1 GB kiintolevy 512 MB RAM	Jos Linux toimii, toimii tämäkin
Muuta		Pohjimmiltaan ”vain” erillinen ohjelma	Perusteellinen ohjelmistopohjainen reititin/ palomuuri	Usein vakiona mukana Linux-jakeluissa

3.2.2 Sisäverkon palvelut

Koska palomuurit toimivat usein reitittävinä laitteina sisäverkon ja ulkoverkon välillä, löytyy niistä palveluita sisäverkon tarpeeseen. Näin sisäverkkoon ei tarvita näitä varten erillistä palvelinta. Taulukko 3 listaa tarpeellisimmat sisäverkon palvelut.

Taulukko 3. Sisäverkon palvelut

	pfSense	**ShoreWall**	**Vyatta Core**	**iptables**
DNS	Kyllä	Ei	Kyllä	-
DHCP-palvelin	Kyllä	Ei	Kyllä	-
DHCP-välityspalvelin	Kyllä	Kyllä	Kyllä	Kyllä
WWW-välimuisti/-välityspalvelin	Lisäosien kautta	Sopivan ohjelman voi asentaa	Kyllä, sisäänrakennettu	Sopivan ohjelman voi asentaa

3.2.3 Palomuuritoiminnot

Varsinaisia palomuuritoimintoja on vertailtu taulukossa 4. Yleisesti ottaen, jokainen testatuista muureista on käyttökelpoinen melkeinpä mihin tahansa asiaan, eikä suuria eroja toiminnallisuudessa ole. Valinta tapahtuukin lähinnä oman osaamisen mukaan.

Taulukko 4. Palomuuritoiminnot

	pfSense	ShoreWall	Vyatta Core	iptables
IPv6-tuki	Ei kirjoitushetkellä, mutta tulossa	Kyllä	Kyllä	Kyllä
Vyöhykepohjainen palomuuri	Ei	Kyllä	Kyllä	Ei suoraan
Tilallinen/ tilaton	Vain tilallinen	Vain tilallinen	Tilaton ja tilallinen	Tilaton ja tilallinen
MAC-suodatus	Ei	Kyllä	Kyllä	Kyllä
IP-suodatus	Kyllä	Kyllä	Kyllä	Kyllä
Porttisuodatus	Kyllä	Kyllä	Kyllä	Kyllä
Protokollasuodatus	Kyllä	Kyllä	Kyllä	Kyllä
Tason 7 tuki	Osittain lisäosien kautta	Ei	Ei	Ei

3.3 Testausmenetelmät

Palomuureja testattiin luomalla niille omia sisäisiä verkkoja ja mallintamalla niissä joitain oikeissa verkoissa esiintyviä tilanteita ja käyttökohteita. Koska eri palomuurituotteet on suunniteltu eri käyttökohteita varten, ei niitä kaikkia testattu jokaisessa mahdollisessa tilanteessa. Esimerkiksi pfSense on suunniteltu käytettäväksi lähinnä sisäverkon ja ulkoverkon rajalla, mutta Vyatta Core ja ShoreWall toimivat missä tahansa kohtaa.

Testejä varten suunniteltiin kaksi erilaista skenaariota. Ne valittiin lähinnä käytännöllisyyden mukaan. Esimerkiksi pfSenseä testattiin juuri ulkoverkon ja sisäverkon rajalla. Kaikki skenaariot toteutettiin pelkästään IPv4-osoitteita käyttäen, sillä IPv6-tuki oli osittain puutteellinen kirjoitushetkellä. Osoitetapojen välillä ei kuitenkaan ole niin isoja eroja, etteikö työssä tehtyjä skenaarioita voisi soveltaa IPv6-pohjaisissakin verkoissa.

Testatut skenaariot olivat:

1. Kuvitteellisen Laatikko Oy:n DMZ-alue, yrityksen sisäverkko ja yritysten työntekijöiden pääsy tarvittaessa julkiseen internetiin. Tämä toteutettiin sekä yhdellä että kahdella peräkkäisellä palomuurilla.

2. Kuvitteellisen virtualisoidun oppimisverkon alueiden ja kohteiden suojaus sekä liikenteen rajoitus.

Lisäksi ekstrana tehtiin nopeustesti (luku 6, sivulla 76): puhdas ohjelmapalomuuri ei ole koskaan yhtä nopea kuin laitteistopalomuuri, mutta miten paljon ne oikein hidastavat verkkoa? Testissä mitattiin tiedostojen siirtonopeuksia ja -aikoja. Testi ei ollut erityisen tieteellinen eikä mittaus ollut perusteellinen, mutta se antoi hieman suuntaa.

Jollei erikseen ole mainittu, kaikki toteutetut skenaariot ja testit tehtiin täysin virtualisoituna VirtualBox-ohjelman avulla, eikä fyysisiä laitteita käytetty. Kaikkien laitteiden täydelliset konfiguraatiot löytyvät liitteestä 2 (sivulta 105 alkaen).

4 Skenaario 1: Laatikko Oy:n DMZ ja sisäverkko

4.1 Johdanto

Ensimmäinen skenaario oli kuvitteellisen suomalaisen pikkuyrityksen sisäverkko, DMZ-alueella oleva internet-palvelin (sisältö nähtävissä liitteessä 4, sivulla 148) ja näiden suojaus ulkomaailmaa vastaan. Verkon yleinen rakenne osoitteineen selviää kuviosta 1 (sivulla 26).

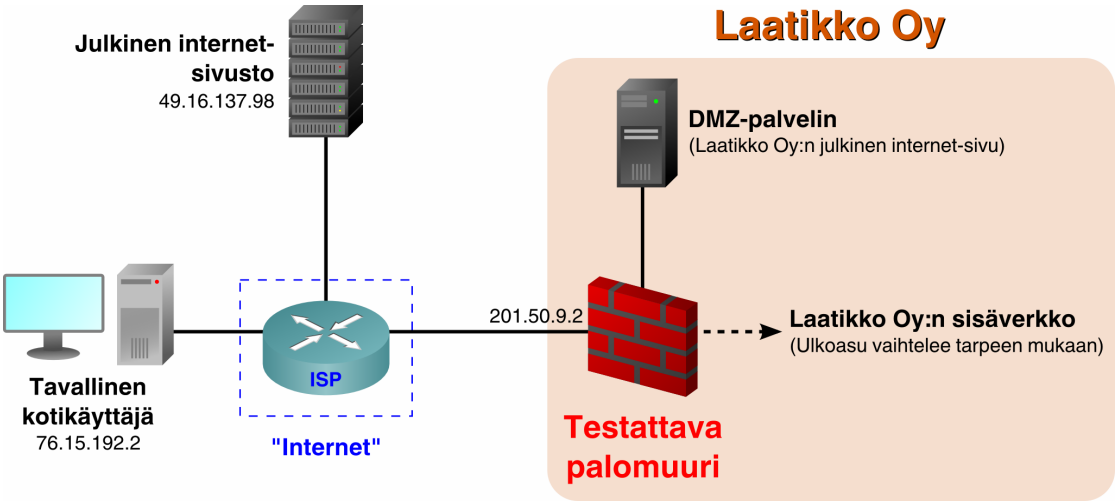
DMZ-alueen palveluksi valittiin kotisivut, koska se oli suoraviivaisin toteuttaa ja testata. Sähköpostipalvelimen rakentaminen olisi vaatinut huomattavasti enemmän työtä. Sähköposti olisi saattanut olla realistisempi vaihtoehto, sillä harva yritys pitää kotisivujaan omilla palvelimillaan. Mutta tämä ei ollut testin kannalta oleellista; DMZ-alueelle piti vain saada jokin palvelu.

Verkon ”runko” koostui yhdestä internet-palveluntarjoajaa simuloivasta Vyatta Core-reitittimestä. Tämä runkoreititin luotiin liitteen 2 (sivulla 100) mukaan ja sen ajonaikainen konfiguraatio löytyy myös liitteestä 2 (sivulla 105).

Runkoreitittimeen kytkettiin yksi Debian 7-pohjainen ”julkinen” internet-palvelin, johon asennettiin Apache-palvelinohjelmisto; sekä yksi Windows XP-kone, joka simuloi palveluntarjoajaan kytketyn tavallisen kotikäyttäjän konetta. Skenaariossa oletettiin, ettei Laatikko Oy ollut onnistunut ostamaan palveluntarjoajalta kuin yhden julkisen IP-osoitteen. Siksi kaikki liikenne, myös DMZ-alueelle menevä, käyttivät vain yhtä osoitetta. Skenaariossa ei myöskään käytetty DNS- eikä sähköpostipalvelimia. Niihin liittyvä liikenne kuitenkin sallittiin.

Yrityksen sisäinen verkko koostui itse testattavasta palomuurista, DMZ-palvelimesta, yhdestä työasemasta ja yhdestä sisäisestä palvelimesta. Koska DMZ-aluetta käytettäessä voi sisäverkon suojauksen rakentaa sekä yhdellä että kahdella palomuurilla, toteutettiin kumpikin tapaus erikseen. Sekä yhden että kahden palo-

muurin verkkojen rakenne selviää kuvioista 2 (sivulla 27) ja 3 (sivulla 27). Syy kahden palomuurin käyttämiseen on yksinkertaisesti tietoturva: yhden palomuurin tapauksessa laite muodostaa yhden vikaantumispisteen ja yhden tunkeutumispisteen. Kahden palomuurin käyttö parantaa tietoturvaa ja luotettavuutta. Uloin palomuuri rajoittaa DMZ-alueen liikennettä ja sisempi suojelee sisäverkkoa. Näin saatiin aikaan usean kerroksen suojaus.



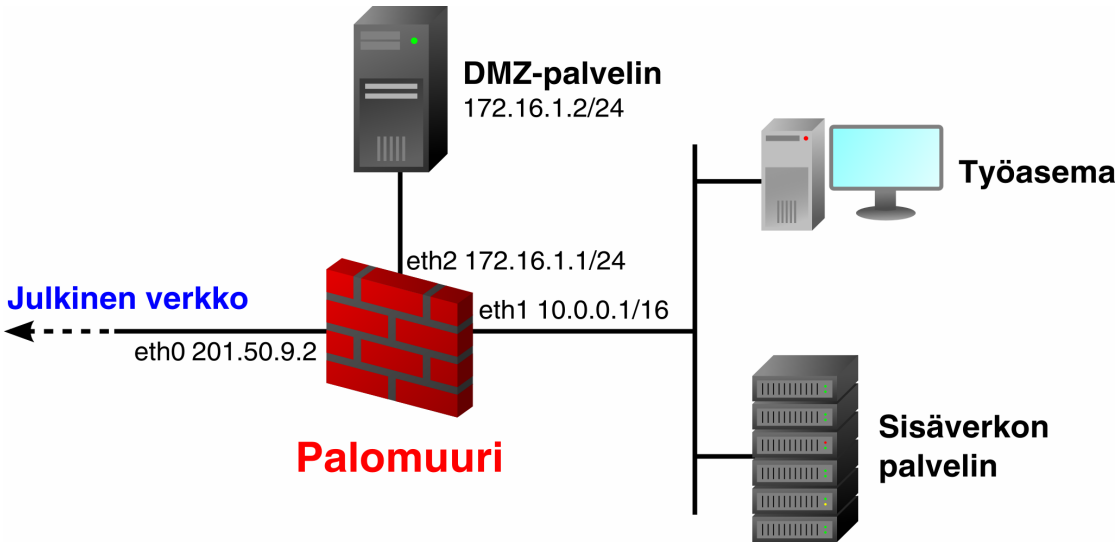
Kuvio 1. Skenaarion 1 verkko

Kuten jo aiemmin on todettu, pfSense on suunniteltu käytettäväksi lähinnä sisäverkon ja ulkoverkon rajalla. Siksi sitä ei käytetty kertaakaan sisempänä palomuurina, vaan tähän tarkoitukseen valittiin joka kerralla Vyatta Core.

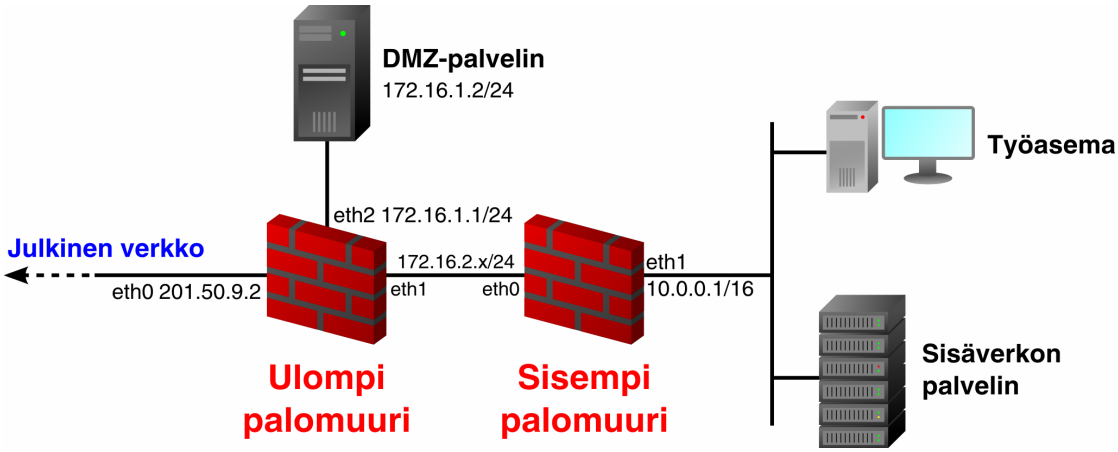
Yhden palomuurin tapauksessa sama laite hoitaa palomuurin lisäksi reitityksen ja toimii DHCP-palvelimena sisäverkolle.

Kahden palomuurin tapauksessa uloin laite hoitaa reititystä ulkoverkon, sisäverkon ja DMZ-alueen välillä; näiden lisäksi se tekee NAT-muunnokset eri verkkojen välillä. Sisempi palomuuri toimii DHCP-palvelimena ja voi sisältää lisäsääntöjä liikenteen suodattamiselle. On huomattava, että ulompi palomuuri ei tiedä sisemmän palomuurin takana olevan varsinaisen sisäverkon olemassaoloa. Siksi ulompaan palomuuriin on luotava jollain tapaa reittitieto, jotta laite osaa reitittää sisä-

verkkoon. Vastaavasti sisemmälle laitteelle on kerrottava koko ulkomaailman ja DMZ-alueen olemassaolosta. Tämä tehdään luomalla oletusreitti kohti ulompaa palomuuria.



Kuvio 2. Laatikko Oy:n sisäverkko yhdellä palomuurilla



Kuvio 3. Laatikko Oy:n sisäverkko kahdella palomuurilla

Kahden palomuurin välille jäävä yhteys on tavallaan ”ei kenenkään maa”, sillä sinne ei voida kytkeä mitään verkkolaitteita. Se olisi otollinen paikka esimerkiksi tunkeutumisentunnistus- ja estojärjestelmälle. Tätä mahdollisuutta ei kuitenkaan tässä työssä hyödynnetty.

Huomautus komentolistauksista

Seuraavat luvut sisältävät useita komentolistauksia. Koska annettavat komennot ovat toisinaan hyvinkin pitkiä, ne eivät aina mahdu yhdelle riville. Siksi niitä on pitänyt jatkaa seuraavalla rivillä. Rivit, jotka jatkuvat seuraavalla rivillä on merkitty taaksepäin olevalla kenoviivalla \. Kenoviiva vain merkitsee täsmälleen sen kohdan, josta rivi on katkaistu kahtia, joten sitä ei pidä koskaan kirjoittaa itse komentoriville. Yhdessäkään tässä työssä käytetyssä komennossa ei esiinny taaksepäin olevaa kenoviivaa, joten sekaannuksen vaaraa ei ole.

4.2 Sisempi palomuuri

Kahden palomuurin toteutuksissa sisempi muuri oli joka kerralla sama Vyatta Core-reititin. Sen olisi voinut toteuttaa myös muillakin, mutta Vyatta Coreen päädyttiin sen suoraviivaisen konfiguroitavuuden vuoksi. Tässä luvussa käydään läpi, miten se konfiguroitiin. Reititin asennettiin liitteen 1 (sivulla 100) ohjeiden mukaisesti ja sen täydellinen konfiguraatio löytyy liitteestä 2 (sivulta 107).

Sisempään palomuuriin ei kuitenkaan, ehkä hieman harhaanjohtavasti, konfiguroitu mitään palomuuriin liittyvää. Tämä siksi, että sisempi muuri haluttiin pitää esimerkin vuoksi mahdollisimman yksinkertaisena. Siihen voisi tarvittaessa konfiguroida joko erillisiin palomuurisääntöihin perustuvan palomuurin, tai vyöhykepohjaisen palomuurin. Kaikki riippuu tarpeesta. Sisempi muuri asetettiin tekemään ainoastaan reititystä ja toimimaan sisäverkon DHCP-palvelimena.

Aluksi tuli asettaa rajapintojen IP-osoitteet kuvion 3 (sivulla 27) mukaisesti. Tämä tapahtui konfiguraatiotilassa seuraavilla komennoilla: (LAN Interfaces 2013.)

```
set interfaces ethernet eth0 address 172.16.2.2/24
set interfaces ethernet eth1 address 10.0.0.1/16
```

Jotta sisempi palomuuri pystyisi reitittämään ulkoverkkoon päin, asetettiin sen oletusyhdyskäytävä osoittamaan kohti ulompaa palomuuria: (Basic System Configuration: Default Gateway 2013.)

```
set system gateway-address 172.16.2.1
```

Tämän jälkeen voitiin luoda sisäverkon DHCP-palvelin. Tarvittavat komennot olivat lähes samat kuin myöhemmin luvussa 4.5.1 (sivulla 46) luotavan DHCP-palvelimen vaatimat komennot (samaisessa luvussa on selitetty tarkemmin DHCP-palvelimen luonti): (Services: Configuring DHCP Address Pools 2013.)

```
set service dhcp-server
set service dhcp-server shared-network-name InternalPool \
subnet 10.0.0.0/16 default-router 10.0.0.1
set service dhcp-server shared-network-name InternalPool \
subnet 10.0.0.0/16 start 10.0.0.10 stop 10.0.255.254
```

Asetusten tallentamisen jälkeen sisempi palomuuri oli valmis käytettäväksi.

4.3 pfSense

pfSense asennettiin liitteen 1 (sivulla 94) mukaisesti. Laitteen rajapinnat konfiguroitiin jo asennusvaiheessa seuraavasti:

- em0: WAN (ulkoverkko), osoite 201.50.9.2/30
- em1: LAN (sisäverkko), osoite 10.0.0.1/16 tai 172.16.2.1/24
- em2: OPT1 (DMZ), osoite 172.16.1.1/24.

4.3.1 Yksi palomuuri

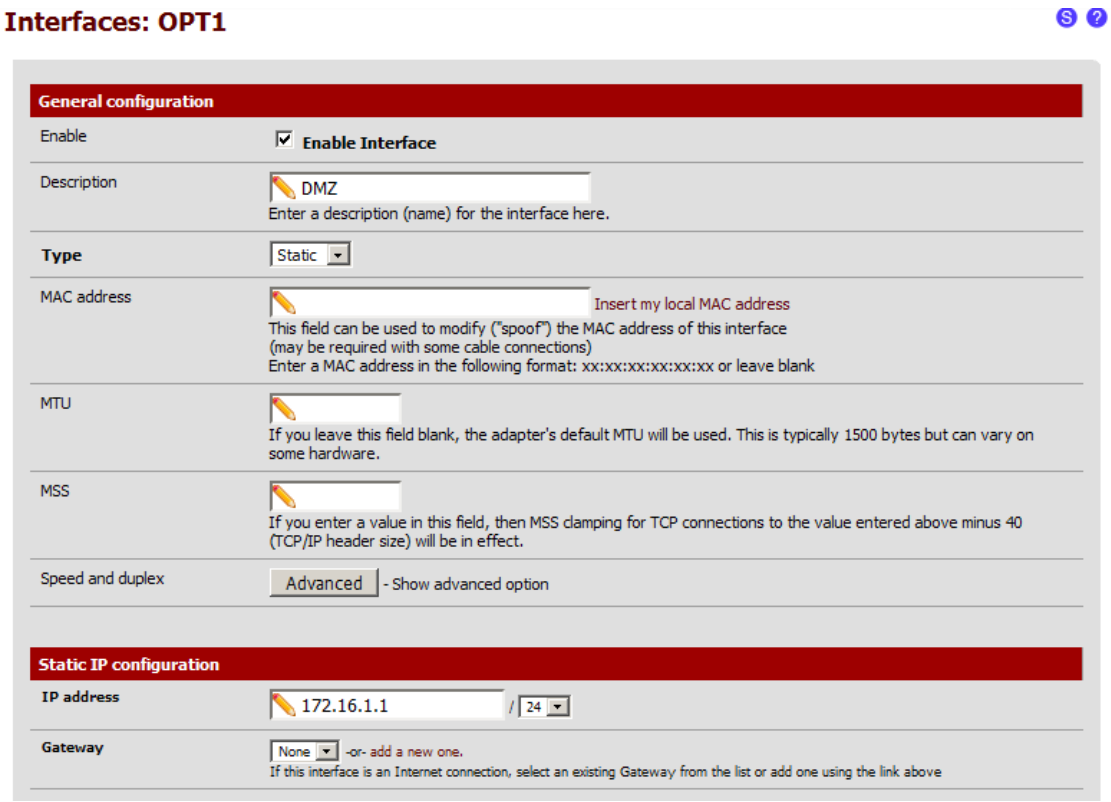
pfSense on lähes sellaisenaan täysin valmis yhden palomuurin (kuvio 2 sivulla 27) toteutusta varten. Ainoat asennuksen jälkeen tarvittavat asiat ovat DMZ-portin

avaus ja konfigurointi (jollei sitä konfiguroitu jo asennusvaiheessa), sekä NAT-muunnoksen luonti DMZ-alueen liikenteelle.

DMZ-rajapinnan avaus

DMZ-rajapinta konfiguroitiin valikon *Interfaces/OPT1* kautta. Rajapinta tuli kääntää päälle ruksaamalla *Enable Interface* -kohta. Vasta tämän jälkeen rajapinta voitiin konfiguroida.

Jotta rajapinnalle pystyttiin antamaan oikea IP-osoite, tuli osoitteen tyyppi vaihtaa *Type* -valikosta kohtaan *Static*. Tämän jälkeen kenttään *Static IP configuration/IP address* voitiin kirjoittaa haluttu osoite 172.16.1.1/24. Tämä näkyy myös kuviossa 4. Kun muutokset oli tehty, ne tallennettiin painamalla alareunasta *Save* ja sen jälkeen *Apply changes*. (Interface Settings 2012.)



Kuvio 4. DMZ-alueen rajapinnan avaus ja konfigurointi

NAT-muunnos DMZ-alueen liikenteelle

Jotta ulkoverkosta saisi DMZ-alueen palvelimeen yhteyden, täytyy palomuurissa tehdä NAT-muunnos saapuville ja lähteville paketeille. Palomuuri tunnistaa DMZ-alueelle tarkoitetun paketin ja muuttaa sen kohdeosoitteen DNAT-muunnoksella DMZ-alueen osoitteeksi. Vastaavasti paluupaketin osoite muutetaan SNAT-muunnoksella takaisin julkiseksi osoitteeksi.

pfSense tallentaa nämä muunnokset portin uudelleenohjauksen alle, eikä erillisiä SNAT- ja DNAT-sääntöjä tarvitse tehdä. Tarvittava portti (eli 80, HTTP) uudelleenohjataan ja pfSense luo itse tarvittavat NAT-säännöt.

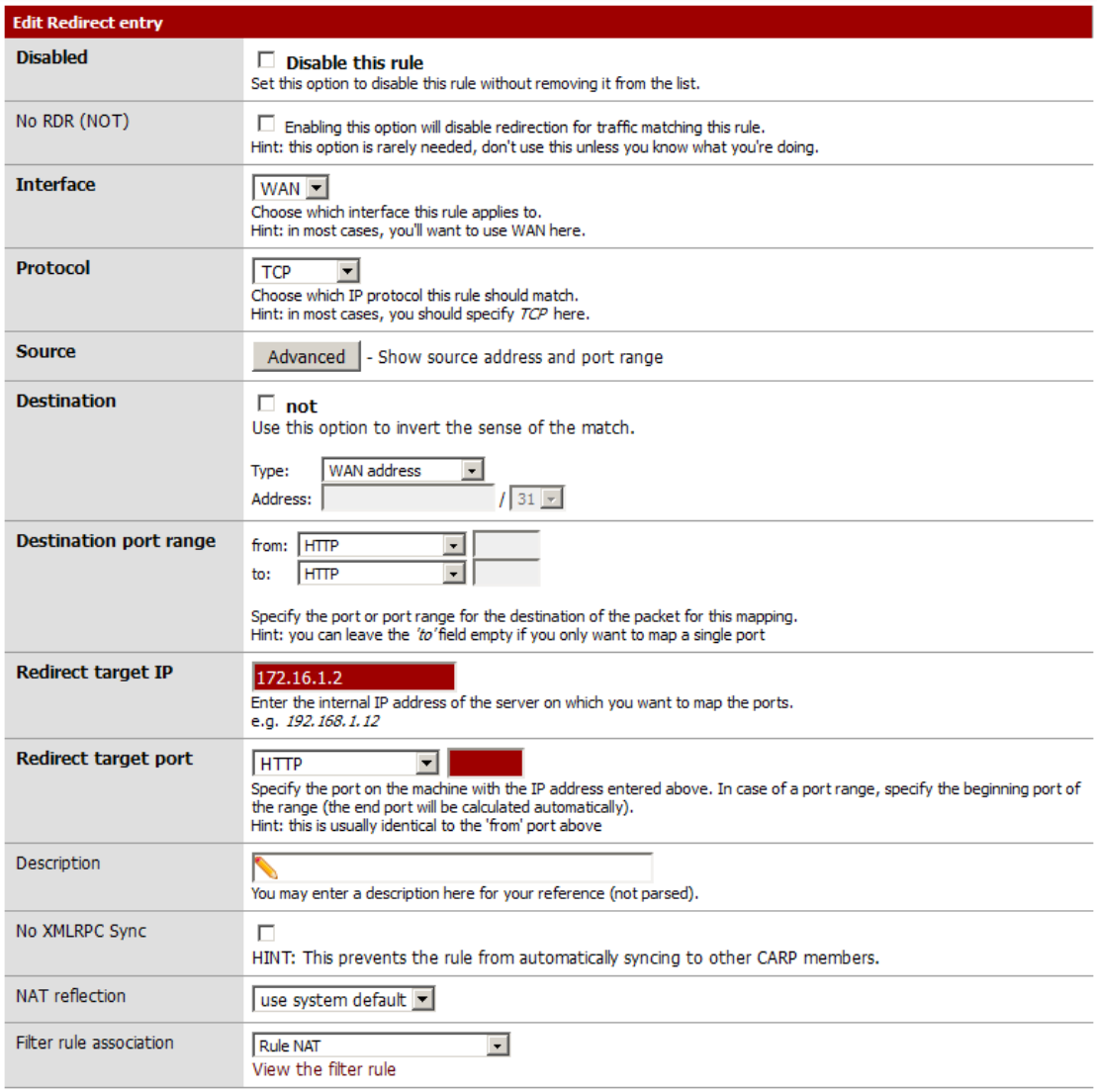
Kokeellisesti havaittiin, että pfSense estää jo oletusasetuksilla DMZ-alueelta lähtöisin olevan liikenteen, oli sen kohde mikä tahansa. Se kuitenkin sallii DMZ-alueelta tulevat vastaukset pyyntöihin, jotka ovat alkujaan lähtöisin DMZ-alueen ulkopuolelta. Tämä on juuri kuten pitääkin, joten muita sääntöjä ei tarvitse luoda.

Uudelleenohjaus luotiin valikon *Firewall/NAT* ja sen välilehden *Port Forward* kautta. Sivun oikeassa reunassa olevaa plus-merkkiä painamalla avautui kuvion 5 (sivulla 32) mukainen ikkuna. Se täytettiin seuraavasti: (How can I forward ports with pfSense? 2011.)

- **Interface**: WAN
- **Protocol**: TCP
- **Destination**: WAN address
- **Destination port range**: HTTP – HTTP
- **Redirect target** IP: 172.16.1.2
- **Redirect target port**: HTTP.

Nämä tarkoittavat, että WAN-rajapinnasta porttiin 80 (HTTP) tulevan TCP-paketin kohdeosoite muutetaan DMZ-alueen palvelimen osoitteeksi, mutta portti pysyy ennallaan.

Asetukset tallennettiin painamalla *Save* ja *Apply Changes* -painikkeita. Yhden palomuurin toteutus pfSensellä oli valmis.



Kuvio 5. NAT-sääntö DMZ-alueelle menevälle HTTP-liikenteelle

4.3.2 Kaksi palomuuria

Sisempänä palomuurina käytettiin luvussa 4.2 (sivulla 28) luotua Vyatta Corea. Se kytkettiin kiinni pfSenseen ja pfSense konfiguroitiin uudelleen.

Rajapinnan IP-osoite

Ensin sisäverkkoon osoittavan rajapinnan IP-osoite tuli vaihtaa. Se tapahtui valikon *Interfaces/LAN* kautta. Osoite muutettiin vanhasta 10.0.0.1/16 -osoitteesta osoitteeksi 172.16.2.1/24. Tässä vaiheessa täytyi konfigurointiin käytettävän koneen IP-osoite vaihtaa aliverkkoon 172.16.2.0/24. Muuten selainpohjaiseen konfiguraatio-ohjelmaan ei enää saanut yhteyttä asetusten tallentamisen jälkeen. Muutos oli kuitenkin vain hetkellinen.

Kiinteä reitti sisäverkkoon

Jotta ulompi palomuuri tietäisi sisemmän muurin takaa löytyvästä verkosta, on sille kerrottava sen olemassaolosta jollain tavalla. Yleensä reitittimissä tämä tapahtuu kiinteällä reitillä. pfSense ei kuitenkaan anna luoda kiinteitä reittejä suoraan. Kiinteitä reittejä luotaessa tarvittavaa **seuraavan hypyn** osoitetta (engl. *next-hop*) ei voi määrittää suoraan osoitteena, vaan ainoastaan **yhdyskäytävien** (engl. *gateway*) kautta. Yhdyskäytäviä voi luoda helposti, mutta jostain syystä ohjelma antaa luoda yhdyskäytäviä vain verkkoihin, joiden osoite on jo jossain pfSensen tuntemassa rajapinnassa. Aivan satunnaisiin verkkoihin osoittavia reittejä ei siis voi luoda (ainakaan ilman toisten reitittimien apua).

Tarvittava yhdyskäytävä luotiin valikon *System/Routing* takaa löytyvän *Gateways*-välilehden kautta. Sivun oikean reunan plus-napista aukeaa kuvion 6 (sivulla 34) mukainen ikkuna. Se täytettiin seuraavasti: (Gateway Settings 2011.)

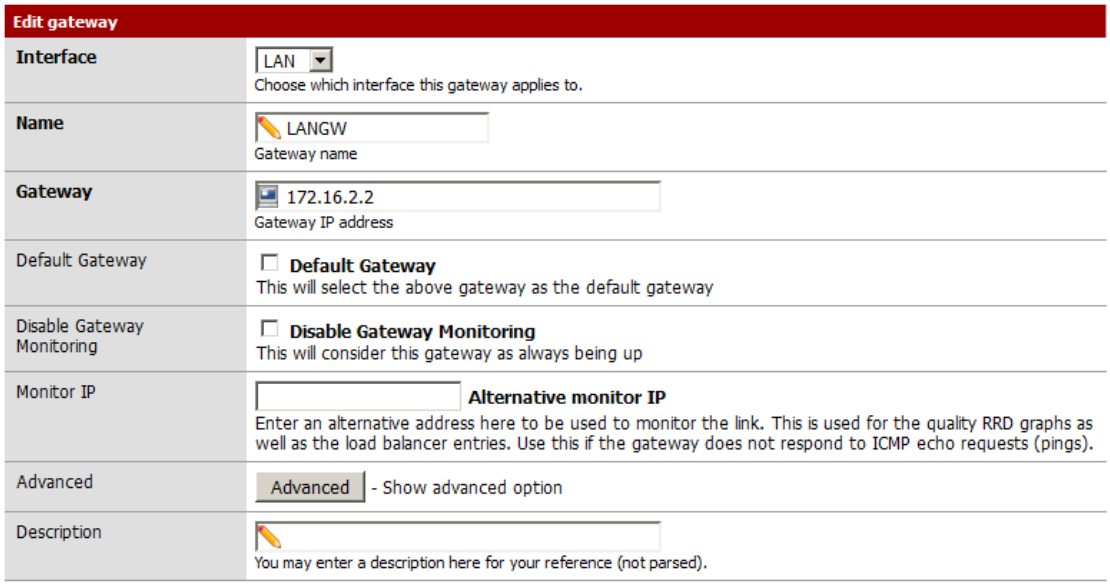
- **Interface**: LAN
- **Name**: yhdyskäytävän nimi, esimerkiksi ”LANGW”
- **Gateway**: sisemmän palomuurin osoite, tässä kuvion 3 (sivulla 27) mukaan 172.16.2.2.
- Ruksi pois kohdasta *Default Gateway*. (Oletusyhdyskäytävä on jo olemassa, eikä tämä ole se, joten ruksi on otettava pois.)

Yhdyskäytävän tallennuksen jälkeen voitiin luoda itse kiinteä reitti. Se tapahtui välilehden *Routes* kautta. Jälleen kerran oikean reunan plus-merkistä päästiin luo-

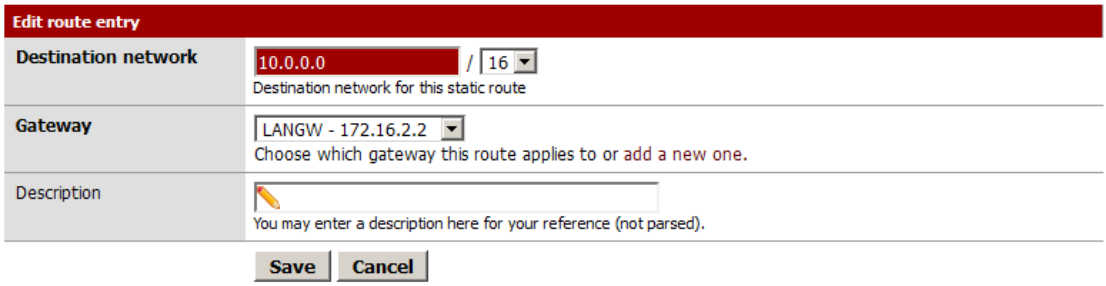
maan uutta reitti (kuvio 7). Ikkunan tiedot täytettiin seuraavasti: (Static Routes 2009.)

- **Destination Network**: 10.0.0.0/16 (kohdeverkon osoite)
- **Gateway**: ”LANGW - 172.16.2.2” (seuraavan hypyn yhdyskäytävä)

Tämä tarkoittaa, että kohdeverkko 10.0.0.0/16 löytyy yhdyskäytävän LANGW takaa, eli verkkoon 10.0.0.0/16 mentäessä on paketti ensin ohjattava osoitteeseen 172.16.2.2 (jossa odottaakin sisempi palomuuri).



Kuvio 6. Uuden yhdyskäytävän luonti



Kuvio 7. Kiinteän reitin luonti sisäverkkoon päin

DHCP-palvelimen sammutus

Ulompi palomuuri ei enää tarvinnut DHCP-palvelinta, joten se käännettiin pois päältä. Tämä tehtiin valikon *Services/DHCP server* kautta. Sen LAN-välilehdeltä otettiin ruksi pois kohdasta *"Enable DHCP server on LAN interface*" ja asetukset tallennettiin.

Sisäverkon sallinta palomuurissa

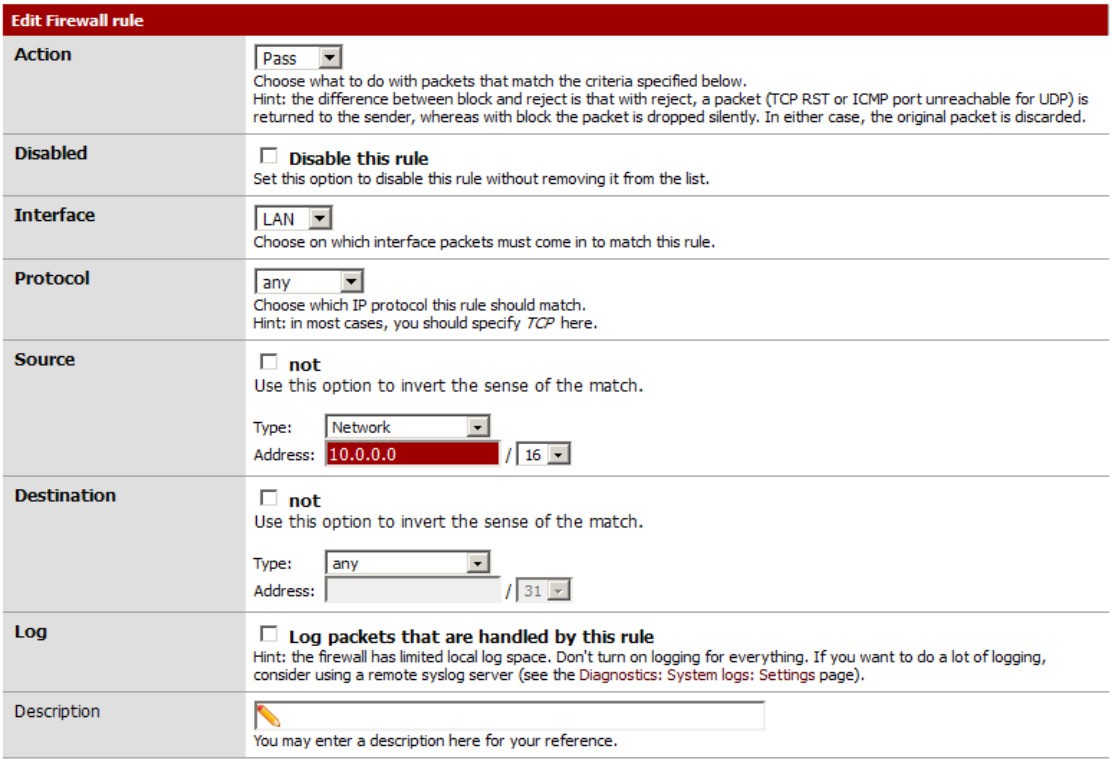
pfSense sallii oletuksena LAN-rajapinnan takaa tulevan liikenteen, mutta koska sisäverkon osoite oli 10.0.0.0/16, ja LAN-rajapinnassa oli osoite 172.16.2.0/24, pysäytti se oletuksena sisäverkon liikenteen. Siksi ulompaan palomuuriin oli luotava sääntö, joka salli sisäverkon liikenteen.

Sääntö luotiin valikon *Firewall/Rules* kautta, LAN-välilehdeltä. Plus-merkin painamisen jälkeen avautunut sivu (kuvio 8 sivulla 36) täytettiin seuraavasti: (Firewall Rule Basics 2012.)

- **Action:** pass
- **Interface:** LAN
- **Protocol**: joko TCP tai "any", riippuen siitä haluaako päästää esimerkiksi ping-viestit läpi. Tämä on mietittävä tapauskohtaisesti. Ei vaikuta sisäverkon protokolliin millään tavalla.
- **Type:** Network
- **Source**: 10.0.0.0/16.

Tämä sääntö salli verkosta 10.0.0.0/16 LAN-rajapintaan tulevat paketit, oli niiden kohde mikä tahansa.

Asetusten tallennuksen jälkeen oli kahden palomuurin toteutus pfSensellä valmis.



Kuvio 8. Sisäverkon liikenteen salliminen uloimmassa palomuurissa

4.4 ShoreWall

ShoreWall asennettiin liitteen 2 (sivulla 99) ohjeiden mukaan virtuaalikoneeseen. Rajapinnat konfiguroitiin tiedostoon `/etc/network/interfaces` seuraavasti (tiedosto löytyy täydellisenä liitteestä 2, sivulta 110):

- eth0: osoite 201.50.9.2/30, oletusyhdyskäytävä 201.50.9.1
- eth1: osoite 10.0.0.1/16 tai 172.16.2.1/24
- eth2: 172.16.1.1/24.

ShoreWallin konfiguraatiotiedostot ovat puhtaita tekstitiedostoja. Niiden syntaksi on melko yksinkertainen:

- Risuaita aloittaa aina kommentin, oli se missä kohtaa tahansa.
- Joka rivillä määritetään yksi asia ja jokainen rivi jakaantuu sarakkeisiin, joiden tarkoitus ja järjestys riippuu tiedostosta. Yleensä tiedoston alussa on lueteltu sarakkeet ja niiden tarkoitus.

Tiedostoja kirjoitettaessa on huomioitava, että ensimmäinen täsmäävä sääntö määrittää mitä paketille tapahtuu. Sääntöjen järjestyksestä on siis pidettävä tarkasti kirjaa.

ShoreWall on käyttövalmis heti asennuksen jälkeen, mutta se ei tee mitään (eikä edes käynnisty), ennen kuin se on konfiguroitu. Ohjelma lukee konfiguraatiotiedostot hakemistosta `/etc/shorewall`, mutta jakelusta ja asennustavasta riippuen hakemisto saattaa olla tyhjä. Tämä ei ole ongelma: ShoreWallin mukana tulee useita toimiva esimerkkejä. Käytetyssä Debian-pohjaisessa koneessa valmiit esimerkit löytyivät hakemistosta `/usr/share/doc/shorewall`. Sen alta löytyi hakemisto `examples/three-interfaces/`, jonka sisältöä on käytetty mallina tässä työssä. On huomattava, ettei kaikkia ShoreWallin tukemia tiedostoja tarvita jokaisessa tilanteessa. Seuraavissa aliluvuissa onkin käyty läpi vain tarvittavien tiedostojen sisältö. Muut tiedostot voi jättää huoletta pois.

4.4.1 Yksi palomuuri

zones

Tiedostossa `zones` määritetään itse vyöhykkeet. Ensimmäinen sarake sisältää vyöhykkeen nimen ja toinen sen tyypin. Valinnainen kolmas sarake voi sisältää muita parametreja. Usein käytettyjä tyyppejä ovat seuraavat:

- **firewall**: palomuuri itsessään
- **ipv4**: IPv4-pohjainen verkko
- **ipv6**: IPv6-pohjainen verkko. (shorewall-zones n.d.)

Kuvion 2 (sivulla 27) kaltaisen verkon vaatimat vyöhykkeet määritettiin seuraavasti:

```
# NIMI TYYPPI
fw firewall
net ipv4
loc ipv4
dmz ipv4
```

Tämä luo vyöhykkeet ”fw” itse laitteelle ja IPv4-vyöhykkeet ”net”, ”loc” ja ”dmz”.

interfaces

Tiedosto `interfaces` liittää rajapinnat eri vyöhykkeisiin. Tiedosto sisältää kolme saraketta: vyöhykkeen nimi, siihen kuuluva rajapinta (tai rajapinnat), ja mahdolliset parametrit pilkuilla erotettuina. Tiedoston alussa on asetettava versionumero ”`FORMAT 2`” -määreellä. Työssä käytetty tiedosto näytti tältä:

```
FORMAT 2
# VYÖHYKE RAJAPINTA PARAMETRIT
net eth0 tcpflags,nosmurfs,logmartians
loc eth1 tcpflags,nosmurfs,dhcp
dmz eth2 tcpflags,nosmurfs
```

Tämä liitti yhden rajapinnan jokaiseen luotuun vyöhykkeeseen: eth0-rajapinta kuului ”net”-vyöhykkeeseen, eli ulkomaailmaan; eth1 kuului ”loc”-vyöhykkeeseen, eli sisäverkkoon; ja eth2-rajapinta kuului DMZ-vyöhykkeeseen.

Rivien loppuun tulevia parametreja on vaikka kuinka paljon, mutta tässä työssä käytetyt olivat:

- **dhcp**: Rajapinta saa osoitteen DHCP:llä, tai sen takaa löytyy verkko jossa käytetään DHCP:tä. Sallii automaattisesti DHCP-liikenteen (katso myöhemmin `rules`-tiedoston kohdalla oleva maininta tästä). Ei käytetty julkisen verkon rajapinnassa, koska sen osoite oli kiinteä.

- **logmartians**: Kirjaa lokiin tiedon paketeista joiden lähdeosoite on sellainen, jollaista ei voi esiintyä julkisessa verkossa. Ei pidä käyttää rajapinnassa jossa on yksityinen osoite.
- **nosmurfs**: suodattaa pois paketit joiden lähdeosoite on verkon levitysosoite. Suositellaan pidettäväksi päällä.
- **tcpflags**: Pakottaa rajapintaan saapuvien TCP-pakettien lippujen tarkistuksen. Lipuista on olemassa tiettyjä yhdistelmiä, joita ei saisi esiintyä. Suositellaan käytettäväksi julkiseen verkkoon kytketyissä rajapinnoissa. (shorewall-interfaces n.d.)

masq

Tiedosto `masq` määrittää NAT-muunnokset. Sarakkeet määrittävät lähtörajapinnan ja pilkuilla erotellun listan aliverkoista, joista tähän rajapintaan tulevat osoitteet muunnetaan. (shorewall-masq n.d.) Työssä käytettiin seuraavanlaista tiedostoa:

```
# RAJAPINTA VERKOT
eth0 10.0.0.0/16,172.16.1.0/24
```

Rivi kertoo, että aliverkoista 10.0.0.0/16 (sisäverkko) ja 172.16.1.0/24 (DMZ) tulevien pakettien osoite on ajettava NAT-muunnoksen läpi niiden lähtiessä eth0-rajapinnasta ulkoverkkoon. Jotta muunnos toimisi, on NAT vielä erikseen kytkettävä päälle. Tämä on selitetty luvussa ”*Muut konfiguroitavat asiat*”, sivulla 43.

policy

Tämä tiedosto määrittää korkean tason säännöt vyöhykkeiden väliselle liikenteelle. Tiedostossa määritettyjä sääntöjä käytetään silloin, jos `rules`-tiedostosta (katso seuraava aliluku) ei löydy yhtään sopivaa sääntöä.

Tiedoston sarakkeet määrittävät lähde- ja kohdevyöhykkeet, toiminnon näiden väliselle liikenteelle ja joukon parametreja. Tiedostossa on käytössä vyöhykkeen nimenä erikoistapaus ”all”, joka tarkoittaa kaikkia vyöhykkeitä. Toimintoja on useita, mutta hyödyllisimmät ovat:

- DROP (”pudota”): paketti tuhotaan ilmoittamatta siitä lähettäjälle
- REJECT (”hylkää”): paketti pudotetaan, mutta lähettäjä saa tästä tiedon
- ACCEPT (”salli”): paketti sallitaan ja päästetään kohteeseensa. (shorewall-policy n.d.)

Työssä käytettiin seuraavanlaista tiedostoa. Se estää kaiken liikenteen vyöhykkeiden välillä:

```
# LÄHDE KOHDE TOIMINTO
# estä ulkoverkosta tuleva liikenne kokonaan
net all DROP

# estä DMZ-vyöhykkeeltä tuleva liikenne kokonaan
dmz all DROP

# estä sisäverkosta tuleva liikenne, mutta ilmoita siitä
loc all REJECT

# estä palomuurista lähtevä liikenne, mutta ilmoita siitä
fw all REJECT

# estä kaikki muu liikenne kokonaan, ilmoita estosta
all all REJECT
```

rules

Tämä tiedosto määrittää täsmälliset säännöt vyöhykkeiden väliselle alkavalle liikenteelle. Se on usein isoin tiedosto, jonka ShoreWallia varten joutuu kirjoittamaan. ShoreWall rajoittaa ainoastaan alkavaa liikennettä, eli se päästää jo alkaneen liikenteen vastauspaketit läpi.

Tiedosto jakaantuu useaan osioon, jotka on eroteltu toisistaan suuraakkosilla kirjoitetuilla otsikoilla. Otsikko alkaa sanalla ”SECTION” ja sen perään kirjoitetaan varsinainen osion nimi. Osioiden on oltava aina ennalta määrätyssä järjestyksessä, mutta ne voivat olla tyhjiä. Osioiden nimet (ja järjestys) ovat:

- ALL (”kaikki”): Tässä osiossa määriteltyjä sääntöjä sovelletaan aina, oli paketin tila mikä tahansa.
- ESTABLISHED (”luotu”): Säännöt, joita sovelletaan paketteihin, joiden liikennevirta on jo luotu.
- RELATED (”liittyvä”): Säännöt joita sovelletaan paketteihin, jotka liittyvät jo luotuihin liikennevirtoihin.
- NEW (”uusi”): Paketit jotka ovat luomassa uutta liikennevirtaa, tai ovat muuten vain läpikulkevia paketteja.

ShoreWallin ohje suosittelee, että aloittelevat käyttäjän sijoittaisivat kaikki säännöt NEW-osioon, eivätkä käyttäisi muita osioita lainkaan. Näin meneteltiinkin tässä työssä. Uudemmissa ShoreWallin versioissa on mukana joitain lisäosioita, kuten ”INVALID” ja ”UNTRACKED”; näitä osioita ei käytetty.

Varsinaiset sääntörivit määrittävät jollain tavalla lähteen, kohteen, protokollan, sekä toiminnon. ShoreWall kelpuuttaa useita eri syntakseja näiden määrittämiseen. Tässä käytettiin kahta eri syntaksia:

TOIMINTO lähdevyöhyke kohdevyöhyke protokolla [muut parametrit]
PROTOKOLLA(TOIMINTO) lähdevyöhyke kohdevyöhyke [muut parametrit]

Toiminnot ovat samat kuin aiemmin policy-tiedostossa käytetyt DROP, REJECT ja ACCEPT. Myös muitakin toimintoja on olemassa, kuten DNAT. (shorewall-rules n.d.)

Vaihe 1. Sallimme aluksi ping-komennon toiminnan, jotta voimme varmistaa yhteyksien toiminnan. Sallimme sisäverkosta ping-viestit palomuuriin, DMZ-alueelle ja ulkoverkkoon:

```
# sallitaan sisäverkosta itse palomuuriin tapahtuva ping
ACCEPT loc fw ICMP

# sallitaan sisäverkosta ping DMZ-alueelle
ACCEPT loc dmz ICMP
```

```
# sallitaan sisäverkosta ping ulkoverkkoon (vaatii NAT:in)
ACCEPT loc net ICMP

# sallitaan palomuurista ping joka paikkaan
ACCEPT fw loc ICMP
ACCEPT fw dmz ICMP
ACCEPT fw net ICMP
```

ICMP on ping-komennon käyttämä protokolla. ShoreWall antaa määrittää tätäkin tarkemmin minkä tyyppiset ICMP-paketit sallitaan läpi, mutta tässä sallimme ne kaikki. Asia on hieman kaksijakoinen: toisaalta voisimme hieman parantaa tietoturvaa rajoittamalla turhaa ICMP-liikennettä, mutta liian rajoitettu liikenne voi tehdä verkon vikatilojen selvittämisen hyvin vaikeaksi.

Vaihe 2. DHCP-liikennettä ei tarvitse erikseen sallia, koska `interfaces`-tiedostossa on sisäverkon rajapinnalle annettu parametri ”dhcp”. Tämä sallii DHCP-liikenteen automaattisesti. Ratkaisu toimi, koska työssä DHCP-palvelin sijaitsi samassa koneessa ShoreWallin kanssa; jos DHCP-palvelin on muualla, se olisi vastaavasti sallittava erikseen. Esimerkiksi seuraava komento sallisi DHCP-viestien menon sisäverkosta kuvitteelliseen ”palvelimet”-vyöhykkeeseen:

```
DHCP(ACCEPT) loc palvelimet
```

DNS on aina sallittava erikseen. Koska esimerkin Laatikko Oy:llä ei ollut omaa DNS-palvelinta, sallittiin palveluntarjoajalle lähteneet DNS-kyselyt seuraavasti:

```
DNS(ACCEPT) loc net
```

Sallimme vain sisäverkon DNS-kyselyt, emme DMZ-alueen, sillä tässä työssä DMZ-alueella ei ole koskaan mitään mikä tarvitsisi DNS:ää. Jos palomuuri itse tarvitsee DNS:ää (esimerkiksi päivityksen yhteydessä), sen voi sallia hetkellisesti komennolla:

```
DNS(ACCEPT) fw net
```

Vaihe 3. Seuraavaksi sallittiin normaali HTTP-liikenne. Ensin sallittiin sisäverkosta tulevat HTTP- ja HTTPS-pyynnöt DMZ-alueelle:

```
ACCEPT loc dmz TCP http,https
```

Samalla tavalla sallittiin yhteydet ulkoverkkoon:

```
ACCEPT loc net TCP http,https
```

Vaihe 4. Koska DMZ-alueella olevaan palvelimeen on tarkoitus päästä käsiksi ulkoapäin, sallittiin HTTP-liikenne ja määritettiin DNAT-muunnoksella mihin se tuli ohjata:

```
ACCEPT net dmz TCP http
DNAT net dmz:172.16.1.2 TCP http
```

Paluuliikenne DMZ-alueelta hoituu jo `masq`-tiedostossa määritetyn muunnoksen avulla, joten muuta ei tarvita DMZ-aluetta varten.

Työssä ei käytetty sähköpostia lainkaan, mutta sen salliminen kävisi komennolla (portit 25 ja 110 ovat sähköpostin SMTP- ja POP3-protokollien portit):

```
ACCEPT loc net TCP 25,110
```

Valmis työssä käytetty `rules`-tiedosto löytyy liitteestä 2 sivulta 111.

Muut konfiguroitavat asiat

Jotta NAT-muunnos sisäverkosta ulkoverkkoon olisi toiminut, ei pelkkä `masq`-tiedoston kirjoitus riittänyt. ShoreWallille oli vielä kerrottava sen pääkonfiguraatiotiedostossa, että pakettien uudelleenlähetys on otettava käyttöön. Tämä asetus tehtiin tiedostoon `shorewall.conf`, ja tarvittava rivi oli nimeltään ”`IP_FORWARDING`”. Se oli oletuksena asetettu arvoon ”`Keep`”, joka piti järjestelmän oletustilan käytössä. Koska oletustilassa uudelleenohjaus ei ollut käytössä, oli rivi muutettava seuraavanlaiseksi:

```
IP_FORWARDING=On
```

Lisäksi ShoreWall oli asetettava käynnistymään koneen käynnistyksen yhteydessä. Se mihin tämä asetus tehdään, vaihtelee hieman käytetyn Linux-jakelun mukaan, mutta työssä käytetyssä Debianissa tarvittava tiedosto oli `/etc/defaults/shorewall`. Sen alusta löytyvä asetus ”`startup=0`” vaihdettiin muotoon ”`startup=1`”.

Koska sisäverkko tarvitsi DHCP-palvelimen, asennettiin ShoreWall-koneeseen laajalti käytetty ISC DHCP Server -ohjelmisto. Se asentui komennolla:

```
apt-get install isc-dhcp-server
```

ISC DHCP Server -palvelimen tarvitsema konfiguraatiotiedosto `/etc/dhcp/dhcpd.conf` löytyy täydellisenä liitteestä 2 (sivulla 111).

ShoreWallin käynnistys

Kun ShoreWall on konfiguroitu, se voidaan käynnistää joko kokonaan, tai pelkästään uudelleen. ShoreWall sisältää komennon jolla voi tarkistaa konfiguraatiotiedostojen syntaksin oikeellisuuden. Komento on lyhykäisyydessään:

```
shorewall check
```

Jos kaikki tiedostot ovat kirjoitettu oikein, saa ilmoituksen ”*Shorewall configuration verified*”. Muutoin komento kertoo, mikä on kirjoitettu väärin. Kun tiedostot ovat valmiit, voi ShoreWallin käynnistää joko kokonaan komennolla:

```
shorewall start
```

Tai kokonaan uudelleen komennolla:

```
shorewall restart
```

Käynnistyksen jälkeen yhden palomuurin toteutus on valmis. Konfiguroinnin jälkeen ShoreWall käynnistyy koneen yhteydessä normaalisti, jos asennusmekanismi

lisäsi sen käynnistykseen. Ilman konfiguraatiota saa vain virheilmoituksen, eikä ShoreWall käynnisty.

Tässä luotu palomuuri on ping-komennon vaatimaa ICMP-protokollaa lukuun ottamatta erittäin tiukka. Se ei päästä läpi mitään muuta kuin normaalin internet-selaimen liikenteen. Muut mahdolliset protokollat on avattava erikseen.

4.4.2 Kaksi palomuuria

Kahden palomuurin toteutus on pitkälti samanlainen kuin yhden palomuurin. On suositeltavaa rakentaa ensin yksi palomuuri toimivaksi, ja vasta sitten lisätä toinen. Käytetty sisempi palomuuri oli luvussa 4.2 (sivulla 28) luotu Vyatta Core-reititin. Se kytkettiin sellaisenaan kiinni ShoreWall-koneeseen.

Kun ShoreWall on konfiguroitu yhden palomuurin toteutukselle, sen muuttaminen kahden palomuurin toteutukselle on helppoa:

1. ShoreWall-koneen eth1-rajapinnan IP-osoite muutettiin osoitteeksi 172.16-.2.1/24. Muokattu tiedosto `/etc/network/interfaces` löytyy myöskin täydellisenä liitteestä 2 (sivulla 110).

2. Uloimman palomuurin ei enää tarvitse toimia DHCP-palvelimena, koska sisempi muuri on käytössä. Jos DHCP-palvelinohjelmiston ehti jo asentaa, sen voi poistaa komennolla:

```
apt-get remove isc-dhcp-server
```

3. Ulommalle palomuurille tuli kertoa sisemmän palomuurin takaa löytyvästä sisäverkosta kiinteällä reitillä. Komentoriviltä reitin luova komento on:

```
route add -net 10.0.0.0 netmask 255.255.0.0 gw \
172.16.2.2 dev eth1
```

Tämä lisättiin (hieman muokattuna) kohdassa 1 mainittuun rajapintojen konfiguraatiotiedostoon, jolloin se säilyi koneen uudelleenkäynnistyksen yli.

Lopuksi uloin palomuurikone käynnistettiin uudelleen, jotta kaikki muutokset tulivat varmasti voimaan. Uudelleenkäynnistys ei ole pakollista, mutta sillä takaa sen, ettei mikään vanha asetus jää kummittelemaan mihinkään.

4.5 Vyatta Core

Vyatta Core asennettiin liitteen 1 (sivulla 100) ohjeiden mukaan.

Vyatta Core tukee useita eri tapoja rakentaa palomuuri. Palomuurisäännöt voi määritellä yksi kerrallaan joka rajapintaan liikenteen suunnan mukaan. Ne voi myös rakentaa vyöhykepohjaisesti. Vyöhykepohjainen tapa on tässä järkevämpi kuin suuntapohjainen, sillä oletuksena vyöhykkeet estävät kaiken liikenteen ja sallittava liikenne on erikseen kerrottava. Tämä on tietoturvan kannalta parempi. Perinteisiin palomuurisääntöihin perustuva palomuuri on käyty läpi luvussa 5.4 (sivulla 64).

4.5.1 Yksi palomuuri

Skenaariossa luotu palomuuri salli seuraavat asiat:

- HTTP-pyynnöt sisäverkosta DMZ-alueelle ja niiden takaisin tulevat vastaukset
- HTTP-pyynnöt ulkoverkosta DMZ-alueelle ja niiden takaisin tulevat vastaukset
- sisäverkosta ulkoverkkoon tapahtuva HTTP-liikenne ja takaisin
- DHCP- ja ping-viestit sisäverkosta palomuuriin
- SSH-yhteyden yhdeltä sisäverkon koneelta palomuuriin.

Kaikki muu estettiin. Palomuurin täydellinen konfiguraatio löytyy liitteestä 2 (sivulla 112).

Rajapintojen IP-osoitteet

Vyatta Coren konfigurointi alkoi rajapintojen osoitteiden määrityksellä kuvion 2 (sivulla 27) mukaisesti. Tämä tapahtui konfiguraatiotilassa seuraavilla komennoilla: (LAN Interfaces 2013.)

```
set interfaces ethernet eth0 address 201.50.9.2/30
set interfaces ethernet eth1 address 10.0.0.1/16
set interfaces ethernet eth2 address 172.16.1.1/24
```

Sisäverkon DHCP-palvelin

Seuraavaksi luotiin sisäverkolle DHCP-palvelin. Vaikka osoitealue oli 10.0.0.0/16, ei sitä otettu kokonaan käyttöön, vaan pelkästään ensimmäiset 256 osoitetta. Lisäksi alueen alusta jätettiin kaksi osoitetta vapaaksi. Toinen niistä, 10.0.0.01, annettiin rajapinnalle eth1 ja osoite 10.0.0.2 varattiin sisäverkon palvelimelle.

DHCP-palvelin piti ensin kääntää päälle seuraavalla komennolla: (Services 2013.)

```
set service dhcp
```

Vyatta Core ei anna määrittää jaettavia osoitealueita suoraan, vaan sille pitää luoda yksi tai useampi **osoiteallas** (engl. *address pool*). Kun allas on luotu, siihen liitetään käytettävä osoitealue, sekä sen muut mahdolliset asetukset. (Services 2013.)

Sisäverkon osoitealtaan nimeksi annettiin ”InternalPool” ja se luotiin seuraavilla komennoilla: (Services 2013.)

```
set service dhcp-server shared-network-name InternalPool
set service dhcp-server shared-network-name InternalPool \
subnet 10.0.0.0/16 start 10.0.0.3 stop 10.0.0.255
```

Myös altaan oletusyhdyskäytävä tuli asettaa. Ilman sitä verkosta ei pystynyt liikennöimään mihinkään:

```
set service dhcp-server shared-network-name InternalPool \
subnet 10.0.0.0/16 default-router 10.0.0.1
```

Oletusyhdyskäytävän asetus

Koska ulompi palomuuri ei tiedä ulkoverkon rakenteesta mitään, sille tuli kertoa minne sen tulisi lähettää paketit joiden kohde on ulkoverkossa. Tämä tapahtui helposti seuraavalla komennolla: (Basic System Configuration 2013.)

```
set system gateway-address 201.50.9.1
```

NAT

Oletusyhdyskäytävän asetuksen jälkeen sisäverkossa voi liikennöidä ja sieltä voi ottaa yhteyden DMZ-alueella olevaan palvelimeen. Sisäverkosta ei kuitenkaan pääse vielä ulkoverkkoon, koska NAT-määrityksiä ei ole tehty.

Vyatta Core käyttää NAT-muunnoksiin **sääntöjä** (engl. *rule*). Jokainen sääntö on numeroitu juoksevasti ja ne käydään läpi nousevassa järjestyksessä. Jokaiseen sääntöön liitetään yksi tai useampi **suodatin** (engl. *filter*), joka määrittää minkä tyyppiseen liikenteeseen sääntö täsmää. Säännön kaikkien suodattimien on täsmättävä, jotta koko sääntö täsmäisi. Ensimmäinen kokonaan täsmäävä sääntö päättää paketin kohtalon. Säännöt tallennetaan kahteen eri listaan, joiden nimet ovat *source* (lähde) ja *destination* (kohde). Lähdesäännöt tekevät SNAT-muunnoksia ja osoitteen piilotusta, kohdesäännöt tekevät DNAT-muunnoksia. Se mitä paketille tapahtuu, mikäli yksikään sääntö ei täsmää, riippuu tilanteesta: paketti saatetaan tuhoa kyselemättä, tai se saatetaan lähettää sellaisenaan oletusyhdyskäytävälle. Yleensä paketti tuhotaan kyselemättä.

Vyatta Core tekee NAT-muunnoksia sekä ennen reititystä, että sen jälkeen. DNAT-muunnokset tehdään ennen reititystä, koska DNAT muuttaa kohdeosoitetta ja siitä ei olisi enää reitityksen jälkeen mitään hyötyä. SNAT tapahtuu reitityksen jälkeen, koska kohteen ei tarvitse tietää alkuperäistä osoitetta. (NAT 2013.)

Skenaariossa tarvittavia NAT-määrityksiä oli kolme:

1. **Sisäverkosta ulkoverkkoon tapahtuva osoitteen piilotus**
   Salli sisäverkon koneiden ottaa yhteys ulkoverkon kohteisiin.
2. **Ulkoverkosta DMZ-alueelle tapahtuva DNAT**
   Muutti DMZ-alueelle kohdistetun liikenteen julkisen osoitteen sisäverkon osoitteeksi.
3. **DMZ-alueelta ulkoverkkoon tapahtuva SNAT**
   Muutti DMZ-alueen sisäisen osoitteen julkiseksi osoitteeksi.

Kun nämä määritykset oli luotu, pystyi ulkoverkosta ottamaan yhteyden DMZ-alueen palvelimeen. Samoin sisäverkosta sai yhteyden ulkoverkon palvelimeen.

Tässä käytettyihin NAT-määrityksiin olisi mahdollista lisätä lisäsuodattimia, jotka rajaisivat liikennettä vieläkin enemmän. Tämä olisi kuitenkin turhaa: NAT ei ole palomuuri. Varsinainen palomuuri luodaan myöhemmin.

1. Sisäverkon osoitteen piilotus

Koska julkisia IP-osoitteita on vain yksi, on sisäverkon osoitteille pakko tehdä osoitteen piilotus. Muuten ne eivät saa yhteyttä ulkoverkkoon. Sisäverkon osoitteen piilotus konfiguroitiin näin: (NAT 2013.)

```
set nat source rule 10 outbound-interface eth0
set nat source rule 10 source address 10.0.0.0/16
set nat source rule 10 translation address masquerade
```

Nämä komennot luovat SNAT-säännön numero 10. Sen suodattimet täsmäävät liikenteeseen jonka lähdeosoite (”*source address*”) on verkossa 10.0.0.0/16 ja jonka kohderajapinnaksi reititys on valinnut rajapinnan eth0 (”*outbound-interface eth0*”). Tähän täsmäävän paketin lähdeosoitteeksi muutetaan saman rajapinnan IP-osoite ja alkuperäinen osoite ja portti tallennetaan taulukkoon (”*translation address masquerade*”).

Jos komentojen antamisen jälkeen antoi komennon ”`show nat`”, sai seuraavanlaisen tulosteen:

```
nat {
    source {
        rule 10 {
            outbound-interface eth0
            source {
                address 10.0.0.0/16
            }
            translation {
                address masquerade
            }
        }
    }
}
```

Listauksesta näkyy hyvin Vyatta Coren sisäisesti käyttämä hierarkkinen tapa tallentaa konfiguraatioita, ja myös miten komennot täsmäävät tähän rakenteeseen.

2. Ulkoverkosta DMZ-alueelle tapahtuva DNAT

Jotta ulkoverkosta voitaisiin ottaa yhteys DMZ-alueelle on luotava sääntö, joka kertoo että ulkoverkosta porttiin 80 (HTTP) tulevien pyyntöjen kohdeosoite on muutettava ensin sisäverkon osoitteeksi. Tätä varten luotiin DNAT-sääntö 10 seuraavilla komennoilla: (NAT 2013.)

```
set nat destination rule 10 destination address 201.50.9.2
set nat destination rule 10 destination port http
set nat destination rule 10 inbound-interface eth0
set nat destination rule 10 protocol tcp
set nat destination rule 10 translation address 172.16.1.2
```

Luotu sääntö 10 täsmää jokaiseen rajapintaan eth0 saapuvaan pakettiin (”*inbound-interface eth0*”) jonka kohdeosoite on 201.50.9.2 (”*destination address 201...*”) ja joka sisältää TCP:n (”*protocol tcp*”) päällä kulkevan HTTP-paketin (”*destination port http*”). Niiden kohdeosoite muutetaan osoitteeksi 172.16.1.2 (”*translation address 172...*”).

Osoitteen piilotus käyttää samaa julkista osoitetta kuin DMZ-palvelin, mutta koska se ei koskaan muuta lähdeportin numeroksi HTTP:n porttia 80, erottuvat ulkoverkosta tulevat DMZ-alueen HTTP-pyynnöt sisäverkon liikenteestä. Näin ne pystytään tunnistamaan ja niiden osoite vaihdetaan ennen reititystä. Kun osoite on vaihdettu, lähetetään paketti uuden IP-osoitteen avulla DMZ-alueen palvelimelle.

3. DMZ-alueelta ulkoverkkoon tapahtuva SNAT

Jotta DMZ-alueen palvelin voisi vastata ulkoverkosta tuleviin pyyntöihin, on sen paketeille tehtävä päinvastainen operaatio kuin saapuville paketeille: lähdeosoite on muutettava julkiseksi osoitteeksi. Tämä tapahtuu SNAT-muunnoksen avulla. Sille luotiin sääntö numero 20 (sääntö 10 on jo osoitteen piilotukselle) seuraavilla komennoilla: (NAT 2013.)

```
set nat source rule 20 outbound-interface eth0
set nat source rule 20 source address 172.16.1.2
set nat source rule 20 translation address 201.50.9.2
```

Sääntö 20 täsmää liikenteeseen, jonka lähdeosoite on 172.16.1.2 (”*source address 172...*”) ja joka on menossa ulos rajapinnasta eth0 (”*outbound-interface eth0*”). Sen lähdeosoite muutetaan osoitteeksi 201.50.9.2. Näin ulkoverkon kone ei koskaan näe sisäverkon osoitetta 172.16.1.2.

Vyöhykepohjaisen palomuurin luonti

Varsinaisen palomuurin luonti oli huomattavasti isompi työ kuin muiden vaiheiden teko. Vyöhykepohjaisen palomuurin rakentaminen jakaantui kolmeen osa-alueeseen:

1. Vyöhykkeiden luonti, nimeäminen ja rajapintojen liittäminen niihin.
2. Vyöhykkeiden välisten liikennöintisääntöjen luonti.
3. Liikennöintisääntöjen liittäminen vyöhykkeisiin.

1. Vyöhykkeiden luonti

Vyöhykkeet kuuluvat `zone-policy` -osion alle. Jokaiselle vyöhykkeelle on annettava yksilöllinen nimi, kerrottava saapuvan liikenteen oletustoiminto, liitettävä sii-

hen rajapinta (tai rajapinnat) ja kerrottava sen palomuurisäännöt. (Firewall reference guide 2013.)

Luodut vyöhykkeet olivat taulukon 5 mukaisia. Varsinainen verkko jaettiin kolmeen vyöhykkeeseen. Niiden lisäksi luotiin yksi lisävyöhyke, joka oli palomuurilaite itsessään. Näin pystyttiin rajaamaan liikennettä, jonka kohteena oli palomuuri (tämä on selitetty tarkemmin myöhemmin tässä luvussa).

Taulukko 5. Luodut vyöhykkeet

Vyöhyke	Tarkoitus
dmz	DMZ-alue
internal	Sisäverkko
public	Ulkoverkko
vyatta	Itse reititin (paikallinen vyöhyke)

Vyöhyke ”public” luotiin seuraavilla komennoilla. Komennot asettivat oletustoimen (kaikki vyöhykkeelle saapuva liikenne tuhotaan) ja liitti rajapinnan eth0 vyöhykkeeseen:

```
set zone-policy zone public default-action drop
set zone-policy zone public interface eth0
```

Muut vyöhykkeet luotiin vastaavalla tavalla:

```
set zone-policy zone internal default-action drop
set zone-policy zone internal interface eth1
set zone-policy zone dmz default-action drop
set zone-policy zone dmz interface eth2
set zone-policy zone vyatta default-action drop
set zone-policy zone vyatta local-zone
```

Viimeinen vyöhyke merkittiin käskyllä ”*local-zone*” tarkoittamaan itse palomuurilaitetta. Palomuuria kuvaava vyöhyke on nimeltään paikallinen vyöhyke, ja niitä

voi olla vain yksi per laite. Paikallinen vyöhyke on käytännössä pakko luoda, koska ilman sitä laite ei vastaa mihinkään verkon viesteihin (tämä estää ping-komennot ja pysäyttää mm. reititysprotokollien toiminnan). Ilman paikallista vyöhykettä Vyatta Core vastaa ulkoverkosta tuleviin SSH-yhteyspyyntöihin ja tämä on erittäin huono asia tietoturvan vuoksi.

Vyöhykkeet olivat nyt valmiita. On huomioitava, että jos asetukset tässä vaiheessa ottaa käyttöön, lakkaa kaikki liikenne vyöhykkeiden välillä. Oletuksena mikään liikenne ei kulje, vaan palomuurisäännöillä on erikseen kerrottava mikä saa kulkea. Tämä kannattaa muistaa, jos laitetta konfiguroi esimerkiksi sisäverkosta SSH-yhteyden yli, sillä SSH katkeaa ilman sitä sallivaa sääntöä. (Firewall reference guide 2013.)

2. Liikennöintisääntöjen luonti

Palomuurisäännöt ovat samoja jotka luotaisiin silloin kun käytettäisiin perinteistä ei-vyöhykepohjaista palomuuria. Ainoa ero on se, miten säännöt otetaan käyttöön: vyöhykepohjaisessa palomuurissa säännöt liitetään vyöhykkeisiin, kun perinteisesti ne liitettäisiin rajapintoihin.

Palomuurisäännöt muistuttavat NAT-sääntöjä. Säännöt numeroidaan juoksevasti ja niille kerrotaan eri lähde- ja kohdesuodattimien avulla mihin ne täsmäävät. Sääntöön liitetään myös toiminto, joka suoritetaan, jos sääntö täsmää johonkin. Säännöt ryhmitellään nimettyjen listojen alle. Jokainen lista sisältää oletustoiminnon, sekä yhden tai useamman säännön. Listojen tarkoitus on ryhmitellä sallittavat asiat tyypeittäin yhteen nippuun. Yksi lista voi esimerkiksi sallia sekä ping-viestit ja SSH-yhteydet. Tai se voi sallia ping-viestit, mutta estää SSH-yhteydet. Tai päinvastoin. Kaikki riippuu siitä, miten listan oletustoiminnon ja itse sääntöjen toiminnot määrittelee. Listan oletustoiminto on pudotus (*drop*), mutta se asetetaan tässä aina erikseen. Näin ei pääse syntymään ristiriitatilanteita, eivätkä toiminnot jää koskaan epäselviksi.

Vyöhykkeiden välinen liikenne on laitteen näkökulmasta aina yksisuuntaista. Jos kahden vyöhykkeen halutaan voivan viestiä kumpaankin suuntaan tahansa, on

pakko luoda lista kummallekin suunnalle ja ottaa se käyttöön kummassakin vyöhykkeessä. Muutoin liikenne voi kulkea vain yhteen suuntaan. (Firewall reference guide 2013.)

Koska vyöhykkeitä oli skenaariossa neljä, tarvittiin listoja seitsemän kappaletta. Vyatta Core ei oletuksena rajoita laitteesta itsestään lähtevää liikennettä. Siksi kahdeksatta palomuurilistaa, joka sallisi paluuviestit sisäverkkoon, ei tarvittu. Tarvittiin vain lista, joka sallii sisäverkosta paikalliselle vyöhykkeelle tulevan liikenteen. (Firewall reference guide 2013.) Taulukko 6 sisältää tarvitut vyöhykkeet ja niiden käyttötarkoituksen.

Taulukko 6. Sääntölistat vyöhykkeiden väliseen liikenteeseen

Vyöhykkeen nimi	**Liikennöinti**	
	Mistä	**Mihin**
dmz_to_internal	DMZ	Sisäverkko
dmz_to_public	DMZ	Ulkoverkko
internal_to_dmz	Sisäverkko	DMZ
internal_to_public	Sisäverkko	Ulkoverkko
internal_to_vyatta	Sisäverkko	Paikallinen vyöhyke
public_to_dmz	Ulkoverkko	DMZ
public_to_internal	Ulkoverkko	Sisäverkko

Ensin luotiin lista joka salli sisäverkosta paikalliselle vyöhykkeelle suuntautuvan liikenteen. Tämän jälkeen laitetta pystyttiin konfiguroimaan SSH-yhteyden yli. Koska skenaariossa sisäverkko on ”luotettu”, ei sieltä paikalliselle vyöhykkeelle suuntautunutta liikennettä rajoitettu. Tälle listalle annettiin nimi ”internal_to_vyatta”, ja se luotiin seuraavilla komennoilla:

```
set firewall name internal_to_vyatta default-action accept
set firewall name internal_to_vyatta rule 10 action drop
set firewall name internal_to_vyatta rule 10 destination \
port 22
```

```
set firewall name internal_to_vyatta rule 10 protocol tcp
set firewall name internal_to_vyatta rule 10 source \
address !10.0.0.3
```

Komentojen luoma lista sallii oletuksena kaiken liikenteen (”*default-action accept*”), sillä luotamme sisäverkkoon. Sääntö numero 10 kuitenkin estää SSH-yhteydet (”*destination port 22*”) jotka eivät tule osoitteesta 10.0.0.3 (”*source address !10.0.0.3*”); huutomerkki IP-osoitteen alussa tarkoittaa ”ei”. Tämän seurauksena sisäverkon koneelta jonka IP-osoite on 10.0.0.3 voi ottaa SSH-yhteyden palomuuriin, mutta mistään muualta ei. On huomattava, että tämä toimii vain, jos sama kone saa joka kerta osoitteen 10.0.0.3. Tämän voi pakottaa käyttämällä koneessa kiinteää IP-osoitetta, tai konfiguroimalla DHCP-palvelimen siten, että se antaa aina tuolle tietylle koneelle saman osoitteen.

Seuraava lista luotiin sallimaan sisäverkosta liikenne ulkoverkkoon. Koska skenaariossa tätä liikennettä ei rajoitettu millään tavalla, tuli tästä listasta yksinkertainen:

```
set firewall name internal_to_public default-action accept
```

Sääntöjä tällä listalla ei ole, ainoastaan oletustoiminto, joka sallii kaiken. Vyatta Core ei tue tason 7 palomuuria, joten työntekijöiden työaikana tapahtuvaa Facebook-selailua ei tällä voi estää. Joka tapauksessa, täsmälleen samalla tavalla luotiin lista joka salli kaiken ulkoverkosta sisäverkkoon:

```
set firewall name public_to_internal default-action accept
```

Tässä on hyvä muistaa, että sisä- ja ulkoverkon rajalla on NAT-muunnos, joten se että palomuuri päästää kaiken läpi ei tarkoita, että sisäverkko olisi suoraan kaikkien nähtävillä. Jos NAT-muunnosta ei olisi, tarvittaisiin tähän kohtaan tilallinen palomuuri.

Asiat mutkistuivat hieman, kun luotiin listoja sisäverkon ja DMZ-alueen välille. Oletuksena kaikki liikenne vyöhykkeiden välillä estettiin, mutta HTTP-liikenne

haluttiin päästää läpi. Siksi sääntöihin tuli määrittää kohdeportiksi HTTP:n portti 80. Sisäverkosta DMZ-alueelle HTTP-liikenteen salliva lista ”internal_to_dmz” luotiin seuraavasti:

```
set firewall name internal_to_dmz default-action drop
set firewall name internal_to_dmz rule 10 action accept
set firewall name internal_to_dmz rule 10 destination \
port http
set firewall name internal_to_dmz rule 10 protocol tcp
```

Vastaavasti toiseen suuntaan paluuliikenteen salliva lista ”dmz_to_internal” oli lähes samanlainen, mutta tällä kertaa porttia ei määritetty. Sen paikalla määritettiin TCP-protokollaan liittyvät tilatiedot ”*established*” ja ”*related*”. Koska DMZ-alueelta on tarkoitus aina tulla vain paluuviestejä, suodattavat nämä kaksi tila-asetusta muun liikenteen pois. Toisin sanoen, DMZ-alueen ei anneta luoda yhteyttä, se voi vain vastata ulkoa tulleisiin yhteyspyyntöihin.

```
set firewall name dmz_to_internal default-action drop
set firewall name dmz_to_internal rule 10 action accept
set firewall name dmz_to_internal rule 10 protocol tcp
set firewall name dmz_to_internal rule 10 state \
established enable
set firewall name dmz_to_internal rule 10 state \
related enable
```

Kaksi viimeistä tilatiedot asettavaa komentoa luovat siis tilallisen palomuurin. Palomuuri ei päästä läpi liikennettä, jotka eivät jo liity ulkoapäin tulleeseen pyyntöön. Näin DMZ-alueelta ei voi luoda yhteyksiä ulkoverkkoon, koska palomuurin näkökulmasta ne eivät ole vastauksia mihinkään. Luvussa 5.4.3 (sivulla 67) on kerrottu, miten tilallinen palomuuri luodaan ilman vyöhykkeitä.

Lopuksi luotiin kaksi listaa, jotka sallivat liikennettä julkisesta verkosta DMZ-alueelle ja takaisin. Niiden kirjoittamista helpotti se, ettei skenaariossa DMZ-alueella ollut muuta kuin yksi HTTP-palvelin. Jos palveluita olisi ollut useita, olisi listoista tullut pitkiä ja hankalia luoda, sillä jokaiselle palvelulle olisi pitänyt luoda

oma sääntönsä. Ensimmäinen listoista, ”public_to_dmz”, kertoo mitä liikennettä ulkoverkosta DMZ-alueelle saa mennä. Tämä lista luotiin seuraavilla komennoilla:

```
set firewall name public_to_dmz default-action drop
set firewall name public_to_dmz rule 10 action accept
set firewall name public_to_dmz rule 10 destination \
port http
set firewall name public_to_dmz rule 10 protocol tcp
```

Jälleen kerran oletustoimeksi asetettiin pudotus. Mutta sääntö numero 10 päästää läpi TCP-protokollan päällä olevat, HTTP-porttiin menevät paketit. IP-osoitteita ei tarvitse käyttää, koska tässä vaiheessa kohde tiedetään jo (nämä säännöt käydään läpi reitityksen jälkeen). Palomuuri vain tarkistaa saako paketti mennä läpi vai ei. Emme päästä läpi muuta kuin HTTP-viestejä.

Vastaavasti DMZ-alueelta ulkoverkkoon mentäessä päästetään läpi vain ulkoapäin tulleisiin pyyntöihin kuuluvat vastaukset. Lista ”dmz_to_public” onkin täsmälleen samanlainen kuin jo aiemmin luotu lista ”dmz_to_internal”. Mikään ei estäisi saman listan käyttöä useaan kertaan, mutta tämä estäisi vyöhykekohtaisten sääntöjen luonnin. On suositeltavaa aina luoda uusi lista joka suunnalle, ja käyttää jokaista listaa vain yhden kerran. Joka tapauksessa tarvittava lista luotiin komennoilla:

```
set firewall name dmz_to_public default-action drop
set firewall name dmz_to_public rule 10 action accept
set firewall name dmz_to_public rule 10 protocol tcp
set firewall name dmz_to_public rule 10 state \
established enable
set firewall name dmz_to_public rule 10 state related enable
```

Listat olivat nyt valmiit ja ne piti enää liittää vyöhykkeisiin.

3. Sääntöjen liittäminen vyöhykkeisiin

Kun palomuurisäännöt on luotu, voidaan listat liittää vyöhykkeisiin. Tämän jälkeen vain säännöissä määritetty liikenne voi liikkua vyöhykkeiden välillä. Listan

liittäminen vyöhykkeeseen tehdään seuraavalla komennolla: (Firewall reference guide 2013.)

```
set zone-policy zone <kohde> from <lähde> firewall name <lista>
```

Tämä komentoa käyttämällä taulukon 6 (sivulla 54) mukaisesti listat liitettiin luotuihin vyöhykkeisiin seuraavilla komennoilla:

```
set zone dmz from internal firewall name internal_to_dmz
set zone dmz from public firewall name public_to_dmz
set zone internal from dmz firewall name dmz_to_internal
set zone internal from public firewall name \
public_to_internal
set zone public from dmz firewall name dmz_to_public
set zone public from internal firewall name \
internal_to_public
set zone vyatta from internal firewall name \
internal_to_vyatta
```

Asetusten käyttöönoton ja tallennuksen jälkeen yhden palomuurin toteutus Vyatta Corella oli valmis.

4.5.2 Kaksi palomuuria

Jo luotu yhden palomuurin toteutus muuttuu helposti kahden palomuurin toteutukseksi. Käytetty sisempi palomuuri oli luvussa 4.2 (sivulla 28) luotu Vyatta Core-reititin. Se kytkettiin sellaisenaan kiinni ulompaan palomuuriin. Tässä muokatun laitteen konfiguraatio löytyy täydellisenä liitteestä 2 (sivulla 119).

DHCP-palvelimen poisto

Koska sisäverkko sai nyt IP-osoitteet sisemmältä muurilta, ei uloin enää tarvinnut DHCP-palvelinta. Se poistettiin konfiguraatiotilassa seuraavalla komennolla:

```
delete service dhcp-server
```

Sisäverkon rajapinnan IP-osoite

Uloimman palomuurin sisäverkkoon osoittavan rajapinnan eth1 IP-osoite oli vaihdettava kuvion 3 (sivulla 27) mukaisesti osoitteeksi 172.16.2.1. Vanha osoite oli poistettava ensin. Muutokset tehtiin komennoilla:

```
delete interfaces ethernet eth1 address 10.0.0.1/16
set interfaces ethernet eth1 address 172.16.2.1/24
```

Kiinteä reitti sisäverkkoon

Uloin palomuuri ei tiedä sisemmän palomuurin takaisesta sisäverkosta, joten sen olemassaolo on kerrottava sille kiinteällä reitillä. Reitti luotiin komennolla: (Basic Routing 2013.)

```
set protocols static route 10.0.0.0/16 next-hop 172.16.2.2
```

Komento kertoo, että verkkoon 10.0.0.0/16 pääsee osoitteen 172.16.2.2 kautta. Sisempi palomuuri sisälsi jo oletusreitin kohti ulompaa muuria, joten muuta ei tarvinnut tehdä. Kahden palomuurin toteutus Vyatta Corella oli valmis.

5 Skenaario 2: Verkon osien suojaus

5.1 Johdanto

Toisessa skenaariossa suojattiin pientä virtuaalista harjoitteluverkkoa. Verkko koostui kolmesta runkoreitittimestä (”CoreA”, ”CoreB” ja ”CoreC”), joihin kytkettiin kolme varsinaista verkkoa (A, B ja C) työasemille. Lisäksi runkoon liitettiin palvelinverkko ja hallintaverkko. Palomuureilla rajoitettiin verkon eri osiin pääsyä, ja niillä karsittiin pois turhaa liikennettä. Verkon rakenne ilmenee kuviosta 9 (sivulla 61).

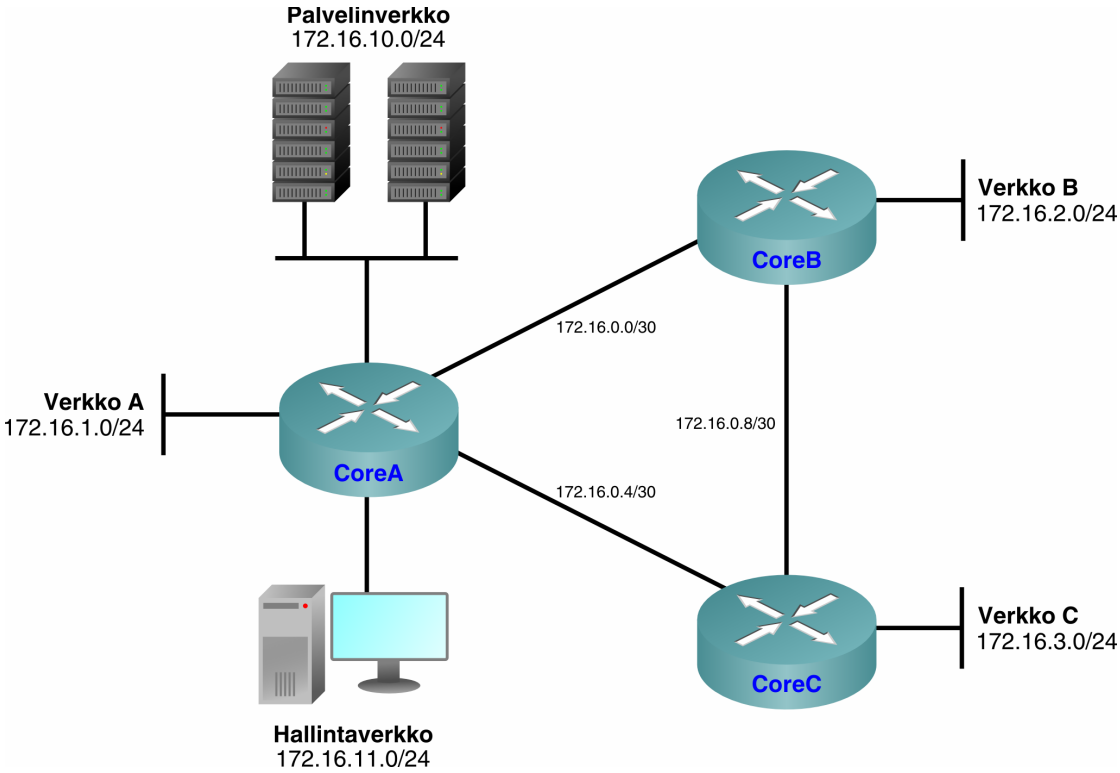
Jokainen runkoreititin toimi palomuurina, ja niiden avulla verkkoon luotiin seuraavat liikennöintisäännöt. Säännöillä simuloitiin erästä verkossa tehtyä koetta:

- verkot A, B ja C saivat vapaasti yhteyden palvelimiin (myös SSH)
- verkot A, B ja C eivät voineet ottaa yhteyttä hallintaverkkoon millään tavalla
- verkot B ja C olivat pahimpia kilpailijoita ja vihollisia, joten ne eivät saaneet viestiä keskenään; verkosta A sai vapaasti yhteyden kumpaakin verkkoon
- verkot A, B ja C pystyivät testaamaan runkoverkon sekä palvelinverkon yhteyttä ping- ja traceroute-komennoilla
- vain hallintaverkosta pystyi ottamaan SSH-yhteyden reitittimiin
- palvelinverkko ei voinut ottaa yhteyttä mihinkään, ainoastaan vastaamaan muualta tulleisiin yhteyspyyntöihin
- lisäksi verkon B takaa löytyi hieman isompi yksityinen verkko, joka käydään läpi luvussa 5.5 (sivulla 72).

Runkoreitittimet koostuivat Vyatta Core-reitittimistä. Toisin kuin skenaariossa 1, ei tässä käytetty vyöhykepohjaisia palomuureja lainkaan. Kaikki palomuurisäännöt tehtiin erillisinä komentoina. Reitittimet konfiguroitiin vaihtamaan reititys-

tietoja OSPF-protokollalla, koska verkon luonteen vuoksi sen rakenne tulisi muuttumaan usein, ja kiinteitä reittejä käyttämällä muutoksiin menisi liian kauan. OSPF käytti aluetta 0. Runkoverkon linkkivälien osoitteet otettiin alueelta 172.16-.0.0/24.

Palvelimet olivat kaikille yhteisiä, eikä niihin pääsyä rajoitettu. Palvelimien rakenne on käyty läpi luvussa 5.2. Varsinaiset verkot sisälsivät joukon tavallisia oletusasennuksilla tehtyjä Windows XP-, sekä XUbuntu-virtuaalikoneita. Verkon kaikkien laitteiden konfiguraatiot löytyvät täydellisinä liitteestä 2 (sivulta 125 alkaen).



Kuvio 9. Skenaarion 2 runkoverkon rakenne

5.2 Yhteisten palvelimien luonti

Verkon yhteiset palvelimet olivat Debian 7-pohjaisia virtuaalikoneita. Niihin asennettiin mm. Apache-palvelinohjelmisto, PHP-tulkki, MySQL-tietokantapalvelin,

NTP-palvelin, sekä joukko muita palveluita. DHCP-palvelinta ei asennettu, koska DHCP jätettiin jokaisen verkon omaksi asiakseen.

5.3 Verkon rungon luonti

Vaikka skenaarion pääasiallisin kohde on palomuurien konfigurointi, käydään tässä luvussa nopeasti läpi miten CoreA-reititin luotiin ja konfiguroitiin. CoreB- ja CoreC-laitteet konfiguroitiin vastaavilla komennoilla.

5.3.1 Viidennen rajapinnan lisäys CoreA-reitittimeen

VirtualBox tukee enintään kahdeksaa rajapintaa per virtuaalikone, mutta sen käyttöliittymä näyttää niistä vain ensimmäiset neljä. Rajapinnat 5-8 on mahdollista luoda ja konfiguroida VirtualBoxin mukana tulevalla komentorivipohjaisella vboxmanage-ohjelmalla. On huomattava, että kohdevirtuaalikone on sammutettava ennen vboxmanage-ohjelman käyttöä. VirtualBox voi olla käynnissä.

vboxmanage tarvitsee avukseen halutun virtuaalikoneen nimen, tai sen yksilöllisen UUID-koodin. Olemassa olevat virtuaalikoneet nimineen ja koodeineen näkee seuraavalla komennolla:

```
vboxmanage list vms
```

Kun halutun koneen nimi ja UUID on tiedossa, voidaan modifyvm-parametrilla suorittaa varsinainen muokkaus. CoreA-laitteen (jonka tunniste skenaariossa sattui olemaan 36c6fc21-af52-4d79-af73-7931df481e45) viides rajapinta luotiin seuraavilla komennoilla:

```
vboxmanage modifyvm 36c6fc21-af52-4d79-af73-7931df481e45 \
–-nic5 intnet
vboxmanage modifyvm 36c6fc21-af52-4d79-af73-7931df481e45 \
--nictype5 82540EM
```

```
vboxmanage modifyvm 36c6fc21-af52-4d79-af73-7931df481e45 \
--cableconnected5 on
vboxmanage modifyvm 36c6fc21-af52-4d79-af73-7931df481e45 \
--intnet5 "management"
```

Komennot asettivat rajapinnan tyypiksi sisäinen (”intnet”), asettivat verkkokortin tyypin (”82540EM” joka tarkoittaa Intel PRO/1000 MT Desktop -verkkokorttia), kytkivät siihen johdon ja liittivät sen ”management”-nimiseen sisäiseen verkkoon. Samaan verkkoon kytkettiin myös verkon hallintakoneet.

Näiden komentojen jälkeen virtuaalikone käynnistettiin normaalisti. Viides rajapinta näkyi virtuaalikoneessa eth4-nimisenä. (VBoxManage 2013.)

5.3.2 Rajapintojen konfigurointi

CoreA-reitittimen rajapinnat konfiguroitiin kuvion 9 (sivulla 61) mukaisiin verkkoihin seuraavilla komennoilla: (LAN Interfaces 2013.)

```
set interfaces ethernet eth0 address 172.16.0.1/30
set interfaces ethernet eth1 address 172.16.0.5/30
set interfaces ethernet eth2 address 172.16.1.1/24
set interfaces ethernet eth3 address 172.16.10.1/24
set interfaces ethernet eth4 address 172.16.11.1/24
```

5.3.3 OSPF

CoreA-laitteen OSPF konfiguroitiin seuraavilla komennoilla: (OSPF 2013.)

```
set protocols ospf passive-interface eth2
set protocols ospf passive-interface eth3
set protocols ospf passive-interface eth4
set protocols ospf area 0 network 172.16.0.0/30
set protocols ospf area 0 network 172.16.0.4/30
set protocols ospf area 0 network 172.16.1.0/24
```

```
set protocols ospf area 0 network 172.16.10.0/24
set protocols ospf area 0 network 172.16.11.0/24
```

Komennot lisäsivät kaikki laitteen tuntemat verkot mainostettavaksi muille reitittimille, ja merkitsi runkoverkkoon kuulumattomat rajapinnat (eth2, eth3 ja eth4) passiivisiksi, eli niihin ei lähetetty protokollan mainosviestejä.

5.4 Runkoverkon palomuurisäännöt

5.4.1 Perinteiset palomuurisäännöt

Kun verkko oli luotu, voitiin tarvittavat palomuurisäännöt luoda runkoreitittimiin.

Toisin kuin Laatikko Oy:n tapauksessa tehdyt palomuurit (luku 4 sivulla 25), ei tässä käytetty vyöhykepohjaisia palomuureja. Tämä siksi, ettei verkkoa voi kovin järkevästi jakaa vyöhykkeisiin runkoreitittimien näkökulmasta. Verkon voisi kyllä jakaa vyöhykkeisiin, mutta joka laite sisältäisi vain kaksi vyöhykettä (”runkoverkko” ja ”ei-runkoverkko”), joten siitä ei olisi mitään käytännön hyötyä. CoreA-reitittimessäkin ainoa käyttökohde vyöhykkeille olisi estää verkko A:n pääsy hallintaverkkoon, mutta vyöhykkeiden luonti tätä yhtä asiaa varten olisi vain ajan tuhlausta.

Koska vyöhykkeillä ei tässä saavutettu mitään, toteutettiin kaikki palomuuritoiminnallisuus ”tavallisilla” sääntölistauksilla, jotka liitettiin rajapintoihin. Sisäverkoissa vyöhykepalomuureja voisi kyllä käyttää ihan hyvin. Lisäksi ei olisi lainkaan huono idea lisätä vielä yksi palomuuri runkoreitittimen ja aliverkkojen väliin lisäsuojaa tuomaan.

Palomuurisäännöt ovat samoja sääntö+suodatin -listauksia, mitä luvun 4.5.1 aliluvussa ”*NAT*” (sivulla 48) luotiin. Erona on se, että ne liitetään suoraan rajapintoihin eikä vyöhykkeisiin. Vyatta Core antaa määrittää joka rajapinnalle kolme sään-

tölistaa. Listat liitetään rajapintoihin liikenteen suunnan mukaan. Listojen nimet ja tarkoitukset selviävät taulukosta 7. (Firewall 2013.)

Taulukko 7. Vyatta Coren palomuurien suunnat

Suunta	Käyttökohde
in	Rajapinnan kautta sisään tuleva, ja järjestelmän läpi kulkeva liikenne.
out	Rajapinnan kautta ulos lähtevä, ja järjestelmän läpi jo kulkenut liikenne.
local	Rajapintaan kohdistuva liikenne. Tarkoittaa laitetta itseään. Vyöhykepohjaisessa palomuurissa vastine on *local zone* (katso luku ”*Vyöhykepohjaisen palomuurin luonti*” sivulla 51).

Koska listoja voi olla korkeintaan kolme rajapintaa kohti, on kaikki tarvittavat säännöt niputettava ja järjesteltävä tarkasti niiden alle.

5.4.2 SSH-yhteyden rajoitus

Vaikkei harjoitusverkko ollut mitenkään erityisen turvallinen, rajoitettiin siinä silti SSH-yhteyksiä. SSH-yhteyksiä sai verkossa ottaa vapaasti (esimerkiksi palvelimiin), mutta runkoreitittimien haluttiin vastaavan vain hallintaverkosta tuleviin SSH-yhteyspyyntöihin. Näin aliverkkojen käyttäjät eivät pystyneet ottamaan niihin yhteyttä ja esimerkiksi sotkemaan palomuurien asetuksia.

Tätä varten luotiin lista nimeltä ”block_ssh”, joka liitettiin jokaisen runkoreitittimen jokaiseen rajapintaan ”local”-suunnan alle. Oletuksena lista päästi kaiken läpi, koska ilman tätä esimerkiksi ping-komennolla ei olisi voinut testata yhteyttä verkosta runkoreitittimeen. Lisäksi, jos oletustoiminto olisi ollut pudotus tai esto, ei OSPF-protokolla olisi toiminut (ja näin tapahtui aivan oikeasti: kokeena CoreB-laitteen sääntöön asetettiin oletuksena pudotus, ja hetkeä myöhemmin CoreA- ja CoreC-laitteet olivat kadottaneet sen reitit; kun sääntö palautettiin ennalleen, rei-

tit palasivat). Koska oletuksena kaikki pääsi läpi, tarvitsi sääntöön vain listata mitä ei haluttu pääsevän läpi. Tietoturvan vuoksi kaikki pitäisi oletuksena estää ja sitten sallia vain haluttu, mutta koska verkko oli harjoituskäytössä, ei sitä haluttu rajata liikaa.

Oletustoiminto asetettiin seuraavalla komennolla:

```
set firewall name block_ssh default-action accept
```

Varsinaisen eston tekevä sääntö luotiin seuraavasti:

```
set firewall name block_ssh rule 10 action drop
set firewall name block_ssh rule 10 destination port 22
set firewall name block_ssh rule 10 protocol tcp
set firewall name block_ssh rule 10 source \
address !172.16.11.0/24
```

Sääntö toimi seuraavasti: pudotetaan jokainen TCP-paketti (”*action drop*”, ”*protocol tcp*”) jonka kohteena on SSH:n portti 22 (”*destination port 22*”) ja jonka lähdeosoite **ei** ole verkosta 172.16.11.0/24 (”*source address !172...*”). Huutomerkki viimeisellä rivillä on koko säännön ydin, se muuttaa komennon vaikutuksen päinvastaiseksi; ilman sitä kaikki paitsi hallintaverkosta 172.16.11.0/24 tulevat paketit pääsisivät läpi ja hallintaverkosta ei saisi lainkaan reitittimeen yhteyttä. Mutta huutomerkki kääntää säännön päinvastaiseksi, ja sallii vain hallintaverkosta tulevat yhteydet. Kaikki muu liikenne pääsee läpi normaalisti. (Firewall 2013.)

Kun sääntö oli luotu, se liitettiin jokaiseen rajapintaan. Tämä tapahtui seuraavalla komennolla:

```
set interfaces ethernet <rajapinta> firewall <suunta> <lista>
```

Täten esimerkiksi CoreB-laitteella palomuurien liittäminen rajapintoihin tapahtui seuraavilla komennoilla:

```
set interfaces ethernet eth0 firewall local name block_ssh
set interfaces ethernet eth1 firewall local name block_ssh
set interfaces ethernet eth2 firewall local name block_ssh
```

Komennot liittivät luodun block_ssh-listan joka rajapintaan paikalliselle laitteelle tulevalle liikenteelle. Vastaavat komennot tuli antaa myös muissakin runkoreitittimissä. Niiden käyttöönoton jälkeen joka verkosta pystyi ping-komennolla testaamaan yhteyttä reitittimiin, OSPF toimi normaalisti ja SSH-yhteydet toimivat hallintaverkosta, mutta muualta ei päässyt reitittimiin käsiksi.

5.4.3 Hallintaverkon suojaus

Koska hallintaverkossa oli koneita, joista sai SSH-yhteyden runkoreitittimiin, ja mahdollisiin muihin tärkeisiin verkon kohteisiin, oli niitä suojattava muilta verkoilta. Tämä tarkoitti tilallista palomuuria, joka esti verkon ulkoa tulevat yhteyspyynnöt, mutta päästi läpi vastaukset alun perin sisältä tulleisiin pyyntöihin.

Suojauksesta vastaavalle listalle annettiin nimeksi ”protect_management”. Se luotiin seuraavilla komennoilla:

```
set firewall name protect_management default-action drop
set firewall name protect_management rule 10 action accept
set firewall name protect_management rule 10 destination \
address 172.16.11.0/24
set firewall name protect_management rule 10 state \
established enable
set firewall name protect_management rule 10 state \
related enable
```

Listan oletustoiminta on pudotus (”*default-action drop*”), koska oletuksena emme päästä mitään läpi. Sääntö numero 10 kuitenkin määrittää, että päästämme läpi liikenteen (”*action accept*”) jonka kohdeosoite on hallintaverkko (”*destination address 172...*”) ja joka liittyy jo johonkin aiempaan hallintaverkosta tulleeseen yhteyspyyntöön (”*state established enable*”, ”*state related enable*”).

Kaksi viimeistä riviä ovat tilallisen palomuurin ydin. Ne toimivat tässä täsmälleen samalla tavalla kuin luvun 4.5.1 (sivulla 46) vyöhykepohjaisten palomuurien vastaavat. Kuten jo luvussa 3.1.3 (sivulla 18) mainittiin, toimii Vyatta Core oletuksena tilattomana, ja sen voi konfiguroida toimimaan pysyvästi tilallisena. Tämä on kuitenkin resurssien tuhlausta, koska voimme kääntää tilallisuuden päälle vain silloin kun sitä tarvitaan.

Kaksi viimeistä komentoa kertovat, että verkkoon tulevien pakettien on jollain tavalla liityttävä hallintaverkosta aiemmin tulleeseen yhteyspyyntöön. Esimerkiksi TCP-protokollan paluuviestit liittyvät, joten ne pääsevät läpi. Mutta yksittäiset ulkoapäin tulevat ICMP-viestit (esimerkiksi ping-komennon luomat) eivät liity mihinkään, joten ne eivät pääse muurista läpi. Jos hallintaverkosta testaa yhteyttä ping-komennolla, sen paluuviesti pääsee läpi, koska se liittyy jo lähetettyyn pingkutsuun.

Tarvittavat palomuurisäännöt liitettiin CoreA-reitittimen rajapinnan eth4 ulosmenevälle liikenteelle. Ulosmenevälle siksi, että laitteen näkökulmasta verkkoon menevä liikenne lähtee laitteesta poispäin, eli ulos. Lista liitettiin rajapintaan luvusta 5.4.2 (sivulla 65) tutulla komennolla:

```
set interfaces ethernet eth4 firewall out name \
protect_management
```

Jos konfiguraatiota nyt tarkasteli komennolla ”`show configuration`”, näki kummankin rajapinnassa käytössä olevan palomuurin suuntineen:

```
…
ethernet eth4 {
    …
    firewall {
        local {
            name block_ssh
        }
        out {
            name protect_management
        }
```

```
    }
    …
}
…
```

Näiden komentojen jälkeen ei hallintaverkkoon enää saanut yhteyttä ulkopuolelta. Kokeena ajettu nmap-verkkoskanneri ei löytänyt verkosta edes koneita, kun ennen palomuurin konfigurointia se löysi sen kaikki koneet ja niiden avoimet palvelut. Rajapinta eth4 vastaa yhä ping-viesteihin, mutta senkin voisi estää lisäämällä sille sopivan ICMP-protokollan estävän säännön.

5.4.4 Palvelinverkon rajoitus

Kuten skenaarion 1 DMZ-alueen palvelin, ei tässäkään skenaariossa haluttu palvelinkoneiden voivan luoda tarpeettomia yhteyksiä aivan mihin sattuu. Palvelimien haluttiin vastaavan ulkoapäin tulleisiin pyyntöihin, mutta niiden ei haluttu pystyvän luoda yhteyksiä itsenäisesti. Tämä toteutettiin tilallisella palomuurilla.

Tarvittava lista (jolle annettiin nimi ”restrict_servers”) luotiin seuraavilla komennoilla:

```
set firewall name restrict_servers default-action drop
set firewall name restrict_servers rule 10 action accept
set firewall name restrict_servers rule 10 state \
established enable
set firewall name restrict_servers rule 10 state related \
enable
```

Lista suodatti liikennettä samalla tavalla kuin edellisen luvun hallintaverkon vastaava, mutta emme määrittäneet mitään lähdettä tai kohdetta, emmekä protokollia. Suurin ero tuleekin suunnasta: hallintaverkon tapauksessa estimme sinne menevää liikennettä, nyt estimme verkosta pois tulevaa liikennettä. Emme rajoittaneet palvelinverkkoon menevää liikennettä mitenkään.

Luotu lista liitettiin palvelinverkkoon menevään eth3-rajapintaan sisäänpäin menevän liikenteen suuntaan. Sisäänpäin siksi, että reitittimen näkökulmasta estettävä liikenne tulee verkosta laitteelle sisään.

```
set interfaces ethernet eth3 firewall in name \
restrict_servers
```

Komennon jälkeen palvelinverkko ei enää nähnyt ulkopuolista maailmaa. Se ei saanut edes ping-viestillä yhteyttä mihinkään. Palvelimiin kuitenkin pystyi ottamaan yhteyttä normaalisti, ja ne vastasivat ulkoa tulleisiin ping-viesteihin.

Tietoturvan kannalta tämä ei ollut kaikkein paras mahdollinen ratkaisu. Palvelimiin menevää liikennettä pitäisi rajoittaa, ja vain halutut protokollat pitäisi päästää läpi. Palomuuri ei rajoita palvelinverkon sisällä tapahtuvaa liikennettä. Palvelimilta voi lisäksi ping-komennolla saada yhteyden CoreA-reitittimen rajapintoihin, koska ne kuuluvat paikalliseen vyöhykkeeseen, ja sen lista estää vain SSH-yhteydet. Palvelinkoneista ei voi ping-komenolla saada yhteyttä muiden runkoreitittimien rajapintoihin. Tämän voisi estää luomalla CoreA-laitteen block_ssh -listaan säännön, joka estäisi vastaamasta palvelinverkosta tulevaan liikenteeseen.

Ulospäin menevää liikennettä rajoitettiin lähinnä siksi, että se esti palvelimien käytön hyökkäyksissä. Hyökkääjä ei pystynyt kierrättämään liikennettä palvelimien läpi. Hyökkääjä voisi myös ottaa jonkun palvelimen haltuunsa, ja tehdä sen avulla jatkohyökkäyksen toista palvelinta vastaan, eikä palomuuri välittäisi asiasta. Lisäksi tällä estettiin verkkojen B ja C välistä liikennöintiä. Kun palvelin ei voi ottaa yhteyttä mihinkään, ei kummastakaan verkosta voi ottaa yhteyttä toiseen kierrättäen sen palvelimen kautta.

5.4.5 Verkkojen välisen liikenteen rajoitus

Verkot B ja C olivat pahoja kilpailijoita ja siksi niiden välille ei haluttu mitään yhteyttä. Verkko A sai liikennöidä vapaasti kumpaankin.

Kun palomuurilla halutaan estää jotain pääsemästä johonkin tiettyyn kohteeseen, on hyvä sijoittaa tarvittavat säännöt mahdollisimman lähelle lähdettä. Näin turhat viestit eivät pääse sisääntulorajapintaa pidemmälle, eivätkä hidasta runkoverkkoa.

Tarvittavat säännöt luotiin CoreB- ja CoreC-laitteille. Kohderajapinta oli kummassakin eth2, ja luotu lista liitettiin aina sen sisääntulevalle liikenteelle. Sisääntuleva, koska pakettien on kuljettava laitteen läpi päästäkseen runkoverkkoon.

Rajoituksen idea oli yksinkertainen: kaikki päästettiin läpi, paitsi jos kohdeosoite oli ei-haluttu. CoreB-laitteelle tehty, liikennettä verkosta B verkkoon C rajoittava lista luotiin seuraavilla komennoilla:

```
set firewall name block_B_to_C default-action accept
set firewall name block_B_to_C rule 10 action drop
set firewall name block_B_to_C rule 10 source address \
172.16.2.0/24
set firewall name block_B_to_C rule 10 destination \
address 172.16.3.0/24
```

Listan sääntö 10 esti liikennettä, jos sen lähde oli verkossa 172.16.2.0/24 ja kohde verkossa 172.16.3.0/24. Kaikki muu pääsi läpi. Lista liitettiin CoreB-laitteen rajapintaan eth2 sisääntulevalle liikenteelle:

```
set interfaces ethernet eth2 firewall in name block_B_to_C
```

Vastaavalla tavalla luotiin CoreC-laitteelle verkosta C verkkoon B estävä lista. Ainoa ero oli se, että lähde- ja kohdeverkot olivat käänteiset:

```
set firewall name block_C_to_B default-action accept
set firewall name block_C_to_B rule 10 action drop
set firewall name block_C_to_B rule 10 source \
address 172.16.3.0/24
set firewall name block_C_to_B rule 10 destination \
address 172.16.2.0/24
```

Lista otettiin käyttöön tutulla tavalla:

```
set interfaces ethernet eth2 firewall in name block_C_to_B
```

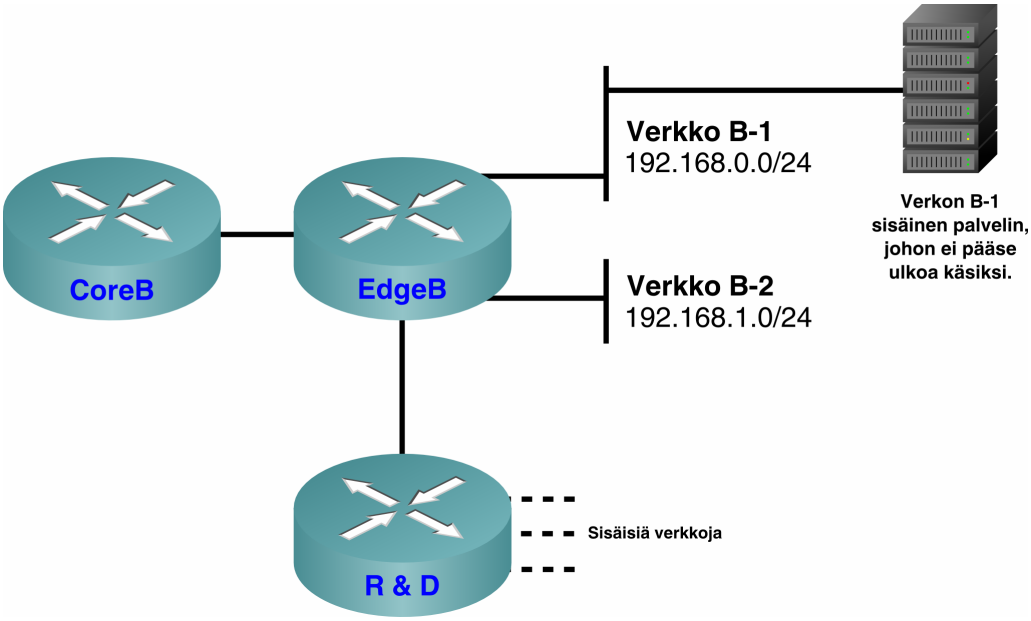
Tämän jälkeen verkkoon C ei enää saanut mitään yhteyttä verkosta B, eikä verkosta B saanut myöskään mitään yhteyttä verkkoon C. Verkosta A sai yhteyden kumpaakin, sekä kaikista sai normaalisti yhteyden palvelimiin.

Hyökkääjän olisi mahdollista kierrättää liikenne verkon A kautta, jolloin verkot B ja C voisivat viestiä keskenään. Tämän estämiseksi verkon A kohteilla olisi hyvä olla omat palomuurinsa.

5.5 Verkon B sisäverkko

Verkossa B ei tyydytty käyttämään jo annettua osoitealuetta 172.16.2.0/24 sellaisenaan kuten muissa verkoissa, vaan käyttäjät halusivat enemmän. Siksi CoreB-laitteen perään kiinnitettiin toinen reititin, joka loi taakseen useamman aliverkon. Nämä aliverkot kytkettiin NAT-muunnoksen avulla runkoverkkoon. Tietyssä mielessä ratkaisu ”tuhlasi” koko annetun 172.16.2.0/24-verkon, mutta käyttäjät eivät tästä välittäneet. Verkon rakenne selviää kuviosta 10 (sivulla 73).

Sisäverkko koostui kahdesta 192.168-alkuisesta verkosta ja vieläkin yksityisemmästä tuotekehitysverkosta. Tuotekehitysverkkoa ei rakennettu, joten reititin R&D sisälsi vain rajapintojen konfiguraatiot ja oletusyhdyskäytävän. Tästäkin huolimatta R&D-reitittimen konfiguraatio löytyy liitteestä 2 (sivulta 142).



Kuvio 10. B-verkon sisäinen rakenne

5.5.1 EdgeB-reitittimen konfigurointi

Reititin EdgeB ei sisältänyt mitään palomuureihin liittyvää NAT-sääntöjen lisäksi. NAT oli jo moneen kertaan läpi käyty osoitteen piilotus. Ainoa ero aiempiin oli se, että tällä kertaa piilotettavia verkkoja oli kolme, eikä yksi. Tämä tehtiin antamalla lähdeosoitteeksi alku- ja loppuosoite. Kaikki osoitteet niiden väliltä tuli ajaa piilotuksen läpi. Muunnettavia verkkoja olivat 192.168.0.0/24, 192.168.1.0/24 ja 192-.168.2.0/24, joten tarvittavat komennot olivat:

```
set nat source rule 10 outbound-interface eth0
set nat source rule 10 source address \
192.168.0.0-192.168.2.255
set nat source rule 10 translation address masquerade
```

Rajapinnat konfiguroitiin tuttuun tapaan:

```
set interfaces ethernet eth0 address 172.16.2.2/24
set interfaces ethernet eth1 address 192.168.0.1/24
set interfaces ethernet eth2 address 192.168.1.1/24
set interfaces ethernet eth3 address 192.168.2.1/30
```

Oletusyhdyskäytävä asetettiin osoittamaan kohti CoreB-laitetta:

```
set system gateway-address 172.16.2.1
```

Lisäksi laitteeseen luotiin kaksi DHCP-osoiteallasta verkoille B-1 ja B-2. Nämä luotiin komennoilla jotka on käyty läpi luvussa 4.5.1 (sivulla 47) ja ne löytyvät myös liitteessä 2 (sivulla 139) olevasta kokonaisesta listauksesta.

5.5.2 Piilotetun palvelimen suojaus

Verkko B-1 sisälsi, ylläpitäjien tietämättä, ”piilotetun” palvelimen. Palvelimelle oli annettu osoite aivan B-1 -verkon osoitealueen lopusta, jolloin EdgeB-reitittimen DHCP ei käytännössä koskaan jakaisi sen osoitetta. Palvelin asetettiin kuuntelemaan yhteyspyyntöjä vain verkon B-1 koneilta. Koska palvelin oli piilossa, ei tarvittavia palomuurisääntöjä voitu tehdä ”näkyville” EdgeB-reitittimeen. Tämän vuoksi tarvittava palomuuri toteutettiin itse palvelimeen pelkän iptablesin avulla.

Kuten jo luvussa 3.1.4 (sivulla 19) todettiin, ovat Vyatta Core ja ShoreWall käytännössä vain Linux-ytimen netfilter-järjestelmää ohjaavia ohjelmia. Ne konfiguroivat sen omien sääntöjensä perusteella. netfilter-järjestelmän voi konfiguroida myös käsin komentoriviltä iptablesin avulla.

iptablesia ei käydä tässä tarkemmin läpi johtuen sen laajuudesta. iptablesista saisi kirjoitettua vaikka kokonaisen kirjan, joten tässä on käyty nopeasti läpi vain muutama perusasia.

iptables käyttää apunaan **ketjuja** (engl. *chain*). Jokainen paketti kulkee yhtä tai useampaa ketjua pitkin ja ketjuun lisätyt säännöt kertovat, mitä paketille tulisi tehdä. Järjestelmä määrittää oletuksena joitain valmiita ketjuja, joihin voi vapaasti liittää omia sääntöjään. Valmiita ketjuja ovat mm. ”INPUT”, johon paikalliselle koneelle saapuvat paketit joutuvat, ja ”FORWARD”, joka sisältää reitityksen läpi menneet paketit ennen niiden eteenpäin lähetystä. Lisäksi on mahdollista luoda **tauluja** (engl. *table*), joihin ketjuja voi liittää. Esimerkiksi NAT-säännöt kuuluvat

omaan ”iptable_nat” -nimiseen tauluunsa. (Iptables tutorial chapter 6: Traversing of tables and chains 2013.)

Koska palvelin ei tehnyt reititystä, ja halusimme estää muualta kuin verkosta 192-.168.0.0/24 tulleet paketit, oli tarvittava komento seuraavanlainen:

```
iptables -I INPUT ! -s 192.168.0.0/255.255.255.0 -j DROP
```

Komento lisäsi INPUT-ketjuun säännön, joka pudotti (”-j DROP”) paketit, joiden lähdeosoite **ei** ollut verkosta 192.168.0.0/24 (”! -s 192.168...”). Huutomerkki teki säännöstä käänteisen, sillä normaalisti ”-s” (s = ”source”, eli lähde) määrittää mihin halutaan täsmäävän, ja tässä määritettiin mihin ei haluttu täsmäävän. (Iptables tutorial 10: IP range match 2013.)

Jotta luodut säännöt selviävät uudelleenkäynnistyksestä, on ne ajettava käynnistyksen yhteydessä. Komento lisättiin tiedostoon `/etc/network/interfaces` eth0-rajapinnan osion alle seuraavassa muodossa (tiedosto löytyy kokonaisena liitteestä 2, sivulta 144):

```
pre-up iptables -I INPUT ! -s 192.168.0.0/255.255.255.0 \
-j DROP
```

Näiden komentojen jälkeen palvelin ei reagoinut millään tavalla verkoista 192-.168.1.0/24 ja 192.168.2.0/24 tulleisiin paketteihin. Verkosta 192.168.0.0/24 sai siihen normaalisti yhteyden.

6 Nopeustesti

6.1 Taustaa

Puhtaat ohjelmapohjaiset reitittimet ja palomuurit eivät ole läheskään yhtä nopeita kuin laitteistopalomuurit. Lisäksi virtualisointi hidastaa niiden toimintaa entisestään. Siksi yhtenä testinä päätettiin suorittaa pienimuotoinen nopeustesti. Ajanpuutteen vuoksi mittausmenetelmät jäivät hieman vajavaisiksi, eikä niitä todellakaan voida luonnehtia tieteellisiksi. Siksi tämän luvun tulokset ovat lähinnä hieman suuntaa antavia, eivätkä ehdottomia totuuksia.

6.2 Mittauksen toteutus

Testissä mitattiin tasan sadan megatavun (104857600 tavua) kokoisen tiedoston siirtoon kulunutta aikaa ja siirtonopeutta verkon yli. Tiedosto oli luotu Debian Linuxin alla dd-komennolla kopioimalla `/dev/urandom` -satunnaislukugeneraattorin luomaa satunnaisdataa tiedostoon. Satunnaisdatalla poistettiin pakkauksen mahdollisuus. Tiedosto luotiin seuraavalla komennolla:

```
dd if=/dev/urandom of=/var/www/tiedosto bs=100M count=1
```

Lähdekoneeseen asennettiin aina Apache 2 -palvelinohjelmisto (jollei sitä jo oltu asennettu siihen), jonka kautta tiedosto kopioitiin wget-ohjelmalla. Kopioinnin ajaksi sekä lähde- että kohdekoneista sammutettiin kaikki mahdolliset muut palvelut ja ohjelmat, jotta ne eivät sotkeneet mittausta. Kopioinnit toistettiin kolme kertaa ja wget-ohjelman raportoimista siirtonopeuksista ja -ajoista laskettiin keskiarvot.

Koska mittaus ei ollut wget-ohjelmaa tarkempi, sisältävät esimerkiksi siirtoajat kiintolevyn käyttöön kuluneen ajan. Tästä olisi päästy eroon tallentamalla lähde- ja kohdetiedostot esimerkiksi RAM-levylle, mutta tätä ei ehditty tehdä. wget-oh-

jelman raportoimien siirtonopeuksien ja siirtoaikojen tarkkuutta ei saatu selville, eikä niihin löytynyt internetistä mitään viitearvoja. Niiden tarkkuudeksi arvioitiin siten yksi sekunti. Lisäksi siirtonopeuksissa on mukana IP- ja TCP-protokollien aiheuttama pieni kaistan tuhlaus.

Jotta virtualisoidun verkon nopeuksia voitaisiin vertailla edes jollain tavalla, toteutettiin siirtotesti myös kahdella fyysisellä koneella. Koneet kytkettiin toisiinsa suoraan verkkojohdolla ja nopeustesti suoritettiin niiden välillä. Toisessa mittauksessa laitteiden välille kytkettiin Telewell EA510 v3 ADSL2+ -modeemi. Näistä kahdesta fyysisestä mittauksesta johtui myös yksi testin tärkeimmistä rajoitteista: kaikki nopeustestit suoritettiin sadan megabitin nopeudella. Toisessa fyysisistä koneista, sekä ADSL-modeemissa, oli vain sadan megabitin verkkokortit (toinen fyysinen kone sisälsi gigabitin verkkokortin, mutta se toimi myös vain sadan megabitin nopeudella). Koska tuloksista haluttiin edes jollain tasolla vertailukelpoisia, muutettiin kaikkien virtuaalikoneiden asetuksia ennen mittauksia. Liitteen 1 (sivulta 94 alkaen) ohjeiden perusteella kaikissa virtuaalikoneissa oli yhden gigabitin verkkokortit. Nämä vaihdettiin nopeustestien ajaksi sadan megabitin kortteja simuloiviin Am79C973-verkkokortteihin. pfSense ei toiminut kyseisellä verkkokortilla, joten sen nopeuksia ei tässä testissä mitattu.

Lisäksi mitattiin kahta toisiinsa kytkettyä virtuaalikonetta, sekä virtuaalikonetta ja fyysistä konetta.

Mittaukset suoritettiin skenaarion 1 aikana kuvion 1 (sivulla 26) verkossa. Pakettien reitti kulki, tilanteesta riippuen, joko kahden tai kolmen reitittävän ja/tai liikennettä suodattavan laitteen läpi.

6.3 Tulokset

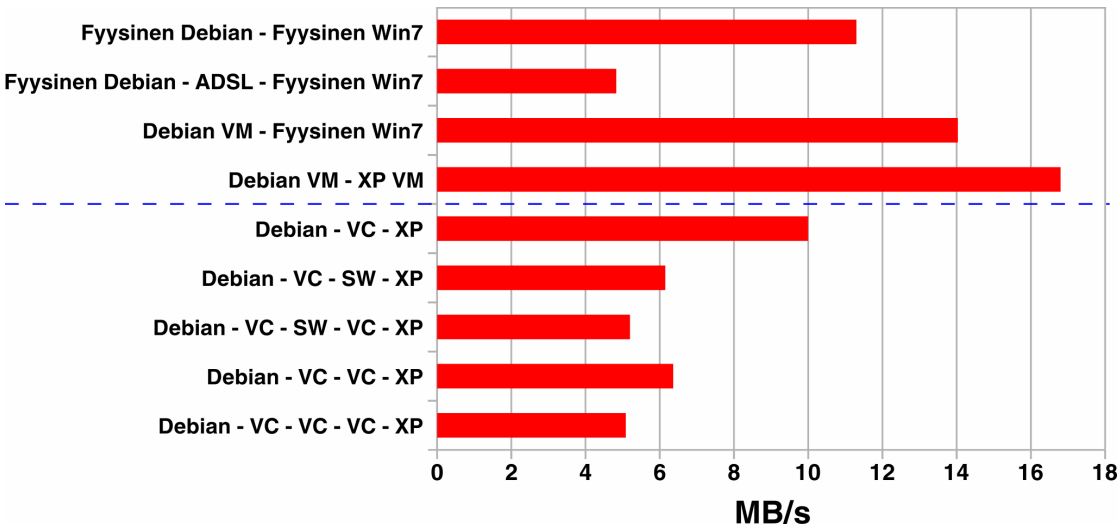
Sadan megabitin nopeudella toimivan verkon suurin teoreettinen nopeus on $(((1024 \times 1024 \times 100) \div 8) \div 1024) \div 1024 = 12{,}5$ MB/s. Tähän ei käytännössä ihan päästä, koska kyseinen nopeus olettaa, että paketeissa liikkuisi pelkkää hyötyda-

taa. Näin ei ole, vaan paketit sisältävät otsaketietoja jotka syövät osan kapasiteetista.

Taulukko 8 (sivulla 79) sisältää raa'at wget-ohjelmasta saadut numerot. Niitä ei ole pyöristetty, eikä käsitelty mitenkään. Jokaisesta kolmesta mittauksesta on annettu sekä siirtonopeus (MB/s) ja siirtoaika (s). Viimeinen sarake sisältää mittauksien keskiarvon. Kuten jo aiemmin mainittiin, pfSense jouduttiin jättämään pois tuloksista. Jotta taulukon ja kuvion 11 otsikot pysyisivät lyhyinä, lyhennettiin niissä esiintyneitä nimiä hieman. Lyhenteet ovat:

- **VC:** Vyatta Core
- **SW:** ShoreWall
- **XP:** Windows XP Professional
- **Win7**: Windows 7 Professional
- **VM:** Virtuaalikone.

Taulukko 8 on hieman hankalalukuinen, koska siinä on kolmen eri mittauskerran tulokset keskiarvoineen. Siksi siitä on poimittu kuvioon 11 se tärkein tieto, eli siirtonopeuksien keskiarvot megatavuina sekunnissa.



Kuvio 11. Siirtonopeudet graafisessa muodossa

Taulukko 8. Nopeustestin raaka numerodata

Pakettien kulkema ketju	Mittaus 1	Mittaus 2	Mittaus 3	Keskiarvo
	Nopeus (MB/s)			
	Aika (s)			
Fyysinen Debian - fyysinen Win7	11,3	11,3	11,3	11,3
	8,8	8,8	8,8	8,8
Fyysinen Debian - ADSL - fyysinen Win7	4,56	4,89	5,04	4,83
	22	22	22	22
Debian VM - fyysinen Win7	14,3	13,8	14	14,03
	7	7,2	6,9	7,03
Debian - XP	16,4	16,8	17,2	16,8
	5,9	5,9	6	5,93
Debian - VC - XP	10,1	9,92	9,97	10
	9,9	10	9,9	9,93
Debian - VC - SW - XP	6,6	5,97	5,88	6,15
	18	17	15	16,67
Debian - VC - SW - VC - XP	5,23	5,25	5,12	5,2
	20	19	19	19,33
Debian - VC - VC - XP	6,41	6,28	6,4	6,36
	16	16	16	16
Debian - VC - VC - VC - XP	4,9	5,29	5,07	5,09
	21	19	19	19,67

Kuvio 11 jakaantuu kahteen katkoviivalla eroteltuun osaan. Ylempi osa sisältää fyysisillä laitteilla tehdyt mittaukset, sekä suoraan virtuaalikoneesta toiseen tehdyn mittauksen. Alempi osa sisältää skenaarion 1 verkossa tehdyt mittaukset.

Tiedoista nähdään, että kahden fyysisen koneen välinen siirtonopeus oli 11,3 MB/s, joka oli vain reilun megatavun alle suurimman teoreettisen nopeuden. ADSL-modeemi pudottaa nopeuden puoleen ja tämän huomasi testin aikana jo ilman tarkkaa ajan mittaustakin.

Mielenkiintoisimmat tulokset kuitenkin saatiin, kun tiedostoa siirrettiin kahden virtuaalikoneen välillä suoraan, ja virtuaalikoneesta fyysiselle koneelle. Näiden testien nopeudet ylittivät verkkokorttien nopeusrajoituksen. Tälle ei löydetty mitään selitystä. Ainoa hypoteesi, joka voisi selittää valtavat nopeudet piilee ehkä VirtualBoxissa itsessään: ohjelma huomaa, että siirto on kahden virtuaalikoneen välillä ja se alkaa kopioida dataa suoraan niiden välillä, ohittaen koko verkkokortin emulaation. Vastaava saattaa tapahtua, kun kopioidaan virtuaalikoneesta fyysiselle koneelle; VirtualBox saattaa jälleen ohittaa verkkokortin. Tämä on kuitenkin puhdas hypoteesi, eikä sille löytynyt etsinnöistä huolimatta mitään tukea. Sama testi toistettiin useita kertoja, eri päivinä ja eri virtuaalikoneilla, mutta samanlainen nopeuspiikki esiintyi joka kerralla. Verkkokorttien ja ohjelman asetukset tarkistettiin useita kertoja, eikä niistä löytynyt mitään mikä olisi voinut selittää asian. Asiaan ei tämän opinnäytetyön tekemiseen käytetyn ajan puitteissa saatu vastausta.

Varsinaiset skenaarion 1 verkossa tehdyt mittaukset paljastivat kaksi mielenkiintoista seikkaa:

1. Puhtaasti virtualisoituna, verkko ylsi jopa yli kuuden megatavun sekuntinopeuteen, ja tämä usean reitittävän laitteen läpi (jotka tekivät samalla palomuuritoimintoja).

2. Laitteiden määrän kasvaessa verkon nopeus ei hidastu mainittavammin. Jopa kolmen Vyatta Core-laitteen läpi mentäessä nopeus pysyi silti viiden megatavun sekuntinopeudessa.

Yksi tuloksista, ”Debian - VC - XP” tapahtui julkiselta palvelimelta ISP-reitittimen kautta tavallista käyttäjää simuloineelle koneelle. Se ylsi noin kymmenen megatavun siirtonopeuteen. Tästä voidaan päätellä, että yksittäinen virtualisoitu reitittävä laite ei juurikaan hidasta verkkoa. Miksi ADSL-modeemi oli sitten niin hidas? Nmap-verkkoskanneri paljasti modeemin sisältävän Linuxin ytimen version 2.4; ehkä Vyatta Coren ja ShoreWall-koneen uudemmat 3.x-ytimet ovat tehokkaampia reitittämään? Tai sitten modeemin laitteisto ei vain pysynyt vauhdissa mukana.

Muita osin eri palomuurien välillä ei ole hirveästi eroja. ShoreWall on käytännössä yhtä nopea kuin Vyatta Core. Niiden pohjalla sama suodatin, joten isoja eroja ei pitäisikään olla. Pienet erot syntynevät ehkä siitä, miten järjestelmät muodostavat omista säännöistään iptablesin varsinaiset sääntöketjut. Jos samat säännöt konfiguroi sekä ShoreWalliin että Vyatta Coreen, muodostuukin ratkaisevaksi tekijäksi se kumpi niistä muuntaa ja optimoi säännöt parhaiten. Voidaankin kysyä, olisiko käsin iptablesille tehty palomuuri nopeampi kuin ShoreWall tai Vyatta Core? Koska iptablesia ei käytetty kuin yhden kerran, ei tähän kysymykseen voida esittää tässä mitään vastausta.

Testiä olisi ollut mielenkiintoinen jatkaa ja katsoa, montako laitetta olisi tarvittu, jotta nopeus olisi pudonnut esimerkiksi alle kahden megatavun. Jos numeroita katsoo, voisi niistä päätellä että jos laitteita on 6-8, nopeus saattaisi painua alle kahden megatavun. Tämäkin on puhdas arvio.

pfSensestä mainittakoon sen verran, että vaikka sitä ei mittauksissa käytetty, ei sen nopeus koskaan työn aikana muodostunut ongelmaksi. BSD-ytimen voidaan olettaa olevan riittävän nopea jo siitäkin syystä, että se on valittu pfSensen alustaksi.

Voiko sitten kotioloissa käyttää pfSenseä, ShoreWallia tai peräti Vyatta Corea palomuurinaan? Kyllä voi, mutta asia riippuu internet-yhteyden nopeudesta. Anekdootit ovat pahoja, varsinkin opinnäytetyössä, mutta tässä välissä voisi mainita kirjoittajan omakohtaisen kokemuksen Soneran 25/3-yhteydestä aivan Jyväskylän keskustassa. Suurin yhteydellä koskaan saavutettu nopeus oli n. 2,1 MB/s (keskiarvo vaihteli usein välillä 1,6 - 1,8 MB/s). Modeemi oli sama TeleWellin ADSL-modeemi, jota nopeustestissä käytettiin. Kyseisellä yhteydellä modeemin voisi huoletta asettaa siltaavaan tilaan ja kytkeä sen perään ohjelmapalomuurin; yhteys ei siitä hidastuisi. 25-megaisen yhteyden teoreettinen maksimi on kolme megatavua sekunnissa, joten tässä työssä käytetyt palomuurit eivät muodostu pullonkaulaksi, vaikka niitä olisi parikin peräkkäin.

Myös pienyrityksille, joiden yhteys ulkomaailmaan on sata megabittiä tai sen alle, voidaan puhtaan ohjelmapohjaisen palomuurin todeta olevan harkittavissa oleva asia. Se vain tarvitsee nopean koneen allensa. Jos sisäverkossa tarvitaan palomuureja, kannattaa pysyä laitteistopohjaisissa palomuureissa; ulkomaailman rajalla ohjelmapalomuuri saattaa olla täysin riittävä. Koska työssä ei huomioitu VPN-yhteyksiä, ei VPN:n aiheuttamaa kuormaa ole otettu tässä huomioon. On mahdollista, että VPN saattaa olla ohjelmapalomuurille liikaa yrityskäytössä.

Jos käytettävissä on sadan megabitin yhteys, palomuurina toimivaan koneeseen kannattaisi asentaa gigabitin verkkokortit (jotta pakettien otsakkeiden kuorma ei hidastaisi liikennettä) ja koneen tulisi muutenkin olla nopea. Tällöin palomuuri ei hidasta yhteyttä. Yli sadan megabitin yhteyksissä on jo pakko harkita laitteistopalomuurin käyttöä. Gigabitin ja sitä nopeammissa sisäverkoissa ohjelmapalomuuri on todennäköisesti liian hidas.

Puhtaasti virtualisoiduissa verkoissa, olivat ne harjoituskäytössä tai ihan oikeassa käytössä, ei yksittäinen ohjelmapalomuuri välttämättä ole niin huono vaihtoehto kuin miltä se kuulostaa. Harjoituskäytössä ohjelmapalomuuri on erinomainen ratkaisu. Varsinkin Vyatta Core on mainio isojenkin verkkojen simulointiin juuri reititysominaisuuksiensa vuoksi.

7 Pohdintaa

7.1 Tavoitteet

Opinnäytetyön tavoitteena oli tutustua avoimen lähdekoodin palomuureihin, niiden toimintaan ja konfigurointiin. Ominaisuuksia tuli vertailla ja palomuureja tuli testata erilaisissa skenaarioissa. Nämä tavoitteet saavutettiin, mutta ongelmilta ei vältytty, ja monia yllättäviä seikkoja nousi esiin työn aikana.

Alkuperäisenä tarkoituksena oli lisäksi käydä läpi VPN-tekniikat palomuurien kanssa, sekä IPv6-pohjaiset palomuuritoiminnot. Nämä kuitenkin jäivät pois ajanpuutteen vuoksi.

7.2 Mikä onnistui ja mikä ei?

Palomuurien vertailu

Vertailu onnistui lopulta ihan hyvin. Se oli aluksi hankala tehdä, koska monet palomuureista käyttivät samaa ydintä ja niiden ominaisuudet olivat identtiset. Eroja tulikin etsiä kokonaiskuvia tarkastelemalla. Niistä saatiin onneksi kokoon tarpeeksi monta asiaa, jotta niistä muodostui asiallisen kokoinen taulukko. Enemmänkin olisi saanut, mutta sisältö olisi alkanut mennä jo tarpeettomaksi nippelitiedoksi.

Skenaarioiden toteutus

Muutamista puutteista huolimatta skenaariot onnistuivat ihan hyvin. Laatikko Oy:n DMZ-alue oli mukana alusta asti ja se onnistui VPN-osiota lukuunottamatta täysin. Skenaario 2 jäi osittain vajaaksi, koska verkko B:n takana ollutta isompaa

verkkoa ei konetehon puutteen vuoksi pystytty tekemään kokonaan. Verkko B:n kanssa oli tarkoitus käyttää kiinteätä NAT-muunnosta, mutta tämä jäi pois. Lisäksi skenaarioita olisi saanut olla ainakin yksi enemmän.

Palomuurit

Palomuurien konfigurointi oli helpompaa kuin mitä ensi alkuun olisi luullut. ShoreWall näytti melko monimutkaiselta, mutta sen toiminnan ja tiedostojen logiikan sisäistämiseen ei mennyt kuin hetki. Sen jälkeen sitä pystyi konfiguroimaan melkein ohjetta lukematta. Vyatta Core oli jo tuttu JYVSECTECin aiemmista harjoituksista, joten sen kanssa ei ollut mitään ongelmia.

Valmiita palomuureja ei testattu ihan loppuun asti. Ne estivät kaikissa testeissä ei-toivotun liikenteen, mutta lisätestausta olisi tarvittu.

Nopeustesti

Nopeustesti ei ollut aivan sellainen kuin sen olisi pitänyt olla. Yksi tiedosto ei ollut riittävä ja mittausmenetelmä (wget-ohjelma) ei ollut tarpeeksi tarkka.

7.3 Tulosten luotettavuus

Ovatko saadut tulokset luotettavia? Tähän on hyvin vaikea vastata. Työn aikana opittiin paljon uusia asioita, eikä niitä kokemuksen puutteen vuoksi pystytty vertaamaan juuri mihinkään. Siksi on mahdotonta sanoa, olivatko luvun 4 (sivulla 25) DMZ-alueen konfiguraatiot täysin oikein tehtyjä. On myös mahdotonta sanoa, jäikö niistä puuttumaan jotain tärkeätä, tai jäikö niihin jokin hakkerin mentävä aukko. Jokainen luotu palomuuri testattiin useilla eri tavoilla (kuten nmap-verkkoskannerilla ja yksinkertaisesti kokeilemalla yhteyksiä kohteeseen jokaisesta verkon laitteesta) ja vertaamalla niitä valmistajien ohjeissa olleisiin esimerkkeihin. Palomuurit saivat näissä testeissä joka kerta puhtaat paperit: ne estivät sen mitä piti, ja sallivat sen mitä piti, eikä aukkoja löydetty. Pieni epäilys jäi silti aina kaiva-

maan. Palomuurit eivät olleet tekijälle täysin tuntemattomia, mutta niitä ei oltu koskaan konfiguroitu näin laajasti.

Vastaus luotettavuuskysymykseen on "todennäköisesti, mutta lue ohjeet ja käy konfiguraatiot läpi ennen niiden käyttöön ottoa ja testaa ne huolellisesti".

Nopeustesti on, kuten jo useaan kertaan todettiin, vain suuntaa antava. Sanomattakin on selvää, että palomuurit hidastavat aina verkon toimintaa (kaikki reitittävät laitteet hidastavat verkkoa), olivat ne ohjelmapohjaisia tai laitteistopohjaisia. Nopeustestiä ei todellakaan suoritettu tieteellistä menetelmää noudattaen. Mittausväline oli epätarkka, eikä kahden fyysisen koneen välillä tapahtunutta testiä voi luonnehtia edes sen kalibroinniksi. Virtualisoinnin tuomaa hidastusta ei mitattu eikä pystytty poistamaan tuloksista. Tuloksista näkyi silti, että verkon nopeus putosi noin puoleen palomuurittoman verkon nopeudesta. Fyysisillä laitteilla tehtynä hidastus ei todennäköisesti olisi tätä luokkaa.

7.4 Jatkokehitysideoita

Skenaariot ja testit

Työtä alun perin suunnitellessa oli tarkoituksena tehdä 3-4 eri skenaariota, joissa palomuureja olisi voinut testata ja konfiguroida. Ajanpuutteen vuoksi skenaarioiden määrä kutistui pelkkään kahteen ja toinenkin niistä kutistui suunnitelmastaan. Alkujaan skenaarion 2 verkko B:n takaa löytyi isompi, vähintään 3-4 reititintä käsittänyt verkko, mutta tämä jouduttiin muokkaamaan pienemmäksi konetehon puutteen vuoksi.

Skenaarioita olisi pitänyt olla useampia, ja jo olemassa olevien olisi pitänyt olla isompia. Skenaariosta 1 pois jäänyt VPN-osio olisi pitänyt tehdä joko skenaarion kanssa, tai sitten kokonaan omana skenaarionaan. Myös IPv6-pohjainen verkko

olisi pitänyt rakentaa ja palomuureja olisi pitänyt testata siinä. IPv6-testin verkko suunniteltiin, mutta sitä ei ehditty toteuttaa.

Skenaariosta 1 olisi voinut tehdä variaation, jossa Laatikko Oy sai dynaamisen julkisen osoitteen. Toinen variaatio olisi ollut kokonaisen julkisen aliverkon käyttö. Kummallekin variaatiolle tehtiin toimivat konfiguraatiot Vyatta Corelle, mutta niitä ei ajanpuutteen vuoksi käyty läpi.

Paremmat palomuurien testaukset

Palomuureja olisi pitänyt testata hieman enemmän. Nmap-skanneri oli hyvä aloitus, mutta verkkoon olisi pitänyt luoda enemmänkin testiliikennettä. On olemassa palomuurien testaukseen tarkoitettuja ohjelmia, kuten hping (www.hping.org) ja nemesis (nemesis.sourceforge.net), joilla voi yrittää sotkea palomuurien toimintaa, mennä niistä läpi, ja esimerkiksi injektoida tahallaan vääriä reititysprotokollien viestejä verkkoon. Näitä työkaluja ei ehditty kokeilla ajanpuutteen vuoksi, mutta jatkossa nämä tulisi ottaa käyttöön.

Tilallisiin palomuureihin on mahdollista puhkoa reikiä, joten olisi mielenkiintoista rikkoa jokin tehdyistä skenaarioista, ja pyrkiä ymmärtämään mitä tapahtui, ja miten aukon voi paikata tulevaisuudessa (TCP hole punching n.d.). Olisi asiallista myös yrittää etsiä aukkoja ja ohjelmointivirheitä itse palomuuriohjelmista ja jopa yrittää niiden avulla kaapata koko laite haltuunsa.

Kaikki työssä käytetyt palomuurit olivat aina reitittävässä tilassa. On kuitenkin mahdollista luoda sillatussa tilassa toimiva palomuuri. Tällöin laite toimii kuin kytkin, eivätkä verkon muut laitteet näe sitä. Siltaava palomuuri kuitenkin näkee kaiken sen läpi kulkevan liikenteen. Siltaavaa palomuuria ei käytetty työssä, mutta sen rakentaminen ja toiminta on selitetty lyhyesti liitteessä 3 (sivulla 145), mikäli sellaisen haluaa rakentaa. Sillattuja palomuureja tulisi tutkia lisää ja niitä tulisi käyttää skenaarioissa.

Muita palomuurituotteita ja kaupalliset tuotteet

Kolme eri palomuuria ei ole paljoa, sillä enemmänkin olisi tarjolla. Laajalti tunnettu SmoothWall Express oli alkujaan listalla ehdokkaana, mutta sitä kokeiltaessa tajuttiin sen olevan sopimaton työhön. Se oli liian helppokäyttöinen ja liian pelkistetty, jotta sillä olisi voinut tehdä skenaarioissa useita esitettyjä asioita. Testiin olisi tarvittu ohjelman maksullinen versio.

Olisi myös mielenkiintoista toteuttaa esimerkiksi skenaarion 1 palomuurit esimerkiksi OpenBSD:llä käyttäen raakaa pf-pakettisuodatinta.

Vastaavia skenaarioita olisi ollut mielenkiintoista kokeilla oikeilla reitittimillä ja laitteistopalomuureilla, ja tutkia niiden toimintojen eroja. Oikeilla laitteilla tehdyn verkon suorituskykyä olisi mukava mitata (paremmilla menetelmillä) ja verrata sitä työn tulosten kanssa. Ciscon ja Juniperin sivuja selatessa löytää helposti reitittimiä, joiden kohdalla mainostetaan monen gigabitin reititysnopeutta (Cisco 4400 Series Integrated Services Routers n.d.). Olisi mielenkiintoista nähdä mitä tapahtuu, jos niihin asettaa palomuurin päälle. Ei olisi myöskään yhtään huono idea testata itsenäisiä kaupallisia ohjelmapalomuureja ja laitteistopalomuureja.

Nopeustesti

Nopeustesti tulisi ehdottomasti uusia parempia menetelmiä käyttäen. Ainakin seuraavat asiat tulisi ottaa huomioon:

- Mittauksessa tulisi käyttää ohjelmaa, joka oikeasti mittaa pelkkää tiedonsiirtoon kulunutta aikaa. Kiintolevyn yms. käyttöön kuluva aika pitäisi poistaa tuloksista.
- Ajan resoluution tulisi olla tarkempi kuin yksi sekunti.
- Vertailudataa tarvitaan ehdottomasti enemmän.
- Samat testit tulisi ajaa fyysisillä koneilla, ei pelkästään virtualisoiduilla.

- Testeissä tulisi käyttää sekä 100 Mbit/s, että 1 Gbit/s nopeutta. Jos mahdollista, jopa 10 Gbit/s nopeutta tulisi testata. Suurilla nopeuksilla olisi mielenkiintoista tutkia, saako ohjelmapalomuurista tarpeeksi nopeita niille.
- Koneita pitäisi olla ketjussa enemmän kuin kolme.
- Enemmän testimateriaalia. Yksi tiedosto ei riitä ja tiedostojen pitäisi olla isompia.
- Pitäisi tutkia, miten paljon suodatuslistojen pituudet ja tarkistettavien asioiden määrä vaikuttavat nopeuteen.

Muita kehitettäviä asioita

VirtualBoxia tulisi tutkia lisää, ja pyrkiä selvittämään kahden virtuaalikoneen välillä tapahtuneen kopioinnin valtavan nopeuden syy. Siirtonopeus ylitti verkon nopeuden. Tälle ei työn aikana löytynyt mitään selitystä.

Virtualisoinnissa käytettävän koneen tulisi olla tehokkaampi. Muistia olisi hyvä olla vähintään 12-16 GB ja enemmänkin. Työssä käytetyn koneen neljän gigatavun muistilla suurin ajettavien koneiden määrä vaihteli 9-12 välillä ja tämä senkin jälkeen, kun virtuaalikoneiden muistien määrät oli pudotettu niin alas kuin mahdollista (osaa Debian-palvelimista ajettiin peräti 64 megatavulla).

Aika on aina ongelma. Ajanpuutteen vuoksi työstä jouduttiin leikkaamaan osia pois. Siksi aikaa olisi hyvä varata näille asioille tarpeeksi.

7.5 Ohjelmapalomuurien tulevaisuus

Onko ohjelmapalomuureilla tulevaisuutta? Kyllä on. Useimmat yhteydet eivät ole niin nopeita, etteikö ohjelmapalomuuri pysyisi mukana niiden vauhdissa. Useat käytössä olevat käyttöjärjestelmät, kuten Windows, eri Linux-jakelut ja Mac OS X

sisältävät kaikki jonkinlaisen ohjelmapalomuurin sisäänrakennettuna, ja ne ovat usein oletuksena päällä.

Ohjelmapohjaiset palomuurit ovat näiden lisäksi melko laajassa käytössä, sillä melkein jokainen kotikäytössä oleva modeemi sisältää sellaisen (The Differences and Features of Hardware and Software Firewalls 2011.). Laitteistoa ei ole käytännössä kuin Ethernet-portit, prosessori ja muistia. Loppu tapahtuu suurimmaksi osaksi ohjelmallisesti.

Periaatteessa pienet yritykset voisivat käyttää (ja ehkä jotkut käyttävätkin?) kalliiden laitteistopalomuurien tilalla puhdasta ohjelmapalomuuria verkkojen rajalla. Ongelmaksi saattavat muodostua VPN-yhteydet, jotka voivat olla hyvinkin raskaita, riippuen käyttäjien määrästä. Jos yritys on todellakin pieni, esimerkiksi muutama kymmenen ihmistä, voi puhdas ohjelmapalomuuri olla oikeasti toimiva ratkaisu.

Virtuaalisissa harjoitusverkoissa ohjelmapalomuurit ovat ohittamattomia. Kuten jo aiemmin mainittiin, on Vyatta Core lähes ylivoimainen kyseisissä verkoissa juuri reititysominaisuuksiensa vuoksi. Niillä voi simuloida hyvinkin isoja verkkoja, ja nopeus on silti aivan riittävä.

Ohjelmapohjaiset palomuurit ovat hieman ristiriitaisia tuotteita. Ne ovat erittäin laajalti konfiguroitavissa ja ne saa tekemään melkein mitä tahansa suodatusta, mutta tämä tulee aina nopeuden kustannuksella. Laitteistopalomuurit saattavat olla kankeampia, mutta vastaavasti ne ovat nopeampia. Rahalla toki saa lisää ominaisuuksia ja nopeutta. Ohjelmapalomuurien etuna on myös niiden hinta: Linux ja BSD ovat ilmaisia. Ainoaksi hinnaksi muodostuu kone, jossa niitä ajetaan.

7.6 Tuloksien hyödyntäminen

Tilallinen palomuuri on tilallinen, oli se ohjelmallinen tai ei. Käytännössä kaikki työssä esitetyt skenaariot ja konfiguroinnit voidaan muokata jollekin toiselle laitteelle sopivaksi. Komennot muuttuvat, mutta ideat ja käsitteet pysyvät samoina. Siksi työn tulokset ovat ainakin joltain osin yleisiä ja käyttökelpoisia kenelle tahansa palomuurien kanssa puuhastelevalle. Ehkä joku saattaa löytää esitetyistä skenaarioista pohjan omille, isommille ja monimutkaisemmille, skenaarioilleen.

Lähteet

Basic Routing. 2013. Vyatta Inc. Viitattu 2.9.2013.
http://www.vyatta.org/documentation

Basic System Configuration. 2013. Vyatta Inc. Viitattu 19.8.2013.
http://www.vyatta.org/documentation

Berkeley Software Distribution. N.d. Wikipedia. Viitattu 20.9.2013.
https://en.wikipedia.org/wiki/Berkeley_Software_Distribution

Bridging Reference Guide. 2013. Vyatta Inc. Viitattu 5.11.2013.
http://www.vyatta.org/documentation

Cheswick, R., Bellowin, S. & Rubin A. 2003. Firewalls and Internet Security, Second Edition. Addison-Wesley.

Cisco 4400 Series Integrated Services Routers. N.d. Cisco. Viitattu 28.9.2013.
http://www.cisco.com/en/US/products/ps12522/index.html

DMZ N.d. Wikipedia. Viitattu 25.9.2013.
https://en.wikipedia.org/wiki/DMZ_%28computing%29

Firewall Debate: Hardware vs. Software. 9.6.2011. Small Business Computing. Viitattu 25.9.2013.
http://www.smallbusinesscomputing.com/webmaster/article.php/3103431/Firewall-Debate-Hardware-vs-Software.htm

Firewall reference guide. 2013. Vyatta Inc. Viitattu 19.8.2013.
http://www.vyatta.org/documentation

Firewall Rule Basics. 2012. pfSense wiki. Viitattu 7.7.2013.
https://doc.pfsense.org/index.php/Firewall_Rule_Basics

Gateway Settings. 2011. pfSense wiki. Viitattu 6.7.2013.
https://doc.pfsense.org/index.php/Gateway_Settings

History of Linux. N.d. Wikipedia. Viitattu 20.9.2013.
https://en.wikipedia.org/wiki/History_of_linux

How can I forward ports with pfSense? 2011. pfSense wiki. Viitattu 6.7.2013.
https://doc.pfsense.org/index.php/How_can_I_forward_ports_with_pfSense%3F

Interface Settings. 2012. pfSense wiki. Viitattu 5.7.2013.
https://doc.pfsense.org/index.php/Interface_Settings

IP Network Address Translation Protocol. 2005. Viitattu 20.9.2013.
http://www.tcpipguide.com/free/t_IPNetworkAddressTranslationNATProtocol.htm

iptables. N.d. Wikipedia. Viitattu 23.9.2013.
https://en.wikipedia.org/wiki/Iptables

Iptables tutorial. 2013. Oskar Andersson. Viitattu 9.9.2013.
http://www.frozentux.net/iptables-tutorial/iptables-tutorial.html

LAN Interfaces. 2013. Vyatta Inc. Viitattu 19.8.2013.
http://www.vyatta.org/documentation

NAT. 2013. Vyatta Inc. Viitattu 19.8.2013.
http://www.vyatta.org/documentation

OSPF. 2013. Vyatta Inc. Viitattu 2.9.2013.
http://www.vyatta.org/documentation

pfSense. N.d. pfSense Open Source Firewall Distribution. Viitattu 20.9.2013.
http://pfsense.com/

Services. 2013. Vyatta Inc. Viitattu 19.8.2013.
http://www.vyatta.org/documentation

ShoreWall. 2013. Shoreline Firewall. Viitattu 20.9.2013.
http://www.shorewall.net/

shorewall-interfaces. N.d. Viitattu 18.8.2013.
http://www.shorewall.net/manpages/shorewall-interfaces.html

shorewall-masq. N.d. Viitattu 18.8.2013.
http://www.shorewall.net/manpages/shorewall-masq.html

shorewall-policy. N.d. Viitattu 18.8.2013.
http://www.shorewall.net/manpages/shorewall-policy.html

shorewall-rules. N.d. Viitattu 18.8.2013.
http://www.shorewall.net/manpages/shorewall-rules.html

shorewall-zones. N.d. Viitattu 18.8.2013.
http://www.shorewall.net/manpages/shorewall-zones.html

Stateful Firewall. N.d. Wikipedia. Viitattu 23.9.2013.
https://en.wikipedia.org/wiki/Stateful_firewall

Static Routes. 2009. pfSense wiki. Viitattu 6.7.2013.
https://doc.pfsense.org/index.php/Static_Routes

TCP hole punching. N.d. Wikipedia. Viitattu 27.9.2013.
https://en.wikipedia.org/wiki/TCP_hole_punching

The Differences and Features of Hardware and Software Firewalls. 7.4.2011. Webopedia. Viitattu 25.9.2013.
http://www.webopedia.com/DidYouKnow/Hardware_Software/2004/firewall_types.asp

The Open Source Definition. N.d. Open Source Initiative. Viitattu 20.9.2013.
http://opensource.org/osd

Understanding Zone Based Firewalls. 2011. Cisco Skills. Viitattu 25.9.2013.
http://ciscoskills.net/2011/03/18/understanding-zone-based-firewalls/

VBoxManage. 2013. Oracle Corporation. Viitattu 30.8.2013.
http://www.virtualbox.org/manual/ch08.html

Vyatta. 2012. Vyatta Inc. Viitattu 25.9.2013.
http://vyatta.org/

WebMin: Standard Modules. N.d. Viitattu 18.8.2013.
http://www.webmin.com/standard.html

What is ShoreWall? 2013. Shoreline Firewall. Viitattu 20.9.2013.
http://www.shorewall.net/Introduction.html

Wikipedia: Firewall. N.d. Wikipedia. Viitattu 25.9.2013
https://en.wikipedia.org/wiki/Firewall_%28computing%29

Liite 1: Asennusohjeita

Tämä liite sisältää asennusohjeita työssä käytettyihin ohjelmiin. Ohjeet eivät suinkaan ole tyhjentävät, oikein tehdyt, tai ainoat mahdolliset asennustavat. Ne vain kertovat täsmälleen, miten mainitut ohjelmat asennettiin tämän työn tekoa varten.

Ohjeet on kirjoitettu VirtualBoxille, mutta vastaavia virtuaalikoneen asetuksia käyttämällä niiden pitäisi asentua myös muiden virtualisointiohjelmien alle, ja myös fyysisiin koneisiin.

pfSense

pfSensen versio 2.0.3 asennettiin taulukon 9 mukaisiin koneisiin. Syyskuussa 2013 julkaistu versio 2.1.0 ei ehtinyt työhön mukaan.

Taulukko 9. pfSense-koneiden asetukset

Asetus	Tyyppi/määrä
Muisti (RAM)	192 MB
Kiintolevyn koko	1 GB
Verkkorajapintojen määrä	2-4 kappaletta, kaikkien tyyppinä ”Intel PRO/1000 MT Desktop”
Muut asetukset	Kaikki virtuaalikoneen asetukset jätettiin oletustiloihinsa, paitsi seuraavat kohdat: • Ei ääntä • Ei USB-ohjaimia

Itse asennus tapahtuu näin:

1. Aloita asennus valitsemalla ensimmäisestä valikosta kohta 1 (”*Boot pfSense [default]*”). Jos asennusohjelma kysyy, haluatko asentaa (I) vai pelastaa jo olemassa olevan järjestelmän (R), vastaa asennus.

2. Seuraava valikko kysyy mahdolliset näppäinkartat ja näytön fontit. Näillä ei ole juurikaan väliä, koska ne ovat käytössä vain asennuksen ajan. Niitä voi vaihtaa jos haluaa, mutta oletuksetkin toimivat. Jatka valitsemalla ”*Accept these Settings*”.

3. Valitse seuraavasta ”*Select Task*”-valikosta kohta ”*Install pfSense*”. Asennusohjelma kysyy kohdekiintolevyn. Oikea levy on jo valittuna, mikäli levyjä ei ole kuin yksi. Valitse levy ja paina Enter.

4. Ohjelma kysyy, halutaanko kiintolevy alustaa. Alustusta ei ole pakko tehdä, jos levyllä on jo toimiva asennus, mutta on suositeltavaa aloittaa aina puhtaalta pöydältä. Valitse siis ”*Format this Disk*”. Seuraavaksi kysytään levyn koko vanhaan kunnon tapaan sylintereinä, päinä ja sektoreina. Näiden pitäisi olla oikein jo oletuksena. Valitse nuolilla ”*Use this Geometry*” ja paina Enter jatkaaksesi. Ohjelma kysyy vielä varmistuksen alustukselle, kuittaa varoitus painamalla Enter.

5. Seuraavaksi kysytään, halutaanko levy osioida (”*Partition Disk?*”). Koska levylle ei asenneta muita käyttöjärjestelmiä, valitse ”*Skip this Step*”.

6. ”*Install Bootblock(s)*”-valikko kysyy mihin osioihin käynnistyslohko asennetaan. Tähän käyvät oletusvalinnat. Valitse siis ”*Accept and Install Bootblocks*” jatkaaksesi.

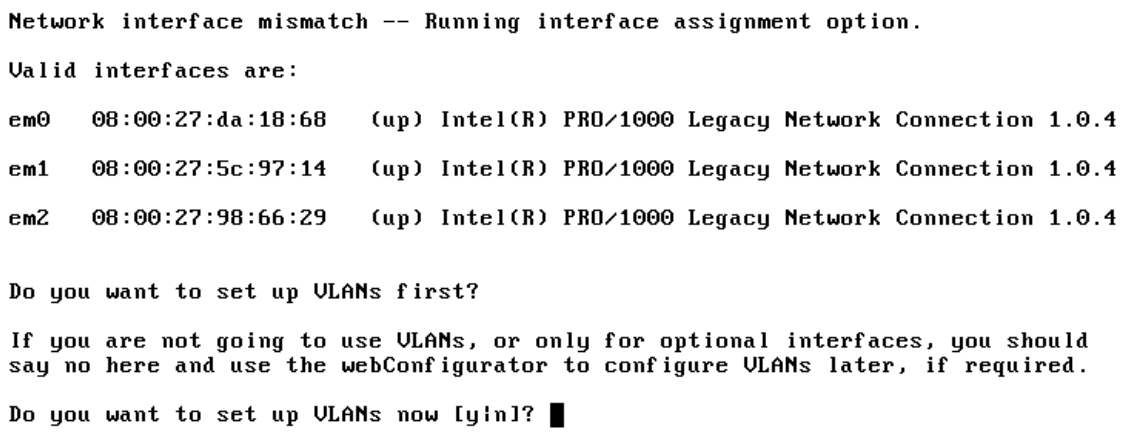
7. Seuraavaksi ohjelma kysyy varsinaisen osion, jolle pfSense asennetaan (”*Select a Partition*”). Koska osioita on vain yksi, tähän riittää pelkkä Enterin painallus. Ohjelma kopioi käynnistystiedostot ja kysyy seuraavaksi aliosioi-

den tietoja (”*Select Subpartitions*”). Tähänkin riittää kun valitsee ”*Accept and Create*”. Ohjelma kopioi lopultakin järjestelmän tiedostot levylle.

8. Seuraavaksi asennus tiedustelee halutun ytimen tietoja. Tähän käy valinta ”*Symmetric multiprocessing kernel*”.

9. Valitse lopuksi ”*Reboot*”. Poista tallennusväline asemasta ja kone käynnistyy uudelleen. Tässä vaiheessa ruudulla vilahtaa järjestelmän oletustunnus ja salasana (admin/pfsense). Nämä on hyvä ottaa ylös.

pfSense käynnistyy normaalisti ja ensimmäisellä kerralla se havaitsee verkkokorttien konfiguraation puuttuvan (”*Network interface mismatch*”). Se esittää kuvion 12 kaltaisen valikon.



Kuvio 12. Rajapintojen liittäminen pfSenseen

Rajapintojen liittäminen tapahtuu seuraavasti:

1. Ensimmäinen kysymys tiedustelee, halutaanko luoda VLANeja. Vastaa ”n”, koska VLANit eivät ole käytössä.

2. Seuraavaksi kysytään, mikä listassa mainituista rajapinnoista on WAN-rajapinta, eli mikä niistä on kytketty ulkomaailmaan. Tässä työssä pfSensen WAN-rajapinta oli aina ”em0”. Kirjoita rajapinnan nimi ja paina Enter.

3. Seuraavaksi kysytään LAN-verkon rajapinta, eli rajapinta johon sisäverkko on kytketty. Tässä työssä käytettiin aina ”em1”-rajapintaa.

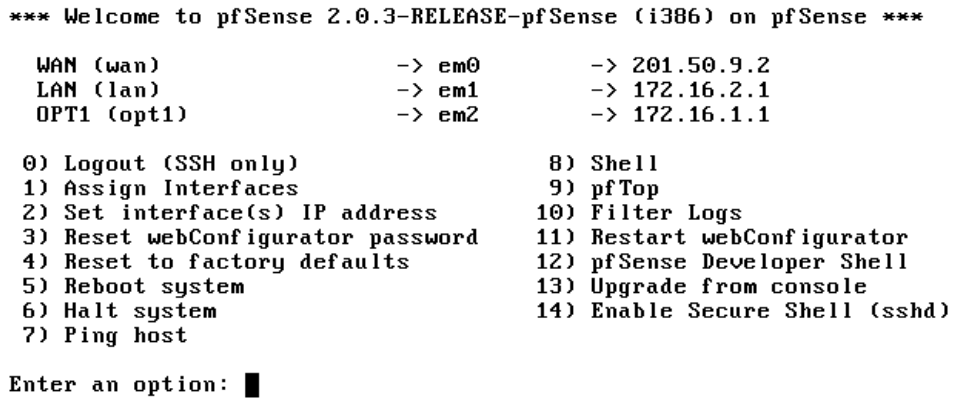
4. Ohjelma kysyy vielä kahden valinnaisen rajapinnan (OPT1 ja OPT2) nimeä. Kun käytössä oli DMZ-alue, kirjoitettiin tähän ”em2”. Jos rajapintoja ei ollut kuin kaksi, voi tämän jättää tyhjäksi, jolloin asennus jatkuu.

5. Ohjelma näyttää nyt listan tehdyistä valinnoista. Jos ne ovat oikein, vastaa ”y” jatkaaksesi. Asennusohjelma luo nyt alkuperäisen konfiguraation.

Tässä kohtaa on mainittava muutama hieman ärsyttävä seikka. Asennusohjelma olettaa, että WAN-rajapinnan toisessa päässä on käytössä DHCP-palvelin. Jos näin ei ole, jää ohjelma jauhamaan ”*Configuring WAN interface...*”-kohtaan turhankin pitkäksi aikaa. Tässä ei voi tehdä muuta kuin odotella. Kiinteän IP-osoitteen voi määrittää, mutta ei ennekuin ohjelma on vähintään kerran käynyt odottelemassa DHCP:tä.

Toinen vastaava tapahtuu hetkeä myöhemmin kun ohjelma jää etsimään NTP-palvelinta. Jos WAN-rajapintaa ei ole vielä kytketty kiinni ulkomaailmaan (tai se ei saanut DHCP:ltä osoitetta), ei NTP-palvelimeen luonnollisesti saada yhteyttä. Tämä vaihe kestää vieläkin kauemmin (n. 5 minuuttia) kuin DHCP-vaihe, eikä sitä voi keskeyttää mitenkään. Ei voi kun odottaa. Jos WAN-rajapinnan takana on DHCP ja oikeasti toimiva yhteys ja NTP-palvelin toimii, jäävät kummatkin odotukset muutamaan sekuntiin.

Lopuksi ohjelman päävalikko kuitenkin tulee esiin (kuvio 13).



Kuvio 13. pfSensen tekstipohjainen päävalikko

Tästä valikosta pfSensen voi sammuttaa, pistäytyä komentorivillä, tai muuten vain tarkkailla ohjelman toimintaa. Sitä kautta voi suorittaa joitain peruskonfiguraatioita. Jotta selaimen kautta toimiva konfigurointiohjelma toimisi, on LAN-rajapinnalle annettava IP-osoite. Oletusosoite on 192.168.1.1, mutta jos tämä ei kelpaa, valitse valikosta kohta 2, ”*Set interface(s) IP address*”. Ohjelma kysyy, mikä rajapinta halutaan konfiguroida, valitse LAN. Syötä sitten haluttu IP-osoite ja etuliitteen pituus (esimerkiksi 24). Jos LAN-rajapintaan haluaa DHCP-palvelimen, tulee syöttää aloitus- ja lopetusosoitteet niitä kysyttäessä.

Nyt pfSensen varsinaiseen konfiguraatio-ohjelmaan pääsee käsiksi sisäverkosta ottamalla selaimella salattu yhteys LAN-rajapinnan IP-osoitteeseen (esimerkiksi https://192.168.1.1). Jos selain varoittaa epäluotettavasta yhteydestä, kuittaa varoitus (pfSense käyttää itse allekirjoitettua varmennetta, joten varoitus on harmiton). Kirjaudu sisään oletustunnuksilla (käyttäjä ”admin”, salasana ”pfsense”). Ensimmäisellä kerralla pfSense ajaa selaimessa lyhyen konfiguraatiovelhon. Velho kyselee mm. DNS-palvelimien osoitteet, verkkotunnuksen, sekä aikapalvelimen (NTP) osoitteet. Näiden jälkeen pfSense käynnistyy uudelleen. Tämän jälkeen pfSense on valmis käytettäväksi.

ShoreWall

Tässä työssä kaikkien käytettyjen ShoreWall-koneiden laitteisto konfiguroitiin taulukon 10 mukaisesti. Alustana ollut käyttöjärjestelmä asennettiin Debianin asennusohjeiden (sivulla 103) mukaan.

Taulukko 10. ShoreWall-koneiden asetukset

Asetus	Tyyppi/määrä
Muisti (RAM)	128 MB
Kiintolevyn koko	1 GB
Verkkorajapintojen määrä	2-4 kappaletta, kaikkien ohjaimena ”Intel PRO/1000 MT Desktop”
Käyttöjärjestelmä	Debian 7 (Wheezy)
Muut asetukset	Kaikki virtuaalikoneen asetukset jätettiin oletustiloihinsa, paitsi seuraavat: • Ei ääntä • Ei USB-ohjaimia

ShoreWall ei ole erillinen Linux-jakelu, vaan tavallinen ohjelma, jota ajetaan koneessa. Siksi se asennetaan kuten mikä tahansa muukin ohjelma. Alustana käytettyyn Debian 7-koneeseen ShoreWall asentuikin helposti komennolla:

```
apt-get install shorewall
```

Kirjoitushetkellä komento asensi ShoreWallin version 4.5.5.3-3, joka oli hieman jäljessä ohjelman kotisivulla työn kirjoitushetkellä saatavilla olleesta versiosta 4.5-.18. ShoreWall on heti asennuksen jälkeen valmis konfiguroitavaksi.

Jos kohdekone on verkossa, josta ei ole yhteyttä internetiin, on yhteys joko hetkellisesti järjestettävä muuta kautta, tai asennustiedostot on siirrettävä koneeseen käsin jollain tavalla (esimerkiksi USB-tikku), ja asennuskin on mahdollisesti tehtävä käsin. ShoreWallin kotisivulla on asennusohjeet eri Linux-jakeluille.

Vyatta Core

Kaikki käytetyt Vyatta Core-koneet asennettiin taulukon 11 mukaisiin virtuaalikoneisiin.

Taulukko 11. Vyatta Core-koneiden asetukset

Kohta	Tyyppi/määrä
Muisti (RAM)	256 MB (toimi myös 128 megatavulla)
Kiintolevyn koko	1 GB
Verkkorajapintojen määrä	2-4 kappaletta, kaikkien ohjaimena ”Intel PRO/1000 MT Desktop”
Muut asetukset	Kaikki virtuaalikoneen asetukset jätettiin oletustiloihinsa, paitsi seuraavat: • Ei ääntä • Ei USB-ohjaimia

Työssä käytetty versio 6.6-R1 asentui seuraavasti:

1. Ensimmäiseen ”*boot:*” -kyselyyn riittää pelkkä Enterin painallus.

2. Järjestelmä käynnistyy livecd-tilassa. Kun login-ruutu tulee näkyviin, kirjaudu sisään oletustunnuksilla ”vyatta”/”vyatta”.

3. Käynnistä asennus komennolla

   ```
   install system
   ```

   Vastaa varmistukseen ”*yes*”.

4. Asennusohjelma kyselee osioita, kohdekiintolevyä, levykuvan nimeä, konfiguraatiotiedoston nimeä ja vaikka mitä muuta. Lähes kaikki näistä menevät sellaisenaan, eli niihin riittää pelkkä Enterin painallus. Ainoa missä oletus

ei kelpaa, on viimeinen varmistus ennen kiintolevyn alustamista. Tähän on vastattava ”*Yes*”, muutoin asennus keskeytyy.

5. Jossain vaiheessa asennusohjelma kysyy järjestelmän vyatta-käyttäjän salasanaa. Tässä täytyy huomioida se, että näppäimistön ulkoasu on amerikkalainen. Esimerkiksi salasanaa ”abcd-1234” käytettäessä väliviiva tulee eri näppäimestä kuin suomalaisella näppäimistöllä. Näppäimistön ulkoasun voi vaihtaa, mutta ei vielä tässä vaiheessa.

6. Kun asennusohjelma on valmis, käynnistä kone uudelleen ja poista asennus-CD järjestelmästä.

Kun laite on käynnistynyt uudelleen, voidaan se konfiguroida loppuun. Kirjaudu sisään uuteen järjestelmään käyttäjätunnuksella ”vyatta” ja asennuksen aikana määrittämälläsi salasanalla.

Järjestelmä on nyt käyttövalmis, seuraavia valinnaisia asioita lukuun ottamatta.

Sarjaporttikonsolin poistaminen

Saatat nähdä virheilmoituksen ”*INIT: Id ”TO” respawning too fast: disabled for 5 minutes*” asennuksen ja käytön aikana. Tästä ei tarvitse huolestua. Ilmoitus liittyy sarjaportin takaa löytyvään konsoliin, mutta virtuaalikoneissa ei sarjaportteja useinkaan ole, joten järjestelmä valittaa asiasta turhaan. Siitä pääsee onneksi eroon seuraavalla komennolla:

```
delete system console
```

Saat varoituksen ”*Warning: access to system console is unconfigured*”. Tästä ei tarvitse välittää. Virtuaalikoneissa ei sarjaporttikonsoleilla tee mitään.

Näppäimistön ulkoasu

Suomalaisen näppäimistökartan saa käyttöön seuraavalla komennolla:

```
sudo dpkg-reconfigure keyboard-configuration
```

Tämän jälkeen valikoista valitaan haluttu ulkoasu. Suomalaisen näppäimistön saa seuraavilla valinnoilla:

1. Generic 105-key (Intl) PC
2. Other
3. Finland
4. Finland - Classic
5. The default for the keyboard layout
6. No compose key

Näiden jälkeen on kone käynnistettävä uusiksi. Ääkköset eivät toimi tämänkään jälkeen, mutta muut erikoismerkit (?, /, jne.) ovat nyt oikeilla paikoillaan.

Muita mukavia pikkuasioita

Oikean aikavyöhykkeen saa komennolla

```
set system time-zone Europe/Helsinki
```

SSH:n saa päälle komennolla

```
set service ssh
```

Sekalaisia huomautuksia

Järjestelmän pääkonfiguraatiotiedosto on `/config/config.boot`. Sitä voi muokata vaikkapa nano-editorilla. Jotkut asiat, kuten rajapintojen IP-osoitteet ja MAC-osoitteiden korjaus (jota tarvitaan jos olemassa olevan järjestelmän kloonaa) käy nopeimmin, kun tiedostoa muokkaa suoraan. Varmuuskopioiden otto ennen muokkausyrityksiä on erittäin suositeltavaa.

Jos Vyatta Core-laitteen kloonaa ja rajapintojen MAC-osoitteet luo uusiksi, ”hukkaa” järjestelmä ne. Vanhat rajapinnat ovat yhä konfiguraatiotiedostossa, mutta niiden MAC-osoitteet eivät täsmää uusiin. Siksi tiedostoa `/config/config.boot` on muokattava käsin ja uudet eth4/eth5/yms. rajapinnat on poistettava, ja uudet MAC-osoitteet tulee kirjoittaa vanhojen tilalle. Tämän jälkeen järjestelmä käynnistetään uudelleen ja vanhat rajapinnat toimivat taas.

Kannattaa muistaa, että Vyatta Core on pohjimmiltaan Debian Linux. Komentorivi **on** muokattu bash-komentorivi. Lähes kaikki normaalit bashin kustomoinnit toimivat myös Vyatta Coressa. Järjestelmän voi siis halutessaan muokata niin värikkääksi kuin haluaa. Siihen voi myös asentaa Debianin deb-paketteja. Tätä varten kannattaa lukea Vyatta Coren omat ohjetiedostot, joissa asia on selitetty. Vyatta Coren mukana tulee valmiina Python-tulkki ja Perl-moduuli konfiguraatiotiedostojen lataamiseen, muokkaamiseen ja tallentamiseen ohjelmallisesti.

Minimaalinen Debian 7-asennus

Kaikki työssä käytetyt palvelimet olivat Debian 7-jakelun päälle rakennettuja. Lisäksi ShoreWall-palomuuri oli myös Debian-pohjainen. Tämä liite sisältää lyhyet ohjeet siihen, miten Debian 7-jakelusta tehdään minimalistinen asennus. Nämä **eivät** ole seikkaperäiset Debianin asennusohjeet!

Jokainen työssä käytetty Debian-kone kloonattiin samasta virtuaalikoneesta, joka oli asennettu Debian netinstall-menetelmällä. Tämä tarkoittaa, että Debianin kotisivuilta (www.debian.org) kopioitiin pieni netinstall-asennus-CD, joka sitten käynnistettiin virtuaalikoneessa. Minimalistinen asennus tehtiin näin:

1. Valitse käynnistysvalikosta ”*Advanced options*” ja ”*Expert Install*”.

2. Käy valikot läpi normaalisti. Sijainnilla, kielellä ja näppäimistön ulkoasulla ei ole mitään väliä. Jos asennusohjelma ehdottaa ”usb-storage”-ajuria, sen voi jättää asentamatta, paitsi jos tuntee tarvitsevansa USB-laitteita virtuaa-

likoneessa. Vastaavasti kannattaa jättää ”virtualbox-ose-guest-x11” -pakketti asentamatta, sillä koneeseen ei tule X:ää.

3. Valitse ”*targeted*” kun asennus kysyy, mitä ajureita ylipäätänsä asennetaan.

4. Kun pääset valikkoon ”*Select and install software*”, ota ruksit pois kaikesta, jopa ”*Standard system utilities*”-valinnasta ja jatka asennusta.

Näillä ohjeilla asennettu järjestelmä vie tilaa noin 420-450 megatavua. Siitä puuttuu joitain oikeasti tarvittavia paketteja, kuten sudo, less, zip ja ssh. Ne kannattaa asentaa heti kun on mahdollista. Näistäkin toimenpiteistä huolimatta järjestelmässä on joitain turhia paketteja, kuten iamerican, ibritish ja wamerican. Nämä ovat ispell-ohjelman paketteja, jotka voi poistaa jollei niitä tarvitse.

Liite 2: Laitteiden ajonaikaisia konfiguraatioita

Tässä liitteessä on listattu kaikkien työssä käytettyjen laitteiden ajonaikaiset konfiguraatiot. Mukana ovat vain ne laitteet, joista kyseiset tiedot voitiin tallentaa.

Kaikki työssä käytetyt salasanat olivat aina ”abcd-1234”. Tämä salasana esiintyy ainakin Vyatta Core-laitteiden konfiguraatioissa, joskin salattuna. Mielenkiintoisena yksityiskohtana mainittakoon, että jokaisen Vyatta Coren salasana on sama, myös salattuna. Tämä johtui siitä, että kaikki Vyatat kloonattiin samasta alkuperäisestä asennuksesta, jolloin ne perivät saman salt-arvon. Tämä kannattaa huomioida tietoturvan osalta, jos Vyatta Corea kloonaa.

On huomattava, että Vyatta Core tallentaa käytössä olleiden rajapintojen MAC-osoitteet konfiguraatioihin (”`hw-id`” -avain). Jos siis kopioit suoraan jonkin tässä olevan listauksen laitteen konfiguraatioksi, muuta osoitteet ennen käyttöönottoa. Muutoin Vyatta Core ei tunnista rajapintoja.

Kuten aiemmin annetut komentolistaukset, myös näissä listauksissa on ylipitkät rivit katkottu \-merkillä. Jos konfiguraatioita kopioi johonkin sellaisenaan, on nämä merkit poistettava ja rivit yhdistettävä ensin.

Skenaarion 1 ISP-reititin

Tämä Vyatta Core-reititin simuloi internet-palveluntarjoajaa luvussa 4 (sivulla 25).

```
interfaces {
    ethernet eth0 {
        address 201.50.9.1/30
        duplex auto
        hw-id 08:00:27:d2:04:50
        smp_affinity auto
```

```
        speed auto
    }
    ethernet eth1 {
        address 76.15.192.1/30
        duplex auto
        hw-id 08:00:27:c0:46:4b
        smp_affinity auto
        speed auto
    }
    ethernet eth2 {
        address 130.17.25.1/30
        duplex auto
        hw-id 08:00:27:44:a3:9c
        smp_affinity auto
        speed auto
    }
    ethernet eth3 {
        address 49.16.137.97/27
        duplex auto
        hw-id 08:00:27:3f:4d:fd
        smp_affinity auto
        speed auto
    }
    loopback lo {
    }
}
system {
    config-management {
        commit-revisions 20
    }
    host-name ISP
    login {
        user vyatta {
            authentication {
                encrypted-password \
$1$LubCkuP6$aW9VO/FodjPDcydX/aWSL.
            }
            level admin
        }
    }
    package {
```

```
        auto-sync 1
        repository community {
            components main
            distribution stable
            password ""
            url http://packages.vyatta.com/vyatta
            username ""
        }
    }
    syslog {
        global {
            facility all {
                level notice
            }
            facility protocols {
                level debug
            }
        }
    }
    time-zone Europe/Helsinki
}


/* Warning: Do not remove the following line. */
/* === vyatta-config-version: "cluster@1:wanloadbalance@3:dhcp-\
relay@1:zone-policy@1:nat@4:config-management@1:webgui@1:ipsec@4:\
system@6:qos@1:conntrack@1:quagga@2:firewall@5:vrrp@1:webproxy@1:\
conntrack-sync@1:dhcp-server@4" === */
/* Release version: VC6.6R1 */
```

Skenaarion 1 sisempi palomuuri

Skenaariossa 1 sisempänä palomuurina (luku 4.2 sivulla 28) käytettiin joka kerta Vyatta Corea. Sen konfiguraatiot olivat samanlaiset joka testissä.

```
interfaces {
    ethernet eth0 {
        address 172.16.2.2/24
```

```
        duplex auto
        hw-id 08:00:27:71:c0:6a
        smp_affinity auto
        speed auto
    }
    ethernet eth1 {
        address 10.0.0.1/16
        duplex auto
        hw-id 08:00:27:5f:f4:b5
        smp_affinity auto
        speed auto
    }
    loopback lo {
    }
}
service {
    dhcp-server {
        disabled false
        shared-network-name InternalPool {
            authoritative disable
            subnet 10.0.0.0/16 {
                default-router 10.0.0.1
                lease 86400
                start 10.0.0.10 {
                    stop 10.0.255.254
                }
            }
        }
    }
}
system {
    config-management {
        commit-revisions 20
    }
    gateway-address 172.16.2.1
    host-name InnerFirewall
    login {
        user vyatta {
            authentication {
                encrypted-password \
$1$LubCkuP6$aW9VO/FodjPDcydX/aWSL.
```

```
            }
            level admin
        }
    }
    package {
        auto-sync 1
        repository community {
            components main
            distribution stable
            password ""
            url http://packages.vyatta.com/vyatta
            username ""
        }
    }
    syslog {
        global {
            facility all {
                level notice
            }
            facility protocols {
                level debug
            }
        }
    }
    time-zone Europe/Helsinki
}


/* Warning: Do not remove the following line. */
/* === vyatta-config-version: "cluster@1:wanloadbalance@3:dhcp-\
relay@1:zone-policy@1:nat@4:config-management@1:webgui@1:ipsec@4:\
system@6:qos@1:conntrack@1:quagga@2:firewall@5:vrrp@1:webproxy@1:\
conntrack-sync@1:dhcp-server@4" === */
/* Release version: VC6.6R1 */
```

Skenaarion 1 ShoreWall-koneen rajapinnat

Alla olevassa listauksessa on tiedoston `/etc/network/interfaces` sisältö luvun 4.4 (sivulla 36) ShoreWall-toteutukseen . Tiedoston alusta on poistettu turhat kommentit ja huomautukset.

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
    address 201.50.9.2
    netmask 255.255.255.252
    gateway 201.50.9.1

auto eth1
iface eth1 inet static
    address 10.0.0.1
    netmask 255.255.0.0

auto eth2
iface eth2 inet static
    address 172.16.1.1
    netmask 255.255.255.0
```

Seuraava listaus sisältää vastaavan tiedoston kahden palomuurin toteutukselle. Ainoat muuttuneet asiat ovat eth1-rajapinnan IP-osoite ja aliverkkomaski, sekä komento joka lisää kiinteän reitin reititystauluun.

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
    address 201.50.9.2
    netmask 255.255.255.252
    gateway 201.50.9.1
```

```
auto eth1
iface eth1 inet static
    address 172.16.2.1
    netmask 255.255.255.0
    # lisää sisäverkko reititystauluun
    up route add -net 10.0.0.0 netmask 255.255.0.0 gw 172.16.2.2

auto eth2
iface eth2 inet static
    address 172.16.1.1
    netmask 255.255.255.0
```

Skenaarion 1 ShoreWall-koneen DHCP-palvelin

ShoreWall ei sisällä DHCP-palvelinta, joten koneeseen asennettiin ISC DHCP server-palvelinohjelmisto (luvussa 4.4.1, sivulla 43). Sen konfiguraatiotiedosto on listattu alla. Koska tiedosto muokattiin ohjelman mukana tulleesta esimerkkitiedostosta, on siitä poistettu sen alussa olleet turhat kommentit ja esimerkit.

```
ddns-update-style none;
default-lease-time 600;
max-lease-time 7200;
authoritative;
log-facility local7;

subnet 10.0.0.0 netmask 255.255.0.0 {
    range 10.0.0.10 10.0.255.254;
    option routers 10.0.0.1;
}
```

Skenaarion 1 ShoreWall-koneen rules-tiedosto

Seuraavassa listauksessa on skenaarion 1 ShoreWall-koneen rules-tiedosto (luvussa 4.4.1 sivulla 37) kokonaisuudessaan, täsmälleen sellaisena kuin se työssä oli

käytössä. Muut tiedostot olivat tarpeeksi lyhyitä ja mahtuivat kokonaan sellaisenaan edellä mainittuun lukuun.

```
#SECTION ALL
#SECTION ESTABLISHED
#SECTION RELATED
SECTION NEW

# PING
ACCEPT loc fw ICMP # salli ping sisäverkosta palomuuriin
ACCEPT loc dmz ICMP # salli ping sisäverkosta DMZ-alueelle
ACCEPT loc net ICMP # salli ping sisäverkosta ulkoverkkoon \
(vaatii NATin)
ACCEPT fw loc ICMP # salli palomuurista ping sisäverkkoon
ACCEPT fw dmz ICMP # salli palomuurista ping DMZ-alueelle
ACCEPT fw net ICMP # salli palomuurista ping ulkoverkkoon

# SISÄVERKON NETTISELAILU
ACCEPT loc dmz TCP http,https # salli alkava HTTP sisäverkosta \
DMZ-alueelle
ACCEPT loc net TCP http,https # salli nettiliikenne ulkomaailmaan
DNS(ACCEPT) loc net # salli sisäverkosta DNS

# DMZ-ALUEELLE PÄÄSY ULKOA PÄIN
ACCEPT net dmz TCP http
DNAT net dmz:172.16.1.2 TCP http
```

Skenaarion 1 Vyatta Core-palomuuri

Seuraava listaus sisältää luvussa 4.5.1 (sivulla 46) luodun, skenaarion 1 ulomman Vyatta Core-palomuurin konfiguraation.

```
firewall {
    all-ping enable
    broadcast-ping disable
    ipv6-receive-redirects disable
    ipv6-src-route disable
    ip-src-route disable
```

```
    log-martians enable
    name dmz_to_internal {
        default-action drop
        rule 10 {
            action accept
            protocol tcp
            state {
                established enable
                related enable
            }
        }
    }
    name dmz_to_public {
        default-action drop
        rule 10 {
            action accept
            protocol tcp
            state {
                established enable
                related enable
            }
        }
    }
    name internal_to_dmz {
        default-action drop
        rule 10 {
            action accept
            destination {
                port http
            }
            protocol tcp
        }
    }
    name internal_to_public {
        default-action accept
    }
    name internal_to_vyatta {
        default-action accept
        rule 10 {
            action drop
            destination {
```

```
                port 22
            }
            protocol tcp
            source {
                address !10.0.0.3
            }
        }
    }
    name public_to_dmz {
        default-action drop
        rule 10 {
            action accept
            destination {
                port http
            }
            protocol tcp
        }
    }
    name public_to_internal {
        default-action accept
    }
    receive-redirects disable
    send-redirects enable
    source-validation disable
    syn-cookies enable
}
interfaces {
    ethernet eth0 {
        address 201.50.9.2/30
        duplex auto
        hw-id 08:00:27:6f:b7:d9
        smp_affinity auto
        speed auto
    }
    ethernet eth1 {
        address 10.0.0.1/16
        duplex auto
        hw-id 08:00:27:c8:a6:a3
        smp_affinity auto
        speed auto
    }
```

```
    ethernet eth2 {
        address 172.16.1.1/24
        duplex auto
        hw-id 08:00:27:93:85:ff
        smp_affinity auto
        speed auto
    }
    loopback lo {
    }
}
nat {
    destination {
        rule 10 {
            destination {
                address 201.50.9.2
                port http
            }
            inbound-interface eth0
            protocol tcp
            translation {
                address 172.16.1.2
            }
        }
    }
    source {
        rule 10 {
            outbound-interface eth0
            source {
                address 10.0.0.0/16
            }
            translation {
                address masquerade
            }
        }
        rule 20 {
            outbound-interface eth0
            source {
                address 172.16.1.2
            }
            translation {
                address 201.50.9.2
```

```
            }
        }
    }
}
service {
    dhcp-server {
        disabled false
        shared-network-name InternalPool {
            authoritative enable
            subnet 10.0.0.0/16 {
                default-router 10.0.0.1
                lease 86400
                start 10.0.0.3 {
                    stop 10.0.0.255
                }
            }
        }
    }
    ssh {
        port 22
    }
}
system {
    config-management {
        commit-revisions 20
    }
    gateway-address 201.50.9.1
    host-name LaatikkoFW
    login {
        user vyatta {
            authentication {
                encrypted-password \
$1$LubCkuP6$aW9VO/FodjPDcydX/aWSL.
            }
            level admin
        }
    }
    package {
        auto-sync 1
        repository community {
            components main
```

```
            distribution stable
            password ""
            url http://packages.vyatta.com/vyatta
            username ""
        }
    }
    syslog {
        global {
            facility all {
                level notice
            }
            facility protocols {
                level debug
            }
        }
    }
    time-zone Europe/Helsinki
}
zone-policy {
    zone dmz {
        default-action drop
        from internal {
            firewall {
                name internal_to_dmz
            }
        }
        from public {
            firewall {
                name public_to_dmz
            }
        }
        interface eth2
    }
    zone internal {
        default-action drop
        from dmz {
            firewall {
                name dmz_to_internal
            }
        }
        from public {
```

```
            firewall {
                name public_to_internal
            }
        }
        interface eth1
    }
    zone public {
        default-action drop
        from dmz {
            firewall {
                name dmz_to_public
            }
        }
        from internal {
            firewall {
                name internal_to_public
            }
        }
        interface eth0
    }
    zone vyatta {
        default-action drop
        from internal {
            firewall {
                name internal_to_vyatta
            }
        }
        local-zone
    }
}


/* Warning: Do not remove the following line. */
/* === vyatta-config-version: "cluster@1:wanloadbalance@3:dhcp-\
relay@1:zone-policy@1:nat@4:config-management@1:webgui@1:ipsec@4:\
system@6:qos@1:conntrack@1:quagga@2:firewall@5:vrrp@1:webproxy@1:\
conntrack-sync@1:dhcp-server@4" === */
/* Release version: VC6.6R1 */
```

Skenaarion 1 ulompi Vyatta Core-palomuuri

Seuraava listaus sisältää luvussa 4.5.2 (sivulla 58) luodun ulomman Vyatta Core-laitteen konfiguraation. Erona edelliseen ovat DHCP-palvelimen poisto, kiinteän reitin luonti ja rajapinnan IP-osoite.

```
firewall {
    all-ping enable
    broadcast-ping disable
    ipv6-receive-redirects disable
    ipv6-src-route disable
    ip-src-route disable
    log-martians enable
    name dmz_to_internal {
        default-action drop
        rule 10 {
            action accept
            protocol tcp
            state {
                established enable
                related enable
            }
        }
    }
    name dmz_to_public {
        default-action drop
        rule 10 {
            action accept
            protocol tcp
            state {
                established enable
                related enable
            }
        }
    }
    name internal_to_dmz {
        default-action drop
        rule 10 {
            action accept
```

```
            destination {
                port http
            }
            protocol tcp
        }
    }
    name internal_to_public {
        default-action accept
    }
    name internal_to_vyatta {
        default-action accept
        rule 10 {
            action drop
            destination {
                port 22
            }
            protocol tcp
            source {
                address !10.0.0.3
            }
        }
    }
    name public_to_dmz {
        default-action drop
        rule 10 {
            action accept
            destination {
                port http
            }
            protocol tcp
        }
    }
    name public_to_internal {
        default-action accept
    }
    receive-redirects disable
    send-redirects enable
    source-validation disable
    syn-cookies enable
}
interfaces {
```

```
    ethernet eth0 {
        address 201.50.9.2/30
        duplex auto
        hw-id 08:00:27:6f:b7:d9
        smp_affinity auto
        speed auto
    }
    ethernet eth1 {
        address 172.16.2.1/24
        duplex auto
        hw-id 08:00:27:c8:a6:a3
        smp_affinity auto
        speed auto
    }
    ethernet eth2 {
        address 172.16.1.1/24
        duplex auto
        hw-id 08:00:27:93:85:ff
        smp_affinity auto
        speed auto
    }
    loopback lo {
    }
}
nat {
    destination {
        rule 10 {
            destination {
                address 201.50.9.2
                port http
            }
            inbound-interface eth0
            protocol tcp
            translation {
                address 172.16.1.2
            }
        }
    }
    source {
        rule 10 {
            outbound-interface eth0
```

```
            source {
                address 10.0.0.0/16
            }
            translation {
                address masquerade
            }
        }
        rule 20 {
            outbound-interface eth0
            source {
                address 172.16.1.2
            }
            translation {
                address 201.50.9.2
            }
        }
    }
}
protocols {
    static {
        route 10.0.0.0/16 {
            next-hop 172.16.2.2 {
            }
        }
    }
}
service {
    ssh {
        port 22
    }
}
system {
    config-management {
        commit-revisions 20
    }
    gateway-address 201.50.9.1
    host-name OuterFirewall
    login {
        user vyatta {
            authentication {
                encrypted-password \
```

```
$1$LubCkuP6$aW9VO/FodjPDcydX/aWSL.
            }
            level admin
        }
    }
    package {
        auto-sync 1
        repository community {
            components main
            distribution stable
            password ""
            url http://packages.vyatta.com/vyatta
            username ""
        }
    }
    syslog {
        global {
            facility all {
                level notice
            }
            facility protocols {
                level debug
            }
        }
    }
    time-zone Europe/Helsinki
}
zone-policy {
    zone dmz {
        default-action drop
        from internal {
            firewall {
                name internal_to_dmz
            }
        }
        from public {
            firewall {
                name public_to_dmz
            }
        }
        interface eth2
```

```
    }
    zone internal {
        default-action drop
        from dmz {
            firewall {
                name dmz_to_internal
            }
        }
        from public {
            firewall {
                name public_to_internal
            }
        }
        interface eth1
    }
    zone public {
        default-action drop
        from dmz {
            firewall {
                name dmz_to_public
            }
        }
        from internal {
            firewall {
                name internal_to_public
            }
        }
        interface eth0
    }
    zone vyatta {
        default-action drop
        from internal {
            firewall {
                name internal_to_vyatta
            }
        }
        local-zone
    }
}
```

```
/* Warning: Do not remove the following line. */
/* === vyatta-config-version: "cluster@1:wanloadbalance@3:dhcp-\
relay@1:zone-policy@1:nat@4:config-management@1:webgui@1:ipsec@4:\
system@6:qos@1:conntrack@1:quagga@2:firewall@5:vrrp@1:webproxy@1:\
conntrack-sync@1:dhcp-server@4" === */
/* Release version: VC6.6R1 */
```

Skenaarion 2 CoreA-reititin

Seuraavassa listauksessa on luvussa 5 (sivulla 60) luodun CoreA-reitittimen täysi konfiguraatio.

```
firewall {
    all-ping enable
    broadcast-ping disable
    ipv6-receive-redirects disable
    ipv6-src-route disable
    ip-src-route disable
    log-martians enable
    name block_ssh {
        default-action accept
        rule 10 {
            action drop
            destination {
                port 22
            }
            protocol tcp
            source {
                address !172.16.11.0/24
            }
        }
    }
    name protect_management {
        default-action drop
        rule 10 {
            action accept
            destination {
                address 172.16.11.0/24
```

```
            }
            state {
                established enable
                related enable
            }
        }
    }
    name restrict_servers {
        default-action drop
        rule 10 {
            action accept
            state {
                established enable
                related enable
            }
        }
    }
    receive-redirects disable
    send-redirects enable
    source-validation disable
    syn-cookies enable
}
interfaces {
    ethernet eth0 {
        address 172.16.0.1/30
        description CoreB
        duplex auto
        firewall {
            local {
                name block_ssh
            }
        }
        hw-id 08:00:27:3c:fc:14
        smp_affinity auto
        speed auto
    }
    ethernet eth1 {
        address 172.16.0.5/30
        description CoreC
        duplex auto
        firewall {
```

```
            local {
                name block_ssh
            }
        }
        hw-id 08:00:27:30:a7:1e
        smp_affinity auto
        speed auto
    }
    ethernet eth2 {
        address 172.16.1.1/24
        description "Verkko A"
        duplex auto
        firewall {
            local {
                name block_ssh
            }
        }
        hw-id 08:00:27:40:3c:5a
        smp_affinity auto
        speed auto
    }
    ethernet eth3 {
        address 172.16.10.1/24
        description Servers
        duplex auto
        firewall {
            in {
                name restrict_servers
            }
            local {
                name block_ssh
            }
        }
        hw-id 08:00:27:32:19:c3
        smp_affinity auto
        speed auto
    }
    ethernet eth4 {
        address 172.16.11.1/24
        description Management
        duplex auto
```

```
        firewall {
            local {
                name block_ssh
            }
            out {
                name protect_management
            }
        }
        hw-id 08:00:27:51:5e:64
        smp_affinity auto
        speed auto
    }
    loopback lo {
    }
}
protocols {
    ospf {
        area 0 {
            network 172.16.0.0/30
            network 172.16.0.4/30
            network 172.16.1.0/24
            network 172.16.10.0/24
            network 172.16.11.0/24
        }
        passive-interface eth2
        passive-interface eth3
        passive-interface eth4
    }
}
service {
    dhcp-server {
        disabled false
        shared-network-name managementpool {
            authoritative disable
            subnet 172.16.11.0/24 {
                default-router 172.16.11.1
                dns-server 172.16.10.2
                lease 86400
                ntp-server 172.16.10.2
                start 172.16.11.2 {
                    stop 172.16.11.254
```

```
                }
            }
        }
    }
    ssh {
        port 22
    }
}
system {
    config-management {
        commit-revisions 20
    }
    host-name CoreA
    login {
        user vyatta {
            authentication {
                encrypted-password \
$1$LubCkuP6$aW9VO/FodjPDcydX/aWSL.
            }
            level admin
        }
    }
    package {
        auto-sync 1
        repository community {
            components main
            distribution stable
            password ""
            url http://packages.vyatta.com/vyatta
            username ""
        }
    }
    syslog {
        global {
            facility all {
                level notice
            }
            facility protocols {
                level debug
            }
        }
```

```
    }
    time-zone Europe/Helsinki
}


/* Warning: Do not remove the following line. */
/* === vyatta-config-version: "cluster@1:wanloadbalance@3:dhcp-\
relay@1:zone-policy@1:nat@4:config-management@1:webgui@1:\
ipsec@4:system@6:qos@1:conntrack@1:quagga@2:firewall@5:vrrp@1:\
webproxy@1:conntrack-sync@1:dhcp-server@4" === */
/* Release version: VC6.6R1 */
```

Skenaarion 2 CoreB

Seuraavassa listauksessa on luvussa 5 (sivulla 60) luodun CoreB-reitittimen täysi konfiguraatio.

```
firewall {
    all-ping enable
    broadcast-ping disable
    ipv6-receive-redirects disable
    ipv6-src-route disable
    ip-src-route disable
    log-martians enable
    name block_B_to_C {
        default-action accept
        rule 10 {
            action drop
            destination {
                address 172.16.3.0/24
            }
            source {
                address 172.16.2.0/24
            }
        }
    }
    name block_ssh {
        default-action accept
```

```
        rule 10 {
            action drop
            destination {
                port 22
            }
            protocol tcp
            source {
                address !172.16.11.0/24
            }
        }
    }
    receive-redirects disable
    send-redirects enable
    source-validation disable
    syn-cookies enable
}
interfaces {
    ethernet eth0 {
        address 172.16.0.2/30
        description CoreA
        duplex auto
        firewall {
            local {
                name block_ssh
            }
        }
        hw-id 08:00:27:13:45:ae
        smp_affinity auto
        speed auto
    }
    ethernet eth1 {
        address 172.16.0.9/30
        description CoreC
        duplex auto
        firewall {
            local {
                name block_ssh
            }
        }
        hw-id 08:00:27:21:f3:ab
        smp_affinity auto
```

```
        speed auto
    }
    ethernet eth2 {
        address 172.16.2.1/24
        description "Verkko B"
        duplex auto
        firewall {
            in {
                name block_B_to_C
            }
            local {
                name block_ssh
            }
        }
        hw-id 08:00:27:fc:8a:69
        smp_affinity auto
        speed auto
    }
    ethernet eth3 {
        duplex auto
        hw-id 08:00:27:a1:74:88
        smp_affinity auto
        speed auto
    }
    loopback lo {
    }
}
protocols {
    ospf {
        area 0 {
            network 172.16.0.0/30
            network 172.16.0.8/30
            network 172.16.2.0/24
        }
        passive-interface eth2
    }
}
service {
    ssh {
        port 22
    }
```

```
}
system {
    config-management {
        commit-revisions 20
    }
    host-name CoreB
    login {
        user vyatta {
            authentication {
                encrypted-password \
$1$LubCkuP6$aW9VO/FodjPDcydX/aWSL.
            }
            level admin
        }
    }
    package {
        auto-sync 1
        repository community {
            components main
            distribution stable
            password ""
            url http://packages.vyatta.com/vyatta
            username ""
        }
    }
    syslog {
        global {
            facility all {
                level notice
            }
            facility protocols {
                level debug
            }
        }
    }
    time-zone Europe/Helsinki
}


/* Warning: Do not remove the following line. */
/* === vyatta-config-version: "cluster@1:wanloadbalance@3:dhcp-\
```

```
relay@1:zone-policy@1:nat@4:config-management@1:webgui@1:ipsec@4:\
system@6:qos@1:conntrack@1:quagga@2:firewall@5:vrrp@1:webproxy@1:\
conntrack-sync@1:dhcp-server@4" === */
/* Release version: VC6.6R1 */
```

Skenaarion 2 CoreC

Seuraavassa listauksessa on luvussa 5 (sivulla 60) luodun CoreC-reitittimen täysi konfiguraatio.

```
firewall {
    all-ping enable
    broadcast-ping disable
    ipv6-receive-redirects disable
    ipv6-src-route disable
    ip-src-route disable
    log-martians enable
    name block_ssh {
        default-action accept
        rule 10 {
            action drop
            destination {
                port 22
            }
            protocol tcp
            source {
                address !172.16.11.0/24
            }
        }
    }
    name protect_management {
        default-action drop
        rule 10 {
            action accept
            destination {
                address 172.16.11.0/24
            }
            state {
```

```
                established enable
                related enable
            }
        }
    }
    name restrict_servers {
        default-action drop
        rule 10 {
            action accept
            state {
                established enable
                related enable
            }
        }
    }
    receive-redirects disable
    send-redirects enable
    source-validation disable
    syn-cookies enable
}
interfaces {
    ethernet eth0 {
        address 172.16.0.1/30
        description CoreB
        duplex auto
        firewall {
            local {
                name block_ssh
            }
        }
        hw-id 08:00:27:3c:fc:14
        smp_affinity auto
        speed auto
    }
    ethernet eth1 {
        address 172.16.0.5/30
        description CoreC
        duplex auto
        firewall {
            local {
                name block_ssh
```

```
            }
        }
        hw-id 08:00:27:30:a7:1e
        smp_affinity auto
        speed auto
    }
    ethernet eth2 {
        address 172.16.1.1/24
        description "Verkko A"
        duplex auto
        firewall {
            local {
                name block_ssh
            }
        }
        hw-id 08:00:27:40:3c:5a
        smp_affinity auto
        speed auto
    }
    ethernet eth3 {
        address 172.16.10.1/24
        description Servers
        duplex auto
        firewall {
            in {
                name restrict_servers
            }
            local {
                name block_ssh
            }
        }
        hw-id 08:00:27:32:19:c3
        smp_affinity auto
        speed auto
    }
    ethernet eth4 {
        address 172.16.11.1/24
        description Management
        duplex auto
        firewall {
            local {
```

```
                name block_ssh
            }
            out {
                name protect_management
            }
        }
        hw-id 08:00:27:51:5e:64
        smp_affinity auto
        speed auto
    }
    loopback lo {
    }
}
protocols {
    ospf {
        area 0 {
            network 172.16.0.0/30
            network 172.16.0.4/30
            network 172.16.1.0/24
            network 172.16.10.0/24
            network 172.16.11.0/24
        }
        passive-interface eth2
        passive-interface eth3
        passive-interface eth4
    }
}
service {
    ssh {
        port 22
    }
}
system {
    config-management {
        commit-revisions 20
    }
    host-name CoreA
    login {
        user vyatta {
            authentication {
                encrypted-password \
```

```
$1$LubCkuP6$aW9VO/FodjPDcydX/aWSL.
            }
            level admin
        }
    }
    package {
        auto-sync 1
        repository community {
            components main
            distribution stable
            password ""
            url http://packages.vyatta.com/vyatta
            username ""
        }
    }
    syslog {
        global {
            facility all {
                level notice
            }
            facility protocols {
                level debug
            }
        }
    }
    time-zone Europe/Helsinki
}


/* Warning: Do not remove the following line. */
/* === vyatta-config-version: "cluster@1:wanloadbalance@3:dhcp-\
relay@1:zone-policy@1:nat@4:config-management@1:webgui@1:ipsec@4:\
system@6:qos@1:conntrack@1:quagga@2:firewall@5:vrrp@1:webproxy@1:\
conntrack-sync@1:dhcp-server@4" === */
/* Release version: VC6.6R1 */
```

Skenaarion 2 EdgeB-reititin

Seuraavassa listauksessa on luvussa 5.5.1 (sivulla 73) luodun EdgeB-reitittimen täysi konfiguraatio.

```
interfaces {
    ethernet eth0 {
        address 172.16.2.2/24
        description CoreB
        duplex auto
        hw-id 08:00:27:15:fa:f6
        smp_affinity auto
        speed auto
    }
    ethernet eth1 {
        address 192.168.0.1/24
        description "Verkko B-1"
        duplex auto
        hw-id 08:00:27:d5:f3:9e
        smp_affinity auto
        speed auto
    }
    ethernet eth2 {
        address 192.168.1.1/24
        description "Verkko B-2"
        duplex auto
        hw-id 08:00:27:18:99:4d
        smp_affinity auto
        speed auto
    }
    ethernet eth3 {
        address 192.168.2.1/30
        description R&D
        duplex auto
        hw-id 08:00:27:73:40:f2
        smp_affinity auto
        speed auto
    }
    loopback lo {
    }
```

```
}
nat {
    source {
        rule 10 {
            outbound-interface eth0
            source {
                address 192.168.0.0-192.168.2.255
            }
            translation {
                address masquerade
            }
        }
    }
}
service {
    dhcp-server {
        disabled false
        shared-network-name PoolB1 {
            authoritative enable
            subnet 192.168.0.0/24 {
                default-router 192.168.0.1
                lease 86400
                start 192.168.0.2 {
                    stop 192.168.0.254
                }
            }
        }
        shared-network-name PoolB2 {
            authoritative enable
            subnet 192.168.1.0/24 {
                default-router 192.168.1.1
                lease 86400
                start 192.168.1.2 {
                    stop 192.168.1.254
                }
            }
        }
    }
    ssh {
        port 22
    }
```

```
}
system {
    config-management {
        commit-revisions 20
    }
    gateway-address 172.16.2.1
    host-name EdgeB
    login {
        user vyatta {
            authentication {
                encrypted-password \
$1$LubCkuP6$aW9VO/FodjPDcydX/aWSL.
            }
            level admin
        }
    }
    package {
        auto-sync 1
        repository community {
            components main
            distribution stable
            password ""
            url http://packages.vyatta.com/vyatta
            username ""
        }
    }
    syslog {
        global {
            facility all {
                level notice
            }
            facility protocols {
                level debug
            }
        }
    }
    time-zone Europe/Helsinki
}


/* Warning: Do not remove the following line. */
```

```
/* === vyatta-config-version: "cluster@1:wanloadbalance@3:dhcp-\
relay@1:zone-policy@1:nat@4:config-management@1:webgui@1:ipsec@4:\
system@6:qos@1:conntrack@1:quagga@2:firewall@5:vrrp@1:webproxy@1:\
conntrack-sync@1:dhcp-server@4" === */
/* Release version: VC6.6R1 */
```

Skenaarion 2 R&D -reititin

Vaikka CoreB-reitittimen taakse tarkoitetun tuotekehityksen reitittimen (luku 5.5 sivulla 72) taakse ei luotu mitään verkkoja, on sen konfiguraatio seuraavassa listauksessa.

```
interfaces {
    ethernet eth0 {
        address 192.168.2.2/30
        duplex auto
        hw-id 08:00:27:9b:fd:60
        smp_affinity auto
        speed auto
    }
    loopback lo {
    }
}
system {
    config-management {
        commit-revisions 20
    }
    gateway-address 192.168.2.1
    host-name RD
    login {
        user vyatta {
            authentication {
                encrypted-password \
$1$LubCkuP6$aW9VO/FodjPDcydX/aWSL.
            }
            level admin
        }
```

```
    }
    package {
        auto-sync 1
        repository community {
            components main
            distribution stable
            password ""
            url http://packages.vyatta.com/vyatta
            username ""
        }
    }
    syslog {
        global {
            facility all {
                level notice
            }
            facility protocols {
                level debug
            }
        }
    }
    time-zone Europe/Helsinki
}


/* Warning: Do not remove the following line. */
/* === vyatta-config-version: "cluster@1:wanloadbalance@3:dhcp-\
relay@1:zone-policy@1:nat@4:config-management@1:webgui@1:ipsec@4:\
system@6:qos@1:conntrack@1:quagga@2:firewall@5:vrrp@1:webproxy@1:\
conntrack-sync@1:dhcp-server@4" === */
/* Release version: VC6.6R1 */
```

Skenaarion 2 piilotetun palvelimen rajapinnat

Seuraava listaus sisältää luvussa 5.5.2 (sivulla 74) luodun ”piilotetun” palvelimen tiedoston `/etc/network/interfaces`.

```
auto lo
iface lo inet loopback

allow-hotplug eth0
iface eth0 inet static
    address 192.168.0.254
    netmask 255.255.255.0
    gateway 192.168.0.1
    # estä muun kuin verkon 192.168.0.0/24 pääsy koneeseen
    pre-up iptables -I INPUT ! -s 192.168.0.0/255.255.255.0 -j DROP
```

Liite 3: Sillattu Vyatta Core-palomuuri

Tässä työssä käytettiin ainoastaan reitittävässä tilassa olleita palomuureja. On kuitenkin mahdollista luoda myös siltaavassa tilassa toimivia palomuureja. Reitittävä laite näkyy aina verkkojen laitteille esimerkiksi traceroute-komennolla. Siltaavat laitteet, kuten kytkimet, eivät näy muille verkon laitteille, sillä ne vain välittävät valmiita paketteja eteenpäin muuttamatta niitä mitenkään.

Perinteiset fyysiset Ethernet-kytkimet eivät pysty toimimaan palomuurina, mutta jos Linux-pohjaisen laitteen asettaa siltaavaan tilaan, se pystyy toimimaan samalla palomuurina. Laite näkee jokaisen läpikulkevan paketin ja pystyy suodattamaan niitä tarvittaessa. Tällöin verkkoon muodostuu ”näkymätön” palomuuri.

Siltaavia palomuureja ei käytetty työssä, koska työn tekijä tajusi sillattujen palomuurien olevan mahdollisia vasta, kun kaikki skenaariot oli jo toteutettu, ja työ oli jo muuten valmis. Täydellisyyden vuoksi tähän liitteeseen kuitenkin koottiin lyhyet ohjeet sillatun palomuurin tekoon Vyatta Corella. Vastaavan palomuurin voi toteuttaa myös puhtaalla iptablesilla. ShoreWallin ja pfSensen soveltuvuudesta sillatuksi palomuuriksi ei tekijällä ollut kirjoitushetkellä tietoa.

Tässä on oletettu, että Vyatta Core on jo muuten toimiva ja valmiiksi konfiguroitu.

Kahden tai useamman rajapinnan siltaus

Siltausta varten tarvitaan vähintään kaksi rajapintaa. Niiden on oltava **valikoimattomassa tilassa** (engl. *promiscuous mode*). Tämä siksi, että rajapinnan on päästettävä läpi kaikki paketit, ei vain sille tarkoitettuja paketteja. Siltaaville rajapinnoille ei voi määrittää IP-osoitteita.

VirtualBoxissa tila asetetaan virtuaalikoneen verkkoasetuksista. Asetus ”*Promiscuous Mode*” on asetettava joko tilaan ”*Allow VMs*” tai ”*Allow All*”. On huomattava, että ”NAT”-tilassa olevaa rajapintaa ei voi asettaa valikoimattomaan tilaan.

Kun rajapinnat on asetettu oikeaan tilaan, voidaan Vyatta Core konfiguroida tekemään siltausta niiden välille. Luo ensin itse **silta** (engl. *bridge*):

```
set interfaces bridge <sillan nimi>
```

Liitä siihen sitten sillattavat rajapinnat. Toista seuraava komento jokaiselle halutulle rajapinnalle:

```
set interfaces ethernet <rajapinta> bridge-group \
bridge <sillan nimi>
```

Esimerkkinä luodaan silta ”br0” ja liitetään siihen rajapinnat eth0 ja eth1:

```
set interfaces bridge br0
set interfaces ethernet eth0 bridge-group bridge br0
set interfaces ethernet eth1 bridge-group bridge br0
```

Asetusten tallentamisen jälkeen sillan toiminta voidaan todeta kytkemällä se kahden virtuaalikoneen välille. Virtuaalikoneet pystyvät viestimään keskenään sillan yli näkemättä sen olemassaoloa. (Bridging Reference Guide 2013.)

Palomuurin luonti

Palomuuria luotaessa on huomattava, että se liitetään siltaan, eikä rajapintoihin. Vyöhykepohjaisia palomuureja ei voi käyttää siltojen kanssa, mutta perinteiset palomuurisäännöt toimivat. Kun säännöt on luotu, se liitetään siltaan seuraavalla komennolla: (Firewall 2013.)

```
set interfaces bridge <sillan nimi> firewall <suunta> <nimi>
```

Ikävä kyllä Vyatan omat ohjeet ovat erittäin niukkasanaisia sillatuista palomuureista. Ohjeet eivät kerro mitä rajoituksia niihin liittyy, tai mitä niitä tehdessä tulisi ottaa huomioon. Ne eivät edes kerro, miten liikenteen suunta määräytyy sillatussa tilassa. Tekijän omakohtaisten kokemusten perusteella ”in” on toimiva, mutta tämäkin saattaa vaihdella tilanteesta riippuen. Kokemus on myös osoittanut sen, että virtuaalikonetta ei voi kytkeä VirtualBoxiin pelkän siltaavan laitteen

kautta, vaan VirtualBox vaatii itsensä ja siltaavan laitteen väliin reitittävän laitteen. Verkkoon tarvitaan siis vähintään kolme laitetta peräkkäin (eli: VirtualBox ↔ reititin ↔ silta ↔ itse virtuaalikone).

Lopuksi esimerkkinä kaiken ICMP-liikenteen pysäyttävä palomuurikonfiguraatio (tämä olettaa, että rajapinnat eth0 ja eth1 on liitetty siltaan ”br0”):

```
set firewall block_icmp default-action accept
set firewall block_icmp rule 10 action drop
set firewall block_icmp rule 10 protocol icmp
…
set interfaces br0 firewall in block_icmp
```

Näiden jälkeen kaikki liikenne paitsi ICMP pääsee palomuurin läpi.

Liite 4: Sekalaisia asioita

Tämä liite sisältää joitain sekalaisia asioita, jotka eivät välttämättä ole oleellisia, mutta ovat mukana täydellisyyden vuoksi.

Laatikko Oy:n kotisivut

Laatikko Oy:n DMZ-alueella sijainneiden kotisivujen tarkoitus näkyä luvun 4 (alkaa sivulta 25) kuvankaappauksissa, mutta näin ei koskaan päässyt käymään. Täydellisyyden vuoksi sivujen sisältö on nähtävissä kuviosta 14. Sivuston ulkoasusta ei pidä vetää mitään johtopäätöksiä.

Kuvio 14. Laatikko Oy:n kotisivut

Julkisten internet-palvelimien sisältö

Laatikko Oy:n kotisivuja lukuun ottamatta kaikki julkiset (ja myös sisäiset, jos totta puhutaan) internet-palvelimet koostuivat oikeasti vain yhdestä PHP-skriptistä. Se raportoi takaisin palvelimen kellonajan, sekä vierailijan julkisen IP-osoitteen. Tämän yksinkertaisen, mutta kätevän skriptin avulla voitiin aina varmistaa ulkoisten IP-osoitteiden olevan sellaisia kuin piti, ja että palvelimeen saatiin oikeasti yhteys (kellonaika oli oikein, eikä sivu tullut selaimen välimuistista).

```
<!DOCTYPE html><html><head><meta charset="utf-8"></head><body>

<?php

echo "Client IP: " . $_SERVER["REMOTE_ADDR"] . "<br>";
echo "Server timestamp: " . date("d.m.Y H:i:s", time());

?>

</body></html>
```

Tiedosto tallennettiin palvelimien hakemistoon `/var/www` nimelle `index.php`.