Lähiverkon toteutus palveluiden näkökulmasta IPv4- ja IPv6-ympäristössä

Ammattikorkeakoulututkinnon opinnäytetyö

Tietojenkäsittelyn koulutus

Kevät 2024

Teemu Ruottinen

HAMK
Hämeen ammatti-
korkeakoulu

Tietojenkäsittelyn koulutus Tiivistelmä
Tekijä Teemu Ruottinen Vuosi 2024
Työn nimi Lähiverkon toteutus palveluiden näkökulmasta IPv4- ja IPv6-ympäristössä
Ohjaaja Pentti Ojaniemi

Opinnäytetyön tavoitteena oli selvittää, miten TCP/IP-verkkotekniikka sekä DNS- ja DHCP-verkkopalvelut mahdollistavat älylaitteiden toiminnan tietoverkoissa ja tutkia kaksoispinotekniikan hyödyntämistä IPv4- ja IPv6-verkkojen yhteensopivuuden tukemiseksi. Työssä kehitettiin ja testattiin verkkoympäristö, joka tukee molempia protokollia, sekä mahdollistaa verkkopalveluiden asentamisen, konfiguroinnin ja hallinnan avoimen lähdekoodin sovelluksilla.

Opinnäytetyön teoreettisessa osassa määriteltiin keskeiset käsitteet, kuten TCP/IP-verkkoprotokollat, DNS- ja DHCP-palvelut sekä kaksoispinotekniikka. Teoreettinen tausta perustui verkkotekniikan standardeihin ja tieteellisiin tutkimuksiin ja se loi pohjan tutkimuksen käytännön osuudelle. Tutkimuksessa käytettiin toiminnallista lähestymistapaa ja tutkimusmetodina oli testausympäristön luominen sekä käytettävissä olevien sovellusten, kuten ISC KEA:n ja BIND:n, konfigurointi ja analysointi. Käytännön testaukset suoritettiin laboratorioympäristössä, jossa verkkoyhteydet ja -palvelut asetettiin toimimaan sekä IPv4- että IPv6-ympäristössä.

Tutkimuksessa havaittiin kaksoispinotekniikkaa hyödyntävän ympäristön mahdollistavan sujuvan siirtymän IPv4:stä IPv6:een ja tarjoavan erinomaisen alustan DNS- ja DHCP-palveluiden testaamiseen. Testaus osoitti, että rakennettu ympäristö soveltui erityisen hyvin näiden palveluiden konfigurointiin ja hallintaan avoimen lähdekoodin sovelluksilla. Johtopäätöksenä todettiin, että kaksoispinotekniikka ja avoimen lähdekoodin ohjelmistot tarjoavat kustannustehokkaan ja skaalautuvan ratkaisun lähiverkoille.

Avainsanat TCP/IP, DNS, DHCP, kaksoispino, IPv6
Sivut 51 sivua ja liitteitä 1 sivua

HAMK
Hämeen ammatti-
korkeakoulu

Degree Programme in Business Information Technology Abstract
Author Teemu Ruottinen Year 2024
Subject Implementation of a local area network from the perspective of services
Supervisors Pentti Ojaniemi

The objective of this thesis was to explore how TCP/IP network technology, as well as DNS and DHCP network services, enable the operation of smart devices in networks and to study the utilization of dual-stack technology to support compatibility between IPv4 and IPv6 networks. A network environment was developed and tested, which supports both protocols and allows the installation, configuration, and management of network services using open-source applications.

The theoretical part of the thesis defined key concepts such as TCP/IP network protocols, DNS and DHCP services, and dual-stack technology. The theoretical background was based on networking standards and scientific research, which provided the foundation for the practical part of the study. A functional approach was used in the research, and the research method involved creating a test environment as well as configuring and analyzing available applications such as ISC KEA and BIND. The practical tests were conducted in a laboratory environment where network connections and services were configured to operate in both IPv4 and IPv6 environments.

The study found that an environment utilizing dual-stack technology enabled a smooth transition from IPv4 to IPv6 and provided an excellent platform for testing DNS and DHCP services. The testing demonstrated that the built environment was particularly well-suited for configuring and managing these services with open-source applications. The conclusion was that dual-stack technology and open-source software offer a cost-effective and scalable solution for local networks.

Keywords TCP/IP, DNS, DHCP, Dual-Stack, IPv6
Pages 51 pages and appendices 1 pages

Sanasto

IP	Internet Protocol. Internet-protokollan tarkoituksena on siirtää datagrammeja eli paketteja verkkojen välillä ja varmistaa niiden päätyminen oikeaan kohteeseen käyttämällä internet-osoitteita.
IPv4	Internet Protocol Version 4. Internet-protokollan ensimmäinen versio
IPv6	Internet Protocol Version 6. Internet-protokollan toinen versio, joka laajentaa IP-osoitteiden avaruutta.
ICMP	Internet Control Message Protocol. ICMP-protokolla raportoi verkon virheistä ja auttaa diagnosoimaan verkko-ongelmia.
IETF	Internet Engineering Task Force. Internetin standardeja ylläpitävä organisaatio.
DNS	Domain Name System. Järjestelmä, jolla nimet käännetään IP-osoitteiksi.
TCP	Transmission Control Protocol. TCP-protokolla tarjoaa luotettavan, yhteyksellisen datansiirtopalvelun ja pyrkii varmistamaan pakettien virheettömyyden verkossa.
DHCP	Dynamic Host Configuration Protocol. Verkon automaattisen konfiguroinnin standardi.
DHCPv4	Dynamic Host Configuration Protocol Version 4. DHCP-protokolla IPv4:lle.
DHCPv6	Dynamic Host Configuration Protocol Version 6. DHCP-protokolla IPv6:lle.
UDP	User Datagram Protocol. Yhteydetön tiedonsiirtoprotokolla, nopeampi kuin TCP, koska tiedon perillemenoa ei varmisteta.
MAC	Media Access Control. Verkkolaitteen fyysisen kerroksen osoite.
ARP	Address Resolution Protocol. Protokolla, jolla IPv4-paketit ohjataan paikallisverkossa.
NDP	Neighbor Discovery Protocol. Protokolla, jolla IPv6-paketit ohjataan paikallisverkossa.
NAT	Network Address Translation. Standardi, jonka avulla verkko-osoite voidaan uudelleenohjata.

ISATAP	Intra-Site Automatic Tunnel Addressing Protocol. Protokolla, joka kapseloi IPv6-paketit IPv4-verkossa.
NAPT	Network Address and Port Translation. NAT-laajennus, jossa muokataan myös porttinumeroita.
B4	Border Relay for 4rd Tunnel. Siirtymäkauden komponentti.
AFTR	Address Family Transition Router. Laite, jonka avulla IPv6-protokolla saa yhteyden IPv4-protokollaan.
MAP-E	Mapping of Address and Port using Encapsulation. Mekanismi, joka yhdistää osoitteita ja portteja tunnelointia käyttäen.
NAT64	Network Address Translation from IPv6 to IPv4. Protokolla, joka kääntää IPv6-osoitteita IPv4-osoitteiksi.
CLAT	Customer-side Translator. IPv6-asiakkaan käännöskomponentti NAT64-ympäristössä.
PLAT	Provider-side Translaton. IPv6-palvelun käännöskomponentti NAT64-ympäristössä.
RTT	Round Trip Time. Aika, joka kuluu paketin lähettämiseen ja sen vastauksen saamiseen verkossa.
IPv6NET	IPv6 Network Evaluation Testbed. IPv6-verkon testialusta.
GNS3	Graphical Network Simulator 3. Verkkojen simulointiohjelmisto.
SLAAC	Stateless Address Autoconfiguration. Mekanismi, jolla IPv6-osoitteet luodaan automaattisesti ilman DHCP-palvelua.
WIFI	Wireless Fidelity. Langaton lähiverkkotekniikka.
TFTP	Trivial File Transfer Protocol. Tiedostonsiirtoprotokolla.
DNSSEC	Domain Name System Security Extensions. DNS:n turvalaajennus.
TLD	Top Level Domain. Verkkotunnuksen päätaso.
CVE	Common Vulnerabilities and Exposures. Julkinen tietoturva-aukkojen ja haavoittuvuuksien tietokanta.
OS	Operating System. Käyttöjärjestelmä.
API	Application Programming Interface. Rajapinta, jonka avulla ohjelmat voivat keskustella keskenään.
CLI	Command Line Interface. Tekstipohjainen käyttöliittymä.
GUI	Graphical User Interface. Graafinen käyttöliittymä.
ISC	Internet Systems Consortium. Yritys, joka kehittää avoimen lähdekoodin ohjelmistoja operaattoreille.

IoT	Internet of Things eli esineiden internet viittaa älylaitteisiin, joissa ei ole perinteistä käyttöjärjestelmää. Laitteita voidaan kutsua myös älyesineiksi. Ne ovat yksinkertaisia kodin laitteita, joihin on lisätty älykkyyttä. Laitteet hyödyntävät internetpalvelua toiminnassaan.

Sisällysluettelo

Kuvat, komennot, ohjelmakoodit, taulukot ja kaavat

Liitteet

1 Johdanto

Internetin räjähdysmäinen kasvu viimeisen vuosikymmenen aikana on johtanut siihen, että digitaalinen maailma on tullut osaksi arkea sekä yksityis- että työelämässä. Internetin kulmakiviä ovat erilaiset verkkotekniikat, jotka mahdollistavat älylaitteiden kommunikoinnin digitaalisessa maailmassa. Kiinnostusta on herättänyt tapahtumakulku, kun älylaite kytketään verkkoon. Tämä opinnäytetyö keskittyy TCP/IP-verkkotekniikkoihin ja niihin liittyviin DHCP- ja DNS-verkkopalveluihin.

Opinnäytetyön tarkoituksena on tarjota lukijoille ymmärrys siitä, miten TCP/IP-verkkotekniikka mahdollistaa tietoliikenneyhteyden, mihin DNS- ja DHCP-verkkopalveluita tarvitaan sekä kuinka ne toimivat. Tämän perustiedon pohjalta opinnäytetyö pyrkii vastaamaan seuraaviin tutkimuskysymyksiin:

1. Mitä kaksoispinotekniikka tarkoittaa IPv6:n käyttöönotossa ja mitä muita vaihtoehtoja IPv4:n ja IPv6:n yhteiselolle on?

2. Mitkä DNS- ja DHCP-sovellukset soveltuvat kaksoispinotekniikan tukemiseen ja mikä on paras alusta niiden käytännön asennuksia ja konfiguraatiota varten?

3. Millaisia avoimen lähdekoodin ratkaisuja on olemassa DNS- ja DHCP-palveluiden konfiguraation ylläpidon helpottamiseksi?

Tutkimuskysymyksiin haetaan vastauksia teoriasta ja käytännöstä, joka toteutetaan asentamalla virtuaalinen palvelinympäristö, joka hyödyntää kaksoispinotekniikkaa. Palvelinympäristössä varmistetaan verkon automaattisen konfiguroinnin toimivuus sekä tarkistetaan miten nimipalvelu käyttäytyy.

Lopputuloksena syntyy verkonpalveluiden palvelinympäristö ja niiden konfigurointiin soveltuva työkalu. Tämä ympäristö soveltuu hyvin uusien laitteiden testaamiseen ja opinnäytetyö mahdollistaa vastaavanlaisen ympäristön asentamisen joko isomman yrityksen käyttöön tai pienelle tietokoneelle esimerkiksi kotiympäristöön.

2 Internetin protokollat

TCP/IP-verkkomallia valvotaan ja ohjataan eri organisaatioiden toimesta ja niiden luomilla standardeilla määritellään, miten ykköset ja nollat muodostavat TCP/IP-yhteyden. Yhteys mahdollistaa laitteiden välisen kommunikoinnin maailmanlaajuisesti. Malli koostuu neljästä pääkerroksesta alkaen fyysisestä linkkikerroksesta, joka mahdollistaa suoran datasiirron laitteiden välillä ja päätyen kuljetuskerrokseen, mikä varmistaa datan toimituksen TCP- tai UDP-protokollia käyttäen.

IP-protokollan rooli tietoliikenneyhteydessä sekä sen toimintamekanismit, kuten paketointi ja reititys ovat olennaisia osia verkon toiminnassa. IP-protokollia on kaksi: IPv4, joka kehitettiin ensin, mutta sen kanssa ilmeni ongelmia, minkä johdosta kehitettiin IPv6-protokolla.

Verkon palveluista automaattinen konfigurointi ja nimipalvelu ovat tärkeitä käytön kannalta ja ne on mahdollista toteuttaa myös avoimeen lähdekoodiin perustuvilla sovelluksilla.

2.1 Internet-protokolla

Jokainen Internetiin kytketty laite tarvitsee yksilöllisen IP-osoitteen, joka toimii kuin digitaalinen puhelinnumero tai postiosoite. IP-osoite mahdollistaa laitteiden tunnistamisen ja kommunikoinnin IP-verkossa. Osoitteita on kahta päätyyppiä: vanhempi IPv4 ja uudempi IPv6. IPv4-osoitteet otettiin käyttöön vuonna 1981 (Postel, 1981) ja ne ovat edelleen laajasti käytössä. IPv4-osoitteiden rajoitettu määrä johti kuitenkin osoitteiden loppumiseen, minkä vuoksi IETF kehitti IPv6-osoitteet (Deering & Hinden, 1998). Vaikka IPv6:n käyttöönotto on edelleen meneillään, sen käyttö kasvaa jatkuvasti.

IETF on yksi organisaatioista, joka ohjaa ja valvoo internetin ja TCP/IP-protokollaperheen kehitystä. IETF:n toiminta perustuu standardointiprosessiin, mikä alkaa luonnoksesta, johon kuka tahansa voi antaa omat ehdotuksensa ja ideansa. Tämän jälkeen samaa prosessia noudattaen siitä tehdään ehdotus, luonnos ja lopuksi virallinen standardi, mihin yleisesti viitataan RFC:nä. Kaikki RFC:t löytyvät internetistä IETF:n kotisivuilta. (Kaario, 2002)

IP-protokolla jakaa lähetettävän tiedon pieniksi osiksi, eli paketeiksi. Jokainen paketti sisältää IP-otsikon, mikä kertoo muun muassa lähettäjän ja vastaanottajan osoitteet sekä ohjeet paketin käsittelyyn (Postel, 1981). Vertauskuvallisesti, jos alkuperäinen viesti olisi pitkä kirje,

IP-protokolla jakaisi sen useisiin pieniin kirjekuoriin, joista jokainen sisältäisi osan viestistä sekä ohjeet siitä, missä järjestyksessä osat tulee koota.

Näitä paketteja välitetään lähettäjältä vastaanottajalle prosessissa, jota kutsutaan reititykseksi. Reititystä varten laitteelle täytyy konfiguroida myös lähiverkon peite ja käytettävä reititin eli oletusyhdyskäytävä. Reitityksessä verkon solmupisteet, eli reitittimet, tarkastelevat pakettien otsikoita ja ohjaavat ne seuraavaan kohteeseen (Postel, 1981). Tämä muistuttaa postijärjestelmää, jossa postilähetykset kulkevat eri postikeskusten kautta määränpäähänsä.

Vaikka IP-protokolla ei takaa pakettien saapumisjärjestystä tai sitä, että ne saapuvat perille, se sisältää mekanismeja virheiden raportointiin. Jos yhteys katkeaa, paketti katoaa tai vioittuu matkalla, IP:n sisältämä ICMP-protokolla pyrkii ilmoittamaan virheistä lähettäjälle. Mikäli paketit ovat liian suuria kohdeverkolle, IP-protokolla voi myös jakaa ja koota ne uudelleen varmistaakseen niiden pääsyn perille. (Postel, 1981)

IP-yhteyden visualisointi (Kuva 1) voidaan tehdä käyttämällä traceroute-työkalua, joka näyttää reitityspisteet, mitä pitkin paketit kulkevat lähettäjältä vastaanottajalle.

Kuva 1 reitityspisteiden selvitys traceroute-työkalulla. (*tcptraceroute(1) - Linux man page*, ei pvm.)

```
opiskelija@amk24:~$ tcptraceroute www.traficom.fi
Selected device enp1s0, address 192.168.0.49, port 33837 for outgoing packets
Tracing the path to www.traficom.fi (87.239.122.16) on TCP port 80 (http), 30 hops max
 1  192.168.0.1  3.389 ms  3.156 ms  3.542 ms
 2  91-152-80-2.elisa-laajakaista.fi (91.152.80.2)  18.589 ms  14.725 ms  17.205 ms
 3  * * *
 4  193.229.29.197  14.650 ms  13.685 ms  25.480 ms
 5  csc.ficix2.ficix.fi (193.110.224.14)  21.465 ms  16.612 ms  10.300 ms
 6  193.166.255.33  16.491 ms  8.181 ms  14.848 ms
 7  www.traficom.fi (87.239.122.16) [open]  21.523 ms  10.890 ms  26.788 ms
opiskelija@amk24:~$
```

Kuvassa näkyy, miten traceroute-työkalulla selvitetään pakettien reititys kohteeseen www.traficom.fi sivustolle. Paikallinen laite lähettää ensimmäisen paketin lähiverkon ensimmäiselle reitityspisteelle eli oletusyhdyskäytävälle, joka sitten lähettää paketin edelleen viisi kertaa, ennen kuin se pääsee perille. Jokainen näistä reitityspisteistä on yhdistetty toisiinsa linkkikerroksessa.

2.2 Linkkikerros

Linkkikerroksessa käytettävät laitteet vastaavat datan fyysisestä siirrosta ja toimivat erilaisten yhteysvälineiden, kuten kaapeleiden ja langattomien signaalien kautta. Tieto siirtyy kahden eri laitteen välillä binäärimuodossa ja paketissa on ilmoitettu lähettäjän ja vastaanottajan fyysinen eli MAC-osoite. Reaalimaailman esimerkissä nämä kaapelit ja langattomat verkot voisivat olla teitä tai ilmareittejä, joita pitkin tietoa siirretään erilaisilla välineillä. (Hall, 2000)

Fyysisellä tasolla datan siirtyminen ei kuitenkaan tapahdu IP-osoitteita käyttämällä, vaan jokaisella laitteella, jolle dataa halutaan siirtää, täytyy olla verkkolaite, jolla on fyysinen osoite, mikä tunnetaan myös MAC-osoitteena. IP-osoite ja MAC-osoite liitetään IPv4-verkoissa toisiinsa käyttämällä ARP-protokollaa. Tämä toimii käytännössä niin, että kun laite liittyy lähiverkkoon ja haluaa lähettää paketin johonkin, se kysyy ensin lähiverkon broadcast-osoitteessa, missä fyysisessä osoitteessa kohde sijaitsee. Kun se saa vastauksen, se tallentaa vastauksen välimuistiinsa. (Hall, 2000)

IPv6 ei käytä ARP-protokollaa verkon sisällä fyysisen osoitteen etsimiseen, vaan sitä varten kehitettiin NDP. Kun laite ei tiedä, onko tietty IP-osoite varattu tai kenelle se kuuluu, laite lähettää IPv6 Neighbor Solicitation -viestin verkkoon, missä se kysyy, kenelle kyseinen IP-osoite kuuluu. Jos kyseinen osoite on varattu, laite, jolle se kuuluu, vastaa viestiin Neighbor Advertisement -viestillä. NDP:ssää on myös muita uusia toimintoja: Router Solicitation -viestiä käytetään verkon reitittimien löytämiseen ja reitittimet vastaavat näihin viesteihin Router Advertisement -toiminnolla. Kyseisiä toimintoja tarvitaan verkon dynaamiseen konfigurointiin, jonka tarkoituksena on helpottaa IPv6 käyttöönottoa, sillä osoitteita ei tarvitse konfiguroida manuaalisesti eikä niitä varten tarvitse pystyttää DHCP-palvelua. (Narten ym., 2007)

2.3 TCP ja UDP

IP-protokollan tehtävänä on toimittaa ja pilkkoa paketit. Data paketoidaan kuljetuskerrokseen, jossa käytetään joko TCP- tai UDP-protokollaa. Sovellus tai palvelu käyttää näistä sovellukselle sopivaa protokollaa avaamalla paikallisen portin, jonka kautta tietoa pyydetään siirrettäväksi sovelluksen määrittelemään kohteeseen. Tämän jälkeen tieto pilkotaan osiin ja lähetetään kohti kohdetta pienissä paketeissa, jotka sisältävät sekä IP-protokollan että UDP- tai TCP-protokollan lisäämän osion. (Hall, 2000)

UDP on näistä protokollista yksinkertaisin. Sen otsikossa ilmoitetaan vain lähde- ja kohdeportit, dataosion pituus ja tarkistusnumero. Tämä tekee siitä kevyen ja nopean, mutta myös epäluotettavan sillä protokolla ei varmista, että sen lähettämä paketti menee perille. (Eggert ym., 2017)

TCP-protokollan tehtävä on varmistaa, että paketti saapuu varmasti vastaanottajalle. Tämä tapahtuu siten, että ennen kuin protokolla edes yrittää lähettää mitään tietoa, se suorittaa kättelyn vastaanottajan kanssa varmistaen, että yhteys on mahdollinen. Dataa siirrettäessä protokolla lisää lähettäjän jokaiseen viestiin sekvenssinumeron, johon se odottaa saavansa kuittauksen vastaanottajalta. Sekvenssinumeron perusteella vastaanottaja pystyy kokoamaan datan, koska joskus paketit kiertävät vastaanottajalle hitaampaa reittiä. Mikäli kuittausta ei kuulu, lähettäjän protokolla yrittää lähettää paketin uudelleen. Näillä mekanismeilla TCP-yhteyksistä saadaan kahden koneen välille varmempi, mutta samalla myös raskaampi ja monimutkaisempi yhteys. TCP on tilallinen protokolla, mikä tarkoittaa, että se pitää kirjaa yhteyden tilasta ja varmistaa, että kaikki data päätyy varmasti perille. (Postel, 1981)

Reaalimaailman esimerkissä TCP-paketti voisi olla kuin kirjattu kirje, joka vaatii vastaanottajan kuittauksen, varmistuksen jokaisessa kuljetuksen vaiheessa ja varmuuden siitä, että kirje saapuu perille. Tämä tarkoittaa, että lähettäjä saa tiedon kirjeen vastaanottamisesta ja jos kirje katoaa matkalla, se lähetetään uudelleen.

UDP taas voisi olla tavallinen kirje, joka pudotetaan postilaatikkoon ilman kuittausta tai seurantaa. Tällainen kirje voi saapua perille nopeasti ja vähin kustannuksin, mutta on mahdollista, että se katoaa matkalla, eikä lähettäjä saa tietoa kirjeen toimituksesta.

2.4 IPv4 ja IPv6

IPv4-osoite esitetään numeerisessa muodossa. Se koostuu neljästä erillisestä numerosarjasta, jotka sijoittuvat välille 0–255. Nämä sarjat erotetaan toisistaan pisteellä. Vaikka esitämme IP-osoitteet numeerisesti, tietokoneet käsittelevät tietoa ainoastaan bittien muodossa, joissa jokainen bitti voi olla joko 0 tai 1. IPv4-osoitteessa yksi numerosarja on 8-bittinen ja kun näitä on neljä, muodostuu yhteensä 32-bittinen osoite. Esimerkiksi osoitteen 190.168.0.1 binaarinen muoto olisi 11000000101010000000000000000001, mikä olisi hyvin vaikea lausua tai muistaa. Koska jokainen bitti voi saada arvot 0 tai 1 on mahdollisia yhdistelmiä yhteensä 2^32 (4,294,967,296), eli noin 4,3 miljardia. (Postel, 1981)

Internetin nopean kasvun myötä vapaat osoitteet ovat kuitenkin käyneet vähiin. Tätä voi verrata reaalimaailman postijärjestelmään, jossa käytettävissä olevat postilaatikot loppuisivat, eikä uusia olisi saatavilla. (Bush, 2011)

Väliaikaisena ratkaisuna IPv4-osoitteiden loppumiseen kehitettiin NAT-standardi. Standardin avulla IPv4-osoitteelle tuleva liikenne voidaan uudelleenohjata toiselle IPv4-osoitteelle paikallisverkon sisällä. Tämän ansiosta paikallisverkossa olevat osoitteet, jotka eivät reitity internettiin voivat tarjota internettiin palveluita. Reaalimaailman esimerkkinä voidaan käyttää isoa toimistoa, jossa työntekijöillä on omat huoneet ja postilaatikot. Tässä tapauksessa toimiston sisäinen postipalvelu vastaanottaa postit ja toimittaa ne perille. (Egevang & Francis, 1994)

IPv6-protokolla kehitettiin lopullisena ratkaisuna IPv4-osoitteiden loppumiseen. IPv6-osoite koostuu 128 bitistä ja esitetään heksadesimaalimuodossa, joka on jaettu kahdeksaan 16-bittiseen osoitteeseen. Esimerkiksi osoite 2001:0db8:0000:0000:0000:0000:1428:57ab binaarimuodossa olisi 00100000000000010000110110111000000000000000000000000000000000000000000000000000000000000000101001000101111010111011 ja tämä mahdollistaa 2^128 (noin 340 sekstiljoonaa) erilaista osoitetta. (Deering & Hinden, 2017)

Koska IPv6-osoite on niin pitkä ja se esitetään heksadesimaalina, se tekee niistä huomattavasti pidempiä ja monimutkaisempia kuin IPv4-osoitteet. Tämän vuoksi IPv6:ssa on kehitetty tapoja tiivistää osoitteita, jotta ne olisivat helpompia kirjoittaa ja lukea. (Deering & Hinden, 2017)

IPv6-osoitteessa saattaa esiintyä yksi tai useampi ryhmä, joka koostuu pelkästään nollista. Tällaiset ryhmät voidaan korvata yhdellä kaksoispisteparilla (::). Tämä sääntö voidaan soveltaa vain kerran osoitteessa, jotta osoitteen yksikäsitteisyys säilyy. Esimerkiksi osoite 2001:0db8:0000:0000:0000:0000:1428:57ab voidaan tiivistää muotoon 2001:db8::1428:57ab. (Deering & Hinden, 2017)

Kukin 16-bittinen osio (neljä heksadesimaalinumeroa) IPv6-osoitteessa voidaan kirjoittaa ilman johtavia nollia. Näin osoite 2001:0db8:0000:0000:0000:0000:1428:57ab voidaan kirjoittaa myös muodossa 2001:db8:0:0:0:0:1428:57ab. (Deering & Hinden, 2017)

Vaikka heksadesimaalijärjestelmässä voidaan käyttää sekä isoja että pieniä kirjaimia, tyypillisesti IPv6-osoitteet kirjoitetaan käyttäen pieniä kirjaimia, jotta niitä olisi ihmisen helpompi lukea. (Deering & Hinden, 2017)

2.5 Siirtymäkausi

Siirtymäkaudella tarkoitetaan ajanjaksoa, jolloin IPv4 ja IPv6 ovat käytössä rinnakkain. Tämä johtuu siitä, että protokollat eivät ole yhteensopivia keskenään, joten monet teknologiat, protokollat, laitteet ja palvelut on mukautettava toimimaan oikein uudella IPv6-protokollalla. Tämän vuoksi protokollien on toimittava rinnakkain niin kauan kuin IPv4-laitteita on olemassa.

IETF on kehittänyt useita siirtymäteknologioita helpottamaan IPv6:n käyttöönottoa. Näihin kuuluvat kaksoispinotekniikka, kapselointi ja käännöstekniikat, jotka mahdollistavat IPv4- ja IPv6-protokollien rinnakkaisen käytön siirtymävaiheessa. Kaksoispinotekniikassa laitteet voivat käyttää molempia protokollia ja kapselointi mahdollistaa IPv4-pakettien siirtämisen IPv6-verkon kautta. Käännöstekniikat puolestaan muuntavat paketit IPv4:stä IPv6:een ja päinvastoin, jolloin erilaisten protokollien välinen liikenne on mahdollista. (Al-hamadani & Lencse, 2021)

2.5.1 Kaksoispino

Kaksoispinolla tarkoitetaan verkko- tai päätelaitetta, joka tukee samaan aikaan IPv4- ja IPv6-protokollia. Tämä on mahdollista saavuttaa konfiguroimalla koko käytössä oleva lähiverkko niin, että kaikille laitteille annetaan sekä IPv4- että IPv6-osoite. Yhdellä laitteella voi olla käytössään useampi osoite ja käyttöjärjestelmä määrittää, mitä lähdeosoitetta käytetään. Koska IPv6-protokollaa halutaan saada laajempaan käyttöön, käyttöjärjestelmät yrittävät ensisijaisesti muodostaa yhteyden käyttäen IPv6-protokollaa ja vasta toissijaisesti IPv4-protokollaa. (Al-hamadani & Lencse, 2021)

Mekanismi voi kuitenkin johtaa merkittäviin viiveisiin viestinnässä, kun isäntä yrittää muodostaa yhteyden etäpalvelimeen. Isäntä aloittaa DNS-pyynnön, saa sekä IPv4- että IPv6-osoitteet ja valitsee oletusarvoisesti IPv6-osoitteen. Jos IPv6-yhteys epäonnistuu, se vaihtaa IPv4:ään, mikä aiheuttaa viivettä. Happy Eyeballs -menetelmä parantaa tätä prosessia aloittamalla useita asynkronisia yhteysyrityksiä ja valitsemalla ensimmäisen onnistuneen yhteyden. Kaksoispino lisää kuitenkin monimutkaisuutta, koska sekä IPv4- että

IPv6-protokollia on ylläpidettävä, eikä se ratkaise IPv4-osoitteiden loppumisen ongelmaa. (Al-hamadani & Lencse, 2021)

2.5.2 Kapselointi

Kapselointitekniikka tarkoittaa menetelmää, jossa yhden verkkoprotokollan, kuten IPv4:n, paketti upotetaan toisen verkkoprotokollan, esimerkiksi IPv6:n, paketin sisälle. Käytännössä alkuperäinen paketti säilytetään kokonaisuudessaan ja siihen lisätään uuden protokollan otsikko, mikä mahdollistaa paketin lähettämisen vain uutta protokollaa tukevan verkon kautta (Al-hamadani & Lencse, 2021). Kapselointitekniikoita on useita ja tässä on katsaus niihin:

Manuaalisella tunneloinnilla tarkoitetaan, että IPv6-isännät yhdistetään toisiinsa IPv4-infrastruktuurin kautta kapseloimalla IPv6-liikenne IPv4-paketteihin. Molempien päätepisteiden on oltava kaksoispinomuotoisia ja niiden konfigurointi tehdään manuaalisesti. (Al-hamadani & Lencse, 2021)

6to4 ratkaisee manuaalisen tunneloinnin ongelmat sallimalla IPv6-isännille, joilla on julkinen IPv4-osoite, kommunikoida ilman eksplisiittistä tunnelin asetusta. Se tukee myös viestintää IPv6-isäntien ja alkuperäisten IPv6-toimialueiden välillä rele-reitittimien kautta. (Al-hamadani & Lencse, 2021)

Teredo mahdollistaa IPv4-isäntien, joilla ei ole julkista IPv4-osoitetta ja jotka sijaitsevat NAT takana, yhdistämisen IPv6-solmuihin tunneloimalla IPv6-paketit UDP:n yli. Tämän tehtävän suorittamiseksi Teredo käyttää kahta laitetyyppiä: Teredo-palvelinta ja Teredo-releettä. Teredo-palvelin vastaa Teredo-tunnelin konfiguroinnista, kun taas Teredo-rele toimii IPv6-reitittimenä ja vastaa liikenteen välittämisestä Teredo-asiakkaiden välillä. Jokaiselle Teredo-isännälle annetaan IPv6-osoite, joka alkaa erityisellä palveluprefiksillä (2001:0000:/32). Teredo on pääasiassa kehitetty tarjoamaan ”viimeisen keinon" vaihtoehto solmuille, jotka haluavat yhdistyä IPv6-internetiin, eikä muita IPv6-siirtymäteknologioita ole otettu käyttöön. (Al-hamadani & Lencse, 2021)

ISATAP on suunniteltu yhdistämään kaksoispinosolmut IPv6-solmuihin IPv4-verkkojen kautta automaattista tunnelointia käyttäen. ISATAP-isäntä muodostaa IPv6-osoitteen käyttämällä ennalta määritettyä etuliitettä ja Ipv4-osoitetta. (Al-hamadani & Lencse, 2021)

6rd on johdettu 6to4-teknologiasta, mutta sallii ISP:n käyttää omaa etuliitettään. Tämä antaa ISP:lle enemmän hallintaa verkkoonsa. 6rd perustuu algoritmiseen kartoitukseen IPv6- ja IPv4-osoitteiden välillä. (Al-hamadani & Lencse, 2021)

Tunnel Broker hallitsee IPv4-tunneleita IPv6-isännille käyttämällä omistettuja palvelimia, jotka käsittelevät tunnelipyyntöjä automaattisesti. Tämä helpottaa IPv6-isäntien määrän kasvattamista. (Al-hamadani & Lencse, 2021)

DS-Lite helpottaa IPv6:n asteittaista käyttöönottoa yhdistämällä IPv4- ja IPv6-IP-in-IP-kapseloinnin sekä tilallisen NAPT:n. Se käyttää B4- ja AFTR-laitteita liikenteen kapselointiin ja dekapselointiin. (Al-hamadani & Lencse, 2021)

MAP-E käyttää tilatonta algoritmia IPv4-osoitteiden määrittämiseen ja upottaa nämä tiedot IPv6-etuliitteeseen. MAP-E mahdollistaa viestinnän yhden MAP-alueen sisällä joko suoraan tai BR:n kautta. (Al-hamadani & Lencse, 2021)

Lw4o6 on DS-Lite:n laajennus, joka siirtää NAPT-toiminnon keskitetystä AFTR:stä hajautetuille B4-laitteille. Tämä vähentää kirjauskuormitusta ja vapauttaa AFTR:n käännöstehtävistä. Lw4o6 tukee myös suoraa viestintää kahden lwB4-laitteen välillä. (Al-hamadani & Lencse, 2021)

2.5.3 Käännöstekniikat

Käännöstekniikat jaetaan kahteen pääkategoriaan, joista ensimmäinen on yksittäinen käännös. Tässä menetelmässä esimerkiksi IPv4-paketit muunnetaan suoraan IPv6-paketeiksi. Esimerkkejä ovat NAT64 ja DNS64, joissa IPv6-asiakkaiden liikenne muunnetaan IPv4-palvelimille sopivaksi. Tämä mahdollistaa suoran kommunikoinnin IPv6- ja IPv4-verkkojen välillä. Esimerkiksi, kun IPv6-asiakas haluaa käyttää IPv4-palvelinta, NAT64 muuntaa IPv6-paketit IPv4-paketeiksi. (Al-hamadani & Lencse, 2021)

Toinen pääkategoria on kaksoiskäännös, jossa paketti muunnetaan kahdesti, ensin yhdestä protokollasta toiseen ja sitten takaisin alkuperäiseen. Esimerkiksi 464XLAT-teknologiassa käytetään kahta laitetta, CLAT ja PLAT, jotka mahdollistavat sekä IPv4- että IPv6-liikenteen käsittelyn verkossa. Ensimmäinen käännös tapahtuu asiakkaan CLAT-laitteessa ja toinen käännös operaattorin PLAT-verkossa, mikä mahdollistaa monimutkaisemmat siirtymävaiheet ja eri protokollien rinnakkaiskäytön. (Al-hamadani & Lencse, 2021)

2.5.4 Tekniikoiden vertailu

Széchenyi István -yliopisto analysoi IPv6-siirtymäkauden tekniikoita artikkelissa *Survey on the Performance Analysis of IPv6 Transition Technologies* . Suorituskykyä mitattiin läpäisykyvyllä, viiveellä ja pakettihäviöllä. Läpäisykyvyllä tarkoitetaan sitä, kuinka paljon dataa pystytään siirtämään tietyn ajan kuluessa. Viive mittaa ajan, joka kuluu datapaketin kulkemiseen lähettäjältä vastaanottajalle ja pakettihäviö kuvaa menetettyjen datapakettien määrää verkossa. (Al-hamadani & Lencse, 2021)

Suorituskykytestit osoittivat, että manuaalinen tunnelointi ja 6to4-teknologiat tarjosivat parhaan läpäisykyvyn ja alhaisimmat viiveet. Toisaalta kaksoispino -tekniikka osoittautui suorituskyvyltään heikoksi, sillä se tuotti vähiten läpäisykykyä ja korkeimmat viiveet tässä testivaiheessa. MAP-E-tekniikka puolestaan saavutti parhaan kokonaisverkkosuorituskyvyn. Käännöspohjaiset teknologiat, kuten 464XLAT ja MAP-T, osoittivat parempaa suorituskykyä viiveen osalta, kun taas kapselointipohjaiset teknologiat, kuten MAP-E ja DS-Lite, pärjäsivät läpäisykyvyssä. (Al-hamadani & Lencse, 2021)

Simulaatiot toivat esille, että 6to4-teknologia tarjosi parhaat tulokset, koska sillä oli alhaisin viive ja pakettihäviö, sekä korkein läpäisykyky. NAT-PT puolestaan osoittautui suorituskyvyltään heikoimmaksi kaikilla mittareilla. ISATAP ja 6to4 osoittivat samankaltaista suorituskykyä, kun taas NAT64 tarjosi parhaat tulokset yksisuuntaisen viiveen ja läpäisykyvyn osalta. (Al-hamadani & Lencse, 2021)

GNS3-simulaattorilla tehdyt testit osoittivat kuitenkin, että kaksoispino saavutti korkeimman läpäisykyvyn ja alhaisimman RTT:n, mutta samalla se kulutti eniten CPU-resursseja. 6to4 puolestaan osoitti tässä testissä heikointa suorituskykyä. Lisäksi kaksoispino rekisteröi parhaan suorituskyvyn kokonaisuutena, mutta se vaati enemmän prosessoritehoa kuin muut vaihtoehdot. NAT-PT oli edelleen heikoin kaikissa mittareissa. (Al-hamadani & Lencse, 2021)

NAT64 analysoitiin tehokkaammaksi kuin NAT-PT erityisesti pienissä verkoissa tai verkoissa, joissa on natiivisti IPv6-yhteys IPv4-palvelimeen. Lopputestit osoittivat, että ISATAP-tekniikka saavutti parhaan suorituskyvyn kaikilla osa-alueilla, kun sitä testattiin todellisessa ympäristössä verrattuna muihin teknologioihin. (Al-hamadani & Lencse, 2021)

3 DNS- ja DHCP-verkkopalvelut

Automaattinen verkon konfigurointi on tekniikka, jonka avulla verkkolaitteiden asetukset automatisoidaan käyttämällä joko DHCP-protokollaa tai IPv6:n mukana tullutta SLAAC-tekniikkaa. Lisäksi verkon käyttämiseen tarvitaan nimipalvelu, jonka perustehtävä on muuntaa osoitteet nimiksi.

3.1 Verkon asiakaslaitteiden automaattinen konfigurointi

Jokainen lähiverkkoon kytketty laite tarvitsee tietoliikenneyhteyttä varten IP-osoitteen, lähiverkon peitteen sekä oletusyhdyskäytävän. Näiden asetusten määrittäminen ja ylläpitäminen manuaalisesti olisi työlästä, koska jokaisen älylaitteen verkkoyhteys pitäisi ensin kirjata ylös ja tämän jälkeen konfiguroida laitteelle erikseen. Tämä olisi tehtävä myös jokaisessa verkossa, jossa älylaite mahdollisesti vierailee. Tämän helpottamiseksi on luotu DHCP, joka automatisoi näiden asetusten jakamisen asiakas-palvelin-mallia hyödyntäen. Tämä prosessi on nykyajan laitteissa automatisoitu tapahtumaan tietokoneen käynnistyksen yhteydessä tai, jos kyseessä on matkapuhelin, wifi-yhteyden käynnistyessä, jolloin laitteet liittyvät lähiverkkoon. (Droms, 1997)

IPv4-verkoissa DHCP:n toimintaperiaate on seuraava:

1. DHCP-Discover viestissä asiakaslaite lähettää verkkoon broadcast-viestin tavoittaakseen DHCP-palvelimen. Broadcast-osoite on verkon viimeinen osoite, minkä kaikki verkon laitteet näkevät. (Droms, 1997)

2. DHCP-palvelin seuraa broadcast-osoitteen viestejä ja tunnistaa DHCP-Discover-pyynnöt. Tämän perusteella se tarjoaa asiakkaalle verkon konfiguraatiota DHCP-Offer viestillä. (Droms, 1997)

3. Asiakaslaite valitsee tarjotuista osoitteista yhden ja lähettää siitä kuittauksen DHCP-palvelimelle DHCP-Request viestillä. (Droms, 1997)

4. DHCP-palvelin vahvistaa IP-osoitteen myöntämisen lähettämällä DHCP-Acknowledge -viestin, minkä jälkeen asiakaslaite voi ottaa asetukset käyttöön. (Droms, 1997)

IPv6-verkoissa toimintaperiaate on hieman erilainen:

1. IPv6-verkoissa ei ole broadcast-osoitetta, joten DHCPv6-asiakas lähettää solicit-pyynnön multicast-osoitteisiin. (Mrugalski ym., 2018)

2. DHCPv6-palvelin vastaa solicit-viestiin advertise-viestillä, mikä sisältää tarjouksen verkon konfiguraatiosta. (Mrugalski ym., 2018)

3. Asiakas vastaa tähän tarjoukseen request-viestillä ilmoittaakseen tarjouksen hyväksymisestä. (Mrugalski ym., 2018)

4. DHCPv6-palvelin vahvistaa asiakkaan pyynnön reply-viestillä. (Mrugalski ym., 2018)

DHCP-palvelimille jokainen verkko, mistä IP-osoitteita voidaan jakaa on konfiguroitu erikseen. Konfiguraatiossa määritetään tietoliikenneyhteyteen tarvittavat asetukset: IP-osoite, aliverkon peite ja oletusyhdyskäytävä. Näiden lisäksi tarvitaan myös DNS-palvelimen asetukset, jotta laite voi käyttää nimiä yhteyksien muodostamiseen. Verkot määritellään konfiguraatiossa kokonaisina verkkoina, jotka voivat sisältää useita satoja osoitteita. Kun osoite annetaan verkosta, DHCP-palvelin kirjoittaa varatun tiedon leasing-tietokantaan, eli osoite annetaan vuokralle määräajaksi. Kun määräaika on loppumassa, asiakaslaitteen on pyydettävä vuokrasopimuksen jatkoa, jolloin palvelin voi hyväksyä sen ja päivittää tiedot tietokantaansa. (Kea Administrator Reference Manual, ei pvm.)

Pakollisia tietoja verkon toimivuuteen ovat IP-osoite, aliverkon peite ja oletusyhdyskäytävä. Näillä tiedoilla tietokone tai mikä tahansa älylaite pystyy kommunikoimaan IP-verkon kanssa. Nämä pakolliset tiedot tunnetaan DHCP:ssä termillä optio, ja jokaisella optiolla on oma koodinsa, joka on määritelty RFC 2132 -standardissa. Yhteensä koodeja on 255, ja verkon konfiguroinnin yhteydessä on mahdollista määrittää verkon laitteelle sen rooli. Esimerkiksi DOCSIS-kaapelimodeemeille voidaan DHCP:n avulla antaa lisää laitteen tarvitsemaa tietoa, kuten aikapalvelimen, DNS-palvelimen, lokipalvelimen, laitteen nimen, alueen nimen, TFTP-palvelimen osoitteen ja laitteen konfiguraatiotiedoston nimi. (*Fifty Shades of DOCSIS Device Management* | *Axiros*, ei pvm.)

Toisena esimerkkinä käytetään tulostinta, mikä tarvitsee kiinteän osoitteen verkossa, jotta se löydetään nimen perusteella. Tällöin on tärkeää, että DHCP antaa kyseiselle laitteelle aina saman osoitteen, jotta nimipalvelimet pystyvät sen ratkomaan. DHCP-palvelimelle pitää konfiguroida kyseisen laitteen MAC-osoitteelle staattinen IP-osoite, minkä johdosta laitteen

IP-osoite ei muutu. Koska IP-osoite ei vaihdu, kyseinen IP-osoite voidaan lisätä DNS:n puurakenteeseen, jolloin osoitetta ei tarvitse enää muistaa vaan tulostettavaa aineistoa voi lähettää nimellä. (*Setting the printer's IP addressing parameters*, ei pvm.)

DHCPv4:n ja DHCPv6:n eroavaisuuksista oleellisin on oletusyhdyskäytävän konfiguraatio. Kyseistä optiota ei ole DHCPv6:ssa vaan laitteet tarvitsevat reitittimeltä tiedon reitti-ilmoituksella. (Mrugalski ym., 2018)

DHCPv6-osoitteissa on myös lisäajastin nimeltään preferred-lifetime, jota ei ole olemassa DHCPv4:ssä. Asetus mahdollistaa pidemmät vuokra-ajat ja T1/T2-ajastimet. Pidemmän T1-ajastimen käyttö vähentää DHCP-liikennettä verkossa, mikä voi olla hyödyllistä suorituskykyyn liittyvien ongelmien ehkäisemiseksi. T1 ja T2 ovat ajastimia, mitkä määrittelevät DHCPv6-asiakkaan uusimisprosessin aikataulun. T1-ajastin on ajankohta, jolloin DHCPv6-asiakas aloittaa vuokra-ajan uusimisen samalta DHCP-palvelimelta, miltä se sai alkuperäisen vuokran. Jos asiakas saa vastauksen ennen T2-ajastimen umpeutumista, prosessi päättyy tähän. T2-ajastinta käytetään, jos asiakas ei ole saanut vastausta T1-ajastimen aikana; tällöin se jatkaa uusimista T2-ajastimen umpeutumiseen asti. T2-ajastimen umpeuduttua asiakas yrittää uudelleen uusimista kaikilta saatavilla olevilta DHCP-palvelimilta. (Mrugalski ym., 2018)

3.2 SLAAC

SLAAC on tekniikka, joka mahdollistaa laitteiden automaattisen IPv6-osoitteiden määrittämisen ilman manuaalista konfigurointia tai erillisiä palvelimia. Laitteet luovat itse IPv6-osoitteensa yhdistämällä paikallisesti saatavilla olevan tiedon ja reitittimien mainostamat tiedot. Reitittimet lähettävät verkon etuliitteitä ja laitteet lisäävät näihin oman tunnisteensa muodostaakseen osoitteen. Jos reitittimiä ei ole laitteet voivat luoda vain linkkikohtaisia osoitteita, mitkä toimivat vain saman verkon laitteiden välillä. (Thomson ym., 2007)

SLAAC-prosessissa laite kuuntelee reitittimen lähettämiä Router Advertisement (RA) -viestejä, jotka sisältävät verkon etuliitteen ja mahdolliset muut asetukset. Näiden tietojen pohjalta laite voi luoda yksilöllisen IPv6-osoitteensa käyttämällä joko omaa MAC-osoitettaan tai satunnaistettua tunnistetta. Tämä dynaaminen osoitteenmuodostus vähentää manuaalisen konfiguroinnin tarvetta ja tekee laitteiden verkkoon liittämisestä helppoa ja nopeaa. (Thomson ym., 2007)

Lisäksi SLAAC tukee osoitteiden elinkaaren hallintaa. Reitittimet voivat asettaa osoitteille enimmäiskäyttöajan, jonka jälkeen laitteet joko päivittävät osoitteensa tai poistavat ne käytöstä. Tämä auttaa pitämään osoiteavaruuden ajantasaisena ja ehkäisee vanhentuneiden osoitteiden kertymistä verkkoon, mikä on tärkeää suurten IPv6-verkkojen hallinnan kannalta. (Thomson ym., 2007)

Jos verkossa on käytössä DHCPv6, SLAAC ja DHCPv6 voivat täydentää toisiaan: SLAAC hoitaa perus-IPv6-osoitteiden luomisen, ja DHCPv6 voi toimittaa lisäasetuksia, kuten DNS-palvelinten osoitteet. Tämä mahdollistaa monipuolisen ja joustavan verkonhallinnan, jossa osoitteiden jakelu ja asetukset voidaan mukauttaa tarpeen mukaan. (Thomson ym., 2007)

3.3 Nimipalvelu

Nimipalvelintyyppejä on erilaisia ja niistä yleisimmät ovat rekursiivinen ja autoritatiivinen. Lisäksi nimipalvelimet voivat tukea DNSSEC-protokollaa. Nslookup-komennolla voidaan havainnollistaa, miten tietokone suorittaa nimikyselyn selvittääkseen verkkosivuston IP-osoitteen seuraamalla nimipalvelun puurakennetta.

3.3.1 Nimenselvityksen ketjureaktio

Lähiverkkoyhteyksissä yhteys muodostetaan kahden IP-osoitteen välillä. Ihmisten on kuitenkin vaikea muistaa ja lausua pitkiä numerosarjoja, joten tämä haaste ratkaistiin muuntamalla IP-osoitteet helpommin muistettaviksi nimiksi. Aiemmin osoitteet ja nimet yhdistettiin toisiinsa hosts-tiedostossa, mitä levitettiin kopioimalla tiedostoa laitteelta toiselle esimerkiksi sähköpostin tai levykkeiden avulla. Internetin kasvaessa tämä menetelmä kävi kuitenkin riittämättömäksi, jonka seurauksena kehitettiin DNS. (Mockapetris, 1987)

Esimerkiksi kun avaat selaimessa www.facebook.com-sivuston, tietokoneesi suorittaa taustalla DNS-kyselyn, jonka tarkoitus on selvittää kyseisen nimen IP-osoite.

Komento 1 nslookup-komento, jolla suoritetaan absoluuttinen DNS-kysely

```
nslookup www.facebook.com.
Server:         1.1.1.1
Address:        1.1.1.1#53

Non-authoritative answer:
www.facebook.com    canonical name = star-mini.c10r.facebook.com.
Name:   star-mini.c10r.facebook.com
Address: 157.240.205.35
```

```
Name:   star-mini.c10r.facebook.com
Address: 2a03:2880:f113:81:face:b00c:0:25de
```

Komennossa suoritetaan nslookup-työkalulla absoluuttinen nimikysely ja vastaukseksi saadaan IPv4- ja IPv6-osoite. Kyselystä tekee absoluuttisen sen perään lisätty piste, mikä estää käyttöjärjestelmää lisäämästä osoitteen perään mitään ylimääräistä. Selain siis muodostaa todellisuudessa yhteyden IPv6-osoitteeseen https://2a03:2880:f113:81:face:b00c:0:25de tai IPv4-osoitteeseen https://157.240.205.35 riippuen siitä onko IPv6-protokolla käytössä. Esimerkissä vastaus tulee DNS-nimipalvelimelta 1.1.1.1, joka on Cloudflaren ilmainen nimipalvelu internetissä.

Nimipalvelintyyppejä on tyypillisesti kaksi: rekursiivinen ja autoritatiivinen. Esimerkissä Cloudflaren 1.1.1.1 on rekursiivinen palvelin, jonka tehtävä on selvittää vastaus käyttäjän pyyntöön. Vastauksen selvitys tapahtuu ketjureaktiolla, mitä kutsutaan rekursioksi. Rekursiossa nimipalvelin seuraa DNS:n puurakennetta ja suorittaa useita kyselyitä vastauksen selvittämiseksi. (Mockapetris, 1987)

3.3.2 Nimipalvelun puurakenne

DNS:n puurakenne on kuin reaalimaailman jättimäinen yhteystietokanta, mistä löytyvät jokaisen ihmisen yhteystiedot, kuten esimerkiksi sähköpostiosoitteet, puhelinnumerot ja henkilötunnukset. Puurakenteessa ylimpänä on root-palvelin, jonka tehtävä on kertoa, missä TLD-palvelimet sijaitsevat. Tätä kutsutaan delegoinniksi, jossa juuresta alaspäin tietoa delegoidaan useaan oksaan, missä voi olla lisää haaroja, jotka delegoivat tietoa edelleen. Reaalimaailman esimerkissä yhteystietokannassa root-palvelimen sivulla olisi tiedot siitä, mistä löytyvät maiden ja alueiden paikalliset eli TLD-tiedot. Näillä paikallisilla sivuilla olisi lisää delegointeja, jotka kertovat, missä alueen eri organisaatiot ja yritykset sijaitsevat. Jokainen organisaatio ja yritys puolestaan voi delegoida tietoa edelleen omille alisivuilleen, jotka voivat sisältää esimerkiksi eri osastojen, palveluiden tai yksittäisten henkilöiden tietoja. Delegoinnin tarkoituksena on paikantaa tarkasti ja nopeasti haluttu verkko-osoite tai palvelu. (Mockapetris, 1987)

Käytännössä tämä tarkoittaa, että kun kirjoitat selaimeesi esimerkiksi URL-osoitteen www.facebook.com, älylaitteesi suorittaa DNS-kyselyn verkkosi nimipalvelimelle, joka käynnistää rekursion selvittääkseen vastauksen kyselyyn. Root-palvelin ilmoittaa, missä .com-TLD-palvelin sijaitsee ja TLD-palvelin puolestaan ohjaa kyselyn oikealle palvelimelle, joka tuntee www.facebook.com:n tarkemman osoitteen. Rekursiivinen

nimipalvelin seuraa DNS-puuta tarpeeksi pitkälle löytääkseen lopullisen vastauksen käyttäjän kysymykseen. Saatuaan vastauksen se lähettää sen takaisin käyttäjälle. (Mockapetris, 1987)

Tieto sijaitsee zone-tiedostossa, joka sisältää palvelusta riippuen tarvittavat tiedot. Root-palvelinten zone-tiedostot sisältävät TLD-palvelinten osoitteet ja TLD-palvelinten zone-tiedostoissa on tiedot kyseisen domainin, kuten .com-domainin, alidomaineista. Facebookin zone-tiedosto sisältää tiedot kaikista kyseisessä domainissa tarjolla olevista osoitteista ja palveluista. Esimerkiksi, jos haetaan www-palvelimen osoitetta, vastaus löytyy tästä tiedostosta. Lisäksi zone-tiedosto voi sisältää tietoa muista palveluista, kuten facebook.com-domainin sähköpostipalvelimesta. (Mockapetris, 1987)

Tietotyypeistä yleisimmät ovat A-tietue, joka pitää sisällään IPv4-osoitteen ja AAAA-tietue, joka sisältää IPv6-osoitteen. CNAME on alias-tietue, eli esimerkissä www.facebook.com on alias star-mini.c10r.facebook.com-osoitteelle. Toinen yleinen tietuetyyppi on MX-tietue, joka sisältää alueen sähköpostipalvelimen nimen. DNS-puun hierarkiaan liittyviä tietueita ovat NS-tietue, josta näkyy nimipalvelimien osoitteet kyseiselle domainille ja SOA-tietue, josta löytyy domainiin liittyviä tietoja, kuten yhteyshenkilön sähköpostiosoite. (Mockapetris, 1987)

DNS:n puurakenne perustuu delegointiin, jossa kysymykseen vastataan joko antamalla autoritatiivinen vastaus tai kertomalla, mistä kyseinen vastaus pitäisi löytyä. Tieto voidaan delegoida useampaan osoitteeseen samanaikaisesti, mikä mahdollistaa DNS:n skaalautumisen. Palvelimien lukumäärän lisääminen puurakenteeseen parantaa myös toimintavarmuutta, sillä rekursiossa riittää, että yksi palvelin antaa vastauksen omalta alueeltaan. Mikäli puurakenteessa on kuitenkin alue, jonka autoritatiivisiin palvelimiin ei saada yhteyttä, kysymykseen ei saada vastausta, jolloin kyseessä on vikatilanne. (Barr, 1996)

Esimerkiksi Facebookin nimipalvelimet lakkasivat toimimasta 10.4.2021, joka johtui siitä, että verkko missä ne sijaitsivat kärsi reititysongelmista. Vaikka DNS-palvelimissa ei ollut mitään vikaa kyseessä oli silti DNS-ongelma, koska kaikki Facebookin käyttäjät yrittivät päästä palveluun omien operaattoreidensa kautta, mutta nimipalvelimet eivät saaneet vastausta Facebookilta. Facebookin suosion johdosta käyttäjät yrittivät jatkuvasti käyttää palveluja, mikä johti joissain tapauksissa heidän paikallisten nimipalvelimiensa ylikuormittumiseen. Paikallisten palvelimien ylikuormitus saattoi näkyä viiveinä myös muissa palveluissa. (*Understanding How Facebook Disappeared from the Internet*, 2021)

3.3.3 DNSSEC

DNS-välimuistin myrkyttäminen on tietoturvauhka, jossa rekursiivisen DNS-ratkaisijan välimuistiin tallennetaan haitallista tietoa, jolla hyökkääjä pyrkii ohjaamaan käyttäjän väärään osoitteeseen. Käytännössä tämä tarkoittaa, että kun käyttäjä vierailee esimerkiksi www.facebook.com-sivustolla, DNS-vastauksessa palautetaan myös virheellinen tieto siitä, missä www.google.com sijaitsee. Seuraavan kerran, kun samaa rekursiivista nimipalvelinta käytetään selvittämään www.google.comin sijaintia, käyttäjä saa palvelimen välimuistista väärän osoitteen, joka on usein haitallinen. (Shulman & Waidner, 2014)

DNSSEC kehitettiin ratkaisuksi tähän ongelmaan ja se perustuu jokaisen DNS-alueen digitaaliseen allekirjoitukseen. Tämän avulla rekursiivinen nimipalvelin voi varmistaa tiedon eheyden ja aitouden. DNSSECin käyttöönotto on kuitenkin kesken, johtuen käyttöönoton ja ylläpidon haasteista, jotka voivat aiheuttaa suuria ongelmia. Esimerkiksi tammikuussa 2012 suuren operaattorin Comcastin nimipalvelu lopetti vastausten tarjoamisen nasa.gov-tietueelle, koska tietueessa oli virheellinen kryptografinen allekirjoitus. (Shulman & Waidner, 2014)

DNSSEC-tekniikka toimii siten, että jokaiselle DNS-alueelle luodaan julkinen ja yksityinen avainpari. DNS-alueen tietueet allekirjoitetaan yksityisellä avaimella ja allekirjoitukset tallennetaan DNS:ään RRSIG-tietueina. Rekursiivinen nimipalvelin voi tarkistaa näiden tietueiden aitouden hakemalla julkisen avaimen DNSKEY-tietueesta ja varmistamalla allekirjoitusten oikeellisuuden. Tämä prosessi luo luottamusketjun, joka alkaa DNS-juurialueesta ja ulottuu aina yksittäisiin DNS-alueisiin asti. Näin varmistetaan, että käyttäjälle tarjottu DNS-tieto on oikea ja muuttumaton. (Shulman & Waidner, 2014)

IETF ratkaisi välimuistin myrkytysongelman päivittämällä standardeja, joten nykypäivänä DNSSEC suojaa man-in-the-middle-hyökkäyksiltä, eli käytännössä tilanteilta, joissa jonkun operaattorin tai DNS-palveluntarjoajan DNS kaapataan ja sitä hyödynnetään ohjaamaan käyttäjiä haitallisille sivustoille. (Hubert & Van Mook, 2009)

4 Avoimen lähdekoodin sovellukset

Avoin lähdekoodi perustuu lähdekoodin julkaisuun verkossa, mikä tarjoaa lukuisia hyötyjä. Internetin ja lähiverkkojen palvelut, kuten DNS ja DHCP, voidaan toteuttaa avoimen lähdekoodin sovelluksilla.

4.1 Avoin lähdekoodi

Avoin lähdekoodi tarkoittaa ohjelmistoja, joiden lähdekoodi on vapaasti kaikkien saatavilla tarkasteltavaksi, muokattavaksi ja parannettavaksi. Lähdekoodi on osa ohjelmistoa, jota useimmat käyttäjät eivät näe, mutta ohjelmoijat voivat käyttää sitä muuttaakseen ohjelmiston toimintaa. Pääsy lähdekoodiin mahdollistaa ohjelman parantamisen, kuten uusien ominaisuuksien lisäämisen tai virheiden korjaamisen. (Opensource.com, ei pvm.)

Toisaalta on olemassa omistusoikeudellisia tai suljetun lähdekoodin ohjelmistoja, joiden lähdekoodia voivat muokata vain sen luonut henkilö, tiimi tai organisaatio. Vain alkuperäiset tekijät voivat laillisesti kopioida, tarkastella ja muuttaa omistusoikeudellista ohjelmistoa. Käyttäjien on hyväksyttävä ohjelmiston käyttöön lisenssi, joka rajoittaa tekijöiden nimenomaisesti sallimia toimintoja. Esimerkkejä omistusoikeudellisista ohjelmistoista ovat Microsoft Office ja Adobe Photoshop. (Opensource.com, ei pvm.)

Avoimen lähdekoodin ohjelmisto eroaa suljetun lähdekoodin ohjelmistoista. Sen tekijät tarjoavat lähdekoodin muiden saataville, jolloin käyttäjät voivat tutkia, kopioida, oppia, muuttaa tai jakaa sitä. LibreOffice ja GNU Image Manipulation Program ovat esimerkkejä avoimen lähdekoodin ohjelmistoista. Avoimen lähdekoodin lisenssit sallivat ohjelmiston vapaan käytön, tutkimisen, muokkaamisen ja jakamisen. Joissakin lisensseissä, joita kutsutaan copyleft-lisensseiksi, vaaditaan, että muokatun ohjelman jakava henkilö julkaisee myös sen lähdekoodin. Lisäksi jotkut lisenssit edellyttävät, että ohjelmiston muokkaaja ja jakaja tarjoaa lähdekoodin ilman lisenssimaksuja. (Opensource.com, ei pvm.)

Avoimen lähdekoodin lisenssit edistävät yhteistyötä ja jakamista, koska ne sallivat muiden muokata lähdekoodia ja sisällyttää nämä muutokset omiin projekteihinsa. Ne kannustavat ohjelmoijia käyttämään, tarkastelemaan ja muokkaamaan avoimen lähdekoodin ohjelmistoja, kunhan he sallivat muiden tehdä samoin jakamalla oman työnsä. Monet internetin alkuperäiset keksinnöt perustuvat avoimen lähdekoodin teknologioihin, kuten Linux-

käyttöjärjestelmään ja Apache-webpalvelinsovellukseen, joten kaikki internetin käyttäjät hyötyvät avoimen lähdekoodin ohjelmistoista. (Opensource.com, ei pvm.)

Ihmiset suosivat avoimen lähdekoodin ohjelmistoja monista syistä. Ensinnäkin avoimen lähdekoodin ohjelmistot tarjoavat enemmän kontrollia. Käyttäjät voivat tutkia koodia varmistaakseen, ettei se sisällä ei-toivottuja toimintoja ja he voivat muokata osia, joista eivät pidä. (Opensource.com, ei pvm.)

Toiseksi avoimen lähdekoodin ohjelmistot auttavat kehittämään ohjelmointitaitoja, koska koodi on julkisesti saatavilla ja opiskelijat voivat helposti tutkia sitä oppiessaan luomaan parempia ohjelmistoja. (Opensource.com, ei pvm.)

Kolmanneksi, jotkut pitävät avoimen lähdekoodin ohjelmistoja turvallisempina ja vakaampina kuin omistusoikeudellisia ohjelmistoja. Koska kuka tahansa voi tarkastella ja muokata avoimen lähdekoodin ohjelmistoa, virheet voidaan havaita ja korjata nopeasti. (Opensource.com, ei pvm.)

Neljänneksi avoimen lähdekoodin ohjelmistot inspiroivat ympärilleen yhteisöjä, jotka eivät ole vain fanikuntaa, vaan aktiivisia käyttäjiä ja kehittäjiä, jotka tuottavat, testaavat, käyttävät, edistävät ja parantavat ohjelmistoa. (Opensource.com, ei pvm.)

Avoimen lähdekoodin ohjelmoijat voivat ansaita rahaa luomillaan ohjelmistoilla. Monille ohjelmoijille on kannattavampaa tarjota maksullisia palveluita ja tukea, mikä mahdollistaa ohjelmiston ilmaisen saatavuuden. (Opensource.com, ei pvm.)

Vaikka avoimen lähdekoodin käyttö tarjoaa useita etuja, se ei ole täysin riskitöntä. Tapaus CVE-2024-3094 toi esiin myös niiden haavoittuvuudet ja riskit. XZ Utils -paketista löydettiin haitallinen takaovi, joka mahdollisti luvattoman etäyhteyden SSH:n kautta. Hyökkääjä, joka toimi vuosien ajan avoimen lähdekoodin kehittäjänä, käytti erittäin obfuskoitua koodia välttääkseen havaitsemisen. Tämä toimitusketjuhyökkäys vaikutti useisiin Linux-jakeluihin, kuten Fedoraan, Debianiin ja Alpineen, mutta ei esimerkiksi Ubuntuun ja Red Hatiin. (Adia, 2024)

Tapaus korostaa, että vaikka avoin lähdekoodi tarjoaa monia hyötyjä, se vaatii tarkkaa valvontaa ja jatkuvaa koodin tarkastusta. Yhteisön nopea reagointi oli ratkaiseva haitallisen koodin havaitsemisessa ja poistamisessa, mutta se ei poista tarvetta paremmille turvallisuuskäytännöille ja -työkaluille. Open source -projektien ylläpitäjien ja käyttäjien on

oltava valppaina ja varmistettava, että he käyttävät vain tarkastettuja ja turvallisia ohjelmistoversioita.

4.2 Käyttöjärjestelmä

Käyttöjärjestelmä on ohjelma, joka ladataan älylaitteeseen ensimmäisenä käynnistysvaiheessa ja joka hallitsee kaikkia älylaitteen resursseja. Sovellusohjelmat käyttävät käyttöjärjestelmää pyytämällä palveluja sovellusohjelmointirajapinnan kautta. Lisäksi käyttäjät voivat olla suoraan vuorovaikutuksessa käyttöjärjestelmän kanssa käyttöliittymän kuten komentorivin tai graafisen käyttöliittymän avulla. (*What Is an Operating System (OS)?*, ei pvm.)

Viisi suosituinta käyttöjärjestelmää (Kuva 2) maailmanlaajuisesti ovat Microsoft Windows, joka julkaistiin vuonna 1980 ja on yleisin esiasennettu käyttöjärjestelmä työasemissa. Windowsin kovin kilpailija työasemissa on Applen julkaisema macOS. Mobiililaitteissa suosituimmat käyttöjärjestelmät ovat Googlen kehittämä Android OS ja Applen kehittämä iOS. Windows, macOS (OS X), Android ja iOS tulevat tyypillisesti laitteen mukana ja ne ovat suljetun lähdekoodin ohjelmistoja sekä maksullisia. Näiden kilpailijana työasemissa on Linux, joka on avoimeen lähdekoodiin perustuva käyttöjärjestelmä. (*5 Most Popular Operating Systems*, ei pvm.)

Kuva 2 suosituimmat käyttöjärjestelmät maailmanlaajuisesti (*Operating System Market Share Worldwide*, ei pvm.)

Linux sai alkunsa vuonna 1991 Linus Torvaldsin toimesta, kun hän osti IBM-yhteensopivan tietokoneen, jonka käyttöjärjestelmä oli MS-DOS. Hän opiskeli tuolloin Helsingin yliopistossa tietojenkäsittelytiedettä ja halusi siirtyä UNIX-pohjaiseen käyttöjärjestelmään, mutta lisenssin korkean hinnan takia se ei ollut vaihtoehto, joten hän päätti kehittää oman UNIX-version. Ensimmäiseksi Linus julkaisi pelkän ytimen, joka on käyttöjärjestelmän perusta, mutta sen käyttöön tarvitaan myös muita komponentteja, kuten käyttöliittymä, joka voi olla tekstipohjainen komentotulkki tai graafinen. (Azorin, 2020)

Nykyään on lukuisia Linux-jakeluja, joista monet perustuvat core-järjestelmiin, kuten Debianiin ja Fedoraan. Näiden pohjalta on kehitetty suosittuja jakeluja, kuten Linux Mint ja Ubuntu. Ubuntuun on saatavilla myös maksullista tukea Canonical-yritykseltä. (Azorin, 2020)

Fedora on kehityshaara IBM:n omistamaan Red Hat Linuxiin, joka on maksullinen, mutta suosittu ratkaisu, koska se virallisesti tukee monia eri laitevalmistajien laitteita. Laitetuen lisäksi Red Hatiin paketoidut sovellukset ja niiden tuki sisältyvät sen lisenssiin. (*Hardware Results*, ei pvm.)

4.3 DHCP- ja DNS-sovellukset

Avoimen lähdekoodin DHCP-palvelimia löytyy viisi kappaletta suosituimpien käyttöjärjestelmien kirjastoista. Näistä kaksi on peräisin samalta valmistajalta, eli ISC:ltä. Sovellukset ovat ISC DHCP, jonka kehityksen he ovat lopettaneet ja korvanneet ISC KEA -sovelluksella. Näiden kuvauksen perusteella ne ovat käytetyimpiä DHCP-palvelimia, tarjoten täydellisen avoimen lähdekoodin ratkaisun. (DebianRepository - Debian Wiki, ei pvm.; Fedora Repositories :: Fedora Docs, ei pvm.)

Tämän lisäksi DHCP-sovelluksia ovat dnsmasq, joka kuvauksen perusteella on pieni ja kevyt DHCP- ja DNS-palvelin, tarkoitettu pienille verkoille. Lisäksi valikoimasta löytyy udhcpd, joka liittyy BusyBoxiin, pieneen käyttöjärjestelmään, joka on yleinen esimerkiksi IoT-laitteissa. Viimeisenä on MaraDNS, joka kuvauksen perusteella on myös pieni ja kevyt DNS- ja DHCP-palvelin. (DebianRepository - Debian Wiki, ei pvm.)

DNS-sovellusten suhteen kahden suosituimman avoimen lähdekoodin käyttöjärjestelmän välillä on eroja. Yhteisiä molemmille ovat Bind, joka on ISC:n kehittämä DNS-ratkaisu ja sisältää kaikki DNS-palvelimen tarpeet. Lisäksi on jo aiemmin mainittu dnsmasq, kevyt sovellus, joka on tarkoitettu pienille verkoille. (DebianRepository - Debian Wiki, ei pvm.)

Kolmas yhteinen DNS-sovellus on Unbound, joka on korkean suorituskyvyn validointiin ja tietoturvaan keskittyvä resolveri, suunniteltu tarjoamaan nopea ja turvallinen DNS-ratkaisu. (DebianRepository - Debian Wiki, ei pvm.)

Debianille löytyy tämän lisäksi vielä Knot, autoritatiivinen DNS-palvelin, joka on suunniteltu tehokkaaksi ja skaalautuvaksi. (DebianRepository - Debian Wiki, ei pvm.)

4.4 Reitin- ja palomuurisovellukset

Palomuuri on tietoturvamekanismi, jonka tarkoituksena on suojata verkkoa ja hallita sen liikennettä. Verkon suunnittelussa pitää varmistaa, että kaikki liikenne kulkee palomuurin kautta. Palomuurin ensisijainen tehtävä on sallia tai estää TCP/IP-pakettien liikenne sille määritettyjen sääntöjen perusteella. Palomuuri voi myös reitittää liikennettä ja sillä pystyy toteuttamaan myös NAT-tekniikan. (Mihalos ym., 2019)

Palomuurit voidaan jakaa käyttöympäristönsä mukaan kolmeen eri tyyppiin: bastion-palvelimet, isäntäpohjaiset palomuurit ja henkilökohtaiset palomuurit. Bastion-palvelin on verkon suojapiste, joka tarkistaa kaiken saapuvan ja lähtevän verkkoliikenteen. Isäntäpohjaiset palomuurit asennetaan yleensä palvelimelle ja henkilökohtaiset palomuurit ovat paikallisesti käyttäjän laitteessa. (Mihalos ym., 2019)

Netfilter/Iptables-palomuurijärjestelmässä Netfilter-komponentit sijaitsevat Linuxin ytimen tilassa ja ne mahdollistavat staattisen pakettien suodatuksen sekä tilallisen pakettien tarkastelun. Netfilter tarjoaa myös joustavan ja laajennettavan kehyksen, joka tukee muun muassa verkkonimipalvelua ja muita lisäominaisuuksia. Lisäksi se tarjoaa useita API-rajapintoja kolmansien osapuolten laajennuksille. Netfilteriä käytetään palomuurin, NAT:n ja muiden tietoturvaominaisuuksien, kuten QoS:n ja reitityspolitiikan, rakentamiseen. Iptables puolestaan toimii käyttäjätilassa työkaluna Linuxin palomuurin konfiguroinnissa ja hallinnassa. (Wu, 2012)

5 Kaksoispino-verkon suunnittelu

Lähiverkon suunnittelussa esitellään lähiverkon suunnittelu ja toteutus laboratorioympäristössä, käytetyt työkalut ja teknologiat sekä testisuunnitelma.

5.1 Laboratorioympäristö

Lähiverkon palveluiden testausta varten toteutettiin laboratorioympäristö, jonka tarkoituksena oli tuoda käytännön kokemusta sovellusten asentamisesta ja konfiguroinnista. Testauksen etuna on se, että teoriassa huomaamatta jääneet puutteet tulevat esiin käytännössä. Lisäksi ympäristössä voidaan testata DNS- ja DHCP-konfiguraatioita muutostilanteissa.

Lähiverkon testiympäristön suunnittelussa hyödynnettiin teoriassa tutkittua tietoa IP-verkkojen toiminnasta ja verkon peruspalveluista. Verkon käytön kannalta kaksi keskeisintä tekniikkaa olivat nimipalvelu ja automaattinen verkon konfigurointitekniikka DHCP. Teorian perusteella valittiin käyttöjärjestelmä, jolle oli saatavilla teoriassa esitetyt sovellukset. Valittu ohjelmisto asennettiin ja konfiguroitiin virtuaalipalvelimille, minkä jälkeen sen toimivuutta testattiin.

Teoriapohjaista tutkimusta täydennettiin standardien tutkimisella ja ymmärtämisellä sekä tieteellisten artikkeleiden ja kirjallisuuden avulla. Lisäksi tietoa haettiin hakukoneista.

Valmiin testausympäristön avulla varmistettiin, että teorian perusteella valitut menetelmät toimivat ja kuinka kauan niiden käyttöönotto kestää. Testausympäristössä voitiin myös arvioida protokollien suorituskykyä ja varmistaa, että konfiguraatiomuutokset toimivat käytännössä. Lisäksi testausympäristössä voitiin seurata, miten asiakaslaitteet hyödyntävät kaksoispinotekniikkaa. Ympäristö oli kustannustehokas ja nopea pystyttää. Vastaavanlainen ympäristö voidaan rakentaa myös yrityksen tiloihin ja tämän työn perusteella on helppo määrittää, kuinka kauan ympäristön rakentaminen kestää ja mitä se vaatii verkolta.

5.2 Käytetyt työkalut ja teknologiat

Tutkimuksen ympäristönä toimii Debian-palvelin, jolle on asennettu virtualisointipalvelut, joiden avulla palvelimella voidaan ajaa virtuaalipalvelimia testejä varten. Lisäksi palvelimille

on konfiguroitu erillisiä virtuaalisia reitittimiä, joilla jokaisella on oma lähiverkkonsa. Reitittimien välinen liikenne reititetään toisiinsa iptables-sovellusta käyttäen. Yhdelle palvelimista asennettiin ISC KEA DHCP -palvelu, jonka tarkoituksena on automatisoida verkon asiakkaiden konfigurointi. Samalle palvelimelle asennettiin myös web-palvelin, joka toimii alustana IPv4- ja IPv6-suorituskykytesteille.

Verkonhallintaa varten toteutettiin sovellus, joka pyörii ohjauskoneella. Sen konfigurointia muuttamalla muutokset siirtyvät hallitusti nimipalvelimille. Sovellus toteutettiin Linuxin perustyökaluilla ja Python-ohjelmointikielellä.

5.3 Testisuunnitelma

Työn tarkoituksena on testata, että valittujen sovellusten ja tekniikoiden avulla testilaitteet saavat toimivan kaksoispinotekniikan verkon ja miten asiakaslaitteet ottavat sen käyttöön. Dynaamisen DHCP:n osalta testataan, että laite saa IP-osoitteen, joka voi vaihtua ja staattisen konfiguraation osalta varmistetaan, että laitteen automaattisesti määritetty osoite ei muutu. Tämän jälkeen testataan, esiintyykö protokollien välillä suorituskykyeroja.

Nimenselvitystestauksessa varmistetaan, että automaattinen nimipalvelu toimii odotetusti ja arvioidaan sen hyöty käytännössä, erityisesti ajansäästön näkökulmasta. Lisäksi testataan, miten kaksoispinokonfiguraatio vaikuttaa nimenselvitykseen.

6 Lähiverkon toteutus

Tässä kappaleessa käsitellään opinnäytetyön verkon suunnitelmaa ja käytännön toteutusta, joka sisältää isäntäkoneen virtualisointipalvelun asennuksen, verkon konfiguroinnin ja testipalvelimien asennuksen.

6.1 Verkon suunnittelu- ja konfiguraatio

Verkko suunniteltiin ja konfiguroitiin seuraavasti. Asiakasverkkoihin asennettiin testilaitteet, jotka saavat automaattisen verkon konfiguraation palvelinverkoissa sijaitsevilta palvelimilta. Verkkojen reitittimet on nimetty alkaen br-tunnuksella. Verkon konfiguraatiossa on määritelty verkon osoite ja sen oletusyhdyskäytävä. Palvelinverkkoon on asennettu myös nimipalvelu kahdelle eri palvelimelle. Verkot on yhdistetty samaan isäntäpalvelimeen, johon on asennettu

iptables-sovellus toteuttamaan reititystä verkkojen välillä. Verkkojen nimet ovat käytettyjen reitittimien nimiä eli br2, br3 ja br64.

Asiakasverkot

- br2: dynaamiset asiakkaat (DHCP-määritellyt muuttuvat osoitteet)

- br3: staattiset asiakkaat (DHCP-määritellyt kiinteät IP-osoitteet)

Palvelinverkko

- br64: dual-stack DNS/DHCP-palvelut ja web-palvelin testausta varten

DHCP-verkot

br2:

- IPv6: 2001:99a:807:1002::/120 gw .::1

- IPv4: 192.168.2.0/24 gw .1

br3:

- IPv6: 2001:99a:807:1003::/120 gw .::1

- IPv4: 192.168.3.0/24 gw .1

Isäntäkoneen konfiguraatio

Rajapinnat:

- Virtuaalipalvelinten verkon rajapinta: br64

- Virtuaaliasiakkaiden verkon rajapinnat: br2, br3

- br0: rajapinta internetyhteydelle ISP:n kaapelimodeemin kautta. IPv4-asiakkaat ovat verkossa 192.168.0.0/24, DHCP-alue on 192.168.0.10-192.168.0.149. Alue 150->254 on

DHCP:n ulkopuolella staattista käyttöä varten. br0 on konfiguroitu käyttämään osoitetta 192.168.0.150.

6.2 Palvelinten konfiguraatio

Palvelimet on konfiguroitu verkkoon seuraavasti. Br2-verkossa ovat dynaamiset asiakkaat jotka saavat DHCP-palvelulta dynaamisen osoitteen. Br3-verkossa olevat laitteet saavat DHCP:ltä aina kiinteän osoitteen joka ei siis muutu, vaikka laite olisikin pitkään pois verkosta.

Br64-verkkon palvelimille on konfiguroitu kiinteät osoitteet ja ne eivät käytä verkon asetusten määritykseen DHCP-palvelua.

br2

- asiakkaat: hauki (Linux), lahna (Windows) - dynaaminen osoite

- dhcp-verkko: 192.168.2.0/24

- dhcpv6-verkko: 2001:99a:807:1000::3000/116

br3

- asiakkaat: kuha (Linux), salakka (Windows) - kiinteä osoite

- kuha: IPv4: 192.168.3.50, IPv6: 2001:99a:807:1000::4050

- salakka: IPv4: 192.168.3.51, IPv6: 2001:99a:807:1000::4051

- IPv4-verkko: 192.168.3.0/24

- IPv6-verkko: 2001:99a:807:1000::4000/116

br64

- IPv64 dhcp: lippa, IPv4: 192.168.64.2, IPv6: 2001:99a:807:1000::2002

- IPv64 dns1: bete, IPv4: 192.168.64.3, IPv6: 2001:99a:807:1000::2003

- IPv64 dns2: lotto, IPv4: 192.168.64.4, IPv6: 2001:99a:807:1000::2004

6.3 Isäntäpalvelimen virtualisointiohjelmiston asennus ja konfigurointi

Tässä kappaleessa asennetaan virtualisointiohjelmisto ja verkon konfiguraatio palvelimille testausta varten. Ensimmäisenä suoritetaan komento 2, jolla asennetaan tarvittavat sovellukset isäntäkoneelle.

Komento 2 apt-komento, joka asentaa tarvittavat sovellukset

```
sudo apt install qemu-system libvirt-daemon-system bridge-utils
iptables isc-dhcp-helper
```

APT (Advanced Package Tool) työkalulla otetaan yhteys .deb-pohjaisten Linux-järjestelmien sovelluskirjastoon. Komennon install optiolla avulla voidaan asentaa tarvittavat sovellukset. Paketit qemu-system ja libvirt-daemon-system sisältävät kaikki virtuaalipalvelimien suorittamiseen tarvittavat komponentit. Bridge-utils lisää järjestelmään verkkosiltojen hallintaan tarvittavat työkalut, kun taas iptables tarvitaan pakettien reitittämiseen siltojen välillä. Lisäksi isc-dhcp-helper otetaan käyttöön DHCP-helper-toiminnon tukemiseksi DHCP:n toimintaa varten.

Sovellusten asentamisen jälkeen palvelimille luodaan virtuaaliset verkkosillat komennolla 3.

Komento 3 for-silmukassa ajettava brctl-komento, joka luo virtuaaliset sillat reititystä varten

```
for i in br2 br3 br64;do sudo brctl addbr $i;done
```

Brctl-komennon avulla luodaan for-silmukan avulla kolme virtuaalista verkkosiltaa verkkoliikenteen hallintaa varten. Yhteen siltaan liitetään myöhemmin virtuaalisia palvelimia, jotka voivat kommunikoida keskenään virtuaalisessa linkkikerroksessa. For-silmukassa ajettu komento säästää hieman aikaa tässä tapauksessa, mutta saman toiminnon voisi suorittaa myös kolmella erillisellä komennolla, joissa jokainen silta luodaan erikseen, esimerkiksi: brctl addbr br1.

Siltojen luomisen jälkeen on vuorossa niiden konfigurointi komennolla 4.

Komento 4 vi-komento, jolla muokataan verkon konfiguraatio palvelimelle

```
vi /etc/network/interfaces
```

Verkon asetukset määritellään tiedostossa /etc/network/interfaces, joka sisältää fyysisten ja virtuaalisten verkkokorttien määritykset, niiden osoitteet ja protokollat sekä virtuaaliset sillat.

Ohjelmakoodi 1 isäntäkoneen verkon konfiguraatio

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# Physical interface configuration
iface eno1 inet manual

# Bridge setup
auto br0
iface br0 inet static
   address 192.168.0.151
   netmask 255.255.255.0
   gateway 192.168.0.1
   bridge_ports eno1
   bridge_stp off
   bridge_waitport 0
   bridge_fd 0

iface br0 inet6 static
   address 2001:99A:807:1000::151/64
   gateway fe80::a698:13ff:fe5f:4773

auto br2
iface br2 inet static
    address 192.168.2.0/24
    bridge_ports none
    pre-up brctl addbr br2 || true

iface br2 inet6 static
    2001:99a:807:1000::3000/116
    bridge_ports none
    pre-up brctl addbr br2 || true

auto br3
iface br3 inet static
    192.168.3.0/24
    pre-up brctl addbr br3 || true

iface br3 inet6 dhcp
    address 2001:99a:807:1000::4000/116
    bridge_ports none
    pre-up brctl addbr br3 || true

auto br64
iface br64 inet static
    address 192.168.64.1/24
    bridge_ports none
    pre-up brctl addbr br64 || true

iface br64 inet6 static
    address 2001:99a:807:1000::2001/120
```

```
bridge_ports none
pre-up brctl addbr br64 || true
```

Seuraavaksi luodaan yhteydet virtuaalisten siltojen välille siirtyen linkkikerroksen ulkopuolelle Internet Protocol -kerroksen reititykseen. Tämä toteutetaan tässä virtuaalisessa ympäristössä iptables-sovelluksella hyödyntäen sen reititystoimintoa.

Komento 5 for-silmukassa ajettu iptables komento, jolla lisättiin reitityssäännöt

```
for i in br2 br3 br64;do sudo iptables -A FORWARD -i $i -o br0 -j
ACCEPT;done
```

Tässä for-silmukassa lisätään kolme reitityssääntöä iptables-työkalulla. Iptables-komennossa käytetty -A FORWARD -optio tarkoittaa, että sääntö lisätään FORWARD-ketjuun. Tämä ketju määrittää, miten isäntäkone välittää paketteja siltojen ja verkkokorttien välillä. Optio -i $i määrittää liitännän josta paketit saapuvat ja silmukka asettaa sen arvoksi kussakin kierroksessa yhden silloista: br2, br3 tai br64. Optio -o br0 määrittää ulostulevan liitännän sillaksi br0. Lopuksi -j ACCEPT sallii kaikki ehdot täyttävät paketit, jolloin liikenne voi kulkea määritettyjen siltojen välillä.

Komento 6 iptables-komento, jolla varmistetaan että muutos onnistui

```
sudo iptables --list-rules
-P INPUT ACCEPT
-P FORWARD ACCEPT
-P OUTPUT ACCEPT
-A FORWARD -i br2 -o br0 -j ACCEPT
-A FORWARD -i br3 -o br0 -j ACCEPT
-A FORWARD -i br64 -o br0 -j ACCEPT
```

Iptables --list-rules käskyllä tarkistetaan, mitä sääntöjä on tällä hetkellä käytössä. Kolme alimmaista sääntöä lisättiin aiemmin komennossa 5. Näiden sääntöjen avulla iptables reitittää kaikki paketit, jotka saapuvat silloista br2, br3 tai br64, eteenpäin isäntäkoneen br0-siltaan. Lisäksi tässä tulosteessa näkyvät ketjujen (INPUT, FORWARD, OUTPUT) oletuskäytännöt, jotka on asetettu hyväksymään kaikki liikenne.

Verkon määritellyn jälkeen asennetaan virtuaaliset palvelimet.

Komento 7 virt-install-komento, jolla asennettiin yksi palvelin. Loput asennettiin samalla komennolla muuttaen --name ja --network bridge asetuksia suunnitelman mukaisesti

```
virt-install --virt-type kvm --name lotto --location
http://deb.debian.org/debian/dists/bookworm/main/installer-amd64/ --
os-variant debian10 --disk size=20 --memory 4196 --network bridge=br0
```

Asennus tapahtuu graafisessa käyttöliittymässä samalla tavalla kuin fyysisenkin koneen asennus. Tämä toistettiin kaikille palvelimille.

6.4 DNS-palvelimen asennus ja konfiguraatio

Sovellukseksi valitsin BIND-ohjelmiston, koska se tarjoaa sekä rekursiivisen nimenselvityksen että auktoritatiivisen nimipalvelun. DNS-palvelu asennettiin bete ja lotto palvelimille. Asennus suoritetaan komennossa 8.

Komento 8 apt-komento, jolla DNS-sovellus asennettiin

```
sudo apt install bind9 -y
```

Komennolla apt install asennettiin BIND-ohjelmisto järjestelmään. Optio -y ohittaa vahvistuspyynnöt, jolloin asennus suoritettiin automaattisesti. Koska apt-komennon perustoiminta on jo selitetty aiemmin, tässä keskityttiin BIND-ohjelmiston asennuksen yksityiskohtiin.

Sovelluksen konfigurointi suoritettiin VIM-editorilla.

Komento 9 vi-komento, jolla muokattiin nimipalvelun konfiguraatio

```
vi /etc/bind/named.conf.options
```

BIND-ohjelmiston konfiguraation sijainti määritellään ohjelman käännösvaiheessa ja Debianissa käytettävän version konfiguraatiotiedostoksi on määritelty /etc/bind/named.conf. Tiedostossa named.conf on viittaus, että asetukset luetaan tiedostosta /etc/bind/named.conf.options, joka muokattiin seuraavanlaiseksi:

Ohjelmakoodi 2 bind-sovelluksen konfiguraatio named.conf.options tiedostossa

```
options {
    directory "/var/cache/bind";

    recursion yes;
    allow-recursion { any; };

    dnssec-validation auto;

    auth-nxdomain no;
    listen-on { any; };
};
```

Konfiguraatiossa määritettiin recursion: yes, mikä tarkoittaa, että nimipalvelun tulee seurata nimenselvityksessä puurakennetta. Allow-recursion-asetuksessa määritetään verkot, jotka saavat pyytää nimenselvitystä. Listen-on-asetuksessa puolestaan määritetään mitä paikallista verkkoa sovellus kuuntelee.

6.5 DHCP-palvelimen asennus ja konfiguraatio

DHCP-ohjelmistoksi valittiin ISC:n kehittämä KEA-sovellus, joka asennettiin ja konfiguroitiin komennolla 10.

Komento 10 apt-komento, jolla asennetaan KEA-sovellus

```
sudo apt install kea -y
```

Asennuksen jälkeen palvelun konfigurointi suoritettiin muokkaamalla kahta erillistä tiedostoa, jotka sisältävät DHCPv4- ja DHCPv6-palveluiden asetukset. Tiedostojen muokkaaminen tapahtui VIM editorilla.

Ohjelmakoodi 3 KEA DHCPv4 -palvelun konfiguraatio

```
#{
  "Dhcp4": {
    "interfaces-config": {
      "interfaces": [
        "enp1s0"
      ]
    },
    "lease-database": {
      "type": "memfile",
      "persist": true,
      "name": "/var/lib/kea/kea-leases4.csv"
    },
    "valid-lifetime": 28800,
    "option-data": [
      {
        "name": "domain-name-servers",
        "data": "192.168.64.3, 192.168.64.4"
      }
    ],
    "subnet4": [
      {
        "subnet": "192.168.64.0/24",
        "pools": [
          {
            "pool": "192.168.64.5 - 192.168.64.199"
          }
        ],
        "option-data": [
          {
            "name": "routers",
            "data": "192.168.64.1"
```

```
            }
          ],
          "reservations": [
            {
              "hw-address": "1a:1b:1c:1d:1e:1f",
              "ip-address": "192.168.64.200"
            }
          ]
        }
      ],
      "loggers": [
        {
          "name": "kea-dhcp4",
          "output_options": [
            {
              "output": "/var/log/kea/kea-dhcp4.log",
              "maxver": 10
            }
          ],
          "severity": "INFO"
        }
      ]
    }
}
```

DHCPv4-palvelun asetukset määriteltiin tiedostossa /etc/kea/kea-dhcp4.conf. Konfiguraatiossa palvelu asetettiin kuuntelemaan verkkoliitäntää enp1s0. Vuokraustiedot määriteltiin tallennettavaksi tiedostoon /var/lib/kea/kea-leases4.csv ja IP-osoitteiden vuokrausajaksi asetettiin 28800 sekuntia mikä vastaa kahdeksaa tuntia. DNS-palvelimiksi määritettiin osoitteet 192.168.64.3 ja 192.168.64.4.

Aliverkoksi määriteltiin 192.168.64.0/24 ja osoitealueeksi asetettiin 192.168.64.5 - 192.168.64.199. Lisäksi aliverkkoon määriteltiin reitittimen osoitteeksi 192.168.64.1. Yksi osoite varattiin staattisesti laitteelle jonka MAC-osoite on 1a:1b:1c:1d:1e:1f ja sen IP-osoitteeksi asetettiin 192.168.64.200. Lokitiedostoksi määriteltiin /var/log/kea/kea-dhcp4.log ja lokituksen tasoksi asetettiin INFO.

Ohjelmakoodi 4 KEA DHCPv6 -palvelun konfiguraatio

```
{
    "Dhcp6": {
        "valid-lifetime": 4000,
        "renew-timer": 1000,
        "rebind-timer": 2000,
        "preferred-lifetime": 3000,
        "interfaces-config": {
            "interfaces": [ "enp1s0" ]
        },
        "lease-database": {
            "type": "memfile",
            "persist": true,
            "name": "/var/lib/kea/kea-leases6.csv"
        },
```

```
        "loggers": [
            {
                "name": "kea-dhcp6",
                "output_options": [
                    {
                        "output": "/var/log/kea/kea-dhcp6.log"
                    }
                ],
                "severity": "DEBUG",
                "debuglevel": 99
            }
        ],
        "subnet6": [
            {
                "subnet": "2001:099a:0807:1000::2000/120",
                "pools": [
                    {
                        "pool": "2001:099a:0807:1000::2005-
2001:099a:0807:1000::2020"
                    }
                ],
                "interface": "enp1s0",
                "option-data": [
                    {
                        "name": "dns-servers",
                        "data": "2001:99a:807:1001::3,
2001:99a:807:1001::3"
                    },
                    {
                        "name": "domain-search",
                        "data": "home.arpa"
                    }
                ]
            }
        ]
    }
}
```

DHCPv6-palvelun asetukset määriteltiin tiedostossa /etc/kea/kea-dhcp6.conf. Myös tässä palvelu asetettiin käyttämään verkkoliitäntää enp1s0. Vuokraustiedot määriteltiin tallennettavaksi tiedostoon /var/lib/kea/kea-leases6.csv. Vuokrausajaksi asetettiin 4000 sekuntia ja uusimisajastimeksi määriteltiin 1000 sekuntia kun taas uudelleensidonnan ajastimeksi asetettiin 2000 sekuntia.

IPv6-aliverkoksi määriteltiin 2001:099a:0807:1000::2000/120. Osoitealueena käytettiin 2001:099a:0807:1000::2005-2001:099a:0807:1000::2020. DNS-palvelimiksi määriteltiin osoitteet 2001:99a:807:1001::3 ja 2001:99a:807:1001::3 ja nimipalveluhaun hakemistoksi asetettiin home.arpa. Lokitiedostoksi määriteltiin /var/log/kea/kea-dhcp6.log ja lokituksen tasoksi asetettiin DEBUG jonka yksityiskohtaisuus määriteltiin tasolle 99.

6.6 Web-palvelimen asennus

Web-palvelun ohjelmistoksi valitsin Apache2-sovelluksen, joka on yksi yleisimmistä avoimen lähdekoodin web-palvelimista. Sovellus asennetaan APT-käskyllä.

Komento 11 apt-komento, jolla asennetaan web-palvelimen sovellus

```
sudo apt install apache2
```

Web-sivulle piti luoda koodi, joka yksinkertaisesti osoittaa sivuston toimivan. Tämän lisäksi tavoitteena oli tarkkailla kaksoispino-ympäristön toimintaa, joten lisäsin web-sivustolle koodinpätkän, jolla haetaan senhetkiset logot osoitteista https://www.google.com ja https://www.bing.com.

Web-palvelimen oli tarkoitus tukea kaksoispinotekniikkaa, joten luotiin kaksi erillistä sivua, joiden tarkoitus oli selkeästi osoittaa käytettävä protokolla. Näiden sivujen ainoa ero on tekstipätkä, joka ilmoittaa, käytetäänkö kyseisellä sivulla IPv4- vai IPv6-protokollaa. Ratkaisun avulla voitiin testata web-palvelimen kykyä toimia molemmilla protokollilla ja varmistaa kaksoispino-ympäristön toimivuus.

Asennus toteutettiin luomalla molemmille sivuille omat hakemistot web-palvelimen sisältöä varten. Hakemistojen sisälle lisättiin yksinkertaiset web-sivut, jotka kirjoitettiin index.html-tiedostoihin. Web-palvelimen konfiguraatioon tehtiin muutokset, joissa molemmat sivut määriteltiin erillisiksi sivustoiksi. Konfiguraation avulla web-palvelin ohjaa oikean protokollan mukaiseen hakemistoon käyttäjän pyynnön perusteella. Esimerkkikomennot kuvaavat toteutusta IPv4:n osalta, mutta sama prosessi toistettiin IPv6:n kohdalla vaihtamalla komennossa "ipv4" sanaan "ipv6".

Web-sivustot asennetaan omiin hakemistoihinsa, ja Debian Linuxissa hakemistot luodaan mkdir-käskyllä komennossa 12.

Komento 12 mkdir-komento, jolla luodaan web-sivuille alihakemistot

```
mkdir /var/www/ipv4
```

Web-sivuston aloitussivu luodaan VIM-editorilla komennolla 13.

Komento 13 vim-komento, jolla web-sivu kirjoitettiin

```
vi /var/www/ipv4/index.html
```

VIM-Editorilla avataan tiedosto index.html joka on tyypillisesti oletussivu jonka web-selaimet yrittävät avata ensimmäisenä. Eli kun testissä otetaan yhteys http://www4.home.arpa sivustoon, selain avaa todellisuudessa osoitteeseen http://www4.home.arpa/index.html.

Ohjelmakoodi 5 web-sivun index.html lähdekoodi

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>IPv4 Site</title>
    <style>
        body {
            display: flex;
            flex-direction: column;
            justify-content: center;
            align-items: center;
            height: 100vh;
            margin: 0;
            font-family: Arial, sans-serif;
            background-color: #f0f0f0;
        }
        h1 {
            color: #333;
        }
        img {
            margin-top: 20px;
            width: 200px; /* Adjust size as needed */
        }
    </style>
</head>
<body>
    <h1>You reached IPv4 site</h1>
    <img
src="https://www.google.com/images/branding/googlelogo/1x/googlelogo_c
olor_272x92dp.png" alt="Google Logo">
    <img src="https://www.bing.com/th?id=OHR.WebbedFeet_EN-
US4503740918_1920x1080.jpg&rf=LaDigue_1920x1080.jpg&pid=hp" alt="Bing
Logo">
</body>
</html>
```

Web-palvelimen konfigurointi tehtiin käyttämällä Apache2:n hallintakäskyjä ja VIM-editoria.

Komento 14 vi-komento, jolla luotiin IPv4-sivuston konfiguraatio

```
vi /etc/apache2/sites-available/ipv4.conf
```

Komento 15 vi-komento, jolla luotiin Ipv6-sivuston konfiguraatio

```
vi /etc/apache2/sites-available/ipv6.conf
```

Ohjelmakoodi 6 IPv4-sivun konfiguraatio

```
<VirtualHost 192.168.64.2:80>
```

```
        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/ipv4
        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined
    </VirtualHost>
```

Ohjelmakoodi 7 IPv6-sivun konfiguraatio

```
    <VirtualHost 2001:99a:807:1000::2002:80>
        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/ipv6
        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined
    </VirtualHost>
```

Komento 16 a2ensite-komento, jolla sivujen konfiguraatio ladattiin

```
    a2ensite ipv4
    a2ensite ipv6
```

A2ensite on Apache2-palvelimen hallintaan tarkoitettu komento, jolla otetaan käyttöön määritelty sivustokonfiguraatio. Tässä tapauksessa komento suoritettiin kahdesti, ensin sivustolle ipv4 ja sitten sivustolle ipv6. Kun tämä komento suoritetaan, Apache luo symbolisen linkin määritellystä konfiguraatiotiedostosta hakemistoon /etc/apache2/sites-enabled, mikä mahdollistaa kyseisen sivuston käytön palvelimella.

Komento 17 systemctl-komento, jolla sivut käynnistettiin

```
    systemctl reload apache2
```

Systemctl reload apache2 -komento käskee Apache2-palvelinta lataamaan konfiguraation uudelleen. Tämä ei käynnistä palvelinta uudelleen, vaan soveltaa uudet konfiguraatiomuutokset jo käynnissä olevaan palvelimeen. Komento on tarpeen esimerkiksi silloin, kun uusia sivustokonfiguraatioita otetaan käyttöön a2ensite-komennolla. Lataamalla konfiguraation uudelleen Apache tunnistaa uudet asetukset ja alkaa palvella niitä määriteltyjen ehtojen mukaisesti, ilman palvelun keskeytystä tai käynnissä olevien yhteyksien katkeamista.

6.7 DNS-alueen automatisoitu hallinta ja julkaisu

Testiverkon alueeksi valittiin home.arpa, joten web-palvelin piti lisätä sinne. Yksi opinnäytetyön tavoitteista oli automatisoida tämä prosessi. Tätä varten päätin luoda skriptin, joka generoi DNS-konfiguraation lähdetiedoston perusteella. Nimipalvelun dataa muokataan ohjaustiedostossa, jonka sisältö määrittää konfiguraation. Tämä konfiguraatio haetaan nimipalvelimille ajastetusti siten, että kun muutos tehdään ohjauskoneella, se siirtyy nimipalvelimille automaattisesti.

Ohjauskoneella käytetään kahta sovellusta, jotka luovat ohjaustiedoston perusteella DNS-zone-tiedostot. Ohjaustiedosto sisältää DNS-alueeseen liittyvät tiedot, kuten palvelimet ja niiden IP-osoitteet. Esimerkki ohjaustiedostosta on nähtävissä ohjelmakoodissa 8.

Ohjelmakoodi 8 nimipalvelun ohjaustiedosto

```
home.arpa  SOA  admin.home.arpa        604800  86400  2419200  604800
www4.home.arpa  A       192.168.64.2
www6.home.arpa  AAAA    2001:99a:807:1000::2002
dns1.home.arpa  A       192.168.64.3
dns2.home.arpa  A       192.168.64.4
fixed.home.arpa  A       192.168.3.2
fixed.home.arpa  AAAA       2001:99a:807:1000::4002
```

Ohjaustiedosto luetaan pienellä Python-ohjelmalla, joka käsittelee tiedoston rivit yksi kerrallaan. Tiedoston rivit on eroteltu tabulaattorilla. Rivi, jossa on SOA, generoi zone-tiedostoon tarvittavat alkuosan tiedot ja sitä seuraavat rivit muodostavat itse DNS-tietueet. Esimerkissä on kuusi tietuetta, joista neljä sisältää IPv4-osoitteita ja kaksi IPv6-osoitteita.

Sovellus on dns-zone-management joka on ajastettu cron-ohjelmalla.

Komento 18 cron-komento, jolla tarkistetaan ajastetut komennot

```
crontab -l
*/2 * * * * make -C /opt/dns-zone-management
```

Cron on ajanhallintaohjelma, joka suorittaa komentoja tai skriptejä ajastetusti Linux-järjestelmissä. Yllä oleva ajastus määrittää, että make-komento suoritetaan kahden minuutin välein ohjelman /opt/dns-zone-management.

Ajastetussa tehtävässä käytettävä make on työkalu, joka automatisoi ohjelmien rakentamista tai tiedostojen generointia määritetyn Makefile-tiedoston avulla. Tässä tapauksessa Makefile ohjaa Python3-ohjelman suorittamista sekä hallinnoi siihen liittyviä toimenpiteitä, kuten tiedostojen generointia, julkaisua ja siivousta.

Ohjelmakoodi 9 makefile, joka suorittaa Python3-ohjelman

```
.PHONY: all generate publish clean
PUBLISH_DIR = /opt/dns/myzones/
generate:
	python3 src/dns_zone_tool.py

publish: generate
	@echo "Publishing to $(PUBLISH_DIR)"

clean:
	rm -f $(PUBLISH_DIR)home.arpa $(PUBLISH_DIR)reverse.home.arpa
```

```
	@echo "Cleaned up generated files in $(PUBLISH_DIR)"

all: clean generate publish
}
```

Makefile suorittaa Python 3 -ohjelman (ks. Liite 2), joka luo DNS-zone-tiedoston ohjaustiedoston perusteella. Kun tämä vaihe on suoritettu, hallintapuoli on valmis. Tämän jälkeen voidaan konfiguroida nimipalvelimille skripti ja cron-ajastus, joka hakee muuttuneet tiedostot ja sijoittaa ne oikeaan paikkaan

Komento 19 cron-komento, jolla katsotaan ajastetut käskyt

```
crontab -l
*/2 * * * * /opt/dns-sync/update.sh
```

Skripti (ks. Liite 3) hakee ohjauskoneelta zone-tiedostot ja kopioi ne paikalleen. Skripti tarkistaa myös, onko tiedosto muuttunut ja jos ei ole, se ei tee mitään. Jos muutoksia löytyy, skripti kopioi uuden tiedoston ja lataa BIND-palvelimen uudelleen.

7 Testaus ja tulokset

Kappaleessa esitetään verkon palveluiden toimivuus testien avulla. Palvelimien tulee käyttää niille määriteltyä nimipalvelua. Dynaamisten palvelimien tulee saada vaihtuva IP-osoite ja kiinteiden palvelimien tulee saada muuttumaton osoite. Paikallisten ja internet-yhteyksien tulee toimia.

IPv4- ja IPv6-protokollille suoritetaan suorituskykytestaukset. Kaksoispinon vaikutuksia analysoidaan nimipalvelun ja automaattisen verkon konfiguraation näkökulmasta.

7.1 Dynaamisen verkon palveluiden testaus

Automaattisen verkon konfiguroinnin testaus tapahtui varmistamalla, että asiakaslaitteet saivat IPv4- ja IPv6-verkkojen konfiguraation. Ping-komennolla tarkistettiin, että verkkoyhteydet toimivat sekä paikallisesti että internetiin. Hauki-nimisen palvelimen piti saada br2-verkosta dynaaminen IPv4-osoite alueelta 192.168.2.2–192.168.2.255 ja koska se on ensimmäinen asiakaslaite verkossa, osoitteen pitäisi olla 192.168.2.2. Nimipalvelimien osoitteiden pitäisi olla 192.168.64.3 ja 192.168.64.4. IPv6-osoitteen pitäisi tulla alueelta 2001:99a:807:1000::3002–2001:99a:807:1000::3999 ja DNS-palvelimien osoitteiden pitäisi olla 2001:99a:807:1000::2003 ja 2001:99a:807:1000::2004.

Komento 20 ip-komento, jolla tarkistetaan, mitkä IP-osoitteet on konfiguroitu Linuxin verkkoon

```
ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel
state UP group default qlen 1000
    link/ether 52:54:00:e9:f1:6f brd ff:ff:ff:ff:ff:ff
    inet 192.168.2.2/24 brd 192.168.2.255 scope global dynamic enp1s0
       valid_lft 16053sec preferred_lft 16053sec
    inet6 2001:99a:807:1000::3002/128 scope global dynamic
       valid_lft 3257sec preferred_lft 2257sec
    inet6 fe80::5054:ff:fee9:f16f/64 scope link
       valid_lft forever preferred_lft forever
```

Ip-komennolla voidaan tarkastella verkon nykytilaa ja varmistaa, että asiakaslaitteelle on määritetty oikeat IPv4- ja IPv6-osoitteet. Tuloste näyttää verkkoliitännät, niiden IP-osoitteet sekä muita tietoja, kuten osoitteen elinaika-arvot. Esimerkkitulosteesta voidaan nähdä, että Hauki-palvelin on saanut dynaamisen IPv4-osoitteen 192.168.2.2 ja dynaamisen IPv6-osoitteen 2001:99a:807:1000::3002. Molemmat osoitteet kuuluvat määriteltyihin alueisiin, mikä osoittaa, että DHCP-palvelu toimii odotetusti.

Komento 21 cat-komento, jolla selvitetään /etc/resolv.conf tiedostosta mitä nimipalvelua Linux käyttää

```
cat /etc/resolv.conf
search home.arpa.
nameserver 2001:99a:807:1000::2003
nameserver 2001:99a:807:1000::2004
```

Cat-käskyllä luetaan ja tulostetaan tiedoston sisältö konsoliin. Komennossa 21 tarkistetaan /etc/resolv.conf-tiedoston sisältö, joka sisältää palvelimen nimipalvelun konfiguraation. Tuloksen perusteella tämä laite hyödyntää vain IPv6-nimipalvelua nimien selvitykseen, koska tuloksessa ei näy IPv4-nimipalvelua lainkaan.

Nimipalvelu on helppo testata nslookup-komennolla: ensin testataan lähiverkon alue ilman absoluuttista nimeä ja tämän jälkeen testataan DNS-rekursion toiminta kysymällä www.facebook.com-osoitetta absoluuttisella nimikyselyllä.

Komento 22 nslookup-komento, jolla selvitetään mihin IP-osoitteeseen nimi www4 viittaa

```
nslookup www4
Server:                    2001:99a:807:1000::2003
Address:         2001:99a:807:1000::2003#53

Name: www4.home.arpa
Address: 192.168.64.2
```

Tuloksen perusteella nimi ratkeaa ja vastauksena saatiin web-palvelimen IPv4-osoite. Vastaus tuli IPv6-nimipalvelusta.

Komento 23 nslookup-komento, jolla selvitetään mihin IP-osoitteeseen nimi www.facebook.com viittaa.

```
nslookup www.facebook.com.
Server:                    2001:99a:807:1000::2003
Address:         2001:99a:807:1000::2003#53

Non-authoritative answer:
www.facebook.com canonical name = star-mini.c10r.facebook.com.
Name: star-mini.c10r.facebook.com
Address: 157.240.205.35
```

```
Name: star-mini.c10r.facebook.com
Address: 2a03:2880:f113:81:face:b00c:0:25de
```

Vastauksena saatiin ensin IPv4-osoite, joka on alias star-mini.c10r.facebook.com ja nimenselvitys selvitti kyseiselle osoitteelle sen IPv6-osoitteen.

Komento 24 nslookup-komento, jolla selvitetään onko nimellä IPv6-osoitetta

```
nslookup -q=AAAA star-mini.c10r.facebook.com
Server:                    2001:99a:807:1000::2003
Address:         2001:99a:807:1000::2003#53

Non-authoritative answer:
Name: star-mini.c10r.facebook.com
Address: 2a03:2880:f113:81:face:b00c:0:25de
```

Testin tulos oli yllätys. Palvelimella on käytössään kaksoispinoverkko, mutta se käyttää nimenselvitykseen pelkästään IPv6-nimenselvityspalvelua, joka osaa vastata nimikyselyihin molemmilla osoitteilla. Mistään ei kuitenkaan näy, miksi palvelin käyttäytyy näin. Teorian perusteella Happy Eyeballs -toiminnon pitäisi kysyä ensin IPv6-osoitetta ja jos se saa siihen vastauksen, sen pitäisi käyttää sitä eikä kysellä IPv4-osoitteita.

Toisena testilaitteena käytettiin Lahna-nimistä Windows-käyttöjärjestelmää. Sen pitäisi saada verkosta seuraavat vapaat osoitteet, eli IPv4-osoite 192.168.2.3 ja IPv6-osoite 2001:99a:807:1000::3003.

Komento 25 ipconfig-komento, jolla katsotaan Windows-käyttöjärjestelmän IP-konfiguraatio

```
lahna@DESKTOP-81CQE8K C:\Users\lahna>ipconfig /all

Windows IP Configuration

   Host Name . . . . . . . . . . . . : DESKTOP-81CQE8K
   Primary Dns Suffix  . . . . . . . :
   Node Type . . . . . . . . . . . . : Hybrid
   IP Routing Enabled. . . . . . . . : No
   WINS Proxy Enabled. . . . . . . . : No
   DNS Suffix Search List. . . . . . : home.arpa

Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . : home.arpa
   Description . . . . . . . . . . . : Intel(R) 82574L Gigabit Network
Connection
   Physical Address. . . . . . . . . : 52-54-00-3E-96-EA
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   IPv6 Address. . . . . . . . . . . :
2001:99a:807:1000::3003(Preferred)
   Lease Obtained. . . . . . . . . . : keskiviikko 14. elokuuta 2024
17.41.01
```

```
   Lease Expires . . . . . . . . . . : keskiviikko 14. elokuuta 2024
18.42.59
   Link-local IPv6 Address . . . . . :
fe80::1db3:ea44:dc68:b0c5%5(Preferred)
   IPv4 Address. . . . . . . . . . . : 192.168.2.3(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Lease Obtained. . . . . . . . . . : keskiviikko 14. elokuuta 2024
17.41.00
   Lease Expires . . . . . . . . . . : torstai 15. elokuuta 2024
1.40.59
   Default Gateway . . . . . . . . . : fe80::302b:bfff:fe60:cdd7%5
                                       192.168.2.1
   DHCP Server . . . . . . . . . . . : 192.168.64.2
   DHCPv6 IAID . . . . . . . . . . . : 106058752
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-2E-4E-37-00-52-54-
00-3E-96-EA
   DNS Servers . . . . . . . . . . . : 2001:99a:807:1000::2003
                                       2001:99a:807:1000::2004
                                       192.168.64.3
                                       192.168.64.4
   NetBIOS over Tcpip. . . . . . . . : Enabled
   Connection-specific DNS Suffix Search List :
                                       home.arpa
```

Tulos osoittaa, että automaattinen verkon konfiguraatio toimi odotetusti. Seuraavaksi testataan DNS:n toimivuus.

Komento 26 nslookup-komento, jolla selvitetään nimen perusteella IP-osoite

```
nslookup www.facebook.com.
Server:  2001:99a:807:1000::2003
Address:  2001:99a:807:1000::2003

Non-authoritative answer:
Name:    star-mini.c10r.facebook.com
Addresses:  2a03:2880:f113:81:face:b00c:0:25de
          157.240.205.35
Aliases:  www.facebook.com
```

Komento 27 ping-komento, jolla testaan internet-yhteys IPv4-protokollalla

```
ping 8.8.8.8

Pinging 8.8.8.8 with 32 bytes of data:
Reply from 8.8.8.8: bytes=32 time=18ms TTL=58
Reply from 8.8.8.8: bytes=32 time=18ms TTL=58
Reply from 8.8.8.8: bytes=32 time=10ms TTL=58
Reply from 8.8.8.8: bytes=32 time=10ms TTL=58

Ping statistics for 8.8.8.8:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 10ms, Maximum = 18ms, Average = 14ms
```

Ping on työkalu, jolla voidaan testata verkkoyhteyden toimivuutta lähettämällä ICMP-viestejä määritettyyn kohteeseen ja odottamalla vastausta. Komennossa 27 testataan IPv4-yhteyttä Google Public DNS -osoitteeseen 8.8.8.8. Tulosteesta nähdään, että kaikki lähetetyt paketit ovat saapuneet perille ja laite on vastaanottanut vastaukset ilman pakettihävikkiä. Lisäksi tuloste sisältää tietoa vasteajoista, jotka kertovat kuinka nopeasti laite kommunikoi kyseisen osoitteen kanssa. Tässä tapauksessa vasteajat vaihtelevat 10–18 millisekunnin välillä.

Komento 28 ping6-komento, jolla testaan internet-yhteys IPv6-protokollalla

```
ping6 2001:4860:4860::8888 -c 1
PING 2001:4860:4860::8888(2001:4860:4860::8888) 56 data bytes
64 bytes from 2001:4860:4860::8888: icmp_seq=1 ttl=58 time=14.3 ms

--- 2001:4860:4860::8888 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 14.279/14.279/14.279/0.000 ms
```

Ping6 on vastaava työkalu kuin ping, mutta se toimii IPv6-protokollalla. Komennossa 28 testataan IPv6-yhteyttä Google Public DNS -osoitteeseen 2001:4860:4860::8888. Tuloste osoittaa, että laite pystyy lähettämään yhden ICMPv6-paketin ja vastaanottamaan vastauksen ilman hävikkiä. Vasteaika on 14,3 millisekuntia, mikä osoittaa verkkoyhteyden olevan toimiva myös IPv6-protokollalla.

Molempien komentojen tulosten perusteella dynaaminen verkko toimii odotetusti, sillä yhteydet sekä IPv4- että IPv6-verkoissa toimivat ilman pakettihävikkiä ja vasteajat ovat alhaiset.

7.2 Staattisen verkon palveluiden testaus

Dynaamisessa verkossa testataan, että palvelimet saavat niille tarkoitetut IP-osoitteet käyttöjärjestelmän MAC-osoitteen perusteella. Kuha-nimisen koneen pitäisi saada IPv4-osoitteeksi 192.168.3.50 ja IPv6-osoitteeksi 2001:99a:807:1000::4002.

Komento 29 ip-komento, jolla nähdään Linuxin verkko-asetukset

```
ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel
state UP group default qlen 1000
```

```
link/ether 52:54:00:5d:b4:a9 brd ff:ff:ff:ff:ff:ff
inet 192.168.3.50/24 brd 192.168.3.255 scope global dynamic enp1s0
   valid_lft 28789sec preferred_lft 28789sec
inet6 2001:99a:807:1000::4002/128 scope global dynamic
   valid_lft 3992sec preferred_lft 2992sec
inet6 fe80::5054:ff:fe5d:b4a9/64 scope link
   valid_lft forever preferred_lft forever
```

Nimenselvitys ja internet-yhteydet toimivat kuten dynaamisessa verkossa, joten testin perusteella tämä on todettu toimivaksi.

7.3 Kaksoispinoverkon analysointi

Verkon testauksen yhteydessä havaittiin omituisuuksia nimipalvelun käytössä, joita tutkitaan lisää. Testaus ja valvonta tehdään kaappaamalla verkon liikennettä, minkä perusteella nähdään, mitä erilaiset käyttöjärjestelmät tekevät taustalla kaksoispinoverkossa.

Testausympäristö valmisteltiin siten, että br2-verkkoon jätetään päälle vain yksi testilaite, jonka jälkeen verkkoliikenne otetaan talteen. Nimipalveluliikennettä generoidaan samanaikaisesti lataamalla www4- ja www6-websivustot. Samalla mitataan sivustojen latausaikoja ja verrataan, onko niissä eroja.

Komento 30 tcpdump-komento, jolla seurataan br2-reitittimen DNS-liikennettä

```
root@discovery:/home/ruottte1# tcpdump -nntti br2 port 53
tcpdump: verbose output suppressed, use -v[v]... for full protocol
decode
listening on br2, link-type EN10MB (Ethernet), snapshot length 262144
bytes
1723652183.299501 IP6 2001:99a:807:1000::3002.40688 >
2001:99a:807:1000::2003.53: 6757+ A? www4.home.arpa. (32)
1723652183.299555 IP6 2001:99a:807:1000::3002.40688 >
2001:99a:807:1000::2003.53: 63851+ AAAA? www4.home.arpa. (32)
1723652183.300993 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3002.40688: 6757* 1/0/0 A 192.168.64.2 (48)
1723652183.301071 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3002.40688: 63851* 0/1/0 (79)
1723652183.305066 IP6 2001:99a:807:1000::3002.50941 >
2001:99a:807:1000::2003.53: 59856+ A? www4.home.arpa. (32)
1723652183.305103 IP6 2001:99a:807:1000::3002.50941 >
2001:99a:807:1000::2003.53: 49372+ AAAA? www4.home.arpa. (32)
1723652183.305468 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3002.50941: 59856* 1/0/0 A 192.168.64.2 (48)
1723652183.305522 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3002.50941: 49372* 0/1/0 (79)
```

Tcpdump on Linux sovellus, jolla voidaan kaapata ja analysoida verkon liikennettä. Komennossa 30 tcpdump tarkkailee kaikkia DNS-liikenteen paketteja br2-verkossa portissa 53. Valitulla -nntti-asetuksella komento näyttää tulosteen ilman osoitteiden nimiselvitystä, tulostaa tiivistetyt tiedot ja lisää verkon liitäntätiedot. Tämä auttaa seuraamaan nimipalvelukyselyjä reaaliajassa ja selvittämään, miten testilaitteet käyttävät DNS-palvelua. Tässä tapauksessa tulokset auttavat ymmärtämään, miten kaksoispinoympäristössä tapahtuva liikenne käyttäytyy ja millaisia kyselyjä eri käyttöjärjestelmät tekevät taustalla.

Kaappauksen aikana hauki-palvelimella ladataan www4.home.arpa -websivu.

Komento 31 lynx-komento, jolla ladataan websivusto komentotulkissa

```
lynx www4.home.arpa
```

Lynx on tekstipohjainen web-selain, jota käytetään komentoriviltä. Sitä voidaan hyödyntää testaamaan verkkosivujen latautumista kevyesti ilman graafista käyttöliittymää. Komennolla 31 lynx ladataan websivusto www4.home.arpa. Tämä luo verkkoliikennettä, joka näkyy samanaikaisesti tcpdump-kaappauksessa. Näin voidaan tutkia, miten nimipalvelukyselyt ja web-liikenne tapahtuvat taustalla kaksoispinoympäristössä. Lisäksi lynx tarjoaa mahdollisuuden mitata sivustojen latausaikoja ja vertailla IPv4- ja IPv6-protokollien välillä esiintyviä eroja.

Analysoitaessa tulosta havaitaan heti, että Linux suoritti IPv6-palvelimelle nimikyselyt sekä IPv4- että IPv6-osoitteille. Lopputuloksena voidaan todeta, että kaksoispinoympäristössä nimipalvelukyselyjen määrä tuplaantuu. Tämä tieto on tärkeä huomioida, kun suunnitellaan kaksoispinon käyttöönottoa.

Seuraavaksi analysoidaan, mitä DNS-kyselyitä Windows 10 -käyttöjärjestelmä tekee taustalla, kun se lataa web-sivua kaksoispinoympäristössä. Testissä ajetaan komento 30, jolla kaapataan kaikki DNS-liikenne. Windowsissa avataan sivusto web-selaimeen.

Komento 32 tcpdump-komento, jolla seurataan DNS-liikennettä br2-verkosta

```
tcpdump -nntti br2 port 53
tcpdump: verbose output suppressed, use -v[v]... for full protocol
decode
listening on br2, link-type EN10MB (Ethernet), snapshot length 262144
bytes
1723653055.553828 IP6 2001:99a:807:1000::3003.64456 >
2001:99a:807:1000::2003.53: 38456+ AAAA? www4.home.arpa. (32)
1723653055.554546 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.64456: 38456* 0/1/0 (79)
```

```
1723653055.555631 IP6 2001:99a:807:1000::3003.58117 >
2001:99a:807:1000::2003.53: 26389+ A? www4.home.arpa. (32)
1723653055.556070 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.58117: 26389* 1/0/0 A 192.168.64.2 (48)
1723653055.556639 IP6 2001:99a:807:1000::3003.63452 >
2001:99a:807:1000::2003.53: 51647+ Type65? www4.home.arpa. (32)
1723653055.557157 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.63452: 51647* 0/1/0 (79)
1723653055.558021 IP6 2001:99a:807:1000::3003.49554 >
2001:99a:807:1000::2003.53: 26244+ AAAA? www4.home.arpa. (32)
1723653055.558540 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.49554: 26244* 0/1/0 (79)
1723653055.559182 IP6 2001:99a:807:1000::3003.51892 >
2001:99a:807:1000::2003.53: 50470+ A? www4.home.arpa. (32)
1723653055.559696 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.51892: 50470* 1/0/0 A 192.168.64.2 (48)
1723653055.926793 IP6 2001:99a:807:1000::3003.63353 >
2001:99a:807:1000::2003.53: 14900+ AAAA? nav-
edge.smartscreen.microsoft.com. (52)
1723653055.927300 IP6 2001:99a:807:1000::3003.62740 >
2001:99a:807:1000::2003.53: 12745+ A? nav-
edge.smartscreen.microsoft.com. (52)
1723653055.927791 IP6 2001:99a:807:1000::3003.57370 >
2001:99a:807:1000::2003.53: 8684+ Type65? nav-
edge.smartscreen.microsoft.com. (52)
1723653055.965303 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.57370: 8684 2/1/0 CNAME prod-atm-wds-
edge.trafficmanager.net., CNAME prod-agic-ne-
1.northeurope.cloudapp.azure.com. (238)
1723653055.971393 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.63353: 14900 2/1/0 CNAME prod-atm-wds-
edge.trafficmanager.net., CNAME prod-agic-ne-
1.northeurope.cloudapp.azure.com. (238)
1723653055.971990 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.62740: 12745 3/0/0 CNAME prod-atm-wds-
edge.trafficmanager.net., CNAME prod-agic-ne-
1.northeurope.cloudapp.azure.com., A 20.191.45.158 (177)
1723653055.999948 IP6 2001:99a:807:1000::3003.51877 >
2001:99a:807:1000::2003.53: 61685+ AAAA? browser.events.data.msn.com.
(45)
1723653056.000482 IP6 2001:99a:807:1000::3003.60291 >
2001:99a:807:1000::2003.53: 3916+ A? browser.events.data.msn.com. (45)
1723653056.000974 IP6 2001:99a:807:1000::3003.56144 >
2001:99a:807:1000::2003.53: 37106+ Type65?
browser.events.data.msn.com. (45)
1723653056.173027 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.56144: 37106 2/1/0 CNAME
global.asimov.events.data.trafficmanager.net., CNAME
onedscolprdweu12.westeurope.cloudapp.azure.com. (240)
1723653056.184918 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.51877: 61685 2/1/0 CNAME
global.asimov.events.data.trafficmanager.net., CNAME
onedscolprdeus21.eastus.cloudapp.azure.com. (236)
1723653056.194360 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.60291: 3916 3/0/0 CNAME
global.asimov.events.data.trafficmanager.net., CNAME
onedscolprdeus16.eastus.cloudapp.azure.com., A 52.168.117.171 (175)
1723653058.107506 IP6 2001:99a:807:1000::3003.51508 >
2001:99a:807:1000::2003.53: 65058+ AAAA? www4.home.arpa. (32)
1723653058.108106 IP6 2001:99a:807:1000::3003.62745 >
2001:99a:807:1000::2003.53: 31009+ A? www4.home.arpa. (32)
```

```
1723653058.108172 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.51508: 65058* 0/1/0 (79)
1723653058.108406 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.62745: 31009* 1/0/0 A 192.168.64.2 (48)
1723653058.108807 IP6 2001:99a:807:1000::3003.63463 >
2001:99a:807:1000::2003.53: 26606+ Type65? www4.home.arpa. (32)
1723653058.109041 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.63463: 26606* 0/1/0 (79)
1723653058.110149 IP6 2001:99a:807:1000::3003.57806 >
2001:99a:807:1000::2003.53: 62796+ AAAA? www4.home.arpa. (32)
1723653058.110538 IP6 2001:99a:807:1000::3003.59189 >
2001:99a:807:1000::2003.53: 35381+ A? www4.home.arpa. (32)
1723653058.110588 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.57806: 62796* 0/1/0 (79)
1723653058.110897 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.59189: 35381* 1/0/0 A 192.168.64.2 (48)
1723653058.125978 IP6 2001:99a:807:1000::3003.59382 >
2001:99a:807:1000::2003.53: 3286+ AAAA? www4.home.arpa. (32)
1723653058.126499 IP6 2001:99a:807:1000::3003.52949 >
2001:99a:807:1000::2003.53: 21332+ A? www4.home.arpa. (32)
1723653058.126500 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.59382: 3286* 0/1/0 (79)
1723653058.126709 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.52949: 21332* 1/0/0 A 192.168.64.2 (48)
1723653058.227213 IP6 2001:99a:807:1000::3003.49619 >
2001:99a:807:1000::2003.53: 2242+ AAAA? www4.home.arpa. (32)
1723653058.227672 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.49619: 2242* 0/1/0 (79)
1723653058.230543 IP6 2001:99a:807:1000::3003.53222 >
2001:99a:807:1000::2003.53: 6850+ A? www4.home.arpa. (32)
1723653058.230954 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.53222: 6850* 1/0/0 A 192.168.64.2 (48)
1723653059.249985 IP6 2001:99a:807:1000::3003.51566 >
2001:99a:807:1000::2003.53: 44331+ AAAA? www.google.com. (32)
1723653059.250426 IP6 2001:99a:807:1000::3003.49512 >
2001:99a:807:1000::2003.53: 14193+ A? www.google.com. (32)
1723653059.250693 IP6 2001:99a:807:1000::3003.49749 >
2001:99a:807:1000::2003.53: 36583+ Type65? www.google.com. (32)
1723653059.251266 IP6 2001:99a:807:1000::3003.59308 >
2001:99a:807:1000::2003.53: 30277+ AAAA? www.bing.com. (30)
1723653059.251509 IP6 2001:99a:807:1000::3003.57960 >
2001:99a:807:1000::2003.53: 44521+ A? www.bing.com. (30)
1723653059.251564 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.59308: 30277 8/0/0 CNAME www-
www.bing.com.trafficmanager.net., CNAME www.bing.com.edgekey.net.,
CNAME e86303.dscx.akamaiedge.net., AAAA 2001:998:12:e::c1e5:6dcc, AAAA
2001:998:12:e::c1e5:6dcd, AAAA 2001:998:12:e::c1e5:6dcf, AAAA
2001:998:12:e::c1e5:6dd4, AAAA 2001:998:12:e::c1e5:6dc5 (297)
1723653059.251765 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.57960: 44521 12/0/0 CNAME www-
www.bing.com.trafficmanager.net., CNAME www.bing.com.edgekey.net.,
CNAME e86303.dscx.akamaiedge.net., A 62.115.252.66, A 62.115.252.72, A
62.115.252.59, A 62.115.252.57, A 62.115.252.67, A 62.115.252.58, A
62.115.252.73, A 62.115.252.74, A 80.239.137.136 (301)
1723653059.251815 IP6 2001:99a:807:1000::3003.55411 >
2001:99a:807:1000::2003.53: 17818+ Type65? www.bing.com. (30)
1723653059.252024 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.55411: 17818 3/1/0 CNAME www-
www.bing.com.trafficmanager.net., CNAME www.bing.com.edgekey.net.,
CNAME e86303.dscx.akamaiedge.net. (235)
```

```
1723653059.356639 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.49512: 14193 1/0/0 A 216.58.210.132 (48)
1723653059.356664 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.51566: 44331 1/0/0 AAAA
2a00:1450:4026:804::2004 (60)
1723653059.356788 IP6 2001:99a:807:1000::2003.53 >
2001:99a:807:1000::3003.49749: 36583 1/0/2 Type65 (101)
```

Analyysin perusteella tilanne on sama: Windows yrittää selvittää jokaiselle nimelle sekä IPv4- että IPv6-osoitteen. Windowsin kanssa näkyy myös ketjureaktio, jonka verkkoselaaminen aiheuttaa. Koska web-sivustolla oli koodinpätkä, jolla se haki kuvia Bing- ja Google-sivustoilta, käyttöjärjestelmän piti selvittää näiden sivustojen osoitteet taustalla. Huomionarvoista on, että laite ei missään vaiheessa tehnyt nimikyselyä kaksoispinoverkon IPv4-palvelulle, vaan kaikki kuormitus kohdistui IPv6-palveluun.

7.4 DNS-alueen hallintasovelluksen testaus

Testauksessa nimipalveluun lisättiin kolme uutta osoitetta: testi1, testi2 ja testi3. Muutos tehtiin ensin manuaalisesti ja mittasin siihen kuluneen ajan sekunneissa. Tämän jälkeen tein saman muutoksen sovellusta hyödyntäen ja mittasin ajan myös siinä.

Manuaalinen muutos vaati ensin yhteyden muodostamisen dns1-palvelimelle. Tämän jälkeen zone-tiedostoa täytyi muokata ja muutokset piti ottaa käyttöön. Tähän kului aikaa yhteensä 102 sekuntia.

Sovelluksella sama muutos tehtiin 86 sekunnissa, joten sen hyöty näkyy selvästi. Muutos oli yksinkertainen: lisättiin vain uusia tietueita. Esimerkiksi tietueiden muuttaminen tai muiden kuin osoitetietojen käsittely voisi vaatia lisää harkintaa, joten tällaisen työkalun käyttö on suositeltavaa.

8 Johtopäätökset ja pohdinta

Työn tavoitteena oli tutkia IPv4- ja IPv6-verkkotekniikoita sekä perehtyä meneillään olevaan siirtymäkauteen. Siirtymäkauden tekniikoita on useita ja teoriaosuudessa käsiteltiin näitä tekniikoita. Erityinen painopiste oli kaksoispinotekniikassa, koska yksi tavoitteista oli tutustua avoimen lähdekoodin sovelluksiin, joilla kaksoispinotekniikkaa voitaisiin testata testiympäristössä. Testauksessa keskityttiin verkon toimivuuteen ja verkon peruspalveluiden toimintaan.

Testiympäristö toteutettiin virtualisointitekniikkaa hyödyntäen, sillä sen käyttö tällaisessa projektissa on kustannustehokasta ja nopeaa. Vastaavan ympäristön toteuttaminen vaatii käytännössä vain internet-yhteyden ja tietokoneen. Käytetyt sovellukset olivat avoimen lähdekoodin projekteja, joihin ei liity maksuja. Testiympäristön asennus onnistui ja siellä suoritetut testit Linux- ja Windows-käyttöjärjestelmillä olivat toimivia.

Kolmantena tavoitteena oli tutustua verkonhallintaan liittyviin avoimen lähdekoodin projekteihin, mutta sellaisia ei löytynyt, vaan kaikki vaihtoehdot olivat maksullisia. Niinpä kehitin ohjelmiston, joka lukee verkon konfiguraation ohjauskoneelta ja luo sen perusteella konfiguraation nimipalvelua varten. Sovellus täytti sille asetetut tavoitteet.

Testiympäristön rakentaminen kuvattiin työn osuudessa yksityiskohtaisesti ja sitä hyödynnettiin lopuksi kaksoispinoverkon asiakaslaitteiden käyttäytymisen tutkimiseen. Testiympäristön avulla havaittiin laitteiden välillä eroavaisuuksia ja huomattiin, miten kaksoispinotekniikka vaikuttaa nimipalveluihin siten, että kaikki liikenne suuntautuu IPv6-palvelulle. Tämä havainto on tärkeä laitteiden kapasiteetin suunnittelussa.

IPv6- ja kaksoispinotekniikkaan tutustuminen on käytännössä hyödyllistä ylläpitotehtävissä sekä projekteissa, joissa suunnitellaan kaksoispinon käyttöönottoa. Testiympäristön rakentaminen tuo jatkossa hyötyä myös siksi, että sen pohjalta voidaan arvioida ympäristön rakentamiseen tarvittava aika ja kustannukset.

Laboratorioympäristön ansiosta jatkossa on helppo testata uusia, erilaisia laitteita käytännössä ja varmistaa, miten ne hyödyntävät kaksoispinotekniikkaa. Esimerkiksi laboratorioympäristöön voisi liittää langattoman tukiaseman, johon voidaan yhdistää matkapuhelimia. Näiden laitteiden verkkoliikenteestä voidaan kaapata tietoa siitä, mitä ne

odottavat verkon automaattiselta konfiguraatiolta. Lisäksi voidaan varmistaa, miten nämä laitteet käyttävät nimipalvelua: tulevatko kaikki kyselyt IPv6:llä, vai onko jokin tietty laite mahdollisesti riippuvainen myös IPv4-nimipalvelusta.

Konsepti verkonhallintasovelluksesta, jossa yhdellä palvelimella on tieto verkon konfiguraatiosta ja joka sen perusteella generoi valmiita tiedostoja nimipalvelulle, osoittautui osittain toimivaksi. Käänteisnimipalvelun konfiguraation tuottamisessa on kuitenkin haasteita, jotka täytyy ratkaista. Myös DHCP-konfiguraation generointi jäi ajatusasteelle. Konseptina molempien pitäisi kuitenkin olla mahdollisia: yksi kone, jossa tietoa hallitaan ja sen perusteella generoidaan sovelluksille valmiit konfiguraatiot. Tämä vaatii vain tiedon lukemista ja sen muuntamista sovelluksen käyttämään muotoon. Automatisoinnin hyötyinä ovat esimerkiksi konfiguroinnin nopeuttaminen ja inhimillisistä virheistä johtuvien ongelmien vähentäminen.

9 Yhteenveto

Opinnäytetyössäni tutkin kaksoispinotekniikkaa ja sen soveltuvuutta siirtymävaiheessa IPv4:stä IPv6:een. Tavoitteena oli selvittää tekniikan käytännön toimivuutta. Tutkimuskysymyksiin vastattiin testaamalla kaksoispinotekniikkaa laboratorioympäristössä, jossa selvisi, että tekniikalla tarkoitetaan molempien protokollien rinnakkaista käyttöä. Samalla selvitettiin, kuinka laitteet ja palvelut, kuten DNS ja DHCP, toimivat tässä ympäristössä.

Tutkin myös avoimen lähdekoodin DNS- ja DHCP-sovelluksia, jotka tukevat kaksoispinotekniikkaa. Sovellusten toimivuutta testattiin laboratorioympäristössä ja ne osoittautuivat toimiviksi. Tämä osoitti, että avoimen lähdekoodin ratkaisut voivat tukea siirtymävaihetta ja olla hyödyllisiä verkon peruspalveluiden toteuttamisessa ja testaamisessa.

Kolmas tutkimuskysymys käsitteli avoimen lähdekoodin verkonhallintaratkaisuja. Koska valmiita vaihtoehtoja ei löytynyt, kehitin oman työkalun nimipalvelun konfiguraation automatisointiin. Työkalu vähensi manuaalista työtä ja nopeutti konfiguraatioprosesseja toimien hyvin käytännössä.

Tutkimuksen tulokset tarjoavat hyvän pohjan kaksoispinotekniikan käyttöönotolle ja osoittavat avoimen lähdekoodin ratkaisujen potentiaalin. Tulevaisuudessa kehitetyt työkalut ja testausympäristöt voivat auttaa yrityksiä verkkoinfrastruktuurin hallinnassa.

Työn myötä opin paljon IPv4- ja IPv6-tekniikoista sekä niiden siirtymävaiheen automaatiosta, erityisesti DHCP:n ja DNS-palveluiden osalta. Käytännön kokemus avoimen lähdekoodin sovelluksista syvensi ymmärrystäni verkkopalveluiden automatisoinnista. Tämä tieto auttaa minua tulevissa projekteissa.

Jatkan tulevaisuudessa ympäristön kehittämistä ja laajennan verkonhallintasovelluksen kattamaan myös automaattisen konfiguraation. Tarkoituksena on myös luoda sille web-pohjainen käyttöliittymä.

Lähteet

5 Most Popular Operating Systems. (ei pvm.). Noudettu 4. syyskuuta 2024, osoitteesta https://www.wgu.edu/blog/5-most-popular-operating-systems1910.html

Adia. (2024, maaliskuu 31). CVE-2024-3094 XZ Backdoor: All you need to know. *JFrog*. https://jfrog.com/blog/xz-backdoor-attack-cve-2024-3094-all-you-need-to-know/

Al-hamadani, A. T. H., & Lencse, G. (2021). Survey on the performance analysis of IPv6 transition technologies. *Acta Technica Jaurinensis*, *14*(2), 186–211. https://doi.org/10.14513/actatechjaur.00577

Azorin, P. (2020, toukokuu 15). What is Linux and why you should be using it. *Medium*. https://medium.com/@guestposts_92864/what-is-linux-and-why-you-should-be-using-it-5513d21ecdc0

Barr, D. (1996). *Common dns operational and configuration errors* (No. RFC1912; p. RFC1912). RFC Editor. https://doi.org/10.17487/rfc1912

Bush, R. (Ed.). (2011). *The address plus port (A+p) approach to the ipv4 address shortage* (No. RFC6346; p. RFC6346). RFC Editor. https://doi.org/10.17487/rfc6346

Deering, S., & Hinden, R. (2017). *Internet protocol, version 6 (Ipv6) specification* (No. RFC8200; p. RFC8200). RFC Editor. https://doi.org/10.17487/RFC8200

Mockapetris, P. V. (1987). *Domain names—Concepts and facilities* (No. RFC1034; p. RFC1034). RFC Editor. https://doi.org/10.17487/rfc1034

Mockapetris, P. V. (1987). *Domain names—Implementation and specification* (No. RFC1035; p. RFC1035). RFC Editor. https://doi.org/10.17487/rfc1035

Droms, R. (1997). *Dynamic host configuration protocol* (No. RFC2131; p. RFC2131). RFC Editor. https://doi.org/10.17487/rfc2131

Eggert, L., Fairhurst, G., & Shepherd, G. (2017). *Udp usage guidelines* (No. RFC8085; p. RFC8085). RFC Editor. https://doi.org/10.17487/RFC8085

Fifty Shades of DOCSIS Device Management | Axiros. (ei pvm.). Axiros | Open Device & Service Management. Noudettu 30. heinäkuuta 2024, osoitteesta https://www.axiros.com/blog/2023/01/19/part-2-fifty-shades-of-docsis-device-management

Hall, E. (2000). *Internet core protocols*. O'Reilly Media, Inc.

Hardware results. (ei pvm.). Noudettu 4. syyskuuta 2024, osoitteesta https://catalog.redhat.com/search

Deering, S., & Hinden, R. (1998). *Internet protocol, version 6 (Ipv6) specification* (No. RFC2460; p. RFC2460). RFC Editor. https://doi.org/10.17487/rfc2460

Hubert, A., & Van Mook, R. (2009). *Measures for making dns more resilient against forged answers* (No. RFC5452; p. RFC5452). RFC Editor. https://doi.org/10.17487/rfc5452

Postel, J. (1981). *Internet protocol* (No. RFC0791; p. RFC0791). RFC Editor. https://doi.org/10.17487/rfc0791

Kaario, K. (2002). *TCP/IP-verkot*. Docendo.

Mrugalski, T., Siodelski, M., Volz, B., Yourtchenko, A., Richardson, M., Jiang, S., Lemon, T., & Winters, T. (2018). *Dynamic host configuration protocol for ipv6(Dhcpv6)* (No. RFC8415; p. RFC8415). RFC Editor. https://doi.org/10.17487/RFC8415

Opensource.com. (ei pvm.). *What is open source?* | *Opensource.com*. Noudettu 1. elokuuta 2024, osoitteesta https://opensource.com/resources/what-open-source

Operating System Market Share Worldwide. (ei pvm.). StatCounter Global Stats. Noudettu 4. syyskuuta 2024, osoitteesta https://gs.statcounter.com/os-market-share

School of Social Sciences, Hellenic Open Unoversity, Mihalos, M. G., Nalmpantis, S. I., Dpt of Electrical Engineering, Eastern Macedonia and Thrace Institute of Technology, Kavala, Greece, Ovaliadis, K., & Dpt of Electrical Engineering, Eastern Macedonia and Thrace Institute of Technology, Kavala, Greece. (2019). Design and implementation of firewall security policies using linux iptables. *Journal of Engineering Science and Technology Review*, *12*(1), 80–86. https://doi.org/10.25103/jestr.121.09

Setting the printer's IP addressing parameters. (ei pvm.). Noudettu 30. heinäkuuta 2024, osoitteesta https://www.office.xerox.com/userdoc/PShare4/htmlnet/tcppcap9.htm

Kea Administrator Reference Manual—Kea 2.2.0 documentation. (ei pvm.). Noudettu 25. syyskuuta 2024, osoitteesta https://kea.readthedocs.io/en/kea-2.2.0/

Shulman, H., & Waidner, M. (2014). DNSSEC for cyber forensics. *EURASIP Journal on Information Security*, *2014*(1), 16. https://doi.org/10.1186/s13635-014-0016-2

Narten, T., Nordmark, E., Simpson, W., & Soliman, H. (2007). *Neighbor Discovery for IP version 6 (Ipv6)* (No. RFC4861; p. RFC4861). RFC Editor. https://doi.org/10.17487/rfc4861

tcptraceroute(1)—Linux man page. (ei pvm.). Noudettu 29. syyskuuta 2024, osoitteesta https://linux.die.net/man/1/tcptraceroute

Thomson, S., Narten, T., & Jinmei, T. (2007). *Ipv6 stateless address autoconfiguration* (No. RFC4862; p. RFC4862). RFC Editor. https://doi.org/10.17487/rfc4862

Postel, J. (1981). *Transmission control protocol* (No. RFC0793; p. RFC0793). RFC Editor. https://doi.org/10.17487/rfc0793

Understanding how Facebook disappeared from the Internet. (2021, lokakuu 4). The Cloudflare Blog. https://blog.cloudflare.com/october-2021-facebook-outage

What is an Operating System (OS)? | Definition from TechTarget. (ei pvm.). WhatIs. Noudettu 6. syyskuuta 2024, osoitteesta https://www.techtarget.com/whatis/definition/operating-system-OS

Wu, Q.-X. (2012). The research and application of firewall based on netfilter. *Physics Procedia*, *25*, 1231–1235. https://doi.org/10.1016/j.phpro.2012.03.225

Egevang, K. B., & Francis, P. (1994). *The ip network address translator(Nat)* (Request for Comments No. RFC 1631). Internet Engineering Task Force. https://datatracker.ietf.org/doc/rfc1631/

Liite 1: Aineistonhallintasuunnitelma

Opinnäytetyöni aikana kerätty aineisto koostuu teknisistä tiedoista laboratorioympäristön konfiguroinnista ja siihen liittyvästä dokumentaatiosta. Kaikki dokumentointi ja opinnäytetyöhön liittyvä data on tallennettu HAMKin levytilaan (/hamk) ja siitä on otettu säännöllisesti varmuuskopiot Dropbox-pilvipalveluun. Varmuuskopiot varmistavat, että aineisto on suojattu ja tarvittaessa palautettavissa.

Opinnäytetyössäni ei käsitelty arkaluonteista tai luottamuksellista tietoa, eikä henkilötietoja tai muita salassa pidettäviä tietoja ole tallennettu pilvipalveluihin. Tämän vuoksi aineiston säilytys pilvipalvelussa ei riko tietoturva- tai tietosuojamääräyksiä.

Opinnäytetyöhön liittyvä laboratorioympäristö on asennettu erilliselle Discovery Linux -palvelimelle ja ympäristön konfigurointi on dokumentoitu yksityiskohtaisesti tässä työssä. Tietojenhallinnan osalta palvelimen ympäristön ylläpitoon ja sen dokumentointiin on kiinnitetty huomiota tietoturvan varmistamiseksi.

Kaikki aineisto säilytetään vähintään vuoden ajan opinnäytetyön hyväksymispäivästä alkaen, jotta opinnäytetyön tulokset voidaan tarvittaessa tarkistaa tai vahvistaa. Tämän ajanjakson jälkeen aineisto joko arkistoidaan tai tuhotaan asianmukaisesti.

Tietojen käsittelyssä ja säilytyksessä on huomioitu tietoturva sekä varmuuskopioinnin säännöllisyys. Aineiston käyttöön ovat oikeutettuja vain projektiin osallistuvat henkilöt ja käyttöoikeudet on rajoitettu tarpeen mukaan.

Tutkimusaineistoa ei jatkokäytetä. Opinnäytetyön tekijä säilyttää aineiston tietoturvallisesti vuoden ajan opinnäytetyön hyväksymispäivästä, jotta opinnäytetyön tulokset voidaan tarvittaessa varmistaa ja hävittää tämän jälkeen aineiston tietoturvallisesti.

Liite 2: Python-sovellus, joka generoi DNS-zone-tiedoston ohjaustiedoston perusteella

```
import os
import time

# Configuration
input_file = 'config/my-network'
zone_output_file = 'home.arpa'
reverse_output_file = 'reverse.home.arpa'
publish_directory = '/opt/dns/myzones/'

def get_serial():
    # Generate serial based on the current timestamp
    return int(time.strftime('%Y%m%d%H'))

def parse_zone_file():
    soa_record = ""
    records = []
    with open(input_file, 'r') as file:
        lines = file.readlines()
        soa_record = lines[0].strip()
        for line in lines[1:]:
            line = line.strip()
            if line and not line.startswith("#"):
                records.append(line)
    return soa_record, records

def generate_zone_file(soa_record, records, serial):
    soa_parts = soa_record.split()

    # Assuming the soa_record has the format:
    # home.arpa SOA admin.home.arpa 604800 86400 2419200 604800
    primary_ns = "dns1.home.arpa."
    contact_email = soa_parts[2].replace('@', '.') + '.'

    # Proper SOA record
    soa_with_serial = f"{primary_ns} {contact_email} {serial}
{soa_parts[3]} {soa_parts[4]} {soa_parts[5]} {soa_parts[6]}"

    zone_file_content = f"$TTL 604800\n@    IN    SOA    {soa_with_serial}\
n\n"
    zone_file_content += f"@    IN    NS    dns1.home.arpa.\n"
    zone_file_content += f"@    IN    NS    dns2.home.arpa.\n\n"

    for record in records:
        parts = record.split()
        if len(parts) == 3:
            name, record_type, value = parts
            # Ensure names are fully qualified by appending a dot
            if not name.endswith('.'):
                name += '.'
            zone_file_content += f"{name}    IN    {record_type}
{value}\n"

    with open(os.path.join(publish_directory, zone_output_file), 'w') as
file:
        file.write(zone_file_content)

def generate_reverse_dns_file(records):
```

```
    reverse_records = []
    for record in records:
        parts = record.split()
        if len(parts) == 3:
            name, record_type, value = parts
            if record_type == "A":
                ip = value.split('.')
                if len(ip) == 4:
                    reverse_records.append(f"{ip[3]}.{ip[2]}.{ip[1]}.
{ip[0]}.in-addr.arpa.    IN    PTR    {name}.")

    reverse_content = "$TTL 604800\n"
    for reverse_record in reverse_records:
        reverse_content += f"{reverse_record}\n"

    with open(os.path.join(publish_directory, reverse_output_file), 'w') as
file:
        file.write(reverse_content)

def main():
    if not os.path.exists(publish_directory):
        os.makedirs(publish_directory)

    soa_record, records = parse_zone_file()
    new_serial = get_serial()

    generate_zone_file(soa_record, records, new_serial)
    generate_reverse_dns_file(records)

    print(f"Zone files generated and published to {publish_directory}")

if __name__ == "__main__":


    main()
```

Liite 3: Shell-ohjelma, joka tarkistaa, onko zone-tiedosto muuttunut. Jos ei ole, se ei tee mitään. Jos muutoksia on, se kopioi uuden tiedoston ja lataa BIND:n uudelleen

```
#!/bin/bash

REMOTE_SERVER="lippa.home.arpa"
REMOTE_DIR="/opt/dns/myzones/"
LOCAL_DIR="/etc/bind/zones/"
BIND_CONF="/etc/bind/named.conf.local"
LOG_FILE="/var/log/dns-update.log"
TMP_DIR="/tmp/dns-update/"
ZONE_FILES=("home.arpa" "reverse.home.arpa")

mkdir -p $TMP_DIR

echo "$(date): Starting zone file synchronization." >> $LOG_FILE
for ZONE in "${ZONE_FILES[@]}"; do
    scp "@$REMOTE_SERVER:$REMOTE_DIR$ZONE" "$TMP_DIR"
    if [ $? -ne 0 ]; then
        echo "$(date): Error fetching $ZONE from $REMOTE_SERVER." >>
$LOG_FILE
        exit 1
    fi
done

CHANGES_DETECTED=0
for ZONE in "${ZONE_FILES[@]}"; do
    if ! cmp -s "$TMP_DIR$ZONE" "$LOCAL_DIR$ZONE"; then
        CHANGES_DETECTED=1
        echo "$(date): Changes detected in $ZONE." >> $LOG_FILE
    else
        echo "$(date): No changes detected in $ZONE." >> $LOG_FILE
    fi
done

if [ $CHANGES_DETECTED -eq 0 ]; then
    echo "$(date): No changes detected. Exiting." >> $LOG_FILE
    rm -rf $TMP_DIR
    exit 0
fi

for ZONE in "${ZONE_FILES[@]}"; do
    cp "$TMP_DIR$ZONE" "$LOCAL_DIR"
    if [ $? -ne 0 ]; then
        echo "$(date): Error copying $ZONE to $LOCAL_DIR." >> $LOG_FILE
        exit 1
    fi
done

echo "$(date): Updating BIND configuration." >> $LOG_FILE
cat > $BIND_CONF <<EOL
zone "home.arpa" {
    type master;
    file "$LOCAL_DIR/home.arpa";
};

zone "64.168.192.in-addr.arpa" {
    type master;
    file "$LOCAL_DIR/reverse.home.arpa";
```

```
};
EOL

named-checkconf $BIND_CONF
if [ $? -ne 0 ]; then
    echo "$(date): Configuration validation failed." >> $LOG_FILE
    exit 1
fi

systemctl reload bind9
if [ $? -ne 0 ]; then
    echo "$(date): Failed to reload BIND service." >> $LOG_FILE
    exit 1
fi

rm -rf $TMP_DIR
echo "$(date): DNS update completed successfully." >> $LOG_FILE
exit 0
```