Jesse Tuunanen

CISCO-VERKOT JA IS-IS PROTOKOLLA

Opinnäytetyö
Tietotekniikka

Toukokuu 2012

MIKKELIN AMMATTIKORKEAKOULU
Mikkeli University of Applied Sciences

KUVAILULEHTI

MIKKELIN AMMATTIKORKEAKOULU Mikkeli University of Applied Sciences	**Opinnäytetyön päivämäärä** 18.05.2012
Tekijä(t) Jesse Tuunanen	**Koulutusohjelma ja suuntautuminen** Tietotekniikan koulutusohjelma
Nimeke Cisco-verkot ja IS-IS -protokolla	

Tiivistelmä

Opinnäytetyön aiheena on Cisco-verkot ja IS-IS -protokolla. Opinnäytetyön tavoitteena on rakentaa ja suunnitella kuvitteellinen toimistoverkko käyttäen lähiverkkotekniikan laitteistoa sekä kirjallisuutta apuna yhteistyössä Aki Timosen kanssa. Aihepiirinä tässä opinnäytetyössä on lähiverkkotekniikat sekä reitittimet ja näillä tehty käytännön osuus verkon rakentamisesta.

Opinnäytetyössä käydään läpi myös teoriaa lähiverkkotekniikan perusteita, OSI- ja TCP/IP -malleista ja näiden sisällöstä. Sisältöön kuuluu eri kerrosten tehtävät sekä mitä näihin kuuluu. Malleihin liittyvän teorian lisäksi käydään läpi opinnäytetyössä tarvittavia protokollia sekä näiden protokollien tehtävät ja toimintaperiaatteet tietoliikenneverkoissa. Lisäksi käydään myös läpi reitittimiin liittyvää teoriaa sekä reititysprotokollia.

Opinnäytetyössä toteutetaan myös tämän kuvitteellisen verkon suunnittelu Cisco PacketTracer ohjelman avulla sekä myös verkon rakentaminen käyttäen apuna Mikkelin ammattikorkeakoulun Mikpolin tiloissa sijaitsevaa Ciscon laitteistoa. Tässä verkossa reitittimissä käytetään IS-IS -reititysprotokollaa yhdistämään kaikki reitittimet keskenään. Verkon rakentamisen jälkeen käydään läpi vielä verkon testaus ja varmistus näin toimivaksi.

Asiasanat (avainsanat)

Cisco, Reitittimet, TCP/IP, IS-IS -protokolla

Sivumäärä	**Kieli**	**URN**
34	Suomi	

Huomautus (huomautukset liitteistä)

Ohjaavan opettajan nimi Matti Juutilainen	**Opinnäytetyön toimeksiantaja**

DESCRIPTION

MIKKELIN AMMATTIKORKEAKOULU Mikkeli University of Applied Sciences	**Date of the bachelor's thesis** 18.05.2012
Author(s) Jesse Tuunanen	**Degreeprogramme and option** Information Technology

Name of the bachelor's thesis

Cisco-networks and IS-IS -protocol

Abstract

The subject of this thesis is Cisco-networks and IS-IS -protocol. The goal of this thesis is to build and design an imaginary officenetwork with the help of network devices and literature with the cooperation of Aki Timonen. The whole subject of this thesis contains local area network technology, routers and the building of the netowork using the network devices.

In this thesis there is also theory about the local area network technology, OSI- and TCP/IP -models. Theory about the models contains the different layers and the tasks of the layers. Theory section also contains information about the protocols including the tasks and principles of them in the networkdesign. Also the theory section contains theory about routers and routing protocols.

In the last section of this thesis is the build and design of the imaginary officenetwork with the usage of Cisco Packet Tracer -application and the Cisco-equipment located in Mikpoli of Mikkeli University of Applied Sciences. The network is designed and build to use with IS-IS -routing protocol to connect the different routers. After the build phase I will go through the methods of testing to check if the routing in the network is working correctly.

Subjectheadings, (keywords)

Cisco, Routers, IS-IS –protocol, TCP/IP

Pages	**Language**	**URN**
34	Finnish	

Remarks, notes on appendices

Tutor	**Bachelor's thesis assigned by**
Matti Juutilainen	

SISÄLTÖ

LIITE/LIITTEET

1. Reitittimen R3.3 asetukset
2. Reitittimen R3.2 asetukset
3. Reitittimen R3.1 asetukset
4. Reitittimen R6.3 asetukset
5. Reitittimen R6.2 asetukset
6. Reitittimen R6.1 asetukset

1 JOHDANTO

Tietoliikenneverkot ovat nykyään arkipäivää. Niitä hyödynnetään jokaisella tekniikan osa-alueella. Lähes kaikki nykyaikana itsestään selvänä pitämämme arkipäiväiset asiatkin perustuvat tiedonsiirtoon Internetin avustuksella. Myös yritysmaailmassa kaikista pienimmilläkin yrityksillä on nykyään yrityksen sisäiset verkot, jotka usein perustuvat tässä työssä käsiteltyihin tietoliikenneverkkotekniikan ratkaisuihin.

Tämä opinnäytetyö keskittyy kuvitteellisen yrityksen kahden toimipisteen tietoliikenneratkaisuihin sekä toimipisteiden verkkojen yhdistämiseen. Työssä rakennettu verkko koostuu kahdesta osasta jotka voisivat kuvata mahdollista ratkaisua kahden rakennuksen tai jopa paikkakuntien välisten toimipisteiden yhteen liittämistä sekä verkon rakennetta kummassakin toimipisteessä.

Opinnäytetyön alkuosa koostuu tietoliikenneverkkojen hierarkkisiin malleihin sekä verkon eri osa-alueiden että laitteiden tehtäviin liittyvästä teoriasta. Teorian ensimmäinen osuus käsittelee OSI- sekä TCP/IP-mallien rakenteen sekä tärkeimmät protokollat. Lisäksi on selitetty millaisista laitteista verkko koostuu ja millaisia tehtäviä milläkin laitteella on.

Työn viimeinen osuus sisältää käytännön toteutuksen kuvitteelliselle verkolle Cisco PacketTracer ohjelman avustuksella sekä käyttäen Mikkelin Ammattikorkeakoulun Cisco-laitteistoa. Opinnäytetyö on tehty yhteistyössä Aki Timosen kanssa siten että Akille tässä verkkokokonaisuudessa kuuluu kytkimet sekä niiden konfigurointi, kun taas minulle kuuluvat reitittimet sekä niiden konfigurointi verkkojen yhdistämistä varten.

2 TIETOLIIKENNEVERKKOJEN RAKENNE

Tietoliikenneverkkojen rakenteen mallina ovat pääasiallisesti toimineet OSI- sekä TCP/IP-malli. Tässä kappaleessa selitetäänkin OSI- sekä TCP/IP-mallin osat sekä niiden tehtävät. Sekä käydään läpi myös mallien rakennetta ja rakenteen osien tehtävää verkon reitityksen osalta. TCP/IP-mallissa käydään läpi myös siihen kuuluvat protokollat sekä niiden tehtävät alueittain. Lisäksi käydään läpi lähiverkkotekniikan fyysistä laitteistoa sekä sen laitteiston sijaintia malleissa.

2.1 OSI-malli

OSI-malli (Open Systems Interconnection) on ISO:n (International Organization for Standardization), kansainvälisen standardointiorganisaation 80-luvun alussa kehittämä malli. Mallin perusajatus oli että siitä tulisi ratkaisu lähi- ja muiden verkkotekniikoiden yhteensopivuusongelmiin sekä sillä tultaisiin tulevaisuudessa yhdistämään kaikki maailman laitteet toisiinsa. Kuitenkaan malli ei saanut tavoiteltua levinneisyyttä, mutta sitä usein käytetään referenssinä helpottamaan verkkojen toiminnan ymmärtämistä. Useimmiten verkkojen rakentamisessa käytetään TCP/IP-mallia, johon tässä työssä tullaan enimmäkseen keskittymään. [1, s. 30-31]

OSI-malli koostuu seitsemästä kerroksesta, jotka on suunniteltu siten, että jokainen kerros pystytään myös toteuttamaan omana kokonaisuutenaan. Suunnitellun kerrosmallin etuina ovat ymmärtämisen, suunnittelun ja kehitystyön helppous. Sekä ylemmät että alemmat kerrokset kommunikoivat keskenään niiden välille rakennettujen rajapintojen avulla. Tämä mahdollistaa yksittäisen kerroksen kehittämisen ja päivittämisen ilman että se vaikuttaa muihin kerroksiin. [1, s. 31-32]

Kuva 1. OSI-mallin kerrokset [6]

OSI-mallin ylimpänä kerroksena toimii *sovelluskerros* (Application Layer), joka on suorassa vuorovaikutuksessa sovelluksen kanssa tarjoten sille verkkopalveluja, esimerkiksi verkkoyhteys sähköpostiohjelmalle. *Esitystapakerros* (Presentation Layer) kertoo missä muodossa data esitetään. [1, s. 32-33]

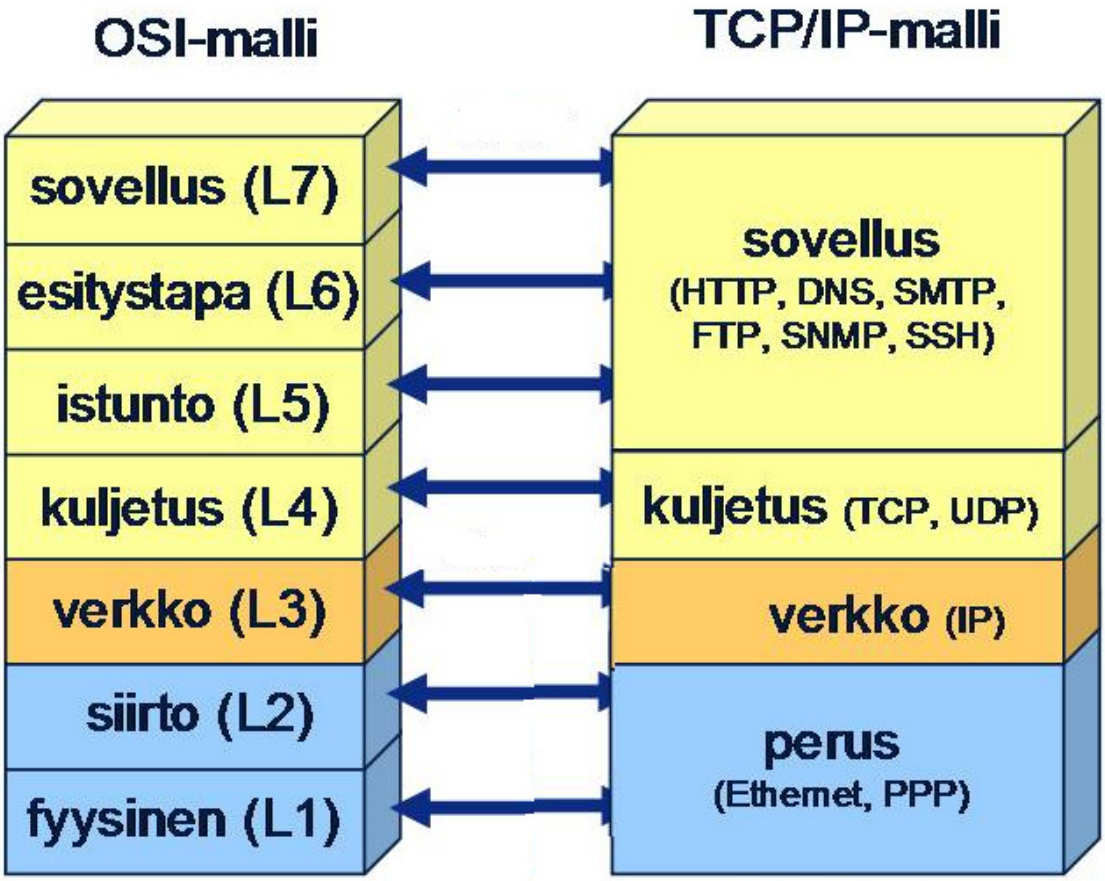
Istuntokerros tai *yhteysjaksokerros* (Session Layer) koordinoi sovellusten toimintoja eri laitteiden välillä, esimerkiksi lähetyksen käynnistäminen ja pysäyttäminen. Istuntokerros myös huolehtii datanvälitysjärjestyksestä. *Kuljetuskerros* (Transport Layer) huolehtii ylemmiltä kerroksilta saadun datan pilkkomisesta ja sen välittämisestä alempiin kerroksiin. Välitys voi tapahtua joko yhteydellisenä, jolloin yhteys muodostetaan kohteeseen ennen datan lähetystä, tai yhteydettömänä, jolloin yhteyttä kohteeseen ei muodosteta ennen datan lähettämistä vaan sen oletetaan menevän määränpäähän. Kuljetuskerroksen tunnetuimpia protokollia ovat TCP (Transmission Control Protocol) sekä UDP (User Datagram Protocol). [1, s. 33-34]

Verkkokerros (Network Layer) pakkaa kuljetuskerrokselta saadun datan paketteihin ja välittää datan verkkokerroksessa sijaitsevaan osoitteeseen. Tämä prosessi tunnetaan nimellä reititys, joka on riippumaton alempien kerrosten tekniikoista. Verkkokerroksella toimivia protokolla ovat esimerkiksi IP (Internet Protocol) sekä IPX (Internetwork Packet Exchange). *Siirtoyhteyskerros* (Data Link Layer) rakentaa kehyksen, jonka sisälle pakataan verkkokerrokselta saatu data. Siirtoyhteyskerros myös lisää

kehykseen otsikot, jotka sisältävät lähettäjän sekä vastaanottajan siirtoyhteyskerroksen osoitteen. Kehystystapoina kerroksessa toimivat esimerkiksi MAC (Media Access Control), LLC (Logical Link Control) sekä HDLC (High-Level Data Link Control). Siirtoyhteyskerros on riippuvainen fyysisestä kerroksesta. *Fyysinen kerros* (Physical Layer) on alin kerros OSI-mallissa, joka määrittelee kaikki fyysiset ominaisuudet sekä käytettävät koodausmenetelmät bittien välitykseen liittyen. [1, s. 34]

2.2 TCP/IP

Toinen yleisesti käytössä oleva kerrosmalli on TCP/IP. Tämä malli koostuu neljästä kerroksesta, joiden suhde OSI-mallin kerroksiin näkyy alla olevassa kuvassa.TCP/IP-mallin nimi tulee sen kahden tärkeimmän protokollan TCP:n ja IP:n lyhenteistä.
TCP/IP-mallin esitteli IETF (Internet Engineering TaskForce) vuonna 1982 ja sitä alettiin käyttää 1.1.1983. [1, s. 14]



Kuva 2. TCP/IP- ja OSI-mallin vertailu [7]

Sovelluskerros (Application Layer) on TCP/IP-mallin ylin kerros, joka sisältää OSI-mallin kolme ylintä kerrosta. Kerroksen tehtävänä on lähettää sekä vastaanottaa dataa jollakin kuljetuskerroksen protokollista. [2, s. 184]

Kuljetuskerros (Transport Layer) siirtää sovelluskerroksen dataa kahden eri sovelluksen välillä. Kuljetuskerroksella lähetettävä tietovirta jaetaan pieniksi paloiksi, joihin lisätään sekä vastaanottavan että lähettävän sovelluksenportin numero. Porttinumerolla määrätään mille sovellukselle paketti ohjautuu. Näiden toimenpiteiden jälkeen kuljetuskerros siirtää paketin seuraavalle kerrokselle. [2, s. 184][1, s. 136]

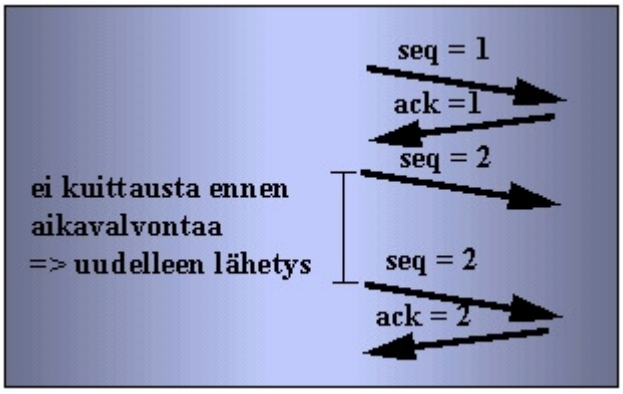
Verkkokerros (Internet Layer) hoitaa laitteiden välisen kommunikoinnin, joka on päästä-päähän (end-to-end) tyyppistä. Kuljetuskerroksesta saapuva paketin lähetyspyyntö vastaanotetaan verkko-kerroksessa, jossa pakettiin lisätään IP-otsikkotiedot. Tämän jälkeen verkko-kerros määrittää reititysalgoritmilla minne paketti lähetetään. Riippuen paketin määränpäästä se ohjataan joko reitittimelle tai suoraan verkossa olevalle päätelaitteelle. Jos paketin määränpää on paikallisessa verkossa, niin verkko-kerros poistaa paketista IP-otsikkotiedot ja lähettää sen kuljetuskerrokselle. [2, s. 185]

Peruskerros (Network Access Layer) on TCP/IP-mallin alin kerros, joka vastaanottaa IP-kehyksen ja siirtää sen oikeaan verkkoliitäntään sekä myös lähettää IP-kehyksen verkkoliitännältä Internet-kerrokselle. Verkkokerroksella tapahtuva tiedonsiirto on paikallista, eli se tapahtuu kahden laitteen välillä, toisin kuin Internet-kerroksella jossa tiedonsiirto tapahtuu päästä-päähän. Yksi tärkeimmistä verkkokerroksen protokollista on Ethernet. Verkkokerros käsittää myös kaikki fyysiset laitteet eli esimerkiksi reitittimet, kytkimet ja päätelaitteet. [2, s. 185]

2.2.1 TCP

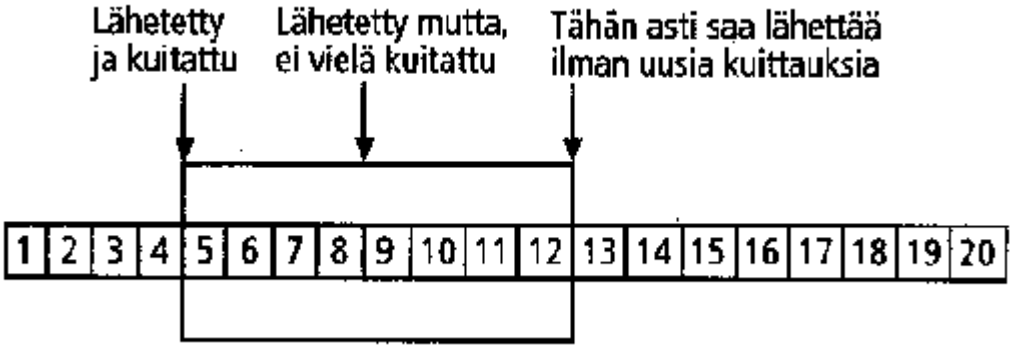
TCP (Transmission Control Protocol) on itsenäinen kuljetusprotokolla, jota on sovellettu luotettavuutensa vuoksi myös muihin kuljetusjärjestelmiin kuten ISO:n TP-4:ään (Transport Protocol Class 4). TCP on yhteydellinen protokolla jossa yhteys muodostetaan päätepisteiden välille ennen tiedonsiirron aloittamista. TCP:n oma luotettava paketinsiirtomenetelmä perustuu yksinkertaiseen kuittausmenetelmään, jossa lähettäjä odottaa paketin lähetyksen jälkeen kuittausta (*acknowledgement, ACK*) vastaanottajal-

ta ennen kuin lähettää uutta pakettia kohteeseen. Paketin lähetyksen lisäksi lähettäjä käynnistää ajastimen paketin lähdettyä ja lähettää saman paketin uudelleen jos aikakatkaisu tapahtuu ennen kuin kuittaus on saapunut vastaanottajalta. Pakettiin liitetään lähetyksessä myös numero, joka saapuu kuittauksen mukana takaisin, jotta pystytään eliminoimaan mahdollisten duplikaattien syntyminen, kun sekä lähettäjä että vastaanottaja tietävät mitä pakettia kuittaus koskee. [2, s. 209-212] [5]



Kuva 3. Yksinkertaista kuittausta käyttävä protokolla [9]

TCP-protokolla käyttää apunaan myös käsitettä nimeltään *liukuva ikkuna*. Toisin kuin yksinkertaista kuittausta käyttävässä protokollassa, niin liukuvassa ikkunassa voi lähettäjä lähettää useamman paketin peräkkäin kuitenkin tarvitsematta odottaa kuittauksia paketeista. Liukuvan ikkunan koon ollessa esimerkiksi 8760, joka on Windows NT –käyttöjärjestelmän oletuskoko Ethernet-verkossa, voidaan lähettää 6 Ethernet kehystä ennen kuin se tarvitsee kuittauksen vastaanottajalta. Tämän mukaisesti ikkuna siirtyy liukuvasti paketti kerrallaan. Liukuvan ikkunan tehokkuuteen vaikuttaa kuitenkin koon lisäksi verkon nopeus eli kuinka nopeasti verkko pystyy paketteja vastaanottamaan. Jokaisella TCP yhteydellä on olemassa neljä kappaletta ikkunoita. Molemmilla osapuolilla on lähetys- sekä vastaanottoikkuna. Näiden kokoa voidaan säätää erikseen siten että toisen laitteen lähetysikkunan koko on sama kuin toisen laitteen vastaanottavan ikkunan koko.[2, s. 213-214] [1, s. 149-150]

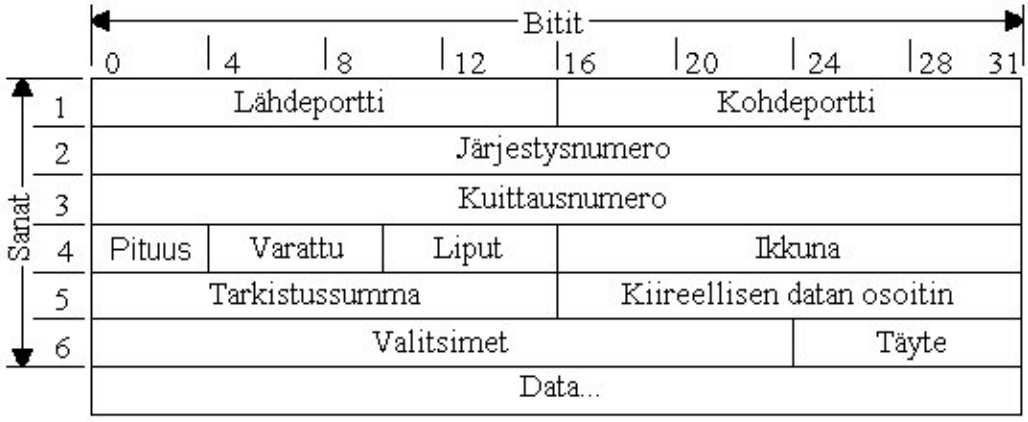


Kuva 4. Liukuva ikkuna. [1, s. 149]

Liukuva ikkuna myös muistaa paketit ja niiden saapuvat kuittaukset. Kuittauksia varten tämä menetelmä pitääkin jokaisella erillisellä paketilla omaa ajastinta aikakatkaisua varten. Jos aikakatkaisu tapahtuu ja paketti lähetetään uudelleen, niin ikkuna siirtyy kaikkien kuitattujen pakettien ohi ja myös vastaanottavassa päässä on samanlainen ikkuna, joka pitää yllä pakettilistaa ja lähettää kuittauksia lähettäjän suuntaan. Eli näin ollen ikkunassa on kolme osaa:

- Ikkunan vasen puoli: Vastaanotetut paketit, jotka on lähetetty, vastaanotettu ja kuitattu.
- Ikkuna: Paketit, joita ollaan siirtämässä.
- Ikkunan oikea puoli: Paketit, joita ei ole vielä siirretty. [2, s. 214-215]

TCP-paketin eli segmentin rakenne



Kuva 5. TCP-segmentti [8.]

TCP-segmentti sisältää kaksi pääosaa eli otsikon ja data-osion. Näistä pääosista otsikko sisältää seuraavat kentät: *Lähde- ja kohdeportti*–kentät, jotka sisältävät sovellusohjelmien porttiosoitteet. *Järjestysnumero*-kenttä, joka sisältää paketin järjestysnumeron datavirrassa. *Kuittausnumero*-kenttä sisältää kuittausnumeron, jota lähettäjä odottaa vastaanottajalta. *Pituus*-kenttä sisältää segmentin otsikon pituuden kertovan luvun sekä tämä kenttä myös vaihtelee *Valitsimet*-kentän pituuden mukaan. *Varattu*-kenttä on sisältää 6-bittiä tulevaa varattua käyttöä varten. 6-bittisen *Liput* eli *koodibitit*-kentän mukaan määritellään segmentin tarkoitus ja sisältö. Nämä bitit määrittävät kuinka otsikko-osan muut kentät tulkitaan. Bitit sekä niiden merkitykset on selitetty taulukossa 1. [2, s. 221-222]

Taulukko 1. TCP-otsikonLiput- eli koodibitit-kentän bitit.

Bitti (vasemmalta oikealle)	**Bitin merkitys, jos bitti on 1**
URG	Kiireellinen datan osoitin-kenttä käytössä
ACK	Kuittausnumerokenttä käytössä
PSH	Datan välittäminen sovellukselle ilman puskurointia
RST	Yhteyden nollaus
SYN	Järjestysnumeroiden tahdistaminen
FIN	Lähettäjä on tullut tavuvirran loppuun

Ikkuna-kentän arvo kertoo kuinka paljon dataa lähettäjä saa lähettää edellisen kuittausnumeron jälkeen. *Kiireellinen datan osoitin*-kenttä ilmaisee jos URG-bitti on asetettu *liput*-kenttään ja myös määrittää näin kiireellisen datan sekä tämän kiireellisen datan sijainnin segmentissä. *Tarkistussumma*-kenttä on 16-bittinen kenttä, jossa sijaitsee kokonaisluku jonka avulla voidaan tarkistaa segmentin eheys. [2, s. 222-224]

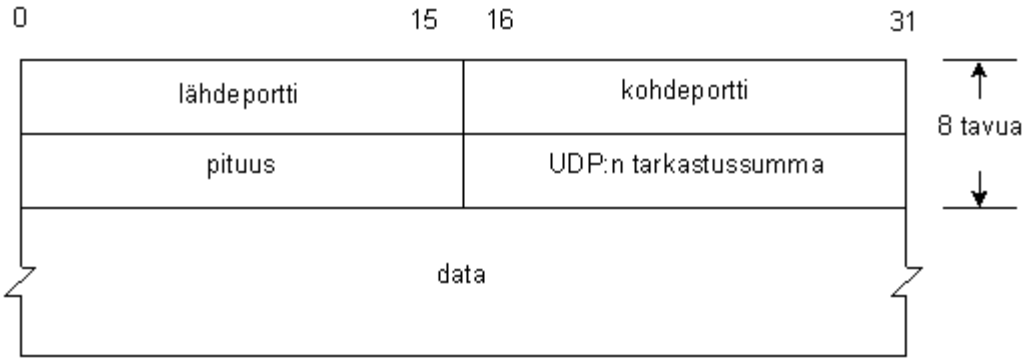
2.2.2 UDP

UDP (User Datagram Protocol) on TCP:n rinnalla toimiva kuljetuskerroksen protokolla. UDP:llä ja TCP:llä on sama päätehtävä, datan kuljettaminen. Tästä huolimatta protokollat ovat hyvin erilaisia toisiinsa verrattuna. TCP:n ominaisuuksiin kuuluu muun muassa yhteydellisyys, luotettavuus, virheenkorjaus ja kuittaus, mutta kaikki edellämainitut ominaisuudet puuttuvat UDP:sta täysin. Multipleksaus on ainut ominaisuus datan välityksen lisäksi johon UDP pystyy. Tämä tarkoittaa sitä että yhdellä fyysisellä

yhteydellä voi olla monta samanaikaista sessiota eri sovellusten kesken. Edellä mainittujen puutteiden vuoksi UDP:tä käytetäänkin sellaisissa tilanteissa joissa ei ole niin kriittistä jos osa tiedosta häviää matkan varrella. Yleisimpänä UDP:n käyttökohteena on nykyään DNS (Domain Name System). Suurinpana etuna UDP:n yksinkertaisuudessa ja ominaisuuksien puutteissa on otsikkokenttien ja muun ylimääräisen datan pieni määrä suhteessa siirrettyyn dataan. Tämän takia lähiaikoina UDP:tä on myös alettu käyttää multimediasovellusten kanssa, koska sen kevyt rakenne mahdollistaa korkean datansiirtokapasiteetin. Lisäksi multimediakäytössä ei ole niin kriittistä jos esimerkiksi äänessä tapahtuu muutaman millisekunnin katkos, sekä esimerkiksi TCP:n käytöstä ei olisi hyötyä koska uudelleenlähetys aiheuttaisi liian suuren viiveen tehden saapuneesta datasta käyttökelvotonta.[2, s. 198-199] [1, s. 167-168]

UDP-sanoman rakenne

UDP-sanoma sisältää kaksi osaa eli otsikon ja data-alueen. Otsikossa on neljä kenttää: lähdeportti, kohdeportti, sanoman pituus ja tarkistussumma. Jokainen näistä otsikko-osan kentistä on 16-bittisiä. [2, s. 199]



Kuva 6.UDP-Sanoman rakenne[10]

Lähde- ja kohdeportti ,kentät sisältävät sovelluksen prosessia koskevan porttinumeron, jonka avulla protokolla pystyy määrittämään, mikä sanoma millekin sovellukselle lähetetään sekä mistä sovelluksesta sanoma on lähetetty.

Sanoman pituus-kentässä oleva arvo ilmoittaa kuinka monta tavua tai oktettia sanoma sisältää kaiken kaikkiaan eli otsikko ja data-alue yhteensä, ja koska otsikon pienin arvo on kahdeksan, niin teoriassa se on myös pienin mahdollinen sanoman pituuden arvo, joka tarkoittaisi sitä että data kentän pituus olisi nolla tavua, ja sanoma sisältäisi

ainoastaan otsikkotiedot. Suurin mahdollinen arvo sanoman pituudelle on 65535 tavua. *Tarkistussumma*-kentän arvolla tarkistetaan onko tietojen siirto tapahtunut virheettömästi ja onko sanoma saapunut oikeaan kohteeseen. Jos tarkistuksessa havaitaan virhe, sanoma hylätään. Tarkistussumma on sanomassa suunniteltu valinnaiseksi, jotta käsittelyssä kuluva aika saataisiin minimoitua ja näin nopeutettua sanomien välitystä. [2, s. 199-200] [1, s. 169-170]

2.2.3 IP

Internet Protocol eli IPon yksi tärkeimmistä verkoissa käytetyistä protokollista. IP käsittää kolme verkoille tärkeää asiaa. Tärkeimpänä niistä on IP-osoite jonka avulla paketit välitetään oikeaan määränpäähän. Toiseksi IP-protokolla suorittaa reitityksen, eli valitsee tiedolle reitin jota pitkin tieto välitetään. Ja kolmanneksi IP-protokolla sisältää säännöt joiden perusteella tiedon kuljetus tapahtuu ja kuinka eri laitteet tietoa käsittelevät. [2, s. 97-98]

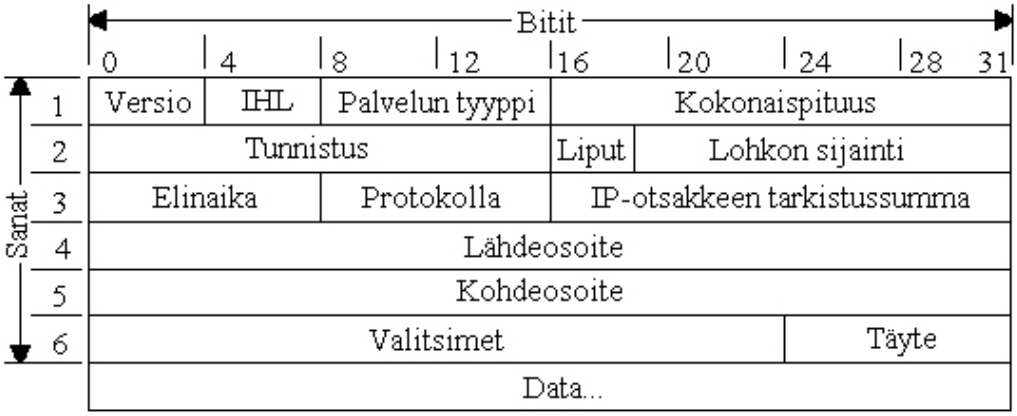
IP-protokollasta on tällä hetkellä käytössä kaksi versiota, IPv4 sekä IPv6. Vielä toistaiseksi lähes kaikkialla käytetään IPv4 versiota mutta IPv6:n osuus kasvaa jatkuvasti koska IPv4 osoitteita ei ole riittävästi, joten siirtyminen IPv6 maailmaan on välttämätöntä.

IPv4 on protokolla jonka tehtävä on välittää paketteja yhteen kytkettyjen verkkojen eri osien välillä. IPv4 on yhteydetön protokolla, joka tarkoittaa sitä, että se ei pidä kirjaa olemassa olevista yhteyksistä, vaan se hoidetaan ylemmillä kerroksilla joko TCP:llä tai sovelluskerroksen protokollilla. IPv4:stä myös puuttuu täysin mahdollisuus hallita liikenteen määrää verkossa suorittamalla vuonohjausta sekä suorittaa virheenkorjausta. Tämä johtuu pääasiassa suorituskyvyllisistä seikoista, koska on turhaa suorittaa samoja tehtäviä monen eri protokollan toimesta. [3, s. 113-114]

IP-liikenne on pakettiliikennettä, ja IP-protokollan tehtäviin kuuluu pakettien käsittely. Näihin kuuluvat muun muassa pakettien osioiminen tarpeen vaatiessa, liikenteen reititys IP-osoitteen perusteella sekä paketin koon määrittäminen. Liikenteen pakettiluontoisuus tarkoittaa sitä että kaikki paketit välitetään erikseen, riippumatta siitä mitä

edelliselle paketille on tapahtunut. Kuten myös vastauspaketit välitetään riippumatta siitä mitä kautta alkuperäinen paketti on saapunut. [3, s. 114-115]

IP-Sanoman rakenne



Kuva 7. IP-Sanoman rakenne[6]

Kuvassa 2 on mallinnettu IPv4 paketin otsikkokenttien rakenne. Paketti sisältää maksimissaan 14 kenttää joista 12 on vakiomittaista ja kaksi määrittelemättömän kokoista kenttää; Valitsimet ja Täyte. *Versio* on 4 bittiä ja se kertoo käytettävän IP-protokollan versionumeron, eli onko kyseessä IPv4 vai IPv6. *IHL (Internet Header Length)* eli otsikon pituus on myös 4 bittiä ja se ilmaisee IP-paketin otsikkokentän pituuden kertomalla kuinka monta 32 bitin jaksoa otsikko sisältää. *Palvelun tyyppi* on 8 bittiä ja sen avulla ylemmän tason protokollat viestivät verkon laitteille minkä tyyppistä palvelua he haluavat verkolta. Tämä tapahtuu lisäämällä pakettiin TOS (Type Of Service) parametrit, joilla voidaan määrittää esimerkiksi että halutaan tieto siirrettäväksi mahdollisimman pienellä viiveellä. *Kokonaispituus* on 16 bittiä ja se kertoo IP-paketin koon oktetteina eli kahdeksan bitin ryhminä. Kentän maksimiarvo, joka on myös maksimikoko IP-paketille, on 2^{16}eli 65535 oktettia. Useimmiten kuitenkin käytetään pienempää arvoa, ja IPv4 standardissa määritelläänkin että laitteiden täytyy kyetä vastaanottamaan ainoastaan 576 oktetin pituisia paketteja.[1, s. 115-117, 123-124] [2, s. 99]

Tunnistus on 16 bitin mittainen ja se toimii paketin ID-numerona. Sitä tarvitaan esimerkiksi pakettien osioinnissa, jotta voidaan tunnistaa mitkä osiot kuuluvat mihinkin pakettiin. *Liput* eli osioimisen määrittely on 3 bittiä ja sillä määrätään voiko paketin

osioida, sekä ilmoitetaan mahdollisista tulevista osioista. *Lohkon sijainti* on 13 bitin mittainen ja ilmoittaa nimensä mukaisesti lohkojen tai osioiden sijainnin paketissa. *Elinaika* on 8 bitin suuruinen ja määrää paketille arvon joka vähenee yhdellä jokaisessa matkan varrella olevassa reitittimessä. Tämä estää pakettien loputtoman kiertämisen ympäri verkkoa koska kun elinaika menee nollaan, tuhoaa reititin paketin ja lähettää siitä ilmoituksen paketin lähettäneelle koneelle. Elinaika yleensä ilmoitetaan lyhenteellä TTL (Time To Live) ja voi olla arvoltaan maksimissaan 255. [1, s. 117-118]

Protokolla on 8 bittiä ja se ilmaisee paketin ylemmän kerroksen protokollan numeron, esimerkiksi numero kuusi ilmaisee että käytettävä protokolla on TCP. *IP-otsikon tarkistussumma* on 16 bitin mittainen ja sillä varmistetaan otsikkokenttien sisältämien tietojen oikeellisuus. Tarkistussumma koskee ainoastaan IP-otsikon sisältäviä tietoja eikä siirrettävää dataa.Jos tarkistussumma ei täsmää lähetettävän datan kanssa paketti hylätään. *Lähdeosoite* on 32 bittiä ja sillä ilmoitetaan lähettäjän IP-osoite. *Kohdeosoite* on myöskin 32 bittiä ja se kertoo vastaanottavan laitteen IP-osoitteen. *Valitsimet* ja *Täyte* kenttiin voidaan määritellä esimerkiksi turvallisuuteen liittyviä lisäasetuksia. Kenttien yhteenlaskettumaksimikoko on määritelty standardissa 40 oktettiin eli 320 bittiin. Kenttien yhteispituus täytyy myös olla aina jaollinen 32:lla, joten *Täyte* kentän avulla lisätään tarvittava määrä nolla-bittejä, jotta lopullinen pituus on jaollinen 32:lla.[1, s. 119-121] [4, s.2]

DHCP -protokolla

DHCP (Dynamic Host Configuration Protocol) protokollan avulla voidaan määrittää laitteille IP-osoitteet automaattisesti. Kun tietokone liitetään verkkoon, verkossa toimiva DCHP-palvelin antaa tietokoneelle IP-osoitteen, aliverkon peitteen (subnetmask), oletusyhdyskäytävän (defaultgateway) sekä nimipalvelimen eli DNS-palvelimen (Domain Name System) osoitteet. [2, s. 450-451]

Aliverkon peite tai aliverkkomaski määrittelee IP-osoitteesta mitkä bitit kuuluvat verkkokenttään ja mitkä isäntäkenttään. Esimerkiksi aliverkon peite 255.255.255.0 tarkoittaa sitä että 3 ensimmäistä oktettia kuuluvat verkkokenttään ja viimeinen oktetti kuuluu isäntäkenttään. Aliverkon peite 255.255.255.0 voidaan myös ilmaista merkinnällä /24 koska se käsittää 24 ensimmäistä bittiä. DNS-palvelin taas vastaa internet

osoitteiden kääntämisestä bittimuotoon. Kun halutaan saada selville esimerkiksi osoitteen http://www.mamk.fi ip osoite, otetaan yhteys DNS-palvelimeen. [15, s. 401-403] [3, s. 226-227]

2.2.4 IPv6

Kuten jo aikaisemmin mainittuna, niin IPv6-verkkoon siirtyminen on välttämätöntä IPv4-osoitteiden loppumisen vuoksi. Tästä syystä IPv6 sekä IPv4 verkoilla onkin muutamia tärkeitä eroja keskenään. Eli toisin kuin IPv4-osoitteessa niin IPv6-osoitteessa on osoitteen pituus 128 bittiä sekä siinä yhdistyvät looginen että fyysinen osoite. Näin ollen jos IPv4:n 32 bittisillä osoitteilla maksimimäärä on vähän yli neljä miljardia osoitetta, niin IPv6-osoitteita voi olla virallisen laskelman mukaan 8×10^{17} – 2×10^{33} laitteella jokaista neliömetriä kohden maapallolla. [3, s. 215-216]

IPv6-osoitteen muoto eroaa myös merkittävästi jo tutuksi tulleesta IPv4-osoitteesta. IPv6-osoitetta ei enää esitetä desimaalilukuina vaan osoitejärjestelmässä on otettu käyttöön heksadesimaalimuoto ja erotinmerkkinä IPV4-osoitejärjestelmän pisteen sijaan IPv6-osoitejärjestelmässä käytetään kaksoispistettä eli esimerkiksi 68E6:8C64:FFFF:FFFF:0:1180:96A:FFFF on IPv6-osoite. On myös tehty tiettyjä sääntöjä IPv6-osoitteen lukemisen helpottamiseksi eli jos esimerkiksi IPv6-osoite on muotoa FF05:0:0:0:0:0:0:B3, niin tästä osoitteesta voidaan suoraan korvata nollat kaksoispisteellä muotoon FF05::B3. [3, s. 217]

<table>
<tr><td>Versio
(4 bit)</td><td>Liikenneluokka
(8 bit)</td><td colspan="3">Vuo (20 bit)</td></tr>
<tr><td colspan="3">Datan pituus (16 bit)</td><td>Seuraava ots. (8 bit)</td><td>Etappiraja (8 bit)</td></tr>
<tr><td colspan="5">Lähdeosoite (128 bit)</td></tr>
<tr><td colspan="5">Kohdeosoite (128 bit)</td></tr>
</table>

Taulukko 2. IPv6-otsikon rakenne.

Sekä myös kuten yllä olevasta kuvasta huomataan, niin otsikon rakenne eroaa olennaisesti IPv4-otsikon rakenteesta. IPv6:n rakenne ei sisälläkään IPv4-rakenteen mukaisia kenttiä vaan nämä kentät ovat IPv6:n rakenteessa sijoitettu *laajennusotsikoihin*. Laajennusotsikot toimivatkin tässä tapauksessa IPv4-rakenteen täyte-kentän mukaisesti eli lähettäjä valitsee mitä laajennusotsikoita käytetään.

Itse IPv6-otsikon rakenne sisältää kuvan mukaiset kentät. *Versio*-kenttä, joka määrittelee protokollan version. *Liikenneluokka*-kenttä, joka toimii IPv4-rakenteen palvelu tyyppi-kentän mukaisesti eli määrittelee mitä palvelua paketti haluaa verkolta. *Vuo*-kenttä, joka sisältää tiedot joiden avulla reititys sovellusten välillä tapahtuu eli toisinsanottuna polku sovellusten välillä, jossa on määritelty tapahtuvat viiveet sekä kaistanleveys. *Datan pituus*-kenttä, joka määrittelee paljonko dataa otsikon lisäksi paketissa on. *Seuraava otsikko*-kentän tietoja tarvitaan reitittimissä ja kohteessa toimivissa ohjelmissa, koska tämä kenttä kertoo seuraavan otsikon tyypin. *Etappiraja*-kenttä sisältää arvon, joka vastaa etappien määrää minkä paketti kulkee. Näiden kenttien lisäksi rakenne sisältää myös *lähde-* ja *kohdeosoite*–kentät, jotka nimensä mukaisesti määrittävät lähettäjän ja vastaanottajan osoitteet. [2, s. 603-606]

2.3 Ethernet

Ethernet on ylivoimaisesti suosituin lähiverkoissa käytetty tekniikka. Yleisesti Ethernetin keksimisestä kunnia annetaan Xeroxille joka kehitti sen 1970-luvun alussa, idea kuitenkin perustui ALOHA-nimiseen hankkeeseen joka kehitettiin Havaijin yliopistolla 1960-luvun lopulla. Vuonna 1980 Ethernetistä julkaistiin 1.0 versio joka syntyi Digitalin, Intelin sekä Xeroxin yhteistyön tuloksena. Alun perin tekniikan nopeus oli 20 megabittiä sekunnissa, mutta se laskettiin myöhemmin 10 megabittiin sekunnissa. Tämä 10 megabitin Ethernet, joka tunnetaan myös nimellä 10Base5, oli tekniikan ensimmäinen laajasti levinnyt versio. IEEE standardoi 10Base5 Ethernetin vuonna 1983 standardi numerolla 802.3.Ethernet on sittemmin yli 30 vuoden aikana kehittynyt uusiin ja nopeampiin versioihin jotka yltävät jopa 100 gigabitin nopeuksiin uuden IEEE 802.3ab standardin myötä. Nykyiset uudet Ethernet standardit ovatkin jo teknologialtaan täysin uudenlaisia ja niissä on vanhoihin enää yhteistä ainoastaan nimi sekä kehysrakenne. [2, s. 20] [1, s. 48-49] [12]

Ethernetin kehysrakenne

Kuva 8. Ethernet-kehyksen rakenne. [13]

Ethernet-kehys sisältää 6 pääosaa: Tahdistusosa, kohdeosoite, lähdeosoite, tyyppi, data sekä tarkistussumman eli CRC (Cyclic Redundancy Check). Tahdistusosan avulla verkkokortti tunnistaa kehyksen alun ja se on pituudeltaan 64 bittiä muodostuen vuorottelevista 1- ja 0-biteistä. Kohde- ja lähdeosoite kentät ovat molemmat 48 bittisiä kooltaan ja sisältävät fyysisen MAC-osoitteen. Tyyppikenttä on kooltaan 16 bittiä ja se sisältää kokonaisluvun, jonka avulla saadaan selville minkä ylemmän tason protokollalle kehyksen data ohjataan eli esimerkiksi jos kentässä on heksadesimaaliluku arvoltaan alle 0600, niin kyseessä on IEEE 802.3 mukainen Ethernet-kehys. Datakenttä sisältää varsinaisen datan ja voi olla kooltaan minimissään 368 bittiä ja enimmiltään 12000 bittiä. Viimeisenä kehyksessä on CRC-kenttä eli tarkistussumma. Tarkistussumma on kooltaan 32 bittiä ja sen avulla voidaan havaita mahdolliset siirtovirheet. Tarkistussumma-kentän luvun laskee lähettäjä kehyksen tietojen mukaan ja vastaanottaja laskee sen taas tarkistaakseen, että tiedot täsmäävät. [2, s. 30-31][3, s. 144-145]

Ethernetin kehitys sekä ominaisuudet

Alun perin Ethernet toimi 10 megabitin nopeudella 10Base5 versiossa. Tästä Ethernet kehittyi muutaman välivaiheen kautta versioon 10BaseT, jossa nopeus säilyi samassa 10 megabitissä mutta kaapeleina käytettiin ensimmäistä kertaa parikaapelia. Kuitenkin Ethernetin 10 megabitin nopeus alkoi muodostua 1990-luvulla monessa verkossa pullonkaulaksi, joten kehitettiin 10 kertaa nopeampi 100 megabitin 100BaseT Ethernet, joka usein tunnetaan nimellä FastEthernet. Ja tästä seuraava versio 1000BaseT Ethernet taas kerran nosti nopeuden kymmenkertaiseksi jatkuvasti kasvavien verkkojen vaatimusten mukaisesti. 1000BaseT eli GigabitEthernet poikkeaa 10BaseT ja 100BaseT standardeista luonnollisesti nopeamman tiedonsiirtonopeutensa takia, mutta

myös siten että vanhoissa standardeissa käytetty kaapelointi ei käy 1000BaseT verkkoon. [2, s. 26-27]

Ethernet-tekniikassa käytetään CSMA/CD (Carrier Sense Multiple Access / Collision Detection) nimistä tekniikkaa estämään mahdollisia törmäyksiä tapahtumasta tiedonsiirrossa pisteestä toiseen. Käytännössä tämä tarkoittaa sitä että laitteen pitäisi lähettää vasta, kun laite huomaa kaapelin olevan vapaa eli jos joku muu laite ei lähetä samassa segmentissä. Kuitenkin jos kaapeli on vapaa ja kaksi laitetta aloittaa lähetyksen täsmälleen samaan aikaan, niin siinä tapauksessa tarvitaan ns. törmäyksen havainnointia. Törmäyksen havainnointi perustuu siihen että jos kaksi laitetta lähettää samaan aikaan ja törmäys tapahtuu jossain kohtaa, niin molemmissa laitteissa havaitaan törmäyksen aiheuttama jännitetasojen muuttuminen. Tällöin laitteet lähettävät verkkoon ns. sotkua (*jam*), jolla segmentin laitteet saavat selville että törmäys on tapahtunut. Tämän jälkeen laitteet lopettavat lähetyksen ja aloittavat uudelleenlähetyslaskurin, joka laskee satunnaisen ajan mukaan milloin voi lähettää. Kun tämä aika on kulunut loppuun niin laite lähettää uudelleen jos kaapeli on vapaa. [1, s. 50-51] [2, s. 28]

2.4 Lähiverkkotekniikan laitteisto

Lähiverkkotekniikassa käytetään tiedonsiirtoon monia erilaisia laitteistoja. Näistä laitteistoista tärkeimmät sijaitsevatkin mallien alimmaisissa kerroksissa eli fyysisessä, siirtoyhteys- sekä verkkokerroksessa.

2.4.1 Fyysisen kerroksen laitteet

Fyysisen kerroksen laitteiden tehtävänä on ainoastaan signaalien siirto paikasta toiseen. Tähän tiedonsiirtoon tarvitaan ensinnäkin kaapeleita tai antenneja jos kyseessä on langaton tiedonsiirto, joita pitkin tieto välittyy eri laitteistoista toiseen. Kuitenkin jokaisella signaalilla mikä liikkuu joko kaapelissa tai langattomasti on maksimimatka minkä se kulkee ja jos tämä maksimimatka ylitetään, signaalissa voi tapahtua liikaa vaimennusta ja näin ollen vastaanottajana oleva laite ei pysty tulkitsemaan saapuvaa dataa. Vaimennuksen takia lähiverkkosegmenttien pituus onkin rajoitettu Ethernet-standardien mukaisesti esimerkiksi eri kaapelityypeittäin. Tähän lähiverkkosegmenttien pituus ongelmaan on kehitetty toistin eli laite, joka toistaa kaapelia pitkin tai lan-

gattomasti saapuneen datan sisääntulosta ja lähettää sen eteenpäin lähtöportista. On myös olemassa moniporttitoistin eli hubi. Hubin tehtävä on samanlainen kuin normaalillakin toistimella eli ottaa dataa vastaan yhteen porttiin ja lähettää data edelleen muihin liitäntöihin. Ongelma toistimissa on se että paketteja ei tarkisteta millään tavalla, joten virheellisiä paketteja ei hylätä vaan ne lähetetään eteenpäin. Kaikki toistimeen liitetyt laitteet jakavat saman kapasiteetin, joten tarvitaan esimerkiksi Ethernetissä käytössä oleva CSMA/CD (Carrier Sense Multiple Access / Collision Detection) joka tarkkailee onko kaapeli vapaa lähetystä varten. [1, s. 38-41]

Samassa OSI-kerroksessa toistimen kanssa on myös verkkokortti, jonka tehtävänä on liittää laitteesta välittyvä dataliikenne verkkoon. Laitteiden kuitenkin tulee tunnistaa tämä liikenne omakseen ja tästä syystä jokaisella verkkokortilla onkin oma MAC-osoitteensa (Media Access Control). Verkkokortin MAC-osoitteet ovat yksilöllisiä 48-bittisiä osoitteita, joista ensimmäiset 24-bittiä ovat valmistajan tunnus ja seuraavat 24-bittiä valmistajan määrittelemiä yksilöllisiä bittejä. [1, s. 41-42]

2.4.2 Siirtoyhteyskerroksen laitteet

Toistimilla kun ei liikennettä pystynyt juuri hallitsemaan, niin tarvittiin kehittyneempi laite, jolla tämä tarvittava ominaisuus pystyttiin toteuttamaan. Tämä laite on nimeltään kytkin, joka toimii siirtoyhteyskerroksessa. Liikenteen hallinnan lisäksi tällä laitteella pystytään liittämään laitteita toisiinsa. Kytkin tutkii jokaisen sille saapuneen kehyksen erikseen ja tämän kehyksen avulla se määrittelee, että onko kehys oikea vai virheellinen sekä kannattaako sitä välittää eteenpäin. Kytkimessä jokainen portti on itsenäinen, joten kytkimeen liitetyt laitteet saavat verkon täyden kapasiteetin käyttöönsä. Kytkimissä käytetään monia erilaisia välitystapoja datansiirtoon esimerkiksi *Suora välitys* (Cut-Through), jossa kehys vain välitetään nimensä mukaisesti suoraan eteenpäin ja *Talleta-ja-välitä* (Store-and-Forward), jossa kehys otetaan vastaan ja tarkistuksen jälkeen lähetetään eteenpäin. [1, s. 43-47]

Kytkimen toimintatapana on kuunnella saapuvaa liikennettä sekä minkä portin takana on mitäkin MAC-osoitteita. Kytkin kerää näistä osoitteista taulukon, jonka perusteella se tekee päätöksen mihin osoitteeseen mitäkin dataa siirretään. [1, s. 44-45]

2.4.3 Verkkokerroksen laitteet

Siirtoyhteyskerrosten laitteet keskustelevatkin seuraavaksi hierarkiassa ylemmän laitteen kanssa ja tätä laitetta kutsutaan reitittimeksi. Reititin toimii verkkokerroksessa ja sen pääasiallinen tehtävä on datan reitittäminen sekä datan suodatus. Datan reitittäminen on ehkä olennaisin osa koko Internetin toiminnasta. Reititin reitittää dataa verkkokerroksen osoitteen avulla, mutta kuitenkaan kaikilla protokollilla ei ole olemassa varsinaista osoitetta ja näitä protokollia kutsutaan ei-reititettäviksi protokolliksi eli esimerkiksi IP/IPX ovat reititettäviä, mutta NetBEUI eli NetBIOS (Network Basic Input/Output System) ei ole. Reititettävistä osoitteista löytyykin kaksi osaa, jonka avulla data reititetään. Nämä kaksi osaa ovat verkko- ja laiteosa, joista verkko-osalla välitetään data oikeaan verkkoon ja laiteosalla välitetään data lopulta oikealle laitteelle. Reititys perustuu reititystauluihin, jotka määrittävät kunkin reitittimen portin (interface) sekä verkkojen IP-osoitteet joihin portista pääsee. Yksi porteista toimii oletusyhdyskäytävänä (defaultgateway) johon data ohjataan jos reititystaulusta ei löydy kyseessä olevaa kohdetta. Reititystaulut voidaan määrittää joko manuaalisesti lisäämällä kaikki reitittimeen kytketyt portit ja aliverkot, tai dynaamisesti reititysprotokollan avulla. [1, s. 47-48] [3, s.200-202]

3 REITITTIMET

Reititin on olennainen osa tietoliikenneverkkoa, jonka päätehtävänä on yhdistää erilaiset verkot toisiinsa. Reititin toimii verkkokerroksessa eli OSI- sekä TCP-mallin mukaisesti tasolla kolme. Reitittimeen voidaan konfiguroida monia erilaisia reititysprotokollia, jotka reititystaulun avulla valitsevat parhaan reitin ja sen jälkeen ohjaavat paketin oikeaan verkkoon oikeasta liitännästä.

3.1 Staattiset ja dynaamiset reitit

Reitittimeen voidaan määritellä sekä staattisia että dynaamisia reittejä. Näiden avulla reititin ohjaa määrättyjä paketteja verkosta toiseen. Staattisien reittien tiedot pitää manuaalisesti ylläpitää eli ne syötetään käsin reitittimeen aina kun verkkoympäristössä tapahtuu jotain päivityksiä tai muutoksia. Dynaaminen reititys taas konfiguroidaan siten että määritellään tarvittavat komennot sekä reititysprotokolla. Tämä protokolla

päivittää reititystiedot aina itse, kun verkossa tapahtuu muutoksia. Reitittimet siis vaihtavat keskenään verkossa tapahtuvia muutostietoja, joiden avulla tarvittavat muutokset tapahtuvat automaattisesti reitittimille. [13, s. 718]

3.2 Reititystaulu

Kaikki laitteet lähettävät IP-paketteja verkossa ja tämän vuoksi niiden pitää tietää mitä kautta sekä mihin paketit pitäisi lähettää. Sen takia esimerkiksi reitittimestä löytyy reititystaulu. Reititystaulu voidaan muodostaa joko käsin käyttäen staattisia reittejä tai antaa reitittimien päivittää sitä eli käyttäen dynaamista reititystä. Reititystaulu sisältää tavallisesti kohdeverkon osoitteen sekä maskin, mahdollisen seuraavan hypyn eli seuraavan reitittimen IP-osoite, liitäntäportin, käytettävän protokollan sekä myös painoarvon. Painoarvo määräytyy protokollan mukaan eli esimerkiksi staattisella reitillä on pienempi painoarvo kuin esimerkiksi dynaamisella protokollalla. Näin ollen ensisijaisesti käytetään staattisesti määriteltyä reittiä kohteesta toiseen. Voidaan myös määritellä siten että jos staattinen reititys jostain syystä lopettaa toimintansa, niin käytetään esimerkiksi painoarvoltaan suurempaa reititysprotokollaa, jonka avulla saadaan vararreitti kohteeseen. [1, s. 269-275]

3.3 Reititysprotokollat

Reititysprotokollat kuuluvat jakautuvat yleisesti kolmeen protokollaperheeseen. Nämä protokollaperheet ovat etäisyysvektoriprotokollat, yhteystilaprotokollat sekä hybridireititysprotokollat.

3.3.1 Etäisyysvektoriprotokollat

Etäisyysvektoriprotokolliin (*Distance Vector Routing Protocol, IGP*) kuuluvat seuraavat protokollat eli RIP (*Routing Information Protocol*), RIPv2 sekä IGRP (*Interior Gateway Routing Protocol*). Näiden protokollien toiminta perustuu naapurireitittimien luottamiseen. Alussa reitittimet mainostavat naapurireitittimille tuntemiaan verkkoja etäisyysvektoreilla eli kertovat mahdollisen etäisyyden kullekin verkolle. Vastaanottava reititin lisää tauluunsa verkon etäisyyden sekä etäisyyden reitittimien välillä sekä myös tutkii löytyykö verkko reititystaulusta jo valmiiksi. Jos tämä etäisyys on suu-

rempi, niin ei tehdä mitään reititystaululle. Jos kuitenkin etäisyys on pienempi, niin päivitetään reititystaulu osoittamaan seuraavaa hyppyä tarjoavaan reitittimeen sekä myös päivitetään liitäntä mitä kautta etäisyysvektori saapui. Sekä myös jos itse reittiä ei aikaisemmin ole reititystaulussa, niin se päivitetään sinne. Kuitenkin jos jo reititystaulussa olevaan reitittimeen ei saada yhteyttä tietyn ajan jälkeen, niin tämä reitti määritellään saavuttamattomaksi ja poistetaan tietyn ajan kuluttua myös reititystaulusta. [1, s. 291-293]

3.3.2 Yhteystilaprotokollat

Yhteystilaprotokolliin (*Link State Routing Protocol*) kuuluvat OSPF (*Open Shortest Path First*) ja IS-IS (*Intermediate System to Intermediate System*) protokollat. Yhteystilaprotokollat soveltuvat paremmin laajojen sekä useampia reittejä sisältävien verkkojen protokollaksi. Yhteystilaprotokollien toiminta perustuu siihen että kaikki verkossa olevat reitittimet tietävät koko verkkotopologian ja laskee sen avulla oman reititystaulunsa. Jokainen reititin, joka käyttää tätä protokollaa lähettävät naapurireitittimilleen yhteystilailmoituksia, joiden avulla muodostuu ns. tietokanta verkon topologiasta. Eli kun reititin konfiguroidaan käyttämään tätä protokollaa, niin se alkaa lähettää naapurilleen sanomia kaikista reitittimen liitännöistä, joilla etsitään naapureita. Kun molemmat reitittimet toteavat toisilleen käyttävänsä samaa protokollaa, niin näiden reitittimien välille muodostuu ns. suhde. Kun tämä suhde on syntynyt niin molemmat reitittimet lähettävät toisilleen verkon topologian sisällön eli tietokannan. Tietokantamuutoksista reitittimet lähettävät toisilleen aika ajoin päivityksiä sekä päivitys myös lähetetään vaikka muutoksia ei olisi tapahtunutkaan. OSPF-protokollassa tämä päivitysaikaväli on noin puoli tuntia. Kuitenkin jos siis muutos tapahtuu, niin se päivitetään välittömästi suhteessa olevalle naapurireitittimelle. [1, s. 319-323]

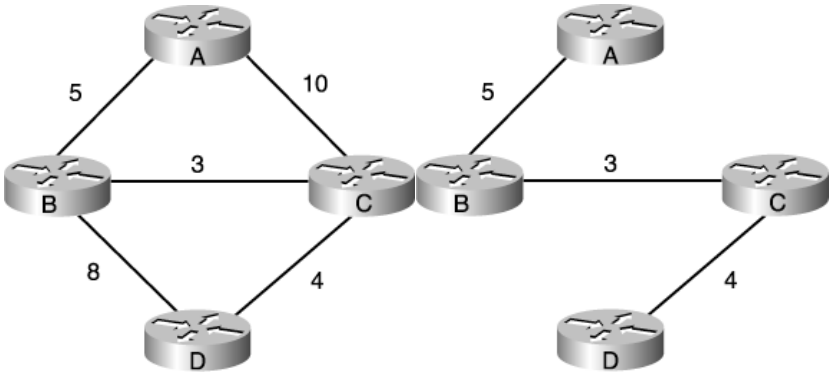
3.3.3 Hybridireititysprotokollat

Hybridireititysprotokolliin (*Balanced Hybrid Routing Protocol*) kuuluuesimerkiksi EIGRP (*Enhanced Interior Gateway Routing Protocol*) -protokolla. Tämmöisiksi protokolliksi kutsutaan sellaisia protokollia, jotka yhdistävät etäisyysvektori- sekä yhteystilaprotokollia keskenään. Eroten kuitenkin siten etäisyysvektoriprotokollasta rikkomalla reititystaulujen päivityksen jos tulee topologiamuutos verkossa. Hybridireititys-

protokolla myös käyttää vähemmän kaistanleveyttä, muistia sekä prosessoria kuin yhteystilaprotokolla. [13, s. 736]

3.4 IS-IS - protokolla

IS-IS -protokolla (*Intermediate System to Intermediate System*) on yhteystilaprotokolla, jota yleisimmin käytetään laajoissa monen alueen verkoissa kuten esimerkiksi operaattorien sisäverkoissa. Tämä siksi, koska IS-IS protokollalla on laajempi skaalautuvuus kuin esimerkiksi OSPF-protokollalla eli IS-IS -verkon kokonaiskoko voi olla maksimissaan noin 2000 reititintä kuin taas OSPF:llä vastaava reititinten määrä voi olla vain noin 50. IS-IS -protokolla kehitettiin DEC:ssä (*Digital Equipment Corporation*) myöhään 1980-luvulla osana ns. DECnettiä ja julkistettiin myös ISO-standardiksi (ISO/IEC 10589.2002). [14, s. 210]



Kuva 8 ja 9. Reititys ennen ja jälkeen Djikstran SPF-algoritmia. [16][17]

IS-IS -protokollan toimintatapa perustuukin OSPF-protokollan tavoin Djikstran SPF-algoritmiin. Tämä algoritmi voidaan kuvailla seuraavanlaisesti. Algoritmissa reititin toimii lähtöpisteenä ja painoarvon mukaan katsotaan mihin tämän reitittimen naapuriin on lyhin etäisyys. Seuraavaksi algoritmi tarkastelee missä on seuraavaksi lyhin etäisyys eli lähtöpisteenä toimivan reitittimen omien naapurien vai lähimmäisen edellisellä kierroksella määritellyn naapurin naapurien. Lähtöpisteen ja lähimmän naapurin etäisyys myös lisätään mukaan kokonaisetäisyyteen tätä laskettaessa. Kun kokonaisuudessaan lyhimmän reitittimen etäisyys on määritelty, niin se merkitään yhdeksi pisteeksi ja tämän jälkeen jatketaan määrittelyä. Tätä jatkuu niin pitkään, että kaikki topologiatietokannassa olevat pisteet ovat määritelty ja tämän jälkeen algoritmin laskeminen päättyy. [1, s. 323-325]

IS-IS -protokollalla voidaan myös määritellä reitittimelle aluetyypit. Nämä aluetyypit ovat Level 1 (*intra*), Level 2 (*inter*) ja Level 1-2 (*intra/inter*). Näillä määrityksillä voidaan rajoittaa miten reitittimet jakavat tietojaan eli Level 1 sekä Level 2 reitittimet voivat jakaa vain saman tason reititinten kanssa informaatiota, mutta molemmat pystyvät kuitenkin jakamaan tietoja Level-1-2 reitittimelle. Tällä voidaan muodostaa alueita, joiden rajat ovat ns. linkeissä toisin kuin OSPF:ssä joissa alueiden rajat ovat reitittimissä. [14, s. 211]

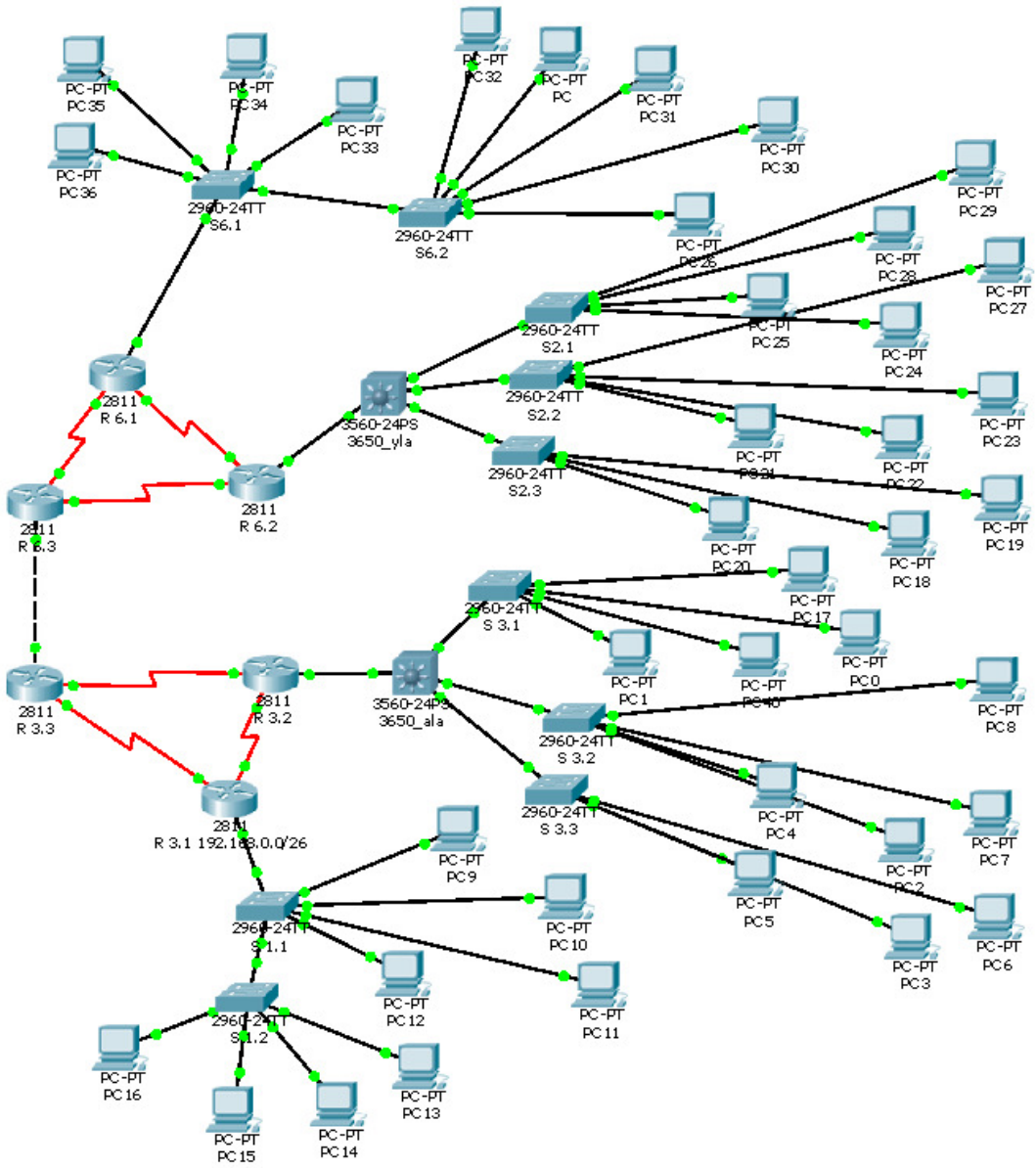
3.4.1 IS-IS -protokolla Cisco IOS:ssa

Cisco IOS (*Internetwork Operating System*) tukee hyvin itse IS-IS -protokollaa ja käydäänkin nyt läpi joitakin komentoja IS-IS -protokollalle tarkempaan protokollan konfigurointiin itse Ciscon käyttöjärjestelmässä. Itse IS-IS -protokolla voidaan siis asettaa reitittimeen Cisco IOS:ssa komennolla *router isis* sekä komennolla *ip router isis*. Näihin komentoihin voidaan myös määritellä alue eli *area* jos kyseessä siis olisi todella laaja verkkokokonaisuus, niin tällä voitaisiin määritellä alueita IS-IS -verkkoon. Tämän lisäksi itse reitittimeen pitää määritellä yksilökohtainen NET-tunniste (*Network Entity Title*), joka siis on heksadesimaalinen numerosarja eli esimerkiksi 49.0001.1111.1111.1111.00. Itse reititysten painoarvoa (*Metric*) voidaan muuttaa komennolla *isismetric* liitännässä, jolla voidaan näin ollen määritellä reittiä mahdollisesti uudelleen. Myös ns. Hello-pakettien lähetysväliä voidaan muuttaa komennolla *isis hello-interval*, jonka arvo tulee määritellä sekunneissa. Tällä Hello-paketilla reitittimet saavat tietää, että onko vielä ns. naapurisuhde kunnossa. [15]

4 VERKON MÄÄRITTÄMINEN IS-IS -PROTOKOLLAAN

Käytännön osuutena tässä opinnäytetyössä oli rakentaa kuvitteellinen toimistoverkko, joka on jaettu kahteen aliverkkoon, 192.168.0.0 ja 192.168.1.0, vaihtuvalla aliverkonpeitteellä konetarpeen mukaan. Tässä käytännön osuudessa keskitymmekin oikeastaan vain verkon reitittimiin ja niiden konfigurointiin sillä verkon kytkimet sekä niiden konfiguroinnin hoitaa Aki Timonen, koska tämä verkko kokonaisuus on tehty hänen kanssaan. Itse verkon suunnittelu on tehty Cisco Packet Tracerin sekä Mikkelin ammattikorkeakoulun laitteiston avulla.

4.1 Verkon kuvaus sekä laitteisto



Kuva 9. Verkon topologia Cisco Packet Tracer -ohjelmassa.

Verkon topologiaa suunniteltiin Cisco Packet Tracerilla ensiksi, että saatiin kokonaisuus hahmoteltua. Verkon ideana olikin alusta asti olla toimistoverkko vähintään kahdella aliverkolla ja nämä aliverkot sitten yhdistettäisiin reititysprotokollan avulla. Itse verkon suunnittelussa tuli ottaa huomioon myös mahdolliset laitteiden ominaisuudet, kun itse verkon käytännön toteuttaminen suoritettiin Mikkelin ammattikorkeakoulun laitteistolla Mikpolissa. Loppujen lopuksi päädyinkin käyttämään reitittimiä malliltaan Cisco 2811. Näissä reitittimissä oli tarvittavat liittimet sekä konfigurointityökalut toteuttamaan tämä suunniteltu verkkokokonaisuus. Tämän verkon toteutukseen tarvit-

sinkin kuusi kappaletta Cisco 2811 mallisia reitittimiä sekä myös tarvittavat serial- ja console-kaapelit. Tämä verkkokokonaisuus myös määriteltiin käyttämään IS-IS – reititysprotokollaa.

Taulukko 3. 192.168.0.0 aliverkon IP-taulu.

Reitittimien väliset verkot	
R3.3-R3.2	192.168.0.240 /30
R3.2-R3.1	192.168.0.244 /30
R3.2-3650_ala	192.168.0.248 /30
R3.1-R3.3	192.168.0.252 /30
R3.1 VLAN aliliitännät (VLAN 10, 20, 30)	
FastEthernet 0/0.10	192.168.0.192 /28
FastEthernet 0/0.20	192.168.0.208 /28
FastEthernet 0/0.30	192.168.0.224 /28
3650_ala VLAN-osoitteet	
VLAN 10	192.168.0.0 /25
VLAN 20	192.168.0.128 /27
VLAN 30	192.168.0.160 /27

Taulukko 4. 192.168.1.0 aliverkon IP-taulu.

Reitittimien väliset verkot	
R6.3-R3.3	192.168.1.192 /30
R6.3-R6.2	192.168.1.196 /30
R6.2-R6.1	192.168.1.200 /30
R6.1-R6.3	192.168.1.204 /30
R6.2-3650_yla	192.168.1.208 /30
R6.1 VLAN aliliitännät (VLAN 10, 20, 30)	
FastEthernet 0/0.10	192.168.1.128 /27
FastEthernet 0/0.20	192.168.1.160 /28
FastEthernet 0/0.30	192.168.1.176 /28
3650_ylaVLAN-osoitteet	
VLAN 10	192.168.1.0 /26
VLAN 20	192.168.1.64 /27
VLAN 30	192.168.1.96 /27

Ylläolevista IP-tauluista (kts. Taulukko 3 ja Taulukko 4) nähdään IP-osoiteavaruudet molemmille aliverkoille eli 192.168.0.0 ja 192.168.1.0. Myös IP-tauluista nähdään aliverkkojen laitteiden väliset kytkennät sekä niiden IP-osoitteet VLAN-kytkennöille.

4.2 Reitittimien konfigurointi

Ensiksi aloitin reitittimien konfiguroinnin peruskomennoilla eli suojauksilla konsolipääte sekä telnet-yhteyksille. Kuitenkin, että näitä päästään konfiguroimaan, niin pitää suorittaa seuraava komento: *Enable* – Tällä komennolla päästään privileged EXEC –tilaan, josta päästään itse globaaliin konfigurointitilaan kirjoittamalla privileged EXEC -tilassa komento *configure terminal*.

Esim.

`Router>enable` – Käyttäjän EXEC –tila (Router>)

`Router#configure terminal` – Privileged EXEC –tila(Router#)

`Router(config)#` – Globaali konfigurointitila

Ensimmäisenä privileged EXEC -tilassa suoritinkin reitittimille käynnistysasetusten tyhjennyksen komennolla: Router#*erase startup-config*. Tämän komennon suorittamisen jälkeen reititin tyhjentää mahdolliset käynnistysasetukset ja käynnistää itsensä oletusasetuksin. Tämän jälkeen suoritinkin suojauskomennot konsolipääte, telnet sekä privileged EXEC –tilaan pääsyä varten globaalissa konfiguraatiotilassa.

Salasana konsolipäätteelle:

```
Router(config)#line console 0
Router(config-line)#password cisco
Router(config-line)#login
```

Salasana Telnet-yhteydelle:

```
Router(config)#linevty 0 4
Router(config-line)#password cisco
Router(config-line)#login
```

Salasana privileged EXEC –tilaan:

```
Router(config)#enable secret cisco
```

Näillä edellämainituilla suojausasetuksilla on estetty pääsy itse reitittimelle muilta kuin ns. verkonhaltijalta, joka nämä salasanat tietää. Suojausasetusten jälkeen nimesin reitittimet komennolla *hostname*, joka myös suoritetaan globaalissa konfigurointitilassa eli Router(config)#hostname. Edellä mainittu komento helpottaa itse reitittimien konfigurointia osoittamalla selkeästi aina mistä reitittimestä on kysymys sekä myös auttaa yksilöimään reitittimen verkkotopologiassa, kun katsotaan itse reittejä.

Perusasetusten laittamisen jälkeen onkin siirryttävä konfiguroimaan itse liitäntöjen verkko-osoitteita sekä mahdollisia protokollia. Tulenkin selittämään tässä vain reitittimien R3.3 ja R3.1 asetusten määritykset sillä reitittimet R3.2, R6.3, R6.2 sekä R6.1 on tehty vastaavanlaisella tavalla, mutta vaan eri aliverkon alle. Itse laitteiden tarkemmat määritellyt asetukset löytyvät liitteinä.

Reitittimen R3.3 asetusten määritys

Reitittimen R3.3 liitäntöjen ja protokollan konfigurointi pitää aloittaa verkko-osoitteiden määrittelystä. Kuten verkkokuvauksessa sekä IP-taulussa huomataan niin tähän reitittimeen pitää konfiguroida yksi FastEthernet liitäntä sekä kaksi Serial-liitäntää.

Ensimmäisenä määrittelin R3.3-reitittimen käyttämään IS-IS –reititysprotokollaa, jolla saataisiin kaikki verkot yhdistettyä toisiinsa. Tämän reititysprotokollan perusmääritys tapahtuu seuraavilla komennoilla:

```
R3.3(config)#router isis
R3.3(config-router)#net 49.0001.1111.1111.1111.00
R3.3(config-router)#is-type level-1-2
```

Näillä komennoilla määriteltiin reitin käyttämään IS-IS –reititysprotokollaa. NET-osoitteella määriteltiin alue sekä reitittimen yksilöllinen tunniste. Tämän lisäksi *is type*–komennolla määriteltiin mitä IS-IS –tyyppiä reitin käyttää. Is-tyyppi level-1-2 on myös oletusasetuksena, kun määritellään reitin käyttämään tätä protokollaa.

Seuraavaksi konfiguroin FastEthernet 0/0 –liitännän saman verkon alle kuin reitittimen R6.3 vastaavan liitännän sekä myös määrittelin tämän kytkennän käyttämään jo edellä mainittua reititysprotokollaa. Tämä saadaan aikaan seuraavilla komennoilla

globaalissa konfiguraatiotilassa eli määritellään näillä komennoilla liitäntä sekä liitäntään haluttu IP-osoite sekä aliverkonpeite. Sekä myös käynnistetään itse liitäntä *no shutdown*–komennolla.

```
R3.3(config)#interface fa0/0
R3.3(config-if)#ip address 192.168.1.193 255.255.255.255.252
R3.3(config-if)#ip router isis
R3.3(config-if)#no shutdown
```

Samanlaiset määritykset piti myös tehdä Serial-liitännöille S0/0/0 ja S0/0/1, että R3.3 olisi myös kytköksissä reitittimiin R3.2 sekä R3.1.

```
R3.3(config)#int s0/0/0
R3.3(config-if)#ip address 192.168.0.241 255.255.255.252
R3.3(config-if)#ip router isis
R3.3(config-if)#no shutdown
```

Reitittimen R3.1 asetusten määritys

Reitittimessä R3.1 tehtiin muuten samanlaiset perusmääritykset sekä liitäntämääritykset kuin reitittimessä R3.3, mutta eri ip-osoitteilla Serial-liitäntöihin S0/0/0 sekä S0/0/1. Kuitenkin tässä reitittimessä piti tehdä VLAN (*Virtual LAN*) määritykset aliliitännöille sekä DHCP (*Dynamic HostConfiguration Protocol*) alueet näille VLAN-alueille eli dynaamiset IP-osoiteavaruudet.

Aliliitännät määrittelin FastEthernet 0/0 –liitännälle. Koska VLAN-alueita oli kolme kappaletta (VLAN 10, VLAN 20 ja VLAN 30), niin myös aliliitäntöjä tälle liitännälle piti tehdä kolme kappaletta. Näiden määritys tapahtui seuraavanlaisilla komennoilla:

VLAN 10 määritys:

```
R3.1(config)#int fa0/0.10
R3.1(config-subif)#encapsulation dot1Q 10
R3.1(config-subif)#ip address 192.163.0.193 255.255.255.240
```

VLAN 20 määritys:

```
R3.1(config)#int fa0/0.20
R3.1(config-subif)#encapsulation dot1Q 20
R3.1(config-subif)#ip address 192.163.0.209 255.255.255.240
```

VLAN 30 määritys:

```
R3.1(config)#int fa0/0.30
R3.1(config-subif)#encapsulation dot1Q 30
R3.1(config-subif)#ip address 192.163.0.209 255.255.255.240
```

Komennolla *int fa0/0.10* päästään konfiguroimaan aliliitäntää FastEthernet 0/0.10. Tämän jälkeen komennolla *encapsulation dot1Q 10* määritellään aliliitäntä käyttämään IEEE802.1Q:ta, joka yksinkertaisuudessaan tarkoittaa sitä että tässä liitännässä voidaan käyttää VLAN-alueita. Tämän lisäksi VLAN-alueille määriteltiin dynaamiset osoitealueet eli DHCP ”altaat”. Nämä saatiin määriteltyä seuraavin komennoin:

VLAN 10 DHCP-alue:

```
R3.1(config)#ip dhcp pool VLAN10
R3.1(dhcp-config)#network 192.168.0.192 255.255.255.240
R3.1(dhcp-config)#default-router 192.168.0.193
```

VLAN 20 DHCP-alue:

```
R3.1(config)#ip dhcp pool VLAN20
R3.1(dhcp-config)#network 192.168.0.208 255.255.255.240
R3.1(dhcp-config)#default-router 192.168.0.209
```

VLAN 30 DHCP-alue:

```
R3.1(config)#ip dhcp pool VLAN30
R3.1(dhcp-config)#network 192.168.0.224 255.255.255.240
R3.1(dhcp-config)#default-router 192.168.0.225
```

Komennolla *ip dhcp pool<nimi>* määritellään DHCP-alue sekä tämän alueen nimi. Tämän jälkeen siirrytään itse DHCP:n asetusten määritykseen. *Network*-komennolla määritellään alue sekä alueen aliverkonpeite. Komennolla *default-router* määritellään tämän alueen oletusyhdyskäytävä.

Kuitenkaan pelkkä DHCP-alueiden määritys ei riitä. Pitää vielä poistaa koneille jaettavasta osoitejakelusta nämä edellä määritellyt oletusyhdyskäytävät. Tämä tapahtuu seuraavalla komennolla:

```
R3.1(config)#ipdhcp excluded-address 192.168.0.193
R3.1(config)#ipdhcp excluded-address 192.168.0.209
R3.1(config)#ipdhcp excluded-address 192.168.0.225
```

4.3 Verkon testaus

Verkon testauksessa käytin apunani konsolipäätettä sekä sen komentoja ja myös verkossa olevilla koneilla *ping*-komentoa komentorivin kautta. Käytin testauksessa myös tietoja mitä sain reitittimiltä R3.1 sekä R6.3.

Ensimmäisenä tarkastetaan, että on reititys onnistunut ja löytävätkö verkot toisensa. Tähän käytin ensimmäisenä komentoa *show ip route* reitittimillä R6.3 sekä R3.1.

```
R6.3#show ip route
     192.168.0.0/24 is variably subnetted, 10 subnets, 4 masks
i L1    192.168.0.0/25 [115/40] via 192.168.1.193, FastEthernet0/0
i L1    192.168.0.224/28 [115/30] via 192.168.1.193, FastEthernet0/0
i L1    192.168.0.248/30 [115/30] via 192.168.1.193, FastEthernet0/0
i L1    192.168.0.252/30 [115/20] via 192.168.1.193, FastEthernet0/0
i L1    192.168.0.240/30 [115/20] via 192.168.1.193, FastEthernet0/0
i L1    192.168.0.244/30 [115/30] via 192.168.1.193, FastEthernet0/0
i L1    192.168.0.192/28 [115/30] via 192.168.1.193, FastEthernet0/0
i L1    192.168.0.208/28 [115/30] via 192.168.1.193, FastEthernet0/0
i L1    192.168.0.160/27 [115/40] via 192.168.1.193, FastEthernet0/0
i L1    192.168.0.128/27 [115/40] via 192.168.1.193, FastEthernet0/0
     192.168.1.0/24 is variably subnetted, 9 subnets, 4 masks
i L1    192.168.1.0/26 [115/30] via 192.168.1.198, Serial0/0/0
i L1    192.168.1.200/30 [115/20] via 192.168.1.205, Serial0/0/1
                         [115/20] via 192.168.1.198, Serial0/0/0
C       192.168.1.204/30 is directly connected, Serial0/0/1
C       192.168.1.192/30 is directly connected, FastEthernet0/0
C       192.168.1.196/30 is directly connected, Serial0/0/0
i L1    192.168.1.208/30 [115/20] via 192.168.1.198, Serial0/0/0
i L1    192.168.1.160/28 [115/20] via 192.168.1.205, Serial0/0/1
i L1    192.168.1.176/28 [115/20] via 192.168.1.205, Serial0/0/1
i L1    192.168.1.128/27 [115/20] via 192.168.1.205, Serial0/0/1
```

Tällä komennolla huomataankin, että reititin R6.3 löytää 192.168.0.0 verkosta 10 aliverkkoa ja 192.168.1.0 verkon alta 9 aliverkkoa. Sekä usea verkko on reitittynyt oikein käyttäen IS-IS – reititysprotokollaa. Sama voidaan huomata myös reitittimestä R3.1.

```
R3.1#show ip route

     192.168.0.0/24 is variably subnetted, 10 subnets, 4 masks
i L1    192.168.0.0/25 [115/30] via 192.168.0.245, Serial0/0/1
C       192.168.0.224/28 is directly connected, FastEthernet0/0.30
i L1    192.168.0.248/30 [115/20] via 192.168.0.245, Serial0/0/1
C       192.168.0.252/30 is directly connected, Serial0/0/0
i L1    192.168.0.240/30 [115/20] via 192.168.0.254, Serial0/0/0
                         [115/20] via 192.168.0.245, Serial0/0/1
C       192.168.0.244/30 is directly connected, Serial0/0/1
C       192.168.0.192/28 is directly connected, FastEthernet0/0.10
C       192.168.0.208/28 is directly connected, FastEthernet0/0.20
i L1    192.168.0.160/27 [115/30] via 192.168.0.245, Serial0/0/1
i L1    192.168.0.128/27 [115/30] via 192.168.0.245, Serial0/0/1
     192.168.1.0/24 is variably subnetted, 9 subnets, 4 masks
i L1    192.168.1.0/26 [115/50] via 192.168.0.254, Serial0/0/0
i L1    192.168.1.200/30 [115/40] via 192.168.0.254, Serial0/0/0
i L1    192.168.1.204/30 [115/30] via 192.168.0.254, Serial0/0/0
i L1    192.168.1.192/30 [115/20] via 192.168.0.254, Serial0/0/0
i L1    192.168.1.196/30 [115/30] via 192.168.0.254, Serial0/0/0
i L1    192.168.1.208/30 [115/40] via 192.168.0.254, Serial0/0/0
i L1    192.168.1.160/28 [115/40] via 192.168.0.254, Serial0/0/0
i L1    192.168.1.176/28 [115/40] via 192.168.0.254, Serial0/0/0
i L1    192.168.1.128/27 [115/40] via 192.168.0.254, Serial0/0/0
```

Seuraavaksi tarkastellaan lähemmin IS-IS – protokollaan liittyviä komentoja, joilla voidaan katsoa toimiiko itse reititys oikein. Ensimmäisenä voidaan katsoa komennolla *show isis neighbors* mahdollisia naapurisuhteita tässä protokollassa.

```
R3.1#show isis neighbors

System Id   Type Interface   IP Address     State Holdtime Circuit Id
R3.3        L1L2 Se0/0/0     192.168.0.254   UP    26       01
R3.2        L1L2 Se0/0/1     192.168.0.245   UP    27       01
```

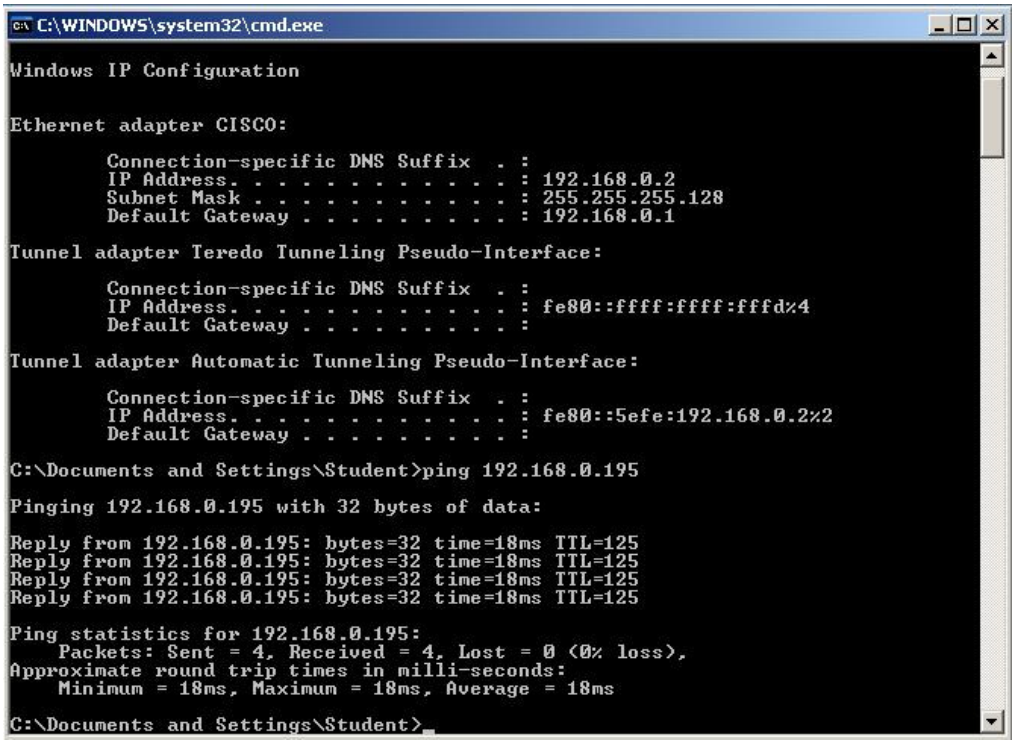
Tästä huomataankin, että reitittimellä R3.1 on kaksi IS-IS – protokolla naapuria listoillaan eli reitittimet R3.3 sekä R3.2. Tämän jälkeen voidaankin tarkastella itse verkkotopologiaa, jonka pitäisi näyttää *show ip route*:n tavoin määritellyt verkot.

```
R3.1#show isis topology

IS-IS paths to level-1 routers
System Id         Metric     Next-Hop           Interface   SNPA
R3.3               10         R3.3                Se0/0/0     *HDLC*
R3.2               10         R3.2                Se0/0/1     *HDLC*
R3.1               --
3650_yla           20         R3.2                Se0/0/1     *HDLC*
R6.3               20         R3.3                Se0/0/0     *HDLC*
R6.2               30         R3.3                Se0/0/0     *HDLC*
R6.1               30         R3.3                Se0/0/0     *HDLC*
3650_ala           40         R3.3                Se0/0/0     *HDLC*
```

Tällä komennolla (*show isis topology*) nähdään kaikki verkot mitä reitittimet mainostavat käyttäen apunaan IS-IS -protokollaa. Taulukosta nähdään laitteiden nimet (*System Id*), reitin painoarvo (*metric)*, seuraava mahdollinen yhdyskäytävä kyseiselle laitteelle (*Next-Hop*) sekä myös liitäntä (*Interface*). Tästä taulukosta voidaan myös päätellä, että kaikki verkot löytyvät IS-IS –tietokannasta.

Seuraavaksi testattavana oli, että säilyykö verkkoyhteys vaikka välistä katkaistaisiin yksi kannatteleva kaapeli. Kuitenkin ensin testataan toimiiko *pingaus* tietokoneesta PC1 (IP: 192.168.0.2) tietokoneeseen PC9 (IP: 192.168.0.195).

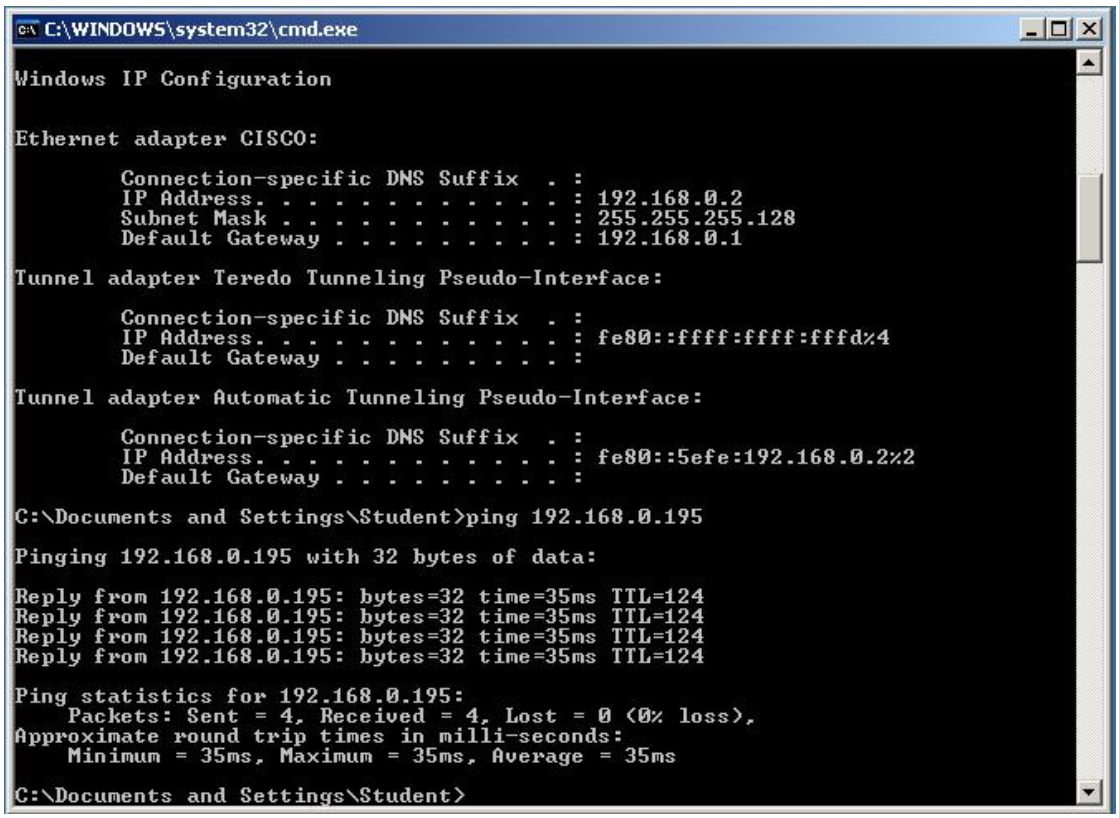


Tämän jälkeen irroitettiin R3.1 ja R3.2 välillä oleva serialkaapeli. *Show isis topology*–komennolla huomataan, että vaikka kannatteleva kaapeli katkaistiin, niin siirryttiin käyttämään automaattisesti varareittiä reitittimien välillä eli kaikki reititys kulkee nyt R3.3 reitittimen kautta kohteesta R3.1 kohteeseen R3.2.

```
R3.1#show isis topology

IS-IS paths to level-1 routers
System Id          Metric     Next-Hop             Interface   SNPA
R3.3                10         R3.3                 Se0/0/0     *HDLC*
R3.2                20         R3.3                 Se0/0/0     *HDLC*
R3.1                --
3650_yla            30         R3.3                 Se0/0/0     *HDLC*
R6.3                20         R3.3                 Se0/0/0     *HDLC*
R6.2                30         R3.3                 Se0/0/0     *HDLC*
R6.1                30         R3.3                 Se0/0/0     *HDLC*
3650_ala            40         R3.3                 Se0/0/0     *HDLC*
```

Myös tietokoneelta tietokoneelle *pingaus* onnistui taas aivan normaalisti.



Kokeilin myös irrottaa kaapelin reitittimien R3.1 ja R3.3 välillä sekä koettaa näin ottaa yhteyttä verkkoon 192.168.1.0. Kaapelin irrottamisen jälkeen huomattiin seuraava topologia muutos.

```
R3.1#show isis topology

IS-IS paths to level-1 routers
System Id          Metric     Next-Hop             Interface   SNPA
R3.3                20         R3.2                 Se0/0/1     *HDLC*
R3.2                10         R3.2                 Se0/0/1     *HDLC*
R3.1                --
3650_yla            20         R3.2                 Se0/0/1     *HDLC*
R6.3                30         R3.2                 Se0/0/1     *HDLC*
R6.2                40         R3.2                 Se0/0/1     *HDLC*
R6.1                40         R3.2                 Se0/0/1     *HDLC*
3650_ala            50         R3.2                 Se0/0/1     *HDLC*
```

```
R3.1#ping 192.168.1.3
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.1.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 68/85/148
ms
```

Eli protokolla oli havainnut muutoksen ja määritellyt reitit R3.2 kautta. Sekä myös *pingaaminen* onnistui tietokoneeseen PC29 (IP: 192.168.1.3) normaalisti, niin kuin pitikin. Näistä voidaan päätellä että IS-IS -protokollan määrittäminen reitittimiin oli onnistunut.

5 YHTEENVETO

Kiinnostukseni kohteena on aina ollut tietoliikennetekniikka ja muutenkin manuaalinen konfigurointi tietoliikennelaitteistolla. Siksi valitsinkin opinnäytetyöni aiheeksi juurikin cisco-verkot sisältäen sen laitteiston sekä mahdolliset laitteistojen protokollat. Ciscon reitittimillä sekä Cisco Packet Tracerilla olen aikaisemminkin tehnyt tehtäviä kursseilla sekä myös vapaa-aikana, joten niiden kanssa ei suurempia ongelmia ollut verkon suunnittelussa. Sen takia peruskomennotkin palailivat nopeasti mieleen eikä niiden kanssa syntynyt ongelmia opinnäytetyötä tehdessäni.

Kun valitsin IS-IS -protokollan reititystä varten tiesin, että tulisin näin oppimaan uutta teoriaa tietoliikenteestä sekä myös uutta asiaa Cisco-laitteistoista. IS-IS -protokollan kanssa tulikin ongelmia sen konfiguroimisesta reitittimiin sekä teorian osalta sen vähäiseksi jääneen kirjallisen materiaalin löytämisestä. Tämän vuoksi jouduinkin turvautumaan löytämääni Ciscon verkkomateriaaliin konfiguroimisen helpottamiseksi. Materiaalin avulla kuitenkin konfigurointi onnistui verkkojen välille loppujen lopuksi täydellisesti. Kaikenkaikkiaan opinnäytetyöstä löytyi riittävästi haastetta sen toteuttamiseen ja näin opin myös paljon uusia asioita tietoliikenneverkoista.

6 LÄHTEET

1. Anttila, Aki 2000.TCP/IP-tekniikka. Helsinki: Helsinki Media
2. Comer, Douglas E. 2000. TCP/IP. IT Press
3. Hakala, Mika & Vainio, Mika 2005. Tietoverkon rakentaminen. Jyväskylä: Docendo Finland Oy
4. RFC: 791: Internet Protocol. 1981.http://www.ietf.org/rfc/rfc791.txt . Päivitetty 16.10.1992.Luettu 18.05.2012.
5. RFC: 905: ISO Transport Protocol Specification. http://tools.ietf.org/pdf/rfc905.pdf. Päivitetty 05.06.2007. Luettu 18.05.2012.
6. OSI-malli. http://upload.wikimedia.org/wikipedia/fi/4/4c/OSI-malli.jpg . Päivitetty 17.07.2005. Luettu 18.05.2012.
7. OSI-malli – TCP/IP-mallihttps://www.vahtiohje.fi/image/image_gallery?uuid=14f8c5f5-83f1-44d0-8e86-87fa83c6d983&groupId=10128&t=1291719541517 . Päivitetty 08.12.2010. Luettu 18.05.2012.
8. IP-sähke. http://koti.mbnet.fi/mrin/kuvat/ip_sahke.gif .Päivitetty 14.12.2000. Luettu 18.05.2012.
9. Yksinkertainen kuittaus. http://oppimateriaalit.internetix.fi/fi/avoimet/6tekniikkatalous/verkko/kuvat/yks_kuit.jpg .Päivitetty 30.11.2004. Luettu 18.05.2012
10. TCP-segmentti. http://koti.mbnet.fi/mrin/kuvat/tcp_segm.gif .Päivitetty 14.12.2000. Luettu 18.05.2012.
11. Ethernet Architecture. http://ewh.ieee.org/r6/scv/comsoc/Workshop_101310_StdArch.pdf . Päivitetty 12.10.2010. Luettu 18.05.2012.
12. Ethernet-kehys. http://internetix.fi/opinnot/opintojaksot/6tekniikkatalous/verkko/luku2/eth_keh.jpg .Päivitetty 19.12.1996. Luettu 18.05.2012
13. Holttinen, Jarmo 2002. Cisco Verkkoakatemia 1. vuosi. Helsinki: Edita Prima Oy
14. Sosinsky, Barrie 2009. Networking Bible. Indianapolis: Wiley Publishing
15. Configuring Integrated IS-IS http://www.cisco.com/en/US/docs/ios/12_2/ip/configuration/guide/1cfisis.html .Päivitetty 18.05.2012. Luettu 18.05.2012.

16. SPF algoritmi. http://cisco-press-traffic-engineering.org.ua/1587050315/images/1587050315/graphics/04fig01.gif. Päivitetys tietoja ei saatu. Luettu 18.05.2012.
17. SPF algoritmi jälkeen. http://cisco-press-traffic-engineering.org.ua/1587050315/images/1587050315/graphics/04fig02.gif. Päivitystietoja ei saatu. Luettu 18.05.2012.

Reitittimen R3.3 asetukset

```
R3.3
Current configuration : 927 bytes
!
version 12.4
service timestamps debug datetimemsec
service timestamps log datetimemsec
no service password-encryption
!
hostname R3.3
!
boot-start-marker
boot-end-marker
!
noaaa new-model
!
ipcef
!
multilink bundle-name authenticated
!
archive
logconfig
hidekeys
!
interface FastEthernet0/0
ip address 192.168.1.193 255.255.255.252
ip router isis
duplex auto
speed auto
!
interface FastEthernet0/1
noip address
shutdown
duplex auto
speed auto
!
interface Serial0/0/0
ip address 192.168.0.241 255.255.255.252
ip router isis
clock rate 64000
!
interface Serial0/0/1
```

Reitittimen R3.3 asetukset

```
ip address 192.168.0.254 255.255.255.252
ip router isis
clock rate 64000
!
routerisis
net 49.0001.1111.1111.1111.00
!
ip forward-protocol nd
!
ip http server
!
control-plane
!
line con 0
password cisco
login
line aux 0
linevty 0 4
password cisco
login
!
scheduler allocate 20000 1000
!
end
```

Reitittimen R3.2 asetukset

```
R3.2
Current configuration : 1015 bytes
!
version 12.4
service timestamps debug datetimemsec
service timestamps log datetimemsec
no service password-encryption
!
hostname R3.2
!
boot-start-marker
boot-end-marker
!
noaaa new-model
!
dot11 syslog
!
ipcef
!
multilink bundle-name authenticated
!
voice-card 0
nodspfarm
!
archive
logconfig
hidekeys
!
interface FastEthernet0/0
ip address 192.168.0.249 255.255.255.252
ip router isis
duplex auto
speed auto
!
interface FastEthernet0/1
noip address
shutdown
duplex auto
speed auto
!
interface Serial0/0/0
ip address 192.168.0.242 255.255.255.252
```

Reitittimen R3.2 asetukset

```
ip router isis
!
interface Serial0/0/1
ip address 192.168.0.245 255.255.255.252
ip router isis
!
routerisis
net 49.0001.2222.2222.2222.00
!
ip forward-protocol nd
!
ip http server
noip http secure-server
!
control-plane
!
line con 0
password cisco
login
line aux 0
linevty 0 4
password cisco
login
!
scheduler allocate 20000 1000
!
end
```

Reitittimen R3.1 asetukset

```
R3.1
Current configuration : 1770 bytes
!
version 12.4
service timestamps debug datetimemsec
service timestamps log datetimemsec
no service password-encryption
!
hostname R3.1
!
boot-start-marker
boot-end-marker
!
noaaa new-model
!
dot11 syslog
!
ipcef
noipdhcp use vrf connected
ipdhcp excluded-address 192.168.0.193
ipdhcp excluded-address 192.168.0.209
ipdhcp excluded-address 192.168.0.225
!
ipdhcp pool VLAN10
network 192.168.0.192 255.255.255.240
default-router 192.168.0.193
!
ipdhcp pool VLAN20
network 192.168.0.208 255.255.255.240
default-router 192.168.0.209
!
ipdhcp pool VLAN30
network 192.168.0.224 255.255.255.240
default-router 192.168.0.225
!
multilink bundle-name authenticated
!
voice-card 0
nodspfarm
!
archive
```

Reitittimen R3.1 asetukset

```
logconfig
hidekeys
!
interface FastEthernet0/0
noip address
duplex auto
speed auto
!
interface FastEthernet0/0.10
encapsulation dot1Q 10
ip address 192.168.0.193 255.255.255.240
ip router isis
!
interface FastEthernet0/0.20
encapsulation dot1Q 20
ip address 192.168.0.209 255.255.255.240
ip router isis
!
interface FastEthernet0/0.30
encapsulation dot1Q 30
ip address 192.168.0.225 255.255.255.240
ip router isis
!
interface FastEthernet0/1
noip address
duplex auto
speed auto
!
interface Serial0/0/0
ip address 192.168.0.253 255.255.255.252
ip router isis
no fair-queue
!
interface Serial0/0/1
ip address 192.168.0.246 255.255.255.252
ip router isis
clock rate 64000
!
routerisis
net 49.0001.3333.3333.3333.00
!
ip forward-protocol nd
```

Reitittimen R3.1 asetukset

```
!
ip http server
noip http secure-server
!
control-plane
!
line con 0
password cisco
login
line aux 0
linevty 0 4
password cisco
login
!
scheduler allocate 20000 1000
!
end
```

Reitittimen R6.3 asetukset

```
R6.3
Current configuration : 915 bytes
!
version 12.4
service timestamps debug datetimemsec
service timestamps log datetimemsec
no service password-encryption
!
hostname R6.3
!
boot-start-marker
boot-end-marker
!
noaaa new-model
!
ipcef
!
multilink bundle-name authenticated
!
archive
logconfig
hidekeys
!
interface FastEthernet0/0
ip address 192.168.1.194 255.255.255.252
ip router isis
duplex auto
speed auto
!
interface FastEthernet0/1
noip address
shutdown
duplex auto
speed auto
!
interface Serial0/0/0
ip address 192.168.1.197 255.255.255.252
ip router isis
no fair-queue
!
interface Serial0/0/1
ip address 192.168.1.206 255.255.255.252
```

Reitittimen R6.3 asetukset

```
ip router isis
clock rate 125000
!
routerisis
net 49.0001.5555.5555.5555.00
!
ip forward-protocol nd
!
ip http server
!
control-plane
!
line con 0
password cisco
login
line aux 0
linevty 0 4
password cisco
login
!
scheduler allocate 20000 1000
!
end
```

Reitittimen R6.2 asetukset

```
R6.2
Current configuration : 1049 bytes
!
version 12.4
service timestamps debug datetimemsec
service timestamps log datetimemsec
no service password-encryption
!
hostname R6.2
!
boot-start-marker
boot-end-marker
!
!
noaaa new-model
dot11 syslog
!
ipcef
!
multilink bundle-name authenticated
!
voice-card 0
nodspfarm
!
archive
logconfig
hidekeys
!
interface FastEthernet0/0
ip address 192.168.1.209 255.255.255.252
ip router isis
duplex auto
speed auto
!
interface FastEthernet0/1
noip address
shutdown
duplex auto
speed auto
!
interface Serial0/0/0
ip address 192.168.1.198 255.255.255.252
```

Reitittimen R6.2 asetukset

```
ip router isis
no fair-queue
clock rate 125000
!
interface Serial0/0/1
ip address 192.168.1.201 255.255.255.252
ip router isis
!
routerisis
net 49.0001.6666.6666.6666.00
!
ip forward-protocol nd
!
ip http server
noip http secure-server
!
control-plane
!
line con 0
password cisco
login
line aux 0
linevty 0 4
password cisco
login
!
scheduler allocate 20000 1000
!
end
```

Reitittimen R6.1 asetukset

```
R6.1
Current configuration : 1766 bytes
!
version 12.4
service timestamps debug datetimemsec
service timestamps log datetimemsec
no service password-encryption
!
hostname R6.1
!
boot-start-marker
boot-end-marker
!
noaaa new-model
dot11 syslog
!
ipcef
noipdhcp use vrf connected
ipdhcp excluded-address 192.168.1.129
ipdhcp excluded-address 192.168.1.161
ipdhcp excluded-address 192.168.1.177
!
ipdhcp pool VLAN10
network 192.168.1.128 255.255.255.224
default-router 192.168.1.129
!
ipdhcp pool VLAN20
network 192.168.1.160 255.255.255.240
default-router 192.168.1.161
!
ipdhcp pool VLAN30
network 192.168.1.176 255.255.255.240
default-router 192.168.1.177
!
multilink bundle-name authenticated
!
voice-card 0
nodspfarm
!
archive
logconfig
hidekeys
```

Reitittimen R6.1 asetukset

```
!
interface FastEthernet0/0
noip address
duplex auto
speed auto
!
interface FastEthernet0/0.10
encapsulation dot1Q 10
ip address 192.168.1.129 255.255.255.224
ip router isis
!
interface FastEthernet0/0.20
encapsulation dot1Q 20
ip address 192.168.1.161 255.255.255.240
ip router isis
!
interface FastEthernet0/0.30
encapsulation dot1Q 30
ip address 192.168.1.177 255.255.255.240
ip router isis
!
interface FastEthernet0/1
noip address
shutdown
duplex auto
speed auto
!
interface Serial0/0/0
ip address 192.168.1.205 255.255.255.252
ip router isis
!
interface Serial0/0/1
ip address 192.168.1.202 255.255.255.252
ip router isis
clock rate 125000
!
routerisis
net 49.0001.7777.7777.7777.00
!
ip forward-protocol nd
!
ip http server
```

Reitittimen R6.1 asetukset

```
noip http secure-server
!
control-plane
!
line con 0
password cisco
login
line aux 0
linevty 0 4
password cisco
login
!
scheduler allocate 20000 1000
!
end
```