TARTU ÜLIKOOL
MATEMAATIKA-INFORMAATIKATEADUSKOND
ARVUTITEADUSE INSTITUUT

Erkki Laaneoks

Sissejuhatus võrgutehnoloogiasse

2010

Käesoleva raamat on valminud Eesti Infotehnoloogia Sihtasutuse programmi Tiigriülikool toel.

Toimetajad: Meelis Roos ja Erkki Kukk

Sisukord

1. Eessõna

Arvutivõrkude vajaduses ei kahtle vist enam keegi, kes vähegi on kasutanud Internetti. Paraku puudus enne käesoleva raamatu esmatrükki korralik eestikeelne võrgutehnoloogia alane kirjandus. Kuna esimene trükk sai kiiresti otsa, siis tekkis vajadus kordustrüki järele.

Käesolev raamat püüab lühidalt, kuid arusaadavalt katta võimalikult suure osa võrgutehnoloogiast. Seetõttu ei peatuta vanadel tehnoloogiatel või mainitakse neid vaid mööda minnes, kui see on vajalik edaspidise mõistmiseks. Samuti püütakse hästi vähe käsitleda ajalugu, tuues välja olulisemad aspektid, mis on vajalikud käsitletava teema jaoks.

Raamat on mõeldud kasutamiseks Tartu Ülikooli ainete Võrgutehnoloogia I ja II õppematerjalina ning sobilik arvutiga rohkem kokkupuutunud ja teadlikumatele kasutajatele, kes soovib ennast iseseisvalt võrgutehnoloogia alaselt harida või täiendada.

Kuna eestikeelne terminoloogia ei pruugi katta kõiki valdkondi või on mõningast võõristust, siis tõlgitule on lisatud vähemalt esmakordsel esinemisel sulgudes ka kursiivis inglisekeelne termin. Raamatu lõppu on lisatud kiireks lisaks väike sõnastik.

Võimalik, et lugeja leiab mingeid vigu, ebamäärasusi või tekivad omapoolsed ettepanekud või mõtted, millest lugeja saab teada anda e-mailile laaneoks@ut.ee või kasutada anonüümset veebivormi aadressil http://math.ut.ee/~laaneoks/raamat/svt/.

Ülesehitus

Raamat on jaotatud viieks tinglikuks osaks. Raamatu esimene osa (ptk. 1-4) sisaldab eessõna, sissejuhatust, mõningate mõistete seletusi ja meeldetuletust kahendsüsteemis arvutusest, mille oskamist eeldatakse mõnede peatükkide juures. Sissejuhatuses tuuakse sisse OSI mudel, mille võtame raamistikuks järgneva osa peatükkideks jaotamisel (ptk 5-9), luues lugejale ülevaate arvutivõrgu töötamise põhialustest. Lihtsuse ja lühiduse eesmärgil käsitleme ainult Etherneti ja TCP/IP-d, millega inimestel on tavaliselt kõige enam kokkupuudet. Olles saanud esialgse ettekujutuse arvutivõrgu toimimise põhimõtetest, toome kolmandas osas sisse teisi olulisi võrgutehnoloogia mõisteid, protokolle ja teenuseid (nt. DHCP, NAT, DNS), millega puututakse kokku vahetult arvutivõrku kasutades. Neljandas osas laiendame silmaringi teiste võrgutehnoloogia protokollide ja tehnoloogiate käsitlemisega (nt. marsruutimisprotokollid, IPv6, kommutaatorite laiendused, traadita ühendused, fiiberoptika, WAN-tehnoloogiad). Seejärel vaatleme viiendas osas lühemalt mitmeid fakultatiivsemaid tehnoloogiaid (teenusekvaliteet, VPN, UPnP, SNMP). Saanud teoreetilise aluse võrgutehnoloogiast, jõuame kuuendas osas võrguseadistuse konfigureerimiseni operatsioonisüsteemides Windows ja Linux. Olles omandanud nii teoreetilise kui ka praktilise aluspõhja, vaatleme lõpuks turvalisust arvutivõrkudes. Turvalisust ei käsitleta viimasena tema ebaolulisuse tõttu, vaid seetõttu, et antud teema nõuab eelnevat arusaamist protokollidest ja nende tööpõhimõtetest.

Tänusõnad

Tahan tänada Meelis Roos'i ja Erkki Kukk'e ja Eero Vainikko't mitmete märkuste ja paranduste eest. Samuti tahan tänada mind tiivustanud ja jõudu andnuid ning EITSA't (Eesti Infotehnoloogia Sihtasutus) raamatu väljaandmise toetamise eest.

2. Sissejuhatus

Meil on olemas arvutid, millega saame sooritada mitmesuguseid tegevusi. Samas ei ole arvuti enam täisfunktsionaalne, kui ta ei ole ühendatud teiste arvutitega. Meil on vajadus suhelda, vahetada informatsiooni ja jagada ressursse. Informatsiooni vahetamisena võime vaadelda näiteks mingi kirjutise, programmi, pildi jne. ülekandmist teise arvutisse, veebilehekülgede külastamist, e-kirjade lugemist jne. Ressursside jagamisena võime vaadelda printeri, kettaruumi, kesksete teenuste jms. jagamist. Aastaid tagasi kasutati arvutist arvutisse informatsiooni transportimiseks andmekandjaid (näiteks *floppy*-ketas), kuid nii ei ole see enam mõeldav. Meile on tähtis kiirus, mugavus ja töökindlus.

Arvutivõrgu ülesanne on ühendada omavahel arvuteid või muid seadmeid, et nad saaksid edastada andmeid, jagada ressursse jms. Võrgutehnoloogia valdkond on mastaapsem, kui esmapilgul paistab, sest võrgutehnoloogial on väga palju rakendusvõimalusi, millest osadega teeme ka tutvust. Võrgutehnoloogia on tänapäeval väga kiiresti edasi arenemas.

Selleks, et edastada mingit informatsiooni (olgu selleks pilt) ühest arvutist teise, peame ühest masinast teise liigutama mingi hulga bitte, millest pilt koosneb. Saatmaks pilti ühest masinast teise, peame lahendama mitmeid esilekerkivaid probleeme, nagu näiteks:

- Kuidas toimub info vahetus füüsilisel tasemel (näiteks kasutades elektrisignaale)?
- Kuidas määrame, millis(t)e masina(te)ga soovime suhelda?
- Kuidas olla kindel, et saadetud andmed jõudsid lõpp-punkti muutumatul kujul (tulenevalt näiteks kaabliriketest)?
- Kuidas olla kindel, et teine osapool sai saadetud andmed kätte?
- Kuidas teeme vahet infoplokkidel (millal algab pilt ja millal lõpeb pilt)?
- Kuidas saame teada, kellelt antud info tuli?
- Kuidas saaksime üles seada sideseansi osapoolte vahel?
- Kuidas peaks infohulka võrgus õigesse kohta suunama ja mille baasil seda tehakse?
- Mis saab siis, kui kasutatakse erinevat riistvara ja tarkvara, kuidas nad omavahel ühilduvad?
- Kuidas anname teisele osapoolele teada, mida me temalt soovime (arvutis võib olla mitu programmi, mis võivad suhelda võrgu kaudu)?
- Ja paljud teised probleemid.

Eelnevalt tekkinud küsimuste põhjal näeme, et püstitatud probleeme ei ole sugugi vähe ning nad pole lihtsakoelised.

Saamaks kaks osapoolt omavahel suhtlema, peavad nad üksteisest aru saama. Inimestel on selleks keel, mida nad räägivad ning mille abil nad end üksteisele arusaadavaks teevad. Arvutite jaoks on selleks protokoll. **Protokoll** on hulk kokkuleppeid, mille põhjal toimub suhtlus ja üksteisest arusaamine osapoolte vahel. Näiteks füüsilise taseme protokolli poolt määratakse kasutatavate elektrisignaalide tugevuse muutumise sagedus, pingenivood, kaabli üldtakistus, kodeering jne. **Arvutivõrk** on kogum arvuteid ja muid seadmeid, mis kasutavad ühist võrguprotokolli, jagamaks ressursse läbi võrgumeediumi. Muudeks seadmeteks peale arvutite võivad olla veel printerid, võrguseadmed, terminalid jt.

Eelnevalt tekkinud küsimuste põhjal näeme, et andmete saatmine masinast masinasse on kompleksne ülesanne. Igati mõistlik on jagada algne ülesanne alamosadeks, kus alamosade funktsioonid oleksid selgesti eristuvad. Neid osasid nimetatakse **kihtideks** (*layer)*. Mõned protokollid võivad moodustada ühtse terviku, mida nimetatakse **protokollistikuks** (*protocol stack*). Parema ülevaate saamiseks pöördume põgusalt ajalukku.

1980. aastate alguses kasvasid arvutivõrgud väga kiiresti, kusjuures kasutati paljusid erinevaid tehnoloogiaid. 1980-ndate keskel tekkisid probleemid, sest erinevates asutustes kasutatud tehnoloogiad ei olnud omavahel ühilduvad. Seega ei saanud võrke omavahel ühendada. Probleemiga hakkas tegelema teiste seas rahvusvaheline standardiseerimisorganisatsioon ISO (*Inter-*

national Organization for Standardization), kes uuris erinevaid enamkasutatavaid võrgumudeleid (DECnet (*Digital Equipment Corporation network*), SNA (*Systems Network Architecture*), TCP/IP (*Transmission Control Protocol/Internet Protocol*)), leidmaks kasutatavate reeglite kogumikku kõikide võrkude jaoks. Töö tulemusena valmis OSI (*Open Systems Interconnection*) protokollistik, mida prooviti kasutusele võtta eeskätt Euroopas, kuid paljude probleemide tõttu ei saanud OSI protokollistik valitsevaks. Tänaseks on võitjana väljunud TCP/IP protokollistik. Olulisim ISO OSI protokollistiku töö tulemus tänapäeval on OSI mudel, mida kasutatakse TCP/IP protokollistiku baasil käesoleva raamatu raamistikuna.

Kahe inimese omavahelist suhtlemist võiksime vaadata kihiliselt. Kui kaks inimest, näiteks arstid, ei mõista üksteise keelt, siis nad vajavad tõlki. Kui osapooled asuvad eraldi riikides, siis on vaja kasutada telefoni. Olemegi saanud kolmekihilise mudeli. Paneme tähele, et selles lihtsas mudelis suhtlevad samad kihid omavahel. Arst suhtleb teise arstiga, tõlk suhtleb teise tõlgiga ja telefon teise telefoniga.

OSI võrgumudel avaldati 1984. aastal, see aitas luua võrke, mis oleksid ühilduvad teiste võrkudega. OSI mudelist sai peamine mudel, millega kooskõlastati tooteid ja protokolle ning mida kasutatakse võrgutehnoloogia õpetamisel. OSI mudeliga jaotatakse võrgusuhtlus seitsmesse üksteisest eristatavasse kihti.

Tänapäeval on Interneti aluseks TCP/IP protokollistik. TCP/IP protokollistiku mudel on lihtsam kui OSI mudel. TCP/IP loodi USA kaitseministeeriumi poolt, mis soovis võimalikult ekstreemsetes (ka sõja) tingimustes funktsioneerivat võrku. Näiteks võib otspunktide vahel olla mitu ühendusteed, ühendusteed peavad olema sõltumatud riistvarast. Sideliinideks võivad olla traadid, mikrolained, optiline fiiberkaabel, satelliitside jne. Samuti seati tingimuseks, et ühendused peavad olema usaldusväärsed (saatja pidi olema kindel andmete sihtpunkti kohalejõudmises). TCP/IP standardiseeriti 1981. aastal.

Võrreldes TCP/IP mudelit OSI mudeliga, võib öelda, et mõlemad on kihilised ja pakettühendusega tehnoloogiad, kuid nad erinevad kihtide arvu osas. TCP/IP mudelis on osa OSI mudeli ülemisi ja alumisi kihte koondatud üheks kihiks, mille tõttu TCP/IP mudel on lihtsam. TCP/IP mudeli transpordikihis on võimalik kasutada ebausaldusväärset ühendust UDP (*User Datagram Protocol*) protokolli näol. OSI mudelis on kokku seitse kihti, TCP/IP mudelis neli kihti. Need neli kihti on sarnased OSI mudelis olevatele, kus OSI kihi esimene ja teine kiht on TCP/IP kihis üheks kihiks. Samuti on OSI kihi seansi-, esitus- ja rakenduskiht üheks kihiks, mida nimetakse TCP/IP mudelis rakenduskihiks. TCP/IP ja OSI mudelite võrdlus on joonisel 2.1.

<table>
<tr><th>TCP/IP mudel</th><th></th><th>OSI mudel</th></tr>
<tr><td rowspan="3">Rakenduskiht</td><td></td><td>rakenduskiht</td></tr>
<tr><td></td><td>esituskiht</td></tr>
<tr><td></td><td>seansikiht</td></tr>
<tr><td>Transpordikiht</td><td></td><td>transpordikiht</td></tr>
<tr><td>Internet</td><td></td><td>võrgukiht</td></tr>
<tr><td rowspan="2">Võrgu ligipääs</td><td></td><td>kanalikiht</td></tr>
<tr><td></td><td>füüsiline kiht</td></tr>
</table>

Joonis 2.1: TCP/IP ja OSI mudelite võrdlus.

OSI mudel

OSI (*Open System Interconnection*) (ISO 7498) mudel on üldine raam-mudel, mis jagab võrgusuhtluse kihtidesse, kus kirjeldatakse nende funktsioonid üldiselt, kuid ei määratleta protokolle ega protokollistikku. Kihtidesse jaotatult on protokollistik selgepiirilisem ja modulaarsem.

OSI mudelis on oluline kihtide järjekord. Andmete saatmisel liiguvad andmed ülevalt alla ja vastuvõtmisel alt üles.

Järgnevalt vaatleme OSI mudeli kihte, alustades kõige alumisest kihist:

1. Füüsiline kiht (*physical layer*) – siin tegeldakse otseselt füüsiliste asjadega (nt. pinged, voolutugevused, füüsiline signaliseerimine, kaablid, otsikud, taktsagedus, füüsiline kodeering).
2. Kanalikiht (*data link layer*) – siin toimub peamiselt füüsiline adresseerimine ja füüsilises kihis tekkinud vigade avastamine.
3. Võrgukiht (*network layer*) – siia kuulub loogiline adresseerimine.
4. Transpordikiht (*transport layer*) – hoolitseb vookontrolliga seonduva eest ja loob usaldusväärse ühenduse osapoolte vahel.
5. Seansikiht (*session layer*) – kannab hoolt ühenduse seansside ja konversioonidega seonduva eest.
6. Esituskiht (*presentation layer*) – vastutab andmete õigesti loetavuse eest vastuvõtja poolel, formaadib andmeid ja tegeleb andmestruktuuridega, et osapooled saaksid üksteisest üheselt aru.
7. Rakenduskiht (*application layer*) – konkreetse rakendusprogrammi andmed.

Kui osapooled suhtlevad, siis OSI mudeli kihid suhtlevad omavahel. See tähendab, et neljas kiht suhtleb neljanda kihiga, teine kiht teise kihiga jne. Igal kihil on oma protokoll, mille alusel nad suhtlevad.

OSI mudelis alumine kiht pakub teenuseid ülemisele kihile. See tähendab, et OSI kihis allpool on üldisemad küsimused, üleval pool spetsiifilisemad. Võib ka öelda, et alumine kiht pakub ülemisele kihile aluspõhja. Näiteks pakub esimene ehk füüsiline kiht teoreetiliselt lõpmatut bitivoogu teisele ehk kanalikihile, mille tõttu kanalikiht saab edasi tegeleda juba bittide tasemel ega pea tegelema bittide kujutamisega, see töö on juba tema eest ära tehtud alumise kihi poolt.

Andmete saatmisel ühe programmi poolt üle võrgu teisele programmile liiguvad andmed saatjamasinas OSI mudelis ülevalt alla. See tähendab, et esituskiht saab rakenduskihi käest andmed, mille abil viib läbi oma protokollile ette nähtud protseduure (näiteks kodeerib andmed sobivale kujule), mille järel antakse täitmisjärg edasi seansikihile, mis viib läbi seansikihi protokollile vastavaid protseduure (näiteks lisab seansi identifitseerimiseks vajalikud andmed). Edasi annab seansikiht täitmisjärje edasi transpordikihile, mis vajadusel segmenteerib andmed ja lisab protokolli päise. Transpordikiht annab modifitseeritud andmed üle võrgukihile, mis lisab juurde oma päises loogilise adresseerimise jm. protokolli andmed. Edasi antakse võrgukihi poolt modifitseeritud andmed kanalikihile, mis kapseldab andmed kaadrisse, lisades füüsilise aadresseerimise jm. protokollile spetsiifilise. Seejärel antakse modifitseeritud andmed füüsilisele kihile, mis signaliseerib saadud bitid üle füüsilise meediumi. Teine osapool, kes saadetud andmed kätte saab, alustab tegevust vastupidises järjekorras (alt üles). Iga kiht saab omale vajalikud andmed, mille järel ekstraheerib andmed, mis antakse edasi ülemisele kihile.

Iga kihi jaoks on andmed koos protokolli infoga (näiteks päis) jaotatud mingi suurusega tükkideks, mis kannavad nime **protokolli andmeüksus** ehk PDU (*protocol data unit*). Joonisel 2.2 on toodud erinevate kihtide protokollide andmeüksused.

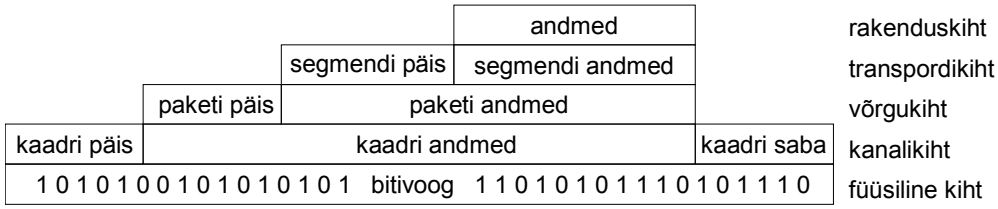
OSI mudeli rakendamise kasuteguritena võiks välja tuua:

- Keerukuse vähenemine.
- Tehnoloogiate koostalitusvõime tagamine.
- Liideste standardiseerimine.
- Kiirema arengu soodustamine.
- Tehnoloogiate modulaarse konstrueerimise soodustamine.
- Õpetamise/õppimise lihtsustumine.

Ülemise kihi protokolli andmed ja päis on alumise kihi protokollile üldiselt ainult andmeteks, millele lisatakse juurde oma päis, osade protokollide puhul ka saba. Vaata ka joonist 2.3.

rakenduskiht	andmed (*data*)
esituskiht	andmed (*data*)
seansikiht	andmed (*data*)
transpordikiht	segment
võrgukiht	pakett
kanalikiht	kaader (*frame*)
füüsiline kiht	bitt

Joonis 2.2: OSI kihtide protokollide andmeüksuste nimetused.



Joonis 2.3: Igas kihis lisatakse protokollispetsiifilised metaandmed.

Seetõttu vaatleme võrgutehnoloogiaga seonduvat lähemalt OSI mudeli kihtide kaupa. Vältimaks liialt abstraktsust ja ähmasust, käsitleme OSI kihtide juures tehnoloogiaid, mis on kõige levinumad ja millega ollakse kindlasti ka kokku puututud. Seega vaatleme OSI füüsilise ja kanalikihi juures konkreetse protokollina Etherneti (keerdpaarkaabli põhjal), võrgukihis protokolli IP (*Internet Protocol*), transpordikihi juures TCP ja UDP protokolle (ehk TCP/IP protokollistikku). Kolm kõige ülemist kihti on hägusemate piiridega ja nendega tegelevad peamiselt otspunktide rakendusprogrammid, need pole võrgutehnoloogia seisukohalt huvipakkuvad, seega vaatleme neid lühemalt. Protokollide vaatlemisel tutvume ka vastava kihi võrguseadmetega. Kui oleme OSI kihtide käsitlemisega saanud esmase aluse võrgutehnoloogia mõistmiseks, siis vaatleme veel mitmeid olulisi tehnoloogiaid, mis on samuti vajalikud, et mõista võrgutehnoloogiat terviklikumalt.

Standardiseerimine

Võrgutehnoloogia ajaloost on selgesti näha, et standardid on väga olulised, sest nad määravad ära riist- ja tarkvaralised fikseeringud, mille alusel erinevate firmade toodang ühildub. See omakorda tekitab suurema konkurentsi, alandades hindasid ja suurendades töökindlust. Standardeid haldavad üldjuhul antud valdkonnale spetsialiseerunud asutused, nendest mõned suuremad on:

- IEEE (*Institute of Electrical and Electronic Engineers*). Arendanud mitmeid kohtvõrkude standardeid. Koduleht http://standards.ieee.org.
- IEC (*International Electrotechnical Commission*). Elektroonikale ja elektrilistele tehnoloogiatele keskendunud standardiseerimisorganisatsioon. Koduleht http://www. iec.ch.
- ISO (*International Organization for Standardization*). Annab välja mitmeid tehnilisi soovitusi. Koduleht http://www.iso.ch.
- EIA/TIA (*Electronics Industry Alliance/Telecommunications Industry Association*). Tegeleb telekommunikatsioonistandardite kehtestamisega (nt. fiiberoptika, traadita ühenduste ja telefoniseadmete standarditega). Koduleht http://www.tiaonline.org.
- ITU (*International Telecommunications Union*). Raadio ja telekommunikatsiooni stan-

dardiseerimisorganisatsioon. Koduleht http://www.itu.int

- ANSI (*American National Standards Institute*). Tegeleb väga laia standardiseerimise valdkonnaga.
- IETF (*Internet Engineering Task Force*). Arendab ja spetsifitseerib internetistandardeid (näiteks: IP, TCP, UDP, OSPF, RIP, ISIS, QoS, MPLS jne). Koduleht http://www.ietf.org.

Veel mõned märkused

Kuna IETF standardid on vabalt ja mugavalt kättesaadavad, siis on lisatud paljude teemade juures sulgudesse viide IETF vastavale RFC (*Request for Comments*) dokumendile. Näiteks "(RFC 1234)". Soovi korral võib neid dokumente alla tõmmata aadressilt "http://www.ietf.org/rfc/rfc<dokumendi nr>.txt". Näiteks RFC 1234 asub aadressil http://www.ietf.org/rfc/rfc1234.txt. HTML versioon on saadaval aadressil "http://tools.ietf.org/html/rfc<dokumendi nr>". Mõningaid viiteid on ka ITU soovitustele, mida võib leida aadressilt http://www.itu.int/rec/T-REC/en.

Paljude protokollide puhul on toodud väljade kirjeldused ja viimasena ka välja pikkus. Väljad on loendis toodud tegelikus järjekorras.

3. Mõisted

autentimine – protsess, mille käigus kasutaja tõendab oma identiteeti.

graaf – kogumik tippe ja neid ühendavaid servi. Võrgutehnoloogias võime graafi tippudena vaadelda marsruutereid, arvuteid, kommutaatoreid ja muid masinaid. Graafi servadena aga sideliine. Graafi serva nimetatakse suunatuks, kui serva saab läbida ainult ühte pidi (võrgutehnoloogias on tegemist simpleks-sideliiniga). Graafi nimetatakse tsükliliseks, kui leidub erinevaid tippe läbiv teekond mingist tipust iseendasse. Graafi nimetakse täielikult sidusaks, kui igast punktist on võimalik liikuda kõikidesse graafi tippudesse. Graafi nimetatakse puuks, kui ta on täielikult sidus ja ei sisalda mitte ühtegi tsüklit.

internet – väikese algustähega internetiks nimetatakse suvalist kinnist ühendatud võrkude kogumit. Võib öelda ka, et internet on võrgustik ehk võrkude võrk.

Internet – suure algustähega Internetiks nimetatakse maailma kõige suuremat internetti, mida meist enamus kasutab.

kontrollkood ka kontrollsumma (*redundancy check*) – andmete saatmisel võib neis tekkida vigu (näiteks sideliini häirete tõttu võib saaja või vahendav võrguseade mõnda bitti valesti tõlgendada). Seetõttu on oluline kontrolli võimalus. Selleks on võimalus andmete juurde lisada kontrollkoodina lisabitte, mille põhjal saaks mingi tõenäosusega kindlaks teha, kas andmed on vastuvõtul samad, mis algselt saadetud. Kontrollkood peaks olema võimalikult väikesemahuline. Kontrollkood on mõeldud tehniliselt toimunud vigade avastamiseks, kuid see ei paku kaitset ründajate vastu. Kontrollkoodi arvutamiseks on kolm põhilist moodust: paarsuskontroll, CRC ja kontrollsumma (*checksum*).

Paarsuskontroll – loetakse kokku andmeühikus olevate bittide "1" arv. Kui bitte "1" oli paarisarv, siis paarsuskontrolli bitiks on "0", paaritu arvu "1" bittide korral on paarsuskontrolli bitiks "1". Selliselt saab avastada paaritu arvu bittide vigu. Oletame, et tahame saata andmeid, mille andmeühiku pikkus paarsuskontrolli bittide jaoks on 7 bitti ja paarsusbitt lisatakse kahekssandaks bitiks. Olgu saadetavad andmed: 1101100 0100000 0010010 1101100 1101000, siis peale paarsusbittide lisamist oleks tulemus järgmine: 1101100<u>0</u> 0100000<u>1</u> 0010010<u>0</u> 1101100<u>0</u> 1101000<u>1</u> (paarsusbittidel on joon all).

Kahemõõtmeline paarsuskontroll – täiendatud paarsuskontroll, mille puhul andmeühikud pannakse kahemõõtmelisse massiivi, kus paarsus leitakse nii ridade kui veergude põhjal. Olgu meil lähteandmeteks 1101100 0010010 1101100 0100000 1101000. Paigutame väiksemad andmeühikud ridade kaupa massiivi ja leiame nii veergude kui ridade paarsusbitid, samuti ridade paarsusbittide paarsusbiti. Seega oleks massiiv järgmine:

1101100<u>0</u>
0100000<u>1</u>
0010010<u>0</u>
1101100<u>0</u>
1101000<u>1</u>
<u>10110100</u>

Andmeteks, mis teele saadetakse, on tekkinud massiiv ridade kaupa koos paarsusbittidega, seega on väljundbitid järgmised: 11011000 01000001 00100100 11011000 11010001 10110100.

CRC (*cyclic redundancy check*) on võimas ja paindlik vigade avastamise meetod. CRC koostab polünoomi, kus termi kordajateks on andmete bitid. Olgu edastatavateks bittideks näiteks "10111101", siis polünoomiks on $1x^7+0x^6+1x^5+1x^4+1x^3+1x^2+0x^1+1x^0$. Seejärel see polünoom jagatakse läbi kindla generaatorpolünoomiga. Selle jagamise jääk ongi CRC kontrollsumma. Erinevate CRC variantide generaatorpolünoome:

- CRC-16 – 16 bitine. Generaatorpolünoomiks on $x^{16}+x^{15}+x^2+1$. Kasutavad näiteks HDLC (*High-Level Data Link Control*), USB (*Universal Serial Bus*).
- CRC-CCITT – 16 bitine. Generaatorpolünoomiks on $x^{16}+x^{12}+x^5+1$. Kasutavad näiteks PPP, Bluetooth.
- CRC-32 – Generaatorpolünoomiks on $x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x$. Kasutab näiteks Ethernet.

CRC on riistvaras palju lihtsam realiseerida kui tarkvaras.

Kontrollsumma. Kontrollsumma leidmisel jagatakse andmed fikseeritud pikkusega osadeks, mis liidetakse arvudena kokku. Saadud tulemuse number lahutatakse jällegi eelnevalt fikseeritud pikkusega osadeks ja liidetakse kokku. Saadud tulemusest võetakse binaarne eitus, mis ongi kontrollsumma väärtus.

Kontrollsumma kontrollimiseks jagatakse jällegi andmed eelnevalt fikseeritud pikkusega osadeks ja liidetakse omavahel arvudena kokku koos kontrollsumma osaga. Saadud tulemus jagatakse jällegi eelnevalt fikseeritud pikkusega osadeks ja liidetakse omavahel kokku ja kui saadud tulemuse binaarne eitus on null, siis kontrollsumma klappis, vastasel korral mitte.

Teeme protsessi läbi kuueteistkümne biti pikkuse kontrollsumma näite varal (seda kasutab näiteks IP protokoll). Lühiduse ja lihtsuse mõttes leiame kaheksa baidi jagu andmetele kontrollsumma. Olgu nendeks andmeteks, millele arvutame kontrollsumma, kuueteistkümnendsüsteemis "71 38 EE F3 0D 12 2B C5". Jaotame andmed 16-bitisteks osadeks: "7138 EEF3 0D12 2BC5" ja liidame nad kokku: 7138+EEF3+0D12+2BC5="0001 9902". Liidame saadud tulemuse osad omavahel kokku: 0001+9902=9903. Leiame 9903 binaarse eituse (~1001 1001 0000 0011=0110 0110 1111 1100), milleks on 66FC, mis ongi kontrollsumma.

Kontrolliks jagame andmed eelnevalt fikseeritud pikkusega osadeks ja liidame nad kokku koos kontrollsummaga: 7138+EEF3+0D12+2BC5+66FC="0001 FFFE". Liidame saadud tulemuse osad omavahel kokku: 0001+FFFE=FFFF. Saadud tulemuse binaarne eitus on tõesti null, seega kontroll oli edukas.

konvergentsus – olukord, kus kõikidel osapooltel on ühesugune arusaam (nt. võrgu topoloogiast).

lipp – tähistab mingit konkreetset bitti. Lipp on püsti, kui biti väärtus on 1, ja lipp on maas, kui biti väärtus on 0.

mastabeeritavus (*scalability*) – süsteemi laienemisvõimelisuse omadus. Süsteem, mida ei ole võimalik laiendada, ei ole mastabeeritav. Mastabeeritavust ei saa üheselt mõõta. Hea näide mastabeeritavuse olulisusest on näiteks asutuse struktuuri ja käsuliini ülesehitus. Oletame, et meil on olemas mingi väiksem asutus mingi struktuuriga. Asutus hakkab aja jooksul kasvama (näiteks firma on edukas ning suurendab ja laiendab tootmist), siis vajatakse uusi inimesi jne. Mida vähem on selles protsessis asutuse struktuurist lähtuvaid probleeme, seda mastabeeritavam on asutuse ülesehitus. Samamoodi on laiendatavusega arvutivõrgus. Näiteks ei pruugi võrguseadmed toime tulla liiklusmahu kasvuga, kasutatavad protokollid ei pruugi olla ühilduvad uute tehnoloogiatega, tekivad uued nõudmised või saavad nüüd oluliseks (näiteks dubleeritud ühendused, varukoopiate tsentraliseeritus).

oktett – spetsifikatsioonides võidakse rääkida terminist oktett. Oktett tähendab kaheksat ja väljade suurustes on ta ekvivalentne baidiga (üks bait on kaheksa bitti).

protokoll (*protocol*) – protokoll on kokkulepitud reeglistik, mille alusel osapooled omavahel suhtlevad. Fikseeritakse näiteks, mida mingi bitt või bittide hulk tähendab. Näiteks võib bitivoos tähendada "01111110" seda, et tegemist on kaadri algusega, näiteks 17.-24. bitid tähendavad lähteaadressi, 25.-32. bitid tähendavad teate tüüpi, kuidas vastavalt protokolli reeglite alusel käitutakse. Igal andmeühikul on olemas oma bittide tähendused. Lisaks sellele võib olla teisi keerulisemaid ja komplekssemaid kokkuleppeid. Teisisõnu, protokoll paneb paika teadete formaadi ja viisi, kuidas osapooled peavad vahetama sõnumeid soovitud tegevuste jaoks (mida protokoll võimaldab). Osapoolteks võivad olla arvutid, võrguseadmed, printerid ja muud masi-

nad, mis toetavad vastavat protokolli. Protokolli võiks nimetada ka osapoolte vaheliseks keeleks ja seadustikuks, mille abil nad suhtlevad ja mida järgivad. Protokollideks on näiteks IP (*Internet Protocol*), TCP (*Transmission Control Protocol*), HTTP (*Hypertext Transfer Protocol*).

protokollistik (*protocol stack, protocol suite*) – protokollide kogumik, mis on omavahel seotud.

ringiratast ehk **round-robin** – lihtne algoritm mingi ressursi jaotamiseks mitme soovija vahel. Round-robini puhul pannakse paika soovijate järjekord, mille alusel kasutatakse ressurssi (näiteks kasutada sideliini mingi aja jooksul). Kui soovija on vastava, määratud koguse ressurssi (näiteks aega) saanud, siis ta läheb järjekorras viimaseks. Seega on ressursi saamine tsükliline.

räsi, **räsifunktsioon** (*hash*) – funktsioon, mis saab sisendina suvalise stringi ja väljastab fikseeritud pikkusega stringi ehk räsi. Räsist ei ole võimalik algset räsifunktsiooni sisendit lihtsalt leida. Mitu räsifunktsiooni sisendit võib teisenduda samaks räsiks (sest sisendhulk on suurema võimsusega kui väljundhulk). Räsifunktsioonid võimaldavad hoida algset parooli salastatult. Nimelt parooli salvestamisel salvestatakse parooli räsi ja autentimisel räsitakse sama räsifunktsiooniga kasutaja poolt pakutud parool. Kui räsid langevad kokku, siis väga suure tõenäosusega on tegu õige parooliga.

skaleeritavus – vt. mastabeeritavus.

volitamine (*authorization*) – autenditud kasutaja õiguste kontroll mingi operatsiooni läbiviimiseks.

4. Kahendarvutus

Kuna arvutid ja võrgutehnoloogia kasutavad kahendsüsteemi, siis peatume natuke ka kahendsüsteemil ja -arvutusel.

Kahendsüsteem on sarnane meil igapäevaselt kasutatavale kümnendsüsteemile, ainult ühe erinevusega – kahendsüsteemis on ainult kaks numbrit kümne numbri asemel. Muu on üldjoontes sama, mõned operatsioonid on isegi lihtsamad. Järgnevalt vaatlemegi lähemalt erinevaid tegevusi, mida meil vaja läheb.

Kahendarvu teisendamine kümnendsüsteemi

Skeem on selline, et järjestame kahendarvu tagantpoolt alates nullist. Liidame kõik arvud, kus a tähistagu arvu, n tähistagu kahendarvu pikkust ja a_i arvu a i-ndat liiget (tagant poolt). Siis on arvu väärtuseks:

$$\sum_{i=0}^{n-1} 2^i \cdot a_i$$

Näidisena võtame kahendarvu 1001110001, mille teisendame kümnendsüsteemi arvuks. Summa oleks järgmine: $2^0 \cdot 1+2^1 \cdot 0+2^2 \cdot 0+2^3 \cdot 0+2^4 \cdot 1+2^5 \cdot 1+2^6 \cdot 1+2^7 \cdot 0+2^8 \cdot 0+2^9 \cdot 1 = 2^0+2^4+2^5+2^6+2^9 = 1+16+32+64+512=625$.

Kuna enamasti läheb vaja kuni kümnekohalisi kahendarve, siis on kasulik meeles pidada kahe astmeid (mida on lihtne ka uuesti genereerida): $2^0=1$ $2^1=2$ $2^2=4$ $2^3=8$ $2^4=16$ $2^5=32$ $2^6=64$ $2^7=128$ $2^8=256$ $2^9=512$ $2^{10}=1024$.

Kümnendsüsteemi arvu teisendamine kahendsüsteemi

Kümnendsüsteemi arvu teisendamiseks kahendsüsteemi on vaja arv jagada kahega, jätta meelde tekkiv jääk ning jagatis uuesti jagada kahega. Jagamist tuleb jätkata, kuni jõutakse seisu, kus jagatakse nulli. Viimane operatsioon, mis tehakse, on ühe jagamine kahega, mis annab vastuseks nulli, jäägiga üks.

Näitena teisendame kümnendsüsteemi arvu 789 kahendsüsteemi: 789/2=394(1) 394/2=197(0) 197/2=98(1) 98/2=49(0) 49/2=24(1) 24/2=12(0) 12/2=6(0) 6/2=3(0) 3/2=1(1) 1/2=0(1).

Tulemuseks saame jääke tagant poolt lugema hakates 1100010101 (sulgudes olevad numbrid).

Alternatiivne variant kahendsüsteemi arvu leidmiseks on järgmine (olgu sisendvahemikuks 0-255):

1. Olgu kõik väljundväärtuse bittide väärtused nullid (eelduste kohaselt kaheksa nulli).
2. Leida suurim kahe aste, mis on väiksem-võrdne etteantud kümnendsüsteemi arvust. Olgu selleks astendajaks `x`. Kui etteantud arvu väärtus on null, siis lõpetame.
3. Kirjutada kahendsüsteemi `x`+1 arvu positsioonile (pööratud järjekorras) üks.
4. Lahutada etteantud arvust maha 2^x, mille väärtus olgu `y`.
5. Määrame etteantud arvu väärtuseks `y` ja kordame punkti 2.

Toome näitena kümnendsüsteemi arvu 134 teisendamise kahendsüsteemi. Väärtustame väljundbitid: "00000000". Leiame suurima kahe astme, mis mahub 134 sisse, selleks on $2^7=128$. Muudame väljundbiti 7+1=8 väärtuse üheks, seega "10000000". Uueks sisendväärtuseks saab 134-128=6. Suurim kahe aste, mis on väiksem-võrdne kuuega, on kaks. Seega väljundbittide väärtuseks saab "10000100". Leiame uue sisendväärtuse: $6-2^2=2$. Suurim kahe aste on üks. Seega väljundbittide väärtuseks saab "10000110". Uueks sisendväärtuseks saab $2-2^1=0$. Kuna sisendväärtus on null, siis tagastame väljundbitid "10000110".

Kuueteistkümnendsüsteemi ja kahendsüsteemi vaheline teisendus.

Kuueteistkümnendsüsteemi arvud koosnevad numbritest 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. Kuna kuusteist on kahe naturaalarvuline aste (2^4=16), siis on teisendamine väga lihtne – ühele kuueteistkümnendsüsteemi üksikule numbrile vastab neli kahendsüsteemi üksikut numbrit. Vastavused on järgmised: 0=0000 1=0001 2=0010 3=0011 4=0100 5=0101 6=0110 7=0111 8=1000 9=1001 A=1010 B=1011 C=1100 D=1101 E=1110 F=1111

Teisendame näitena kuuteistkümnendsüsteemi arvu 78D14F504 kahendsüsteemi arvuks: 78D14F504=01111000110100010100111101010000010 0

Kahendarvu teisendamisel hakkame asendamist tegema arvu tagumisest otsast või kirjutame mõtteliselt juurde piisavalt palju nulle, et kahendsüsteemi arvu pikkus jaguks neljaga. Teisendame näiteks kahendsüsteemi arvu 000110010011111011110100000010111011000111 00 = 193EF40BB1C.

Analoogselt on teostatavad ka teiste arvusüsteemide teisendused kahendsüsteemi vahel, mille aluseks on kahe naturaalarvulised astmed (näiteks 4, 8, 32, 64 jne).

Loogiline korrutamine

Mitmes kohas on vaja loogilist korrutamist. Loogilise korrutamise juures on vaja teada, et 1·1=1; 1·0=0; 0·1=0; 0·0=0. Kui meil on vaja kaks arvu loogiliselt korrutada, siis võtame arvud kahendkujul ja paigutame üksteise alla. Kui üks arvudest on kahendkujul teisest arvust pikem, siis arvestame, et lühema arvu puudu olevad kohad on nullid (näiteks: 10110 on sama mis 010110 ja 0010110). Kahendarvud võime kirjutada üksteise alla kohakuti ja kohakuti jäänud kohtades korrutame numbrid (1 ja 0) kokku.

Võtame näidisena arvud 1001011111101 ja 110100011111101110. Viime üksteise alla ja korrutame kokku (üksteise kohal olevad nullid ja ühed).

Esimene arv:	000001001011111101
Teine arv:	110100011111101110
Vastus:	000000001011101100

Binaarne eitus

Binaarse eituse puhul kirjutatakse nulli asemele üks ja ühe asemele null. Olgu näiteks bitid "1000110010101100". Binaarne eitus eeltoodud bittidele on "0111001101010011".

5. OSI füüsiline kiht

OSI füüsiline kiht pakub ülemisele kihile liidest, mis võimaldab tegeleda ainult bittide saatmise ja vastuvõtmisega võrgust. Teisiti öeldes pakutakse bitivoogu ülemisele kihile (kanalikihile). Kõik vajalik selle realiseerimiseks tehakse OSI füüsilises kihis. See tähendab, et saatmisel peab tõlkima bitid elektrilisteks, elektromagnetilisteks või optilisteks signaalideks, mille teised sideliini otsad peavad suutma uuesti samadeks bittideks tõlkida. Tänu OSI füüsilisele kihile ei pea ülemised kihid otseselt tegelema füüsikaliste probleemidega.

Arvutivõrgu füüsilise taseme komponendid võib jagada kolmeks:

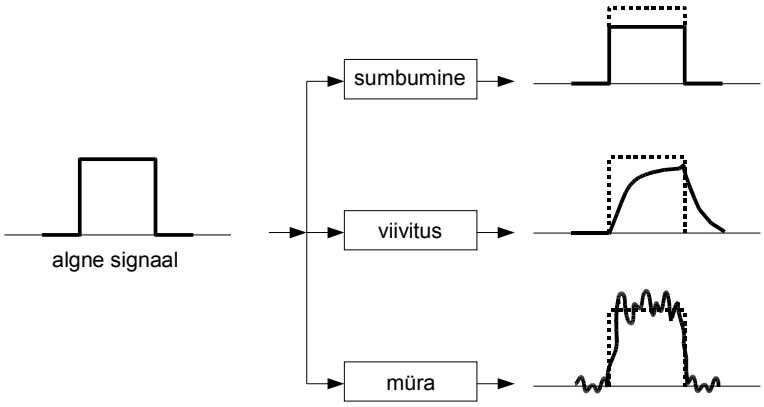
- Sobiv meedium andmete edastamiseks (kaablid, elektromagnetlained jms). Nimetame seda edaspidi sideliiniks.
- Liides arvuti ja sideliini vahel, mis saadaks ja võtaks vastu infot. Arvutis täidab seda ülesannet võrgukaart. Seda kaarti nimetatakse võrguliidese kaardiks ehk NIC (*network interface card*).
- Võrguseadmed, mis suunavad ja võimendavad signaale teekonnal ühest sideliini otspunktist edasi ühte või mitmesse otspunkti.

Antud peatükis vaatleme elektrilist signaliseerimist, kuna see on kõige kasutatavam ja odavam. Elektrijuhiks kasutatakse tavaliselt vaskkaablit, sest normaaltingimustes on vask väikseima eritakistusega peale hõbedat.

Kaabli kui edastava meediumi ülesanne on toimetada saatja signaal ühest otspunktist teise ideaalsel juhul muutumatul kujul. Paraku ei ole praktikas võimalik ideaaljuhtu saavutada. Elektrijuhi kasutamisel andmeedastajana on olulisemateks parameetriteks, mis määravad edastatava signaali kvaliteedi, mahtuvus, üldtakistus ja sumbuvus. Mahtuvus on omadus, mis näitab, kui palju elektrijuht salvestab elektrilaengut. Laengut salvestades muutuvad signaalide sakid ümaramaks. Mida kvaliteetsem kaabel, seda väiksem on mahtuvus. Mahtuvus sõltub ka elektrijuhi kujust, mõõtmetest, lähedal asuvatest elektrijuhtidest ja ümbritsevast dielektrikust. Liiga suure mahtuvuse ja pika kaabli korral muutub signaal vastuvõtjale arusaadamatuks (signaali seisundite erinevused muutuvad liialt väikesteks). Mahtuvuse mõõtühikuks on F (farad). Üldtakistuse mõõtühikuks on Ω (oom). Sumbuvus on signaali tugevuse nõrgenemine. Mida suurem kaabli pikkus või signaali sagedus, seda suurem sumbuvus. Mõõtühikuna kasutatakse detsibelli 100 meetri kohta (mida väiksem, seda paremate omadustega). Joonisel 5.1 on toodud signaali muutumine kaablit läbides.

Kahe otspunkti vahelise kaabli pikkus on piiratud, sest elektrisignaal, mida selle kaudu edastatakse, nõrgeneb ja muutub kaablit läbides lamedamaks. Mida suurem sagedus, seda kiiremini see juhtub. Lisaks sumbumisele raskendab signaalist arusaamist või teeb signaalist arusaamise vastuvõtjale võimatuks müra. Müra on soovimatu ja kõrvaline signaal meediumis. Seda leidub kahel kujul – ümbritsev (kutsutakse ka termiline) ja impulsi müra. Ümbritsev müra on alati olemas, see on tekitatud edastus- ja vastuvõtuseadmetest. Samuti võib müra olla põhjustatud välise allika fluorestseeruvatest valgusmuundajatest, elektrilistest vahenditest, kuumusest ja ümbritsevast kiirgusest. Ümbritseva müra tõttu võib vastuvõtjal olla probleeme bittide eristamisega. Impulsimüra koosneb katkendlikest ja soovimatutest signaalidest, mis on välise allikaga, nagu näiteks elektrimootorid, koopiamasinad, äike jms.

Mürad võime jaotada välimisteks ja sisemisteks. Välised mürad võivad olla põhjustatud nii tehnoloogiatest (mobiiltelefonid, mikrolaineahjud, elektrimootorid jms) kui ka loodusest lähtuvad (näiteks päikese magnettormid). Müra ei saa täielikult elimineerida. Me saame ainult müra eest osalist kaitset pakkuda. Müra allikateks võivad olla näiteks lähedal olevate sidekaablite signaalid, elektromagnetlained, lasermüra saatjas või vastuvõtjas (optilise signaali korral).



Joonis 5.1: Algse signaali muutumine sumbuvuse, viivituse ja müra puhul. Algne signaal on joonisel katkendjoonega.

Elektriimpulsi sumbumisel ja müra tekkimisel on olulisemateks kaablisisesteks põhjusteks:

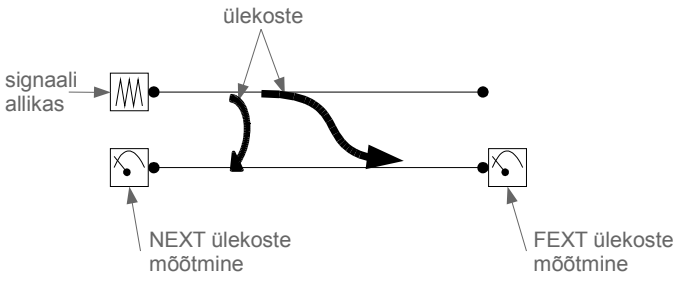
- Elektrisignaal muundub kaablis soojuseks. Mida parem on elektrijuht, seda väiksem on takistus ning seda vähem elekter muundub soojuseks elektrijuhti läbides. Mida kuumem on elektrijuht, seda suurem takistus. Ükski elektrijuht pole ideaalne. Parimad elektrijuhid paremuse järjekorras on hõbe, vask ja alumiinium. Vask on üldiselt kõige kasutatavam, sest hõbe on liialt kallis ja looduses leidub seda vasega võrreldes palju vähem.
- Elektrisignaali energia lekib mingil määral läbi traati ümbritseva dielektriku. Dielektriku ülesanne on olla võimalikult suure takistusega, et elekter ei suudaks läbida antud materjali. Paraku ei ole olemas ka ideaalseid dielektrikuid.
- Ülekoste (*crosstalk*). Kui kasutatakse kaabeldust, mis sisaldab mitut elektrijuhti või kaablid asuvad lähestikku, on võimalik, et ühe elektrijuhi signaal kandub üle teisele elektrijuhile elektromagnetilise välja energia näol. Voolu muutumine tekitab elektromagnetvälja, mis kiirgub välja elektrijuhist nagu raadiosaatjast. Ümbritsevad elektrijuhid püüavad nagu antennid kinni selle signaali, mille energia mõjutab vastuvõtvat elektrijuhti, tekitades seal häireid. Need häired võivad põhjustada vastuvõtjale probleeme, mille tõttu ei pruugi see signaali õigesti tõlgendada. Mida kõrgemaid sagedusi kasutatakse andmete edastamiseks, seda hävitavam on ülekoste mõju.
- Vigastest või vigaselt paigaldatud otsikutest põhjustatud üldtakistus. Halvasti paigaldatud otsikute tõttu suureneb otsikutes üldtakistus, mille tõttu osa elektrisignaali energiast peegeldub tagasi ja hakkab kaablis edasi-tagasi põrkuma (sest kaabli takistus on väiksem otsiku omast). See aga teeb vastuvõtjale niigi nõrgenenud signaalist arusaamise veelgi keerulisemaks, see omakorda võib tekitada vigu bittidest arusaamisel (eriti järgnevate signaalide lugemisel). Seda nimetatakse värinaks.

Ülekosted jagatakse omakorda veel alamliikideks:

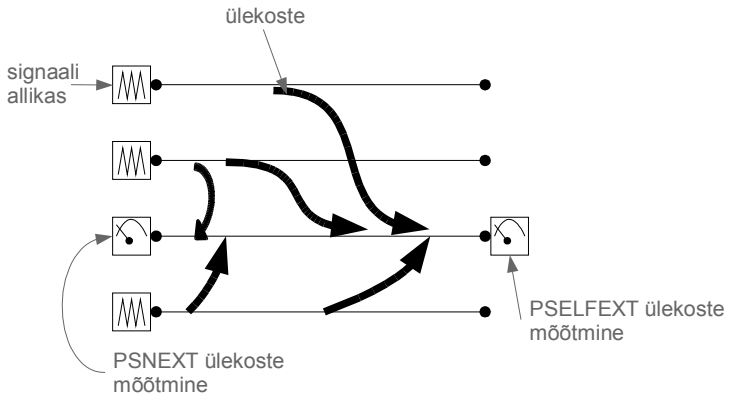
- NEXT (*near-end crosstalk*) – signaali edastamisel kandub ülekoste tõttu traadi alguses osa signaalienergiat üle lähedal olevasse traati, mis on signaali edastaja jaoks vastuvõtu traat. Vaata joonist 5.2. NEXT on üks kõige problemaatilisemaid ülekoste liike, sest signaal on alguses tugev ja seetõttu on võimalik ülekoste suurem. Samas on saadav signaal sumbuvuse jms. tõttu nõrgenenud, mille tõttu võib ülekoste olla piisavalt tugev vale diskreetse lugemi saamiseks. NEXT ülekostet on raske tuvastada.
- FEXT (*far-end crosstalk*) – signaali ülekostet mõõdetakse kõrvaltraadis teisel pool traadi otsa. Toimub kaugemal saatjast. FEXT ei ole nii tõsine probleem kui NEXT, sest eemal on signaal juba nõrgenenud. FEXT on muutunud viimasel ajal olulisemaks teguriks kui ta varem oli, eeskätt üle kahe traadipaari kasutamisel. Vaata joonist 5.2.

- PSNEXT (*power sum near-end crosstalk*) – mitmelt traadilt toimub NEXT ülekoste mingile ühele traadile. Vaata joonist 5.3.
- ELFEXT (*equal level FEXT*) – ELFEXT on erinevus FEXT-i ja juhtmepaari insertsioonikao vahel, mille signaali segab FEXT. Vaata ka joonist 5.4.
- PSELFEXT (*power sum equal level far-end cross-talk*) – mitme traadi ELFEXT ülekoste mingile traadile. Vaata joonist 5.3.

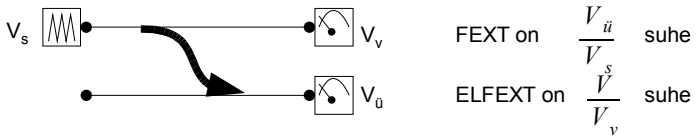
Järgnevalt mõningaid näiteid ülekostete kohta graafiliselt kujutatuna.



Joonis 5.2: NEXT ja FEXT ülekostete mõõtmine.



Joonis 5.3: PSNEXT ja PSELFEXT ülekostete mõõtmine.



Joonis 5.4: FEXT ja ELFEXT mõõtmise erinevus. Kehtib ka seos ELFEXT = FEXT - sumbuvus.

Müra, mis mõjutab kõiki ülekandeks kasutatavaid sagedusi võrdselt, nimetatakse valgeks müraks. Müra, mis mõjutab ainult kitsast riba, nimetatakse kitsasribahäireks (*narrowband interference*).

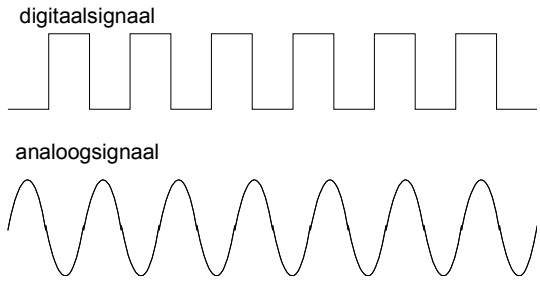
Andmete edastuse järgi sideliinis võib andmeedastuse jagada kolmeks:

- Simpleks-edastus (*simplex transmission*) – üks pool on ainult andmete edastaja ja teine ainult vastuvõtja. Tavaelust on näiteks telemast ja televiisor.

- Poolduppleks-edastus (*half-duplex transmission*) – kõik saavad andmeid edastada, kuid korraga ainult üks osapool. Näiteks inimesed on raadiosaatjatega mingis metsas, samal sagedusel (nad ei saa korraga rääkida omavahel, sest siis ei saa mitte keegi lausest aru).
- Täisduppleks-edastus (*full-duplex transmission*) – mõlemad osapooled saavad andmeid edastada, ka samal ajal mõlemas suunas (saata ja vastu võtta samal ajal). Põhimõtteliselt võiks vaadelda ka kahe simpleks-kanalina, kus üks on ühes suunas ja teine teises suunas.

Vastavalt kaablile saab sideliinis olla kahte liiki edastusviise: jadaedastus (*serial transmission*) ja rööpedastus (*parallel transmission*). Jadaedastus (ka järjestikedastus) on edastusviis, kus bitid edastatakse järjekorras ükshaaval teisele osapoolele üle ühe suhtluskanali. Rööpedastuse puhul saadetakse bitte mitme kanali kaudu paralleelselt. Seega, kui rööpedastuse sideliinil on näiteks 10 kanalit ja üks kanal on sama kiire kui jadaedastuse sideliin, siis rööpedastus on 10 korda suurema läbilaskvusega. Samas on rööpedastusel omad probleemid: vajatakse palju keerukamat sideliini ja liideseid; ülekostete võimalus piirab sideliini pikkust.

Traadis edastatavad signaalid võib jagada kahte liiki: analoog- ja digitaalsignaalina edastamine (vt. joonis 5.5). Analoogsignaali ribalaiust mõõdetakse tavaliselt hertsides (mitu võnget sekundis). Digitaalsignaali ribalaiust mõõdetakse selle kaudu, kui palju informatsiooni (bitte) saab vaadeldavas ajaühikus (tavaliselt sekundis) edastada. Mida suuremat sagedust kasutada, seda suuremat läbilaskevõimet on võimalik saavutada. Digitaalsignaali on keerulisem edastada, kuid samas on see palju töökindlam andmete edastusel, sisaldades vähem müra ja moonutusi kui analoogühendus. Digitaalsignaal on diskreetne. See tähendab, et digitaalsignaali puhul on olemas kindlad seisundid ja puuduvad vahepealsed olekud. Edastamisel kaablis signaal neeldub, hajub, mille tõttu jõuab vastuvõtjani nõrgenenud signaal. Teatavas pingenivoo vahemikus arvestab vastuvõtja digitaalsignaali vastavaks bitiks. Analoogedastuse puhul on läbilaskevõime piirajaks teoreem, mida tuntakse Shannoni teoreemina (ka limiidina) (vaata valemit 5.1). Seetõttu on sissehelistamisteenuse analoogside piiriks 33,6 Kbit/s (56 Kbit/s vajab digitaalsidet).



Joonis 5.5: Näide digitaal- ja analoogsignaalist.

$$MDR = H \cdot \log_2\left(1 + \left(\frac{S}{N}\right)\right)$$

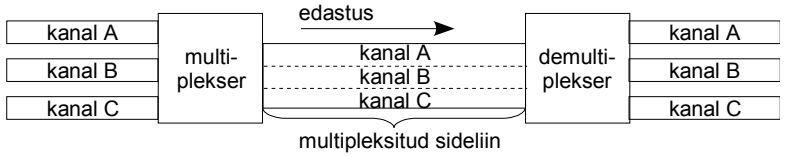
MDR – maksimaalne läbilase (bit/s)
H – ribalaius (Hz)
S – signaali võimsuse nivoo (dB (detsibell))
N – müra võimsuse nivoo (dB)

Valem 5.1: Shannoni limiidi leidmise valem.

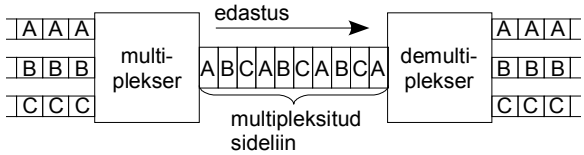
Reaalses elus on tihti vaja kahe punkti ühendamiseks mitut iseseisvat sidekanalit, kuid iga ühenduse jaoks eraldi sideliini paigaldamine oleks väga kallis. Selle asemel võiks kasutada ühte suurema läbilaskevõimega sideliini ja siis saaksid erinevad ühendused üksteist kuidagi segamata sama sideliini kasutada. Sellist ühe suure läbilaskevõimega kaabli jagamist mitme ühenduse vahel nimetatakse multipleksimiseks ja selle teostajat multiplekseriks (vastuvõtjat nime-

tame demultiplekseriks; lähtume edaspidi sellest, et multiplekser sisaldab ka demultiplekserit). Multipleksimise viise on mitmeid erinevaid. Nendest kasutatavamad on:

- **Aegmultipleksimine** ehk **TDM** (*time-division multiplexing*) – igale ühendatud sõlmele määratakse mingi hulk aega ehk pilu (*slot*). Igal juhul saadetakse eraldatud pilu jooksul bitid teele (ka siis, kui pilu omanik ei soovi mitte midagi saata). Seega on TDM-multipleksitud sideliin jagatud võrdselt kõigi osapoolte vahel. Kasutatakse näiteks GSM (*Global System for Mobile Communications*), SDH (*Synchronous Digital Hierarchy*), ISDN (*Integrated Services Digital Network*), SONET (*Synchronous optical networking*) puhul. TDM on oma olemuselt järjestikuline edastusviis. TDM on OSI ülemiste kihtide protokollidest sõltumatu.
- **Sagedusmultipleksimine** ehk **FDM** (*frequency division multiplexing*) – erinevad signaalid edastatakse erinevates sagedusvahemikes. Kasutatakse näiteks telefonivõrkudes, telekanalite puhul (erinevad telekanalid kasutavad erinevaid sagedusvahemikke, kus heli ja pilt omakorda eraldi sagedusvahemikes). FDM-edastus on oma olemuselt paralleelne edastus.
- **WDM** (*wavelength division multiplexing*) – põhimõtteliselt analoogne FDM multipleksimisele, mida kasutatakse optilises fiibris edastamisel (FDM alla loetakse raadiolainete elektromagnetkiirguse sagedusvahemikud). Fiiberoptilises kaablis kasutatakse erinevaid lainepikkusi (ehk erinevat värvi valgusi) erinevate kanalite signaalide edastamiseks.
- **Statistiline multipleksimine** – sarnane TDM-le, kuid ajapilusid jagatakse ainult neile, kellel on vajadus, ning pilude suurus on muutuv, näiteks saadetava kaadri suurune. Tihti toimub edastus FIFO (*first in, first out*) põhimõttel (esimesena saadud andmehulk edastatakse esimesena) või kasutatakse teisi järjekorra määramise võimalusi (ringiratast, juhuslik või mingi muu järjekorra arvestamine). Saatmise otsustamine tehakse kanalikihis või kõrgemal. Otsustamisel eraldatakse ajapilusid ainult neile sõlmedele, mis vajavad andmete saatmist.



Joonis 5.6: FDM multiplekser. Erinevad kanalid asuvad erinevatel sagedustel ja erinevate kanalite andmeid edastatakse paralleelselt.



Joonis 5.7: TDM multiplekser. Erinevad kanalid jagavad omavahel multipleksitud sideliini aega.

Lihtsalt bittide saatmisel sideliini ei pruugi vastuvõtja aru saada, millal mingi takt algab ja millal lõpeb. Ei saa garanteerida, et saatja ja vastuvõtja kellad käiksid ideaalselt samas rütmis. Seega on vaja kasutada mooduseid, edastamaks sünkroniseerimiseks vajalikku infot. Edastusviisid jagatakse sünkroonseteks ja asünkroonseteks. Sünkroonse edastuse puhul sisaldab iga bitiaeg mingil moel sünkroniseerimisinfot. Sünkroonsele edastusele on omane, et üks pool loeb

teisele takti. Sellest tulenevalt määrab see ka läbilaskevõime, sest mida tihedam on takt, seda suurem on läbilase (mõlemad peavad muidugi toetama vastava kiirusega takti). Asünkroonse edastuse puhul lisatakse bitivoogu spetsiaalsed alguse ja lõpu bitid, mis aitavad hoida sünkroonis bittidest arusaamist. Osapoolte kelladele on lubatud mingi maksimaalne takti erinevus. Saatja lisab vajadusel spetsiaalbitte, mille abil saab vastuvõtja hoida ennast saatja taktis. Vastuvõtja pool ei ole teadlik saatmise kestvusest, ta saab teada seda informatsioonist endast (kui tulevad lõppbitid või mingil muul moel). Seejärel näiteks ei saadeta tükk aega enam mitte midagi. Saatma hakkamisel peab uuest edastusest vastuvõtjale teada andma, et alustatakse saatmist. Näitena võiks tuua terminaliühenduse, kus terminal ei pea pidevalt suhtlema keskse süsteemiga (terminalina vaatleme antud juhul arvutit, millel puudub kõvaketas ning programmifailid jms. asuvad keskses masinas ja on otseses sõltuvuses sellest). Seal on vaja teada anda, kui terminal soovib informatsiooni saata ja millal saatmist lõpetada.

Väga oluliseks näitajaks on, kui palju võimaldab sidekanal kindla aja jooksul bitte edastada. Nimetatud omaduse jaoks on kaks natuke erinevat mõistet – ribalaius (*bandwidth*) ja läbilaskevõime (*throughput*). **Ribalaiuseks** nimetatakse teoreetilist edastuse mahtu ehk mitu bitti on võimalik edastada kindla aja jooksul. **Läbilaskevõimeks** nimetatakse reaalselt kasutatavat ribalaiust. Tegelikkuses on pidurdavateks faktoriteks sisend-/väljundprotsessori kiirus, võrgu koormatus, võrguseadmete töötluse ajakulu, operatsioonisüsteemi ajakulu, suhtleva programmi enda ajakulu jms. Mõlema puhul on mõõtühikuks bit/s (bitti sekundis), mida kasutatakse tavaliselt koos eesliitega (kilo, mega, giga, tera). Näiteks võib Interneti-ühenduse ribalaius olla 8 Mbit/s. Tasub tähele panna, et andmed märgitakse bittides, mitte baitides.

Tänapäeval valdavalt kasutusel olevad sideliinid võib jagada üldjoontes kahte liiki: leviedastus- ja punktist-punkti sideliin. Leviedastus-sideliini korral jagavad mitu masinat ühte sideliini ja kõik infoühikud jõuavad lisaks sihtmasinale ka mittesihtmasinateni. Tavaelust oleks analoogiks majasisesed paralleeltelefonid, kus saab korraga sidet pidada ainult üks isik ja teine ei saa kasutada telefoni, kui see (sideliin) on juba kasutusel, kuid ta saab pealt kuulata. Punktist-punkti sideliini korral on sideliinis olemas ainult kaks osapoolt, kes omavahel kasutavad sideliini.

Leviedastuse sideliini võib omakorda jagada kasutamise järgi kaheks: staatiliseks ja dünaamiliseks. Staatilise puhul jagatakse aeg diskreetseteks lõikudeks, mida ringiratast (*round-robin*) algoritmi alusel masinad kasutatavad. Staatilise meetodi probleemiks on raiskamine, sest iga masin ei pruugi iga kord soovida kasutada sideliini, mille tõttu sideliin seisab jõude. Samuti on keerulisem masinate lisandumise ja eemaldumisega arvestamine. Dünaamilise sideliini kasutamise puhul saab jagada sideliini kasutamise tsentraliseeritud ja detsentraliseeritud kanali hõivamiseks. Tsentraliseeritud meetodi puhul määrab mingi kindel masin selle, kes saab sideliini kasutada järgmisena. Detsentraliseeritud meetodi puhul määravad masinad ise selle, millal nad saavad andmeid saata ja millal mitte (erinevad protokollid kasutavad selleks erinevaid algoritme).

Ethernet

Vaatleme järgnevalt protokolli Ethernet elektrijuhtmeediumi baasil. Ethernet on tänapäeval kohtvõrgus kõige sagedamini kasutatav tehnoloogiate perekond. Etherneti kontseptsioon sai alguse Alohanet võrgust (oli vaja raadiolaineala jagada), mida arendati Havai Ülikoolis 1960-ndate lõpus – 1970-ndate alguses ning mille võtsid Etherneti arendusel aluseks Digital, Intel ja Xerox (tuntakse ka DIX akronüümina). Ethernet oli mõeldud kontoriseadmete ühendamiseks. Etherneti algne ja täiendatud teine versioon avaldati vastavalt 1980. ja 1982. aastal. Patendiprobleemide tõttu avaldas IEEE eraldi IEEE 802.3 spetsifikatsiooni, mis oli väga sarnane Ethernet V2.0 spetsifikatsioonile. Kuigi need kaks spetsifikatsiooni on väga sarnased, ei ole nad omavahel ühilduvad (IEEE 802.3 on paindlikum). Käesoleval ajal on uutes paigaldustes Etherneti nime all kasutusel IEEE 802.3 spetsifikatsioon (mis pole tehniliselt päris korrektne, kuid kuna see on selliselt juurdunud, siis edaspidi mõtleme OSI füüsilise kihi Ethernet spetsifikatsioonina

IEEE 802.3, mitte Ethernet V2.0). IEEE 802.3 standardi nimi tähistab samaaegselt algset IEEE poolt välja töötatud Etherneti kui ka kogu IEEE Etherneti edasiarendusi hõlmavat tehnoloogiate perekonda. Edasiarenduste standardite nimedes lisatakse "IEEE 802.3" järel mingi täht või tähed. Näiteks "IEEE 802.3i", "IEEE 802.3ab".

Hiljem on Etherneti standardit laiendatud toetamaks erinevaid kaableid, ribalaiusi, meediumeid jne. Ethernet ise katab endas OSI esimese kihi ja osaliselt ka OSI teise kihi. Praegu vaatleme Etherneti esimese kihi juurde kuuluvat osa. Etherneti erinevatel laiemalt kasutatavamatel variatsioonidel on mitmeid sarnasusi, kuid ka mitmeid erinevusi. Sellest tulenevalt vaatleme alguses neid ühiselt ja hiljem eraldi Etherneti perekonda kuuluvaid levinumaid standardeid.

Tänapäeval on uued Etherneti võrgud üldjuhul täielikult täisdupleksvõrgud, kuid Ethernet töötati välja olukorras, kus meedium oli jagatud. Seetõttu vaatleme praegu lühidalt Etherneti jagatud meediumi kasutamist ehk poolduplekс-Etherneti ühendusi (jagatud meedium on juba oma loomu poolest poolduplekс). See tähendab, et kui jagatud meediumiga võrgus üks masin saadab andmeid, siis teised peavad sellest aru saama, et toimub saatmine ja ei tohi sel ajal meediumit edastamiseks kasutada. Vastasel juhul toimub signaalide segunemine, mille tulemusena ei ole võimalik vastuvõtjal signaalist aru saada. Sellist olukorda, kus vähemalt kaks masinat mingil põhjusel samal ajal kasutavad jagatud meediumit edastamiseks, nimetatakse kollisiooniks. Seetõttu peab jagatud meediumi puhul kindlustama, et ainult üks masin saab korraga meediumi edastamiseks kasutada. Etherneti meediumile juurdepääsumeetodit nimetatakse CSMA/CD (*carrier sense multiple access with collision detection*), mis koosneb kahest osast. CSMA on protokoll, mis saadab andmeid alles siis, kui keegi enam meediumit ei kasuta. CD on täiendus, mille abil saab tekkinud kollisioonid võimalikult kiiresti avastatud ja lõpetatud ning sideliini jälle võimalikult kiiresti kasutatavaks. Ilma CD täienduseta oleks jagatud meediumis kollisioonide tõttu ajalised kaotused palju suuremad (CD vajab keerulisemat elektroonikat, kuid see ei ole tänapäeval üldjuhul probleemiks). Järgnevalt vaatleme, mis moodi CSMA/CD-d kasutav võrguliides käitub (ehk mis moodi poolduplekssühendus toimib). Kui võrguliides soovib hakata andmeid saatma, siis enne kuulab ta sideliini, et ega see ei ole hetkel kasutusel. Kui sideliin on vaba, siis hakkab võrguliides ise seda kasutama andmete edastamiseks. Kui sideliin on kasutusel, siis ootab võrguliides juhusliku aja ja kuulab uuesti sideliini. Kui sideliin on ikka hõivatud, siis oodatakse uuesti jälle mingi juhuslik aeg (mikrosekundites) ning seejärel kuulatakse uuesti. See tsükkel kestab, kuni sideliin on vaba või kui on ebaõnnestunult proovinud järjest teatud arv kordi (Etherneti puhul 16 korda järjest), mille peale tagastatakse ülemisele kihile vastav veateade. Seda võib juhtuda, kui võrk on väga hõivatud. Kui võrguliides on parasjagu andmeid saatmas, siis ta samal ajal jälgib, ega ei ole toimunud kollisiooni (kollisiooni tekkimisest saadakse aru, kui signaalide amplituud kasvab liiga suureks). Kollisiooni tulemusena ei ole võimalik infot välja lugeda, mistõttu läheb edastatu kaduma. Peale kollisiooni avastamist peatatakse koheselt andmete saatmine ning sideliini saadetakse ummistussignaal (*jam signal*), mis on 32 bitti pikk (enamasti bitijärjekord 10101...). Niimoodi rikutakse täielikult parasjagu saadetavad kaadrid ning kõik masinad saavad toimunud kollisioonist aru. Seejärel käivitatakse tagasivõtu (*backoff*) algoritm. Kõik masinad ootavad juhusliku aja (mikrosekundites), enne kui hakkavad uuesti kuulama, kas sideliini kasutatakse ning kui sideliin on vaba, siis hakkavad nad jälle andmeid saatma. Kui sama paketi saatmisel tekib kollisioon, siis iga kord kahekordistatakse ootamise aega. Selle algoritmi pikem nimetus on kärbitud kahend-eksponentsiaalse tagasivõtu algoritm (*truncated binary exponential backoff algorithm*). CSMA/CD on vajalik poolduplekси puhul. Täisduplekси puhul ei ole otsest vajadust CSMA/CD järele, sest kollisiooni kui sellist ei saa tekkida, sest andmeid saadetakse ja võetakse vastu erinevates traatides.

Tavaelust võiks näiteks tuua pimedate koosoleku, kus saab korraga rääkida ainult üks inimene. Rääkima hakatakse siis, kui mitte keegi parasjagu ei räägi. Kui juhtub, et samaaegselt hakkasid rääkima kaks või enam inimest, siis saavad nad ise aru, et nad polnud ainukesed rääkijad ning lõpetavad kõnelemise, sest nad kuulavad pidevalt ka oma rääkimise ajal. Kõneldud lausest ei saanud keegi aru ja seda peab uuesti ütlema, oodates selleks sobivat aega.

Etherneti perekonna tehnoloogiatel on olemas ka lisanimed, mis koosnevad kolmest osast:

- Kaabli ribalaius megabittides, kui numbrile järgneb G, siis gigabittides.
- Määratleb, kas tegemist on põhiriba- (*baseband*) (tähis nimetuses "BASE") või lairibaedastusega (*broadband*) (tähis nimetustes "BROAD"). Põhiribaedastuse puhul kasutatakse ainult ühe moduleerimata ja multipleksimata signaali edastamist. Lairibaedastuse puhul kasutatakse mitme signaali paralleelset edastamist.
- Maksimaalne kaabli pikkus (mitusada meetrit) või kaabli tüüp. Erinevad tähistused: "T" keerdpaarkaabel (maksimaalselt 100 meetrit). "F" fiiberoptiline kaabel. Kui samal meediumitüübil on kasutusel mitu samasuguse kiirusega Etherneti versiooni, siis võidakse lisada X või mingi muu märk.

Näiteid nimetustest: 10BASE-2, 10BASE-T, 100BASE-TX, 100BASE-T4, 10BROAD36 (kasutatakse eelkõige kaabeltelevisiooni puhul, Etherneti jaoks ei võetud eriti kasutusele). Meie vaatleme lähemalt keerdpaarkaableid, täpsemalt 10BASE-T, 100BASE-TX ja 1000BASE-T, mis on ka kõige levinumad (elektrijuhtkaablite osas). Joonisel 5.17 on toodud mitmeid erinevaid Etherneti tehnoloogiaid, nende tähiseid ja kasutatavaid kaableid.

Keerdpaarkaablid

Keerdpaarkaabel koosneb kaheksast peenemast vaskjuhtmest, mille igaühe ümber on õhuke plastikust dielektriku kiht. Kõik traadid on paarikaupa keerdus (keerdude arv pikkusühiku kohta on ka spetsifitseeritud), millest tuleneb ka nimetus. Kaheksat traati (koos dielektriku kihiga) ümbritseb kate, mida nimetatakse mantliks. Materjaliks on tavaliselt polüvinüülkloriid ehk PVC (*polyvinyl chloride*), mis on odav ja sobivate omadustega. PVC puuduseks on tulekahju korral eralduv mürgine gaasiline kloor, millest veega kokku puutudes tekib tugevatoimeline vesinikkloriidhape (ohtlik just silmadele ja hingamisteedele, kuna need piirkonnad on niisked), ja märgatavalt suurem süsinikoksiidi ehk vingugaasi (CO) eraldumine. Alternatiiviks on polüpropüleen ehk LSZH (*low smoke zero halogen*).

Traadid on keerdus paarikaupa, et mõlemas traadis oleks võimalikult sarnane ülekoste. Vastuvõtja saab seda ära kasutada müra väljafiltreerimiseks. Paarikaupa keerdus traadid on samuti vähem vastuvõtlikud ülekostele või mürale, mis lähtuvad naabertraadipaaridest (sest nad on omavahel rohkem risti). Seetõttu nõuavad kõrgema kategooria keerdpaarkaablid rohkem keerde pikkusühiku kohta, et vähendada ülekosteid suurematel ülekandesagedustel.

Et traadid oleksid omavahel paremini eristatavad, siis iga traadi ümber olev isolatsioon on erinevat värvi. Standardi järgi on värvid järgmised: valge-roheline, roheline, valge-oranž, oranž, valge-sinine, sinine, valge-pruun ja pruun. Omavahel on keerdus sama värvi sisaldavad traadid (näiteks valge-roheline ja roheline).

Keerdpaarkaablid võib jagada kaheks:

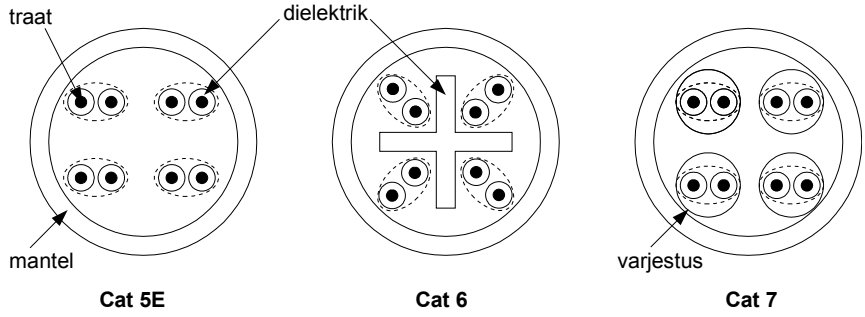
- UTP (*unshielded twisted pair*) – varjestamata keerdpaar. Varjestamata keerdpaar on kõige lihtsama ehitusega ning seetõttu ka kõige odavam ja paindlikum. Praktikas kasutatakse kõige enam UTP kaableid.
- STP (*shielded twisted pair*) – varjestatud keerdpaar. Siin on iga keerdus traadipaari ümber kas punutis või metallikiht (foolium). Lisaks sellele on mantli all samuti kas punutis või metallikiht. Varjestuste ülesanne on pakkuda kaitset häirete ja ülekoste eest, samuti kaitstakse ümbritsevat keskkonda lekkiva signaali eest, mis võib teistele müraks saada. STP kaableid kasutatakse vähe. Keerdpaarkaablite vahel oleva varjestuse peamine ülesanne on takistada traatide omavahelist mõjutamist. STP kaableid kasutatakse peamiselt kohtades, kus on palju häireid. Samas on STP kaablid kallimad ja neid on keerukam paigaldada. Näiteks peavad varjestused STP kaablite otsades olema maandatud. Tänu varjestustele on kaabel ka jämedam, raskem ja kallim. Varjestatud kaableid tasub paigaldada intensiivsemate raadio- ja elektromagnetlainete piirkondadesse.

Lisaks eelmainitutele on olemas ka kombinatsioonid nendest, ainult keerdpaaride ümber varjestusega ja ainult mantli all oleva varjestusega kaablid.

Kuna enamasti kasutatakse UTP kaableid, siis edaspidi keskendumegi rohkem UTP kaablitele. UTP kaablitel on olemas erinevad kategooriad, mis eristuvad erinevate nõudmiste poolest kaablile ja otsikutele. Järgnevalt anname lühiülevaate UTP kategooriatest ("Category" lühendatakse tihti "Cat") (vaata ka joonist 5.8):

- Cat 1 – kasutatakse heli edastuseks (telefoniliinid), ka alarmsüsteemides. Ei ole sobiv andmete edastuseks.
- Cat 2 – sagedus kuni 1 MHz. Kasutati heli ja väiksema läbilaskega andmete transpordiks (näiteks 4 Mbit/s Token Ring võrgus).
- Cat 3 – sagedus kuni 16 MHz. Kasutati andmete ja heli edastuseks. Kasutatav Ethernetis. Uutes installatsioonides ei kasutata.
- Cat 4 – sagedus kuni 20 MHz. Asendati kiiresti Cat 5-ga.
- Cat 5 – sagedus kuni 100 MHz. Kasutati kuni 100BASE-TX kiirusega ühendustes. Uutes paigaldustes ei kasutata.
- Cat 5e – sagedus kuni 125 MHz. Kasutatav ka 1 Gbit/s ühenduse juures. Tänapäeval kõige enam uutes paigaldustes kasutatav UTP kaabel.
- Cat 6 – sagedus kuni 250 MHz. Lisaks on kõrgemad nõudmised ülekoste ja tagasipõrke kao osas. Cat 6 kaabli traadid on võrreldes Cat5e kaabliga natuke jämedamad, mille tõttu on ka kaabel natuke jämedam.
- Cat6a – sagedus kuni 500 MHz. Mõeldud 10 Gbit/s ühenduste jaoks.

Saavutamaks suuremaid läbilaskevõimeid, on vaja teha mitmeid täiustusi ja muudatusi. Üks osa nendest on muidugi kõrgemad nõudmised kvaliteedile, kuid samas on ka teisi viise, näiteks keerdude arvu suurendamine pikkusühiku kohta ja traatide erinev paigutus kaablis, uus signaliseerimine, traatide jämedus jne.



Joonis 5.8: Cat 5E, 6 ja 7 kaablite ristlõiked.

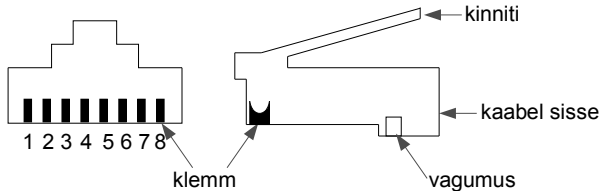
Kaablid võib jagada ühekiuliseks (*solid*) ja mitmekiuliseks (*stranded*). Nimetatakse ka vastavalt jäigaks ja pehmeks kaabliks. Ühekiulise kaabli puhul on ainult üks traat, kuid mitmekiulise kaabli puhul koosneb traat mitmest peenemast traadist. Ühekiuline kaabel on odavam ja sagedamini kasutatavam. Mitmekiuline kaabel kannatab rohkem liigutamist kui ühekiuline, mille tõttu kasutatakse mitmekiulist kaablit näiteks sülearvuti ühendamiseks. Ühekiuline kaabel on natuke paremate elektriliste omadustega kui mitmekiuline, kuna mitmekiulise kaabli ümarate traatide vahel on natuke tühja ruumi. Siinjuures peab kaabli ristlõige olema sama nii ühekiulise kui ka mitmekiulise kaabli puhul, seega mitmekiulise kaabli traatide summaarne ristlõige on väiksem kui ühekiulise kaabli traadi oma. Seetõttu on mitmekiulises kaablis sumbuvus suurem, mistõttu on ka kaabli maksimumpikkus väiksem ühekiulisest.

Olemas on ka Cat 7 (standardi nimetus ISO/IEC 11801 Class F) ja Cat 8 (vastab IEC 61156-7 nõuetele), kuid need ei ole enam UTP kaablid. Cat 7 ja Cat 8 puhul on varjestus igal traadipaaril ja ka kogu kaablil. Sagedus on kuni 600 MHz. Cat 7 on mõeldud 10Gbit/s Ethernet ühenduste jaoks. Võrreldes Cat 6 kaabliga on peamiselt rangemad nõuded ülekostele ja mürale.

Otsikud

Kaablite otsikud on komponendid, mis liidestavad kaabli ja seadme võrguliidese. UTP kaabli standardseks otsikuks on RJ-45 (*Registered Jack*, 45 viitab traatide järjestusele). RJ-45 otsikul on 8 klemmi (*pin*). RJ-45 liidesel on kaabli poolne ots isane ja seadme poolne ots emane, seega arvuti võrgukaartidel olev RJ-45 liides on emane. RJ-45 otsik on sarnane lauatelefoni otsikuga, mille otsikut tuntakse RJ-11 nime all. Cat 6 kaabli otsikuks on 8P8C, milles traadid lähevad otsikusse sik-sakis, mitte ühes tasapinnas, ülejäänud osa on ühilduv RJ-45-ga. Sik-sakis paigutamine oli vajalik, sest Cat 6 kaabli traadid on jämedamad. Cat 7 kaabli puhul on otsikuks GG45 (*Giga Gate*), mis on tagasiühilduvad RJ-45 otsikuga. Teine otsik Cat 7 kaabli jaoks on TERA, mis ei ole enam ühilduv RJ-45-ga, kuid on paremate jõudlusomadustega.

UTP ühendamisel otsikuga RJ-45 peab traadid võimalikult lühikeselt sirgeks tegema ja otsikusse lõpuni lükkama (et kontaktiala oleks võimalikult suur). Kaabli mantel peab ulatuma vagumusest natuke üle. Kui traadid on õiges järjestuses korrektselt otsikusse lükatud, siis fikseerimiseks kasutatakse spetsiaalseid näpitsaid, mida nimetatakse võrgutangideks. Mõned otsikud sobivad nii ühe- kui mitmekiulise kaabli jaoks, osad otsikud sobivad kas ainult ühekiulise või ainult mitmekiulise kaabli jaoks.



Joonis 5.9: Otsiku RJ-45 eest ja küljelt vaade. Numbrid näitavad klemmide lugemise järjekorda (kuidas traadid ühendada).

Info vahetus sideliinis

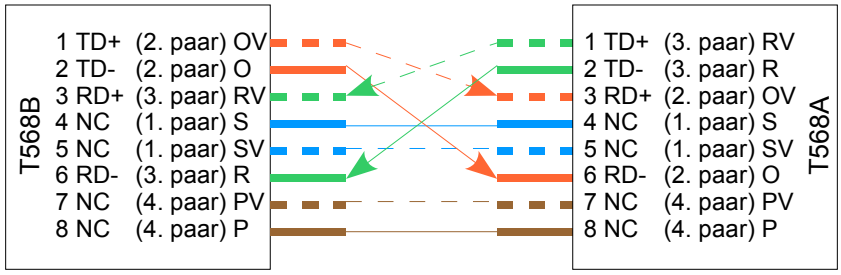
Nüüd on üle vaadatud alused signaalide saatmiseks kaablis ja kaabliotste liidesed, mis ühendavad võrguliidesega. Edasi vaatleme, kuidas täpsemalt toimub infovahetus, mis moodi signaale kasutatakse, mis moodi termineeritakse kaablid jne. Võtame jällegi aluseks kohalikus võrgus enim kasutatava protokolli Ethernet ja keerdpaarkaabli UTP.

Kasutamaks kaablit informatsiooni vahetamiseks, peab olema spetsifitseeritud, mis klemme millekski kasutatakse. Keerdpaarkaablis on kaheksa traati, nende puhul on vajalik määratleda traatide tähendused, et mõlemad otspunktid mõistaksid üksteist. Kuna me kasutame otsikut RJ-45, siis on vaja täpsustada, millist funktsiooni mingi klemm omab. Üldjuhul on keerdpaarkaabli traadid markeeritud erineva värviga. Olenevalt traatide järjestusest mõlemas kaabli otsas jaotatakse kaablid:

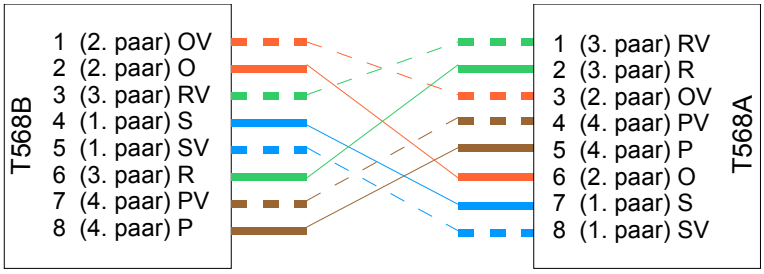
- Otsekaabel (*straight-through*) – kaabli mõlema otsa erinevat värvi traadid on sama järjestusega. Kasutatakse põhiliselt hubi või kommutaatori ühendamiseks mitte hubi või kommutaatoriga (üks ots on hub või kommutaator, teine ots on mingi muu seade).
- Ristkaabel (*crossover*) – kaabli otste traadid on erineva järjestusega, et ühe poole saatja kontakt jõuaks teisele poole vastuvõtvasse kontakti ja vastupidi. Kasutatakse selleks, et ühendada seadmeid, kus ükski otspunkt ei ole kommutaator ega hub või on mõlemad pooled kommutaatorid või hubid.

Etherneti 10 Mbit/s ja 100 Mbit/s ühenduste puhul kasutatakse informatsiooni edastamiseks keerdpaarkaablite esimest ja teist traati ning vastuvõtmiseks kolmandat ja kuuendat traati. Oluline on, et otspunktide vahel oleks traadid õigesti ühendatud. Ka siin on olemas värvide järjestuse standard. Standardis EIA/TIA-568-B.1 spetsifitseeritakse kaks värvikoodi T568A ja

T568B. Soovitatav on kasutada siintoodud värvikoode. Otsekaabli puhul on mõlemad otsad sama värvikoodiga, see tähendab, et mõlemad otsad on T568A või T568B värvikoodi alusel. Kui on vaja ristkaablit, siis peavad mõlemad otsad olema eri värvikoodi järgi tehtud, see tähendab, et üks pool peab olema T568A ja teine pool T568B. Joonisel 5.10 on toodud värvikoodid ja traatide kasutus ristkaabli näitel. Joonisel 5.11 on gigabitise Etherneti puhul kasutatava ristkaabli traatide järjestus.



Joonis 5.10: Ristkaabli traatide paigutused otsikusse. Välja on toodud vertikaalsena värvikoodi standardi nimetus. Veergude kaupa: traadi järjekorra number, traadi kasutus (TD (*transmit data*) saatja, RD (*receive data*) saaja, NC (*no connection*) pole kasutusel), sulgudes paari number ja viimasena värv (O - oranž, R - roheline, S - sinine, P - pruun ja V - valge).



Joonis 5.11: Gigabitist ühendust kasutava ristkaabli traatide järjestus.

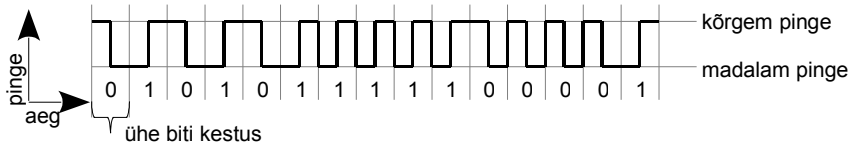
Värvikoodi võib aidata meelde jätta see, et vöödilised ja ühevärvilised traadid on vaheldumisi. Sinised traadid on keskel ja pruunid viimased. Oranžid ja rohelised traadid on ainukesed, mille järjekord on kord ühtemoodi ja kord teistmoodi. Samuti võib abi olla sellest, kui teada, et esimene ja teine traat on saatmiseks ja kolmas ja kuues on vastuvõtmiseks. Numbritega arvutades 1+2=3 ja 2*3=6.

Etherneti erinevad versioonid

10BASE-T on 10 Mbit/s Ethernet, mis oli esimene IEEE Etherneti standard keerdpaarkaabli jaoks. Selle standardi tähiseks on IEEE 802.3i (avaldatud 1990). 10 Mbit/s Ethernet on asünkroonne. Vastuvõtja kasutab 8 baidilist Etherneti preambulat sünkroniseerimiseks, mis sisaldab vaheldumisi bitte 0 ja 1 (10101...) ning mis lõpeb "...01011". 100 Mbit/s ja suurema kiirusega Etherneti versioonid on sünkroonsed. See tähendab, et nad ei kasuta preambulat, seega sünkroniseerimisinfo saatmine ei ole vajalik, kuid ühilduvuse huvides on preambula ikkagi kasutusel. Ethernet 10 Mbit/s puhul kasutatakse sideliinis elektriliseks kodeeringuks Manchesteri kodeeringut. 10Mbit/s puhul on ühe biti kestus 100 nanosekundit.

Manchesteri kodeeringu puhul toimub üleminek ühest pingeolekust teise bitile vastava takti kestel. Kui takti alguses on pinge madalam ning lõpus on pinge kõrgem, siis antud biti väärtus on 1. Kui biti takti alguses on kõrgem pinge, kuid lõpus on madalam, siis biti väärtuseks on 0.

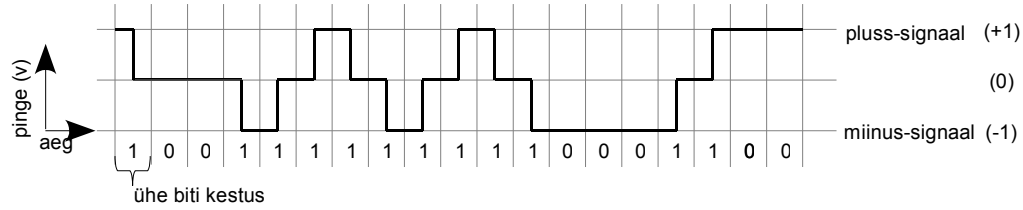
Pinge väärtuste muutmised sobivasse seisundisse toimuvad bititaktide vahepeal (vastavalt sellele, millist bitti on vaja edastada). Manchesteri koodis on vastuvõtjal võimalik ennast sünkroniseerida. Puuduseks on asjaolu, et kodeerimisel toimub signaali pinge muutmine kaks korda tihedamini, kui kodeeritavas andmestikus. Järelikult on sagedusriba kaks korda laiem lähtesignaali sagedusribast.



Joonis 5.12: Manchesteri kodeering. All on kodeeritavad bitid ja üleval on signaali muutused.

100BASE-TX on 100 Mbit/s Ethernet, mida nimetatakse **Fast Ethernet**iks. Selle standarditähiseks on IEEE 802.3u (avaldatud 1995). 100 Mbit/s ühenduskiiruse saavutamiseks võiks tõsta 10BASE-T taktsagedust, kuid paraku on UTP kaabli piiriks selliselt umbes 30 MHz. Seetõttu ei saa kasutada Manchesteri kodeerimist, vaid kasutatakse kahte eraldi kodeerimise etappi. Esimene on 4B/5B kodeerimine, teine on sideliini spetsiifiline kodeerimine, mida keerdpaarkaabli puhul nimetatakse MLT-3 (*multi-level transmit*, 3 näitab tasemete arvu) kodeeringuks. Ühe biti kestuseks on 10 nanosekundit.

4B/5B on plokk-kodeering, kus neli bitti kodeeritakse viieks bitiks (sellest ka nimetus), et vältida üle kolme järjestikulise nulli esinemist voos (iga nelja biti asemele pannakse viis eeldefineeritud bitti). Seda tehakse füüsiliseks signaliseerimiseks paremate eelduste saamiseks. Nendeks eeldusteks on lihtsamini sünkroniseeritavus, saatja ja vastuvõtja lihtsam ehitus ning parem vigade avastamine. 4B/5B puuduseks on 80%-line efektiivsus. Neljast bitist saab moodustada 16 erinevat järjestust, 5 biti puhul 32 erinevat järjestust, mille tõttu viiebitisest osast 16 erinevat bitijärjestust on nelja bitise koodi asendajad, kuus erinevat järjestust on kontrollbittidena kasutusel ja 10 üle jäävat järjestust on kasutamata. 4B/5B koodide vastavused on toodud tabelis 5.1.



Joonis 5.13: MLT-3 kodeerimise näidis. All asuvad kodeeritavad bitid. Tänu eelnevale 4B/5B kodeerimisele ei saa olla korraga rohkem kui kolm järjestikulist nullbitti. Joonisel sulgudes on pingeseisundid.

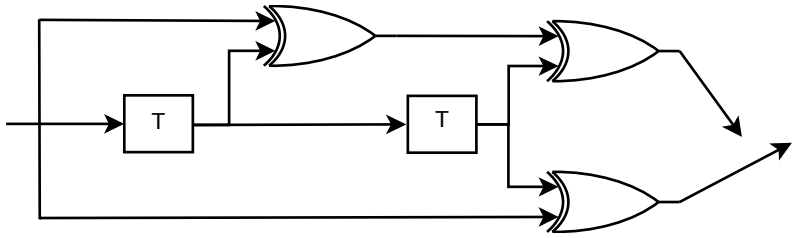
MLT-3 kodeerimise puhul kasutatakse tavapärase kahe pingeseisundi asemel kolme erinevat pingeseisundit. Pingeseisundite muutumine toimub tsükliliselt, kus tsükli olekuteks on 1, 0, -1, 0 (, 1, 0, -1, 0, 1, 0, -1, 0, ...) (vt. joonis 5.13). Kui edastatav bitt on "1", siis toimub pinge seisundi muutumine, kui edastatav bitt on "0", siis seisundi muutust ei toimu. MLT-3 kodeerimisega tekitatakse vähem elektromagnetilist interferentsi, sest signaali tugevus ei muutu nii kiiresti kui Manchesteri kodeerimisega. Sellega saavutatakse sagedus 31,25 MHz (125/4=31,25).

Tabel 5.1: Tabel näitab neljabitisele koodile vastavat viiebitist koodi. Lisaks spetsiaalkasutusega bitijärjendid: *1– "idle" bitijärjend; *2 – voo alguse eraldaja esimene osa; *3 – voo alguse eraldaja teine osa; *4 – voo lõpu eraldaja esimene osa; *5 – voo lõpu eraldaja teine osa. 6 – "halt" bitijärjend. Lisaks võib erinevatel implementatsioonidel olla oma defineeritud spetsiaalkasutusega bitijärjendeid.

4B kood	nimi	5B sümbol	4B kood	nimi	5B sümbol	4B kood	nimi	5B sümbol
0000	0	11110	1000	8	10010	– *1	I	11111
0001	1	01001	1001	9	10011	– *2	J	11000
0010	2	10100	1010	A	10110	– *3	K	10001
0011	3	10101	1011	B	10111	– *4	T	01101
0100	4	01010	1100	C	11010	– *5	R	00111
0101	5	01011	1101	D	11011	– *6	H	00100
0110	6	01110	1110	E	11100			
0111	7	01111	1111	F	11101			

1000BASE-T standardi tähiseks on IEEE 802.3ab (avaldatud 1999). Võrreldes 100 Mbit/s Ethernet ühendusega keerdpaarkaablil, peab 1 Gbit/s ühenduse saavutamiseks kasutama veelgi keerulisemat tehnoloogiat. Samuti on ta veelgi tundlikum mürale. 1 Gbit/s puhul kasutatakse tavapärase kahe traadipaari asemel edastamiseks kõiki nelja paari traate. UTP kaablina nõutakse Category 5e või paremat. 1000BASE-T puhul kasutatakse 4D-PAM5 kodeerimist. Ühe biti kestus on 1 nanosekund.

Esimene etapp andmete saatmisel on konvolutsioonikodeerija (*convolutional*) läbimine (joonis 5.14), mis võimaldab läbi viia veaparandusi, kuid lisab juurde liiasust (ühe sisendbiti asemel väljastatakse kaks väljundbitti).

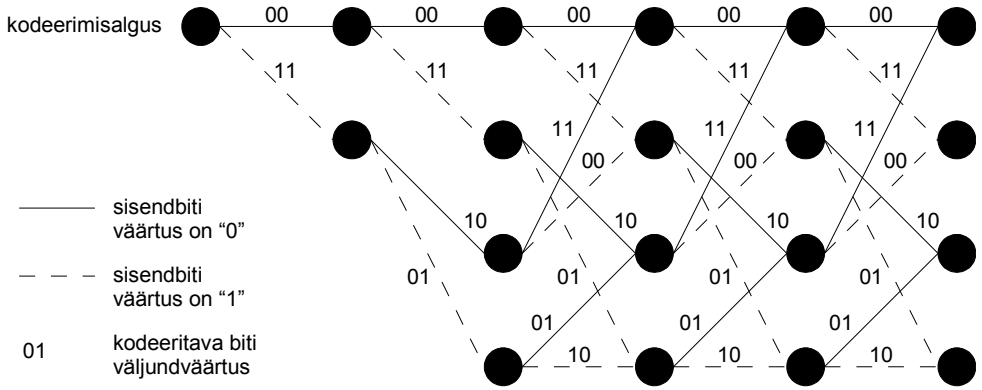


Joonis 5.14: Konvolutsioonkodeerija. Skeemil tähistab ristkülik viivitusega komponenti, mis igal sisendil väljastab eelmise sisendi väärtuse (algväärtusena 0). Kaarjate külgedega kolmnurk, millel üks pool on topeltjoonega, tähistab XOR komponenti (kui sisendiks on 0&1 või 1&0, siis tagastab 1, vastasel juhul tagastab 0). Iga sisendbiti kohta on kaks väljundbitti, millega on hiljem võimalik rakendada veaparandust, kui juhtub, et mingist bitist saadi valesti aru.

Kodeeritud andmetest saadakse algne info tagasi, kasutades Viterbi dekodeerimist. Viterbi dekodeerija võimaldab parandada vigu (mitte ainult neid avastada). Vaatleme Viterbi dekodeerimist Trellise diagrammi põhjal (joonis 5.15).

Viterbi dekodeerimine seisneb selles, et leitakse kõige lähedasem vastus. Seega ollakse üle ka mõningatest sisendandmevigadest (mis võisid võrguhäirete tõttu tekitada vale bitilugemi). Sisendbitte vaadeldakse paarides (nagu kodeerimisel on bitid koos väljastatud). Lisaks definee-

ritakse nn. distantsid. Samasuguste bitipaaride distants on null. Bitipaarid, mis erinevad omavahel ühe biti võrra (arvestades ka järjekorda), on distantsiga üks. Ülejäänud juhtudel erinevad bitipaarid mõlema biti võrra ja nende distants on kaks (vt. tabel 5.2).



Joonis 5.15: Trellise diagramm, mis on lihtsamini haaratav inimesele, võrreldes konvolutsioonikodeerijaga joonisel 5.14.

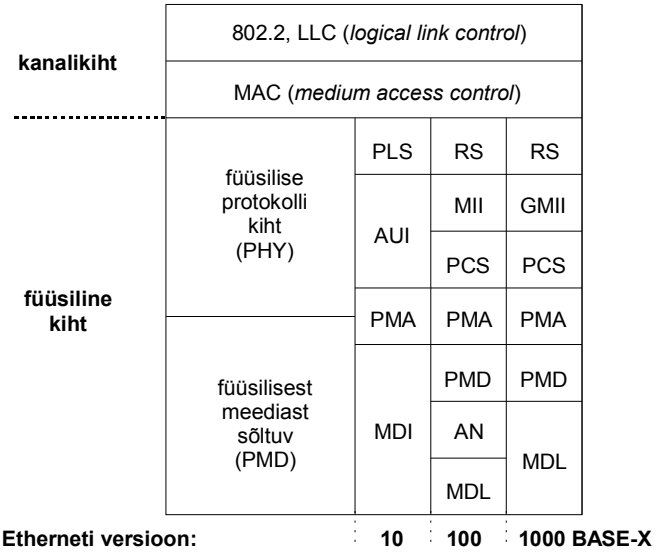
Tabel 5.2: Viterby dekodeerimise võrdluse tabel. Bitipaaride distantsid on omavahel sümmeetrilised, see tähendab, et näiteks 01 ja 10 on sama distantsiga kui 10 ja 01.

bitipaar 1	**bitipaar 2**	**distants**	**bitipaar 1**	**bitipaar 2**	**distants**
00	00	0	10	00	1
00	01	1	10	01	2
00	10	1	10	10	0
00	11	2	10	11	1
01	00	1	11	00	2
01	01	0	11	01	1
01	10	2	11	10	1
01	11	1	11	11	0

Tähistagu Trellise skeemis pidevad jooned nulli ja katkendjoon ühte (väljundbittides). Skeemi koostamine on järgmine. Esimese bitipaari korral leitakse sisendbitipaari ja joonel märgitud bitipaari alusel distantsid (tabeli põhjal ehk mitme biti võrra nad erinesid) teise veeru kõikidesse punktidesse (kuhu joonte kaudu saab) ja jäetakse distants meelde. Järgmiste bitipaaride korral leitakse distantsid kolmanda veeru punktidesse ja liidetakse eelmise etapi distantsi väärtusele juurde. Selliselt minnakse veeru kaupa edasi, jättes meelde kulgemise raja. Mingil hetkel jõutakse kahest erinevast tipust samasse tippu, siis katkestatakse rada, mille distants oli suurem. Jooksvalt on kogu aeg aktiivsed neli rada (v.a. kaks esimest veergu). Selliselt jätkatakse radu, mille distantsid olid kõige väiksemad. Protsessi jätkatakse seni, kuni on sisendbitipaare. Kui sisendbitid on lõppenud, siis leitakse kõige väiksema distantsiga rada (üks neljast rajast). Lühimale rajale vastavate joonte järgi leitakse väljundbittide väärtused ehk millised jooned vastaval rajal asusid. Kui raja distants oli null, siis eeldatavalt ei esinenud vigu sisendandmetes, vastasel

korral, kui distants oli suurem kui null, tekkis ülekandel mingeid vigu, millest nüüd kõige tõenäolisem variant välja selekteeriti (väikseima distantsiga rada).

Andmete saatmisel jaotatakse andmed nelja paralleelsesse voogu, kodeeritakse, saadetakse ja võetakse vastu paralleelselt. 1000BASE-T toetab standardi järgi nii täisdupleks- kui ka poolduplekс-ühendust, kuid poolduplekс-ühendust reaalselt ei kasutata.



Joonis 5.16: Etherneti erinevate versioonide füüsilise kihi alamkihid. Joonise akronüümide lahtikirjutused: AN (*auto negotiation*); AUI (*attachment unit interface*); GMII (*gigabit medium-independent interface*); MDI (*medium-dependent interface*); MDL (*medium dependent layer*); MII (*medium-independent interface*); PCS (*physical coding sublayer*); PLS (*physical layer signalling*); PMA (*physical medium attachment*); RS (*reconciliation sublayer*); PHY (*physical protocol layer*); PMD (*physical medium dependent*).

Kõige uuem Etherneti standard, mis on jätkuks eelnevatele käsitletud Etherneti standarditele, on 10GBASE-T, standardi nimetusega IEEE 802.3an (avaldatud 2006). 10GBASE-T on veel väga uus ja ei ole seetõttu praktikas suurt kasutust leidnud, seetõttu me ei käsitle seda standardit pikemalt.

Ethernetil on olemas mitu versiooni. Reaalselt on oluline tagasiühilduvus – see tähendab, et näiteks võrreldes 1 Gbit/s Etherneti 100Mbit/s Ethernetiga, peab 1 Gbit/s toetav võrguliides olema suuteline suhtlema maksimaalselt 100 Mbit/s kiirust toetava võrguliidesega. Et see toimuda saaks, on osapoolte vahel vaja kokku leppida mõlema poole jaoks parimas ühenduskiiruses ja dupleksis (suurem kiirus ja täisdupleks on prioriteetsem, kui poolduplekс). Seda protsessi nimetatakse **utomaatkätluseks** (*auto-negotiation*). Joonisel 5.16 on toodud erinevate Etherneti versioonide alamkihtide detailsem jaotus.

Kaabelduse testimine

Kui kaabeldus on paigaldatud, siis on soovitatav seda testida (keerdpaarkaablite korral). Selleks kasutatakse kaablitestereid. Kümme peamist parameetrit, mida TIA/EIA standardi järgi testitakse, on järgmised (väärtused sõltuvad protokollist):

- Traadid on vastavalt standardile õigesti omavahel ühendatud.

- Insertsiooni kadu (*insertion loss*) – testitakse ülekostet, saates ühte traati signaali ning mõõtes teises otsas teistes traatides signaali tugevust (ideaalsel juhul ei ole teistes traatides signaali).
- Signaali sumbuvus (*attenuation*) – signaali nõrgenemine kaablit läbides (ideaaljuhul ei ole signaal nõrgenenud).
- Tagasipõrke kadu (*return loss*) – kui elektrisignaal läbib elektrijuhti, mis on kohati erineva takistusega, siis tekivad signaali tagasipõrked. Mida suuremad on takistuste järsud muutused, seda suurema ulatusega on tagasipõrge, mis oleneb ka signaali sagedusest. Tagasipõrge vähendab signaali tugevust. Tihti on tagasipõrgete kohaks otsikud. Mida ühtlasem on takistus, seda väiksem on tagasipõrkekadu. Mõõdetakse detsibellides. Suurem väärtus on parem.
- Signaali levimise kiirus (*propagation delay*). Väiksem väärtus on parem.
- Signaali levimise kiiruse erinevused juhtmetes (*delay skew*). Ideaaljuhul levib signaal kõikides traatides sama kiirusega (samal ajal lähetatud signaalid peaks jõudma teise sideliini otsa samal ajal, mida suurem erinevus, seda halvem).
- Kaabli pikkus ei tohiks olla pikem standardis toodud pikkusest (senivaadeldute puhul 100 meetrit).
- NEXT ülekoste. Mõõdetakse detsibellides. Suurem väärtus on parem.
- PSNEXT ülekoste. Mõõdetakse detsibellides. Suurem väärtus on parem.
- ELFEXT ülekoste. Mõõdetakse detsibellides. Suurem väärtus on parem.
- PSELFEXT ülekoste. Mõõdetakse detsibellides. Suurem väärtus on parem.

Väiksemates asutustes tihti testereid ei kasutata, kuid suuremates enamasti kasutatakse. Odavamatel ja lihtsamatel testeritel on ainult traatide ühendatuse kontrolli funktsionaalsus, natuke kallimate testeritega on võimalik mõõta ka sideliini pikkust või katkestuse ligikaudset asukohta (mõõdetakse tagasipõrkunud signaali alusel). Kallid testerid võimaldavad testida kõiki eelpool toodud parameetreid.

OSI esimese kihi võrguseadmed

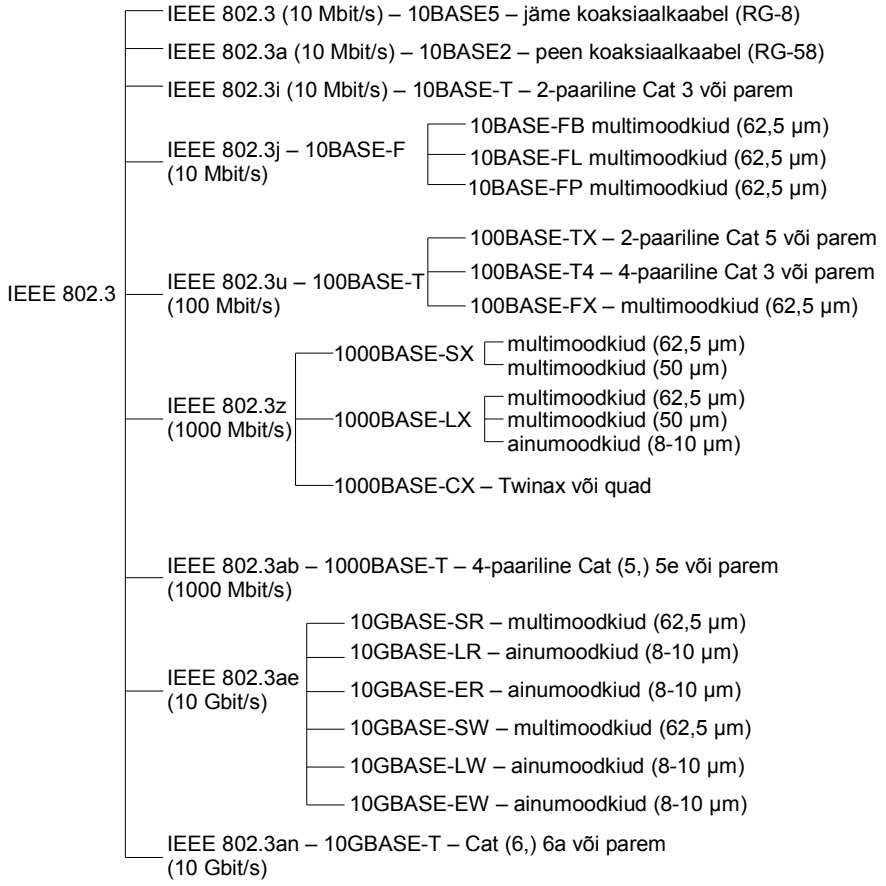
OSI esimese kihi võrguseadmed opereerivad ainult füüsilisel tasemel. Nad võtavad vastu signaali, mille saadavad edasi ühte või mitmesse teise porti, tavaliselt taasgenereerides signaali. Vahendatavad bitid ei oma esimese kihi seadmete jaoks semantilist tähendust. Enamasti kasutatakse esimese kihi seadmeid juhul, kui on vaja ühendada omavahel kaks otspunkti, kuid kaabli maksimaalne lubatav pikkus on väiksem vajatavast (seni vaadeldud Etherneti tehnoloogiate juures ligikaudu 100 meetrit). Kui aga on vaja ühendada kaks otspunkti, mille ühendamiseks kuluks 180 meetrit kaablit, siis see ei õnnestu (signaal lihtsalt sumbub kaablit läbides liiga palju). Saamaks üle taolisest probleemist, on võimalik kasutada võrguseadet, mis võimendab signaali ehk taasgenereerib algse bitivoo signaalid. Neid seadmeid nimetatakse **repiiteriteks** (*repeater*) (nimetatakse ka järgur). Repiiter tasub panna maksimaalselt iga saja meetri taha või siis 180 meetrise ühendamise korral umbes keskele, kuid mõlemale poole peab jääma mitte rohkem kui 100 meetrit kaablit (10 Mbit/s ühenduste korral võib vahemaa olla reaalselt ka mitukümmend meetrit pikem). Repiitereid ja hube ei saa järjestikku palju olla, sest signaali levimine võtab aega (maksimaalselt neli tükki).

Võrgus on meil vaja suhelda rohkem kui ühe osapoolega. Füüsilisel tasemel on seda probleemi lahendavaks seadmeks **hub** (nimetatakse ka jaotur). Hub omab erinevalt repiiterist mitut porti, seetõttu kutsutakse teda ka mitme pordiga repiiteriks. Ta võtab vastu signaali ja saadab selle edasi kõikidesse portidesse, välja arvatud sinna, kust bitijada pärines. Hubide korral saab andmeid saata maksimaalselt ainult üks otspunkt korraga. Kui juhtub, et mitu masinat saadavad andmeid samal ajal, on tulemuseks kollisioon. Saadetud bitid jõuavad kõikide masinateni, mis hubiga on ühendatud, välja arvatud saatjani. See tähendab, et kui hubiga on ühendatud arvutid A, B, C, D ja E ja arvuti A saadab andmeid arvutile B, siis hub saadab andmed masinatele B, C, D ja E (masinad C, D ja E filtreerivad neile mittemõeldud info välja OSI teises kihis).

Ühte jagatud meediumiga osa võrgust kutsutakse **kollisioonidomeeniks**. Esimese kihi seadmed laiendavad kollisioonidomeeni. Mida suurem on kollisioonidomeen ja seal olevate masinate arv, seda suurem on kollisiooni tõenäosus. Võrguseadmete juures peame arvestama, et nad lisavad mingi viivituse, mille tulemusena aeglustub signaali levik kõigi kollisioonidomeenis olevate masinateni. Taolises olukorras ei pruugi üks masin olla teadlik, et meedium on kasutusel. Kui andmete saatmisel tekkis kollisioon, siis peab andmed (kaadri näol) uuesti saatma. Mida enam on kollisioone, seda tugevamalt mõjutab see võrgu jõudlust.

Poolduplekс-edastust kasutades on maksimaalne ühenduskiirus jagatud kõikide võrgus olijate vahel. Paraku on see nii aga ainult ideaaljuhul, sest mida intensiivsemaks muutub võrguliiklus, seda suuremaks muutub kollisiooni tõenäosus. Seega on maksimaalne läbilaskevõime kindlasti tunduvalt madalam ühenduse teoreetilisest kiirusest. Esimese kihi võrguseadmed on oma loomuselt pooldupleksid.

Poolduplekѕvõrgus peab signaal igasse poolduplekѕvõrgu osasse levima mitte kauem kui standardis määratud aja jooksul. Vastasel korral on palju suurem oht kollisioonideks ning need võivad jääda saatja poolt avastamata.Tänapäeval kasutatakse repiiterite ja hubide asemel kommutaatoreid, mida vaatleme järgnevas peatükis "OSI kanalikiht" (lk. 35). Uute võrkude paigaldamisel on soovitatav kasutada kommutaatoreid. ube võib leida vanemates võrkudes või kasutatakse spetsiaalkasutuses (näiteks võrgu pealtkuulamiseks).



Joonis 5.17: IEEE 802.3 Etherneti tehnoloogiate mittetäielik ülevaatlik skeem. Joonisel olevaid kaablitüübi akronüüme: FB (*fiber backbone*), FL (*fiber link*), FP (*fiber passive*), SR (*short range*), ER (*extended range*), LR (*long range*), LW (*long wavelength*), SW (*short wavelength*), EW (*extra long wavelength*).

6. OSI kanalikiht

OSI kanalikiht (*data link layer*) asub füüsilise ja võrgukihi vahel. Kanalikiht kasutab füüsilise kihi poolt pakutavat bitivoo teenust. Kanalikiht ise pakub ülemisele võrgukihile põhiliselt füüsilise adresseerimise ja andmete korrektsuse teenust (füüsilise ülekande käigus tekkinud bittide tõlgendamise vigade avastamiseks). Mida ta konkreetselt teeb, see oleneb protokollist. Arvutiga paralleele tuues võib kanalikihti vaadelda ka seadmedraiverina.

Kanalikihi protokoll piiritleb infoplokkide piirid ja määratleb bittide tähendused kanalikihi jaoks. Piiritletud infoplokki nimetatakse **kaadriks** (*frame*). Kaader võib olla fikseeritud või dünaamilise pikkusega. Dünaamilise pikkusega kaadrite korral kasutatakse piiritlemiseks tavaliselt mingit bittide järjestust, mida kaadri sees välditakse. Lisades näiteks sobivasse kohta juurde nn. farssbiti (*bit stuffing*), et andmete sees ei oleks piiritlemiseks mõeldud bittide järjestust. Näiteks olgu piiritlemiseks kasutatavate bittide järjestuseks "01111110" ja kui soovitakse saata bitte "0110111111111011111", siis näeksid bitid kapseldatult välja järgmiselt "01111110011011111011110111111001111110". Antud näite puhul lisatakse iga viienda "1" järele üks farssbitt. Nõnda viskab vastuvõtja iga viienda biti tagant "0" ära. Kui seal oli "1", siis on tegu kaadri lõpu piiriga.

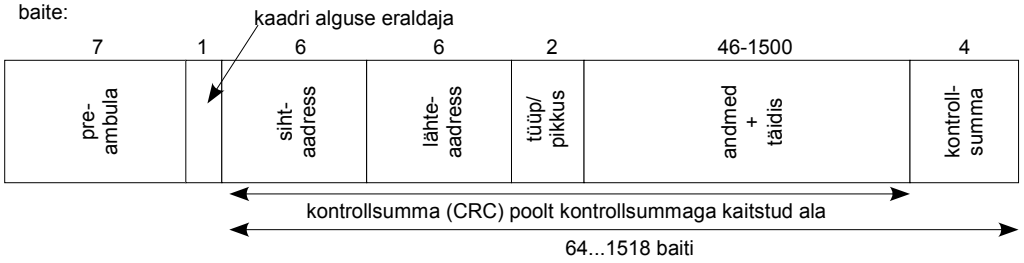
Bittide kogumikku, millel on ühtne tähendus, nimetatakse **väljaks** (*field*) (vaadeldavad bitid koos määravad midagi). Kaadriga saadetavatele andmetele lisatakse juurde päis (*header*) ja olenevalt protokollist võibolla ka saba (*trailer*). Päis sisaldab väljasid, mis annavad andmete kohta informatsiooni, näiteks kaadri puhul sihtaadress, lähteaadress, üle kantavate andmete hulk, kaadri kontrollsumma jms.

IEEE LAN standardi IEEE-802 mudelis jagatakse OSI kanalikiht kaheks, MAC (*Media Access Control*) ja LLC (*Logical Link Control*) (IEEE 802.2) alamkihtideks (LLC asub MAC peal). MAC alamkihi ülesanneteks on meediumile ligipääsu reguleerimine ja füüsiline adresseerimine (osaliselt katab OSI füüsilist kihti). LLC ülesanneteks on kaadri andmete võrgukihi protokolliga seostamine, vookontroll (kui kiiresti toimub edastus – oluline, kui saaja ei suuda saadetavat piisavalt kiiresti töödelda ning tulemusena saavad puhvrid täis, mille järel hakatakse järgnevaid kaadreid minema viskama), kadreerimine (kaadriteks tegemine) jms.

Meie vaatleme kohalikus võrgus enimkasutatavat tehnoloogiate perekonda Ethernet (kunagi olid levinud ka näiteks Token Ring ja FDDI, kuid enam neid praktiliselt ei kasutata). Kanalikihi protokollidena on olemas veel näiteks ATM, Frame Relay, HDLC, PPP jm. Etherneti kaadri ülesehitus on järgmine (lõpus on toodud välja pikkus) (vaata ka joonist 6.1):

- Preambula. Sisaldab vaheldumisi 1 ja 0 (101010...1010). Preambula annab vastuvõtjale teada kaadri algusest. Etherneti võrguliidesekaardid viskavad jooksvalt preambula ära. 7 baiti.
- Kaadri alguse eraldaja. Käsitletakse tihti ka preambula alamosana. Selle välja sisuks on "10101011", kus viimased kaks ühte signaliseerivad kaadri sisu algust. Etherneti võrguliidesekaardid viskavad selle jooksvalt ära. 1 bait.
- Sihtaadress. Sihtmasina füüsiline aadress ehk MAC (*Media Access Control*) aadress. 6 baiti.
- Lähteaadress. Lähtemasina füüsiline aadress ehk MAC-aadress. Kasutatakse selleks, et identifitseerida saatja, millele saab selle alusel vastata. Kaadrit saates pannakse lähteaadressiks masina enda MAC-aadress. 6 baiti.
- Tüüp/pikkus. Kui see on vahemikus 0-1500, siis tähistab see andmete osa pikkust. Kui ≥1536 (ehk 0x600), siis tähistab see tüübiinfot. Tuntumaid protokolle: 0x0800 on IPv4 (*Internet Protocol version 4*); 0x0806 on ARP (*Address Resolution Protocol*); 0x86DD on IPv6 (*Internet Protocol version 6*). Nimekiri tüüpide registreeringutest on aadressil http://standards.ieee.org/regauth/ethertype/eth.txt ja http://www.iana.org/assignments/ethernet-numbers. 2 baiti.

- Andmed. Kaadri andmed, mida edastakse. 0-1500 baiti. Maksimaalset infohulka, mida saab edastada vastava protokolliga andmeüksuses, nimetatakse MTU-ks (*maximum transmission unit*). Etherneti MTU on 1500 baiti.
- Täidis. Kui andmeid on vähem kui 46 baiti, siis lisatakse lõppu juurde nulle, et kaadri andmete osa pikkus oleks minimaalselt 46 baiti. 0-46 baiti.
- Kontrollsumma. Arvutatakse sihtaadressist kuni andmete osa lõpuni. Kui kontrollsumma ei klapi, siis visatakse kaader vastuvõtul minema. Kui hakkab tekkima liiga palju vigase kontrollsummaga kaadreid, siis võib see viidata vigasele Etherneti võrguliidesele, rikutud või vigastele draiveritele, halvasti paigaldatud kaabli otsikutele, vigasele kaablile, välisele mürale vmt. Kontrollsummana kasutatakse CRC-32. 4 baiti.



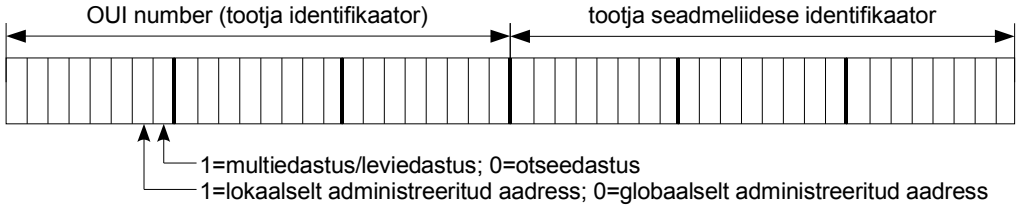
Joonis 6.1: Etherneti kaadri väljad. Väljade kohal olevad numbrid näitavad välja pikkust baitides.

MAC-aadresside üleskirjutamiseks on võimalik kasutada mitut notatsiooni. Meie kasutame kuju, kus kirjutame MAC-aadressi kuueteistkümnendsüsteemi arvudena kahe kaupa grupis, kus eraldajana kasutame koolonit, näiteks "00:19:D2:5F:29:D1".

MAC-aadressid on jaotatud järgnevateks gruppideks (kuuluvuse määravad esimese okteti kaks viimast bitti ehk MAC-aadressi seitsmes ja kaheksas bitt) (vaata ka joonist 6.2):

- Üksikedastusaadress (*unicast*). Need MAC-aadressid on mõeldud masinatel MAC-aadressidena kasutamiseks. Otseaadresside puhul on esimese okteti viimane bitt "0". Otseaadressid on ainukesed legaalsed kaadri lähteaadressid. Otseaadressid jaotatakse kahte alamgruppi:
  - ◇ Globaalselt unikaalsed aadressid. Võrguliidesed omavad üldjuhul MAC-aadressi antud hulgast. Globaalsetel MAC-aadressidel on esimese okteti kaks viimast bitti nullid ja ülejäänud suvalised. Aadressikujud kuueteistkümnendkujul on (tärnid tähistavad suvalisi numbreid vahemikus 0 kuni F): *0:**:**:**:**:**, *4:**:**:**:**:**, *8:**:**:**:**:**, *C:**:**:**:**:**.
  - ◇ Lokaalselt administreeritud aadressid (*locally administered addresses*). Need aadressid on määratud riistvara tootja asemel, võrguadministraatori poolt, et ta saaks teha oma aadressiskeemi. Siin on esimese okteti kaks viimast bitti "10" ja ülejäänud suvalised. Seega MAC-aadressid kuueteistkümnendkujul on: *2:**:**:**:**:**, *6:**:**:**:**:**, *A:**:**:**:**:**, *E:**:**:**:**:**. (Etherneti võrgus kasutatakse harva, kuid kasutati tihti Token Ring võrkudes).
- Leviedastusaadress (*broadcast address*). Leviedastusaadress on "FF:FF:FF:FF:FF:FF". Leviedastusaadressi puhul võtavad kõik masinad kaadri vastu ja annavad edasi OSI ülemisele kihile. Kasutatakse paljude protokollide puhul, kui ei ole teada sihtaadressi või kui kaader peaks jõudma kõigini. Ei ole kasutatav võrgukaardi aadressina.
- Multiedastuse aadressid (*multicast*). Multiedastuse aadressidel on esimese okteti viimane bitt "1" ja ülejäänud suvalised. Seega kuueteistkümnendkujul on multiedastuse aadressid *1:**:**:**:**:**, *3:**:**:**:**:**, *5:**:**:**:**:**, *7:**:**:**:**:**, *9:**:**:**:**:**, *B:**:**:**:**:**, *D:**:**:**:**:**, *F:**:**:**:**:** (välja

arvatud FF:FF:FF:FF:FF:FF). Üldjuhul käituvad kommutaatorid multiedastusaadressi korral sarnaselt leviedastusaadressiga, saates kaadri igasse porti. Masinad võtavad multiedastuse kaadri vastu ja saadavad ülemisele OSI kihile, kui ta kuulub vastavasse multiedastuse gruppi. Vastasel korral visatakse see minema. Multiedastusgruppi kuulutakse, kui võrguliideses on registreeritud vastav multiedastuse aadress (näiteks arvutis draiver registreerib). Etherneti multiedastuse aadresside nimekiri on saadaval lehel: http://www.cavebear.com/CaveBear/Ethernet/ multicast.html



Joonis 6.2: MAC-aadressi bittide tähendused. Kaheksas bitt on määravam kui seitsmes bitt.

Edaspidi MAC-aadressist rääkides lähtume vaikimisi lihtsalt globaalselt unikaalsest MAC-aadressihulgast, kui ei ole öeldud teisiti. MAC-aadressi puhul eeldatakse, et see on globaalselt unikaalne. Võrgukaarte, võrguseadmeid jm. seadmeid valmistavad paljud firmad, mis peavad panema unikaalsed MAC-aadressid igale seadmele. Unikaalsuse tõttu on MAC-aadressiruum jaotatud alamhulkadeks, kus MAC-aadressi 24 esimest bitti on igal tootjal erinevad. Neid numbreid haldab IEEE. Seda osa MAC-ist tuntakse organisatoorse unikaalse identifikaatorina ehk OUI (*Organizational Unique Identifier*) numbrina. Ülejäänud 24 bitti on tootja poolt määratud seerianumber, mis peaks igal seadmel erinev olema. Seega on võimalik OUI numbri järgi määrata võrgukaardi, marsruuteri jms. tootja. Alati ei saa selles kindel olla, sest kasutaja võib MAC-aadressi ise muuta, tootjad võivad edasi delegeerida neile eraldatud OUI alamosasid. Registreeritud OUI nimekiri on saadaval aadressil http://standards.ieee.org/regauth/oui/oui.txt.

Üldiselt on MAC-aadress salvestatud võrgukaardi ROM-i (*read only memory*), mis kaardi initsialiseerimisel kopeeritakse RAM-i (*random access memory*), kust ta kopeeritakse edasi iga väljuva kaadri lähteaadressiks. RAM-is olevat lähteaadressi on võimalik muuta, seega saab saata suvalise saatja aadressiga kaadreid.

Sihtaadressina on lubatavad kõik neli MAC-aadresside hulka (lähtuvalt vajadusest). Lähteaadressina on lubatavad ainult globaalselt unikaalsed ja lokaalselt administreeritavad aadressid.

Iga masin, mis saab võrguliidese kaudu kaadri, kontrollib, kas kaadri kontrollsumma klapib kaadri kontrollsumma välja väärtusega ja kaadri sihtaadressi, et kas see on mõeldud temale (masina enda MAC-aadress, leviedastusaadress või registreeritud multiedastusaadress). Kui see ei ole nii, siis visatakse kaader ära (kontrollil ei koormata arvuti keskset protsessorit, vaid kontrollimised tehakse tavaliselt võrgukaardi siseselt). Kui kontrollsumma klappis ja kaader oli mõeldud temale, siis kaadris sisalduvad andmed antakse edasi ülemisele kihile (võrgukihile).

OSI teise kihi seadmed

OSI teise kihi seadmeteks on **sild** (*bridge*), **kommutaator** (*switch*) ja ka arvuti **võrgukaart**. Tänapäeval on esimese ja teise kihi võrguseadmetest kasutusel kommutaatorid (hube, sildasid ja repiitereid eriti ei kasutata). Kommutaatorid on teise kihi seadmed, mis erinevalt hubidest ei saada kaadreid igasse porti, vaid ainult porti, kus antud sihtmasin asub (kellele kaader mõeldud on). Kaadri saatmiseks ainult ühte, õigesse porti, kasutab Etherneti kommutaator sihtkoha MAC-aadressi. Et teada saada, kuhu porti antud sihtaadressiga kaader saata, kasutab kommutaator vastavat tabelit, mis seab MAC-aadressi vastavusse pordiga. See tabel asub CAM-mälus

(*content-addressable memory*) (nimetatakse ka assotsiatiivmäluks), mille tõttu kutsutakse seda tabelit ka CAM-tabeliks. CAM-mälu töötab vastupidiselt tavalisele mälule (nt. arvuti RAM-mälu). CAM-mälule antakse ette andmed, mille peale tagastatakse tema andmete aadress. Kommutaatorite puhul antakse ette MAC-aadress ning saadakse kiiresti kätte vastav port, kasutamata spetsiaalseid otsingualgoritme. Tavamälu puhul antakse teatavasti ette mälupesa aadress ja saadakse tagasi seal aadressil olevad andmed.

CAM-tabel ehk MAC-tabel sisaldab üldjuhul kolme veergu: MAC-aadress, port ja aeg. Aeg näitab, millal viimati antud MAC-aadressilt liiklust täheldati. Üks MAC-aadress saab olla korraga ühes pordis ning ühes pordis võib olla üldjuhul mitu MAC-aadressi.

Öeldakse ka, et kommutaator loob kahe suhtleja vahel virtuaalse sideliini ehk punktist-punkti (*point-to-point*) sidekanali (jagatud meediumi puhul on iseloom üks mitmele). Samal ajal saavad ka teised pordid omavahel suhelda. Selliselt eraldab kommutaator kollisioonidomeenid. Kollisioonidomeenideks tükeldamist nimetatakse ka kollisioonidomeenide segmenteerimiseks.

Kommutaatorid toetavad täisdupleks- ja poolduplekss-ühendusi, eelistades esimest. Seega on kommutaatoritega võrgus kõik ühendused täisdupleksid ning seega ei ole sisuliselt ka kollisioonidomeene ja teoreetiliselt saaks kommutaatori igast pordist ja igasse porti andmed liikuda vastavalt ühenduskiirusele.

Kommutaator otsustab sihtaadressi põhjal temale saabunud kaadri edastamise. Kui sihtaadressiks on leviedastus- või multiedastusaadress, siis saadetakse kaader kõikidesse portidesse (v.a port, kust kaader pärines). Seda nimetatakse ka üleujutamiseks (sellest ka leviedastusdomeeni nimetus). Vastasel korral vaadatakse, kas taoline MAC-aadress eksisteerib MAC-tabelis. Kui eksisteerib, siis kasutatakse selle MAC-tabeli kirjesse märgitud porti, kus taoline MAC-aadress asub ja saadetakse kaader sellesse porti edasi. Kui MAC-tabelis sihtaadressina märgitud MAC-aadressi ei leidu, siis saadab kommutaator antud kaadri kõikidesse portidesse, välja arvatud porti, kust see kaader pärines. Kui kaadri sihtaadress asub saatjaga samas pordis, siis kommutaator viskab antud kaadri minema (tegu võis olla hubiga või oli tegu mingi kommutaatoriga, millel puudus vastav MAC-aadress CAM-tabelis). Mõned kommutaatorid võivad toetada ainult ühte MAC-aadressi ühe pordi kohta, kuid tavaliselt toetatakse mitut.

Kommutaator peab teadma, millises pordis mingi masin asub. Selleks kasutab Etherneti kommutaator kaadri lähteaadressi. See tähendab, et iga kaadri puhul, mida kommutaator vahendab, kontrollib ta kaadri edastamisel lähteaadressi olemasolu CAM-tabelis. Kui lähteaadress tabelis puudub, lisatakse antud MAC-aadress CAM-tabelisse, pordiks pannakse port, kust kaader vastu võeti ja vastuvõtu hetke aeg. Kui kommutaatoril oli CAM-tabelis lähteaadress olemas, kuid erineb port, siis asendatakse port CAM-tabelis ja värskendatakse aega. Kui CAM-tabelis oli vastav MAC olemas ja port oli ka sama, siis uuendatakse ainult aega. Seda protsessi nimetatakse õppimiseks. Kommutaator õpib dünaamiliselt teda läbiva liikluse põhjal. Võrguosa, mis on omavahel ühendatud ainult OSI esimese ja teise kihi seadmetega nimetatakse leviedastusdomeeniks (*broadcast domain*). See tähendab, et kaader võib oma teekonnal läbida mitut kommutaatorit ning muid esimese ja teise kihi seadmeid. Leviedastusdomeenis olevad masinad suhtlevad omavahel otse, väljaspool asuvate masinatega nad ei saa otse suhelda (suhtlus toimub läbi OSI kolmanda kihi võrguseadme).

Leviedastusdomeenis võib olla palju masinaid, mis võivad vahetuda suvaliselt, mille tõttu tekib vajadus vananenud kirjete eemaldamiseks CAM-tabelist (iga mälu on lõplik). Selleks kasutataksegi aja märkimist. Kui mingi MAC-aadressi pealt ei ole saadetud mitte ühtegi kaadrit fikseeritud aja jooksul, siis vastava MAC-aadressiga rida kustutakse CAM-tabelist. Seda fikseeritud aega kutsutakse aegumise ajaks. Tavaliselt on kommutaatoritel aegumise ajaks mitu minutit.

Olukorda, kus saadetakse ülisuurtes kogustes leviedastusaadressiga kaadreid, nimetatakse leviedastustormiks (*broadcast storm*). Leviedastustorm võib ära kasutada kogu ribalaiuse, mistõttu võib võrgus tekkida ummistusi, samuti peab leviedastuskaadreid töötlema kõikide masinate keskne protsessor, sest kaadri andmed antakse edasi kolmandale kihile.

Kommuteerimine jaotatakse kolme liiki:

- **Vahehoidega edastus** (*store-and-forward*) – võetakse vastu kogu kaader, kontrollitakse kontrollsumma alusel, ega kaader vigane pole. Kui on vigane, siis visatakse see ära. Kui kaader pole vigane, saadetakse kaader edasi vastavasse porti. Mida suurem on kaader, seda suurem on viivitus.
- **Vooledastus** (*cut-through*) – kõige kiirem moodus kaadrite edastamiseks. Selles laadis töötav kommutaator ootab ära, kuni jõuab kohale sihtaadress, mille järel hakkab kaadrit edasi saatma vastavasse porti, kus antud MAC asub. Seega on viivitus 8+6 baidi jagu. Enne edastamist veakontrolli ega midagi muud ei tehta.
- **Fragmenditu** (*fragment-free*) – erinevalt vooledastus laadis töötavast kommutaatorist oodatakse ära kaadri esimesed 64 baiti, mis on minimaalne kaadri pikkus. Vastasel juhul on tegemist vigase/pooliku kaadriga (näiteks kollisiooni tagajärg).

Lisaks neile kolmele tüübile on olemas ka hübriidkommuteerimine, mis töötab vaikimisi vooledastuse laadis, kuni vigade sagedus ületab fikseeritud piiri. Seejärel lülitub kommutaator ümber vahehoidega edastuslaadi. Kui vigade arv langeb alla fikseeritud piiri, siis lülitutakse ümber vooledastuslaadile. Seda nimetatakse kohanduvaks vooledastuse (*adaptive cut-through*) laadiks. Vooledastusega ja fragmenditu edastusega kommutaatorid on keerulisema riistvaraga kui vahehoidega edastusega kommutaatorid.

Kommutaatorite puhul on olemas ka väljundpordi puhvrid, mida kasutatakse, kui port on hetkel hõivatud (näiteks mingi muu port saadab antud porti andmeid). Puhver on üldiselt tavaline mälu, mis võib olla pordipõhine või jagatud mälu. Jagatud mälu puhul on võimalik kasutada suuremat puhvrit, kuid samas võib teiste portide jaoks puhver täis olla ning edastamist ei saa läbi viia (konkurents mälu osas).

Kommutaatorid võib jagada sümmeetrilisteks ja asümmeetrilisteks. Sümmeetrilise kommutaatori puhul toimub ühest pordist teise toimuv infovahetus sama pordikiirusega. Asümmeetrilise kommutaatori puhul saavad pordid olla erineva läbilaskevõimega (sama läbilaskevõimega portides toimub infovahetus nagu sümmeetrilisel kommutaatorilgi). Näiteks võib üks port olla 10 Mbit/s ja teine 100 Mbit/s. Asümmeetrilise edastuse puhul on vajalik vahehoidega edastuse kasutamine. Suurema läbilaskega pordid on kasulik jätta serverite ja magistraalühenduste tarvis, kus on rohkem liiklust.

Veel üheks OSI teise kihi Etherneti seadmeks on sild (*bridge*). Sild on sarnane kommutaatorile, kuid erineb selle poolest, et sild on arhitektuuriliselt mõeldud töötama jagatud meediumiga LAN-des (sild on tavaliselt ühendatud hubiga), kuid kommutaator on mõeldud otse arvuteid ühendama. Sild ei ole mõeldud vahendama paralleelselt toimuvat liiklust. Sild opereerib ainult vahehoidega edastuse laadis (sest antud hetkel võib teises liideses olla liiklus või toimub kollisioon). Silda kasutati varasemal ajal, mil kommutaatoreid ei olnud võimalik kasutada, sest tehnoloogia ei olnud selleks veel võimeline või olid kommutaatorid liiga kallid. Üldjuhul oli leviedastusdomeenis valdav enamus liiklusest piirkondade sisene, millest väljapoole minevat liiklust oli vähem ja selle viivitus võis olla ka natuke pikem. Tänapäeval üldjuhul enam sildasid uutes installatsioonides ei kasutata. Silla kasutuseesmärk oli kahe kollisioonidomeeni eraldamine, et nad ei oleks liiga suured, ja ka turvalisuse kaalutlustel (liikluse pealtkuulamise ohu vähendamiseks). Kommutaatorit nimetatakse ka mitme pordiga sillaks.

Tavalise lihtsa kommutaatori võib panna võrku ilma seadistamiseta. Lihtsaks kommutaatoriks nimetatakse kommutaatorit, millel ei ole kasutaja poolt seadistamise võimalust. Nad hakkavad võrgus olevate masinate asukohtasid automaatselt ise õppima. Kommutaatori kaadrite edastus on üldiselt teostatud riistvaraliselt (mitte tarkvaraliselt), mis on tunduvalt kiirem. Kommutaatorid ühendatakse teiste masinatega sarnaselt hubiga. See tähendab, et ühendades kommutaatorit teise hubi või kommutaatoriga, kasutatakse ristkaablit. Teiste seadmetega (näiteks arvuti, printeri, marsruuteriga) ühendamiseks kasutatakse otsekaablit.

Paljud uued võrguseadmed ja võrguliidesed on automaatse MDI/MDIX võimekusega, mis

võimaldab ühendada vastavat võrguliidest suvalise muu masinaga nii otse- kui ristkaablit kasutades (riistvaraliselt automaatselt muudetakse, milliseid traate kasutatakse saatmiseks ja milliseid vastuvõtmiseks).

Ainult lihtsaid esimese ja teise kihi seadmeid ei tohi ühendada tsükliliselt. Kui esimese kihi seadmed ühendada tsükliliselt, siis on lihtsasti nähtav, et näiteks leviedastusaadressiga kaadri saatmisel tsükliga võrku jääbki ta liikuma ringiratast. Kui leviedastusdomeenis on kaks või enam tsüklit, siis leviedastusaadressiga kaader "paljuneb" ja kaadreid tekib aina juurde. Ringliikluse tõttu ei ole võrk enam varsti praktiliselt üldse kasutatav. Sellises olukorras aitab tsükli lõhkumine ehk mingi tsüklis oleva sideliini eemaldamine (osas "STP", lk. 114 vaatleme selle probleemi lahendust STP-protokolli näol).

7. OSI võrgukiht

Kanalikihti kasutatakse kaadri saatmiseks järgmisele füüsilisele osapoolele või osapooltele leviedastusdomeeni piires. Nüüd vaatleme loogilist adresseerimist, mille ülesanne on toimetada andmed suvalisse võrgupunkti (mis Interneti puhul võib asuda üle kogu maailma). Selle käigus on vaja andmeid mingite kriteeriumite alusel suunata sihtpunkti poole ehk marsruutida. Andmeühikuid kolmandas OSI kihis nimetatakse pakettideks.

Kõige enamkasutatav kolmanda kihi protokoll tänapäeval on IP (*Internet Protocol*) (RFC 791, 823). IP protokoll on TCP/IP mudeli südameks. Loogilist ehk kolmanda kihi aadressi ei ole masinasse sisse kirjutatud, vaid see määratakse muud moodi.

IP on adresseerimisel kasutatavaks protokolliks Internetis. IP on marsruuditav protokoll (*routed protocol*). See tähendab, et edastatav andmeühik ehk pakett on võrgusõlmedel suunatav ühelt sõlmelt teisele, kuni pakett jõuab sihtpunkti (võrgusõlmed suunavad paketti järjest lähemale sihtpunktile). Peale IP on marsruuditavateks protokollideks veel näiteks IPX/SPX (*Internetwork Packet Exchange/Sequenced Packet Exchange*), AppleTalk, DECnet (*Digital Equipment Corporation Network*), XNS (*Xerox Network Services*). Mittemarsruuditavaks protokolliks on näiteks NetBEUI (*NetBIOS* (*Network Basic Input/Output System*) *Extended User Interface*). IP võrgus olevat masinat, millel on unikaalne IP-aadress, nimetatakse **hostiks**. Hostiks võib olla tavaline arvuti, kuid ka server, võrguprinter, võrguseade, külmkapp jne. – kõik seadmed, millele saab määrata IP-aadressi ja millega saab suhelda.

Võrgukihiaadressi järgi pakette edasisuunavaid võrguseadmeid nimetatakse marsruuteriteks. Erinevalt teise kihi seadmetest kommutaatoritest, mis õpivad läbiva liikluse pealt (missugune MAC-aadress kuhu porti kuulub), on marsruuteritel vajalik seadistada, millise sihtaadressivahemiku puhul millisele naabrile pakett edasi saata. Marsruuterid on võrguseadmed, mis ühendavad erinevaid võrke.

IP protokollist on kasutusel kaks versiooni. Esimene on IPv4 (*IP version 4*), mis on praegu valdavalt kasutusel. Teine kasutusel olev protokoll on IPv6, mis peaks tulevikus tasapisi asendama IPv4. Meie keskendume praegu IPv4 kasutavale võrgule. IPv6 vaatleme eraldi peatükis "IPv6" (lk. 103) (IPv5 oli ebaõnnestunud eksperiment). IP protokolli pakett on olemuselt datagramm (*datagram*). **Datagramm** on pakett, mis sisaldab saatja poolt määratud siht- ja lähteaadressi ning mida saab iseseisvalt suunata sihtkohta.

Abiprotokollid, mis kuuluvad IP juurde, on ICMP (*Internet Control Message Protocol*), IGMP (*Internet Group Management Protocol*), ARP (*Address Resolution Protocol*) ja RARP (*Reverse Address Resolution Protocol*). ICMP on põhiliselt probleemidest teavitamiseks. IGMP on multiedastusgruppide haldamiseks (vaatame lähemalt peatükis "Multiedastus" (lk. 86)). ARP ja RARP on IP ja kanalikihi aadresside sidumiseks. ARP ja RARP protokolle nähakse tihti ka iseseisvate protokollidena, sest nad ei kasuta IP datagramme.

Iga arvuti, mis IP võrgus asub, peab antud võrgus olemise ajal omama unikaalset aadressi. IP-aadress on midagi sarnast näiteks postiaadressile või telefoninumbrile. Igaüks, kellel on telefon, omab mingit unikaalset identifikaatorit, millele helistades jõuab kõne just ainult temani või jõuab kiri üheselt määratud aadressile. See tagab, et suvaline host saaks suhelda suvalise teise hostiga, kui ta teab tema aadressi. IP-aadresside kasutust reguleerib organisatsioon nimega IANA (*Internet Assigned Numbers Authority*). IANA koduleht: http://www.iana.org.

IP protokoll on ühenduseta, see tähendab, et mingit eraldi ühendust ei looda enne paketi teele saatmist. Seega paketi saatja ei pruugi IP protokolli tasemel ette teada, kas pakett jõuab kohale või mitte või kas sihtpunkti üldsegi on olemas (näiteks IP-aadress ei ole kasutusel või on arvuti välja lülitatud). Samuti ei pruugi paketi liikumistee olla alati samasugune, see tähendab, et üks pakett võib liikuda ühte teed pidi, aga teine pakett teist teed pidi sihtpunkti. Sellest lähtuvalt võib hiljem teele pandud pakett jõuda sihtpunkti varem kui temast varem teele saadetud pakett, sest iga pakett on iseseisev ja sõlmed teevad edastamisotsused jooksvalt antud hetkel eeldatavalt

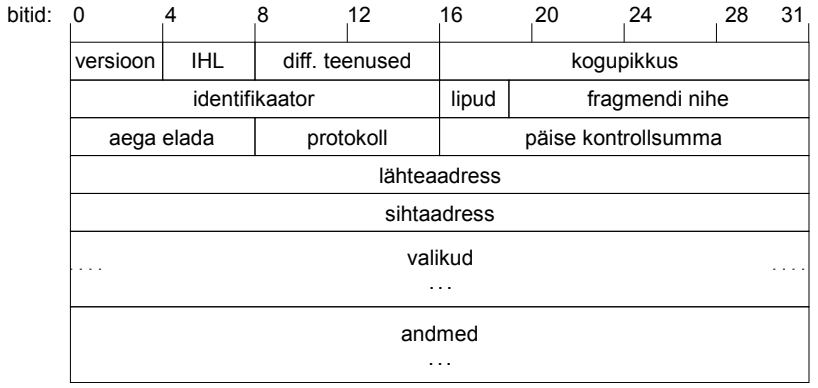
parimat teed valides (mingite kriteeriumite alusel). Ühendusele orienteeritud ühendusviisi puhul luuakse ühenduskanal mõlema otspunkti vahel, seejärel saab andmeid vahetada osapoolte vahel. Paralleelina on tavalise paberkirja saatmine ühenduseta andmete saatmise viis (eelnevalt ei looda osapoolte vahel kanalit, seega võib kiri ära kaduda, adressaat mitteeksisteeriv olla jne.). Ühendusele orienteeritud ühendus on seevastu tavaline telefoniühendus, kus luuakse otspunktide vahel ühendus, mille järel hakatakse sõnumeid vahetama. Ühenduseta ühendust kutsutakse tihti ka pakettkommuteeritud (*packet-switched*) meetodiks ja ühendusega ühendust lülituskommuteeritud (*circuit-switched*) meetodiks.

IP protokolli pakett sisaldab järgmisi välju (vt. ka joonist 7.1):

- Versioon. Näitab IP paketi päise formaati. IPv4 puhul on siin bittideks "0100" ehk kümnendsüsteemis 4. IPv6 puhul bitid "0110" ehk kümnendsüsteemis 6. Numbrid tulenevad versiooni numbritest. Praktikas ei kasutata seda välja eristamaks IPv4 ja IPv6 pakette, eristamine tehakse OSI teise kihi protokollis. 4 bitti.
- Päise pikkus ehk IHL (*Internet header length*). Näitab IP paketi päise pikkust, kus ühikuks on 32 bitti ehk mitmest 32-bitisest sõnast päis koosneb (sest üks päise väli on muutuva suurusega). Minimaalne väärtus on 5. 4 bitti.
- Diferentseeritud teenused (*differentiated services*) (RFC 2474). Pakette on võimalik jagada erinevatesse teenusteklassidesse, mida siis vastavalt klassile koheldakse erinevalt (teenuseklass määratakse kuue esimese bitiga). Mõnedel pakettidel on vaja väikest viivitust, kuid teiste puhul kohale jõudmise usaldusväärsust. Kaks viimast bitti on ummistustest teavitamiseks (ECN (*Explicit Congestion Notification*) (RFC 3168) näol). Kasutusele võeti nad seoses reaalajalise heli ja video edastamise vajadusega. Kõik seadmed ei pruugi seda veel toetada. Algselt nimetati seda välja teenuse tüübiks ehk ToS (*type of service*), mis tähistas paketi tähtsuse taset. Tavaliselt seda ei kasutatud, sest kasutaja programm saab paketile märkida kõige kõrgema tähtsuse taseme, mistõttu kadus välja mõte. 8 bitti.
- Kogupikkus. Määrab kogu paketi suuruse baitides (päis ja andmed kokku). 16 bitti.
- Identifikaator. Identifitseerib paketi, olles järjekorranumbriks. Fragmenteeritud paketid omavad ühist identifikaatorit kõigis fragmentides. 16 bitti.
- Lipud. Kokku 3 bitti. Lippude tähendused:
  - ◇ Reserveeritud bitt, mis peab olema null.
  - ◇ Ära fragmenteeri ehk DF (*don't fragment*) – määrab, kas paketti lubatakse fragmenteerida (teha mitmeks jupiks). Fragmenteerida on vaja näiteks siis, kui pakett on suurem kui edastataval sideliinil määratud MTU. Kui lipp on püsti, aga on vaja fragmenteerida, siis visatakse pakett minema ja saadetakse saatjale vastavasisuline ICMP teade (sihtpunkt kättesaamatu).
  - ◇ Veel fragmente ehk MF (*more fragments*) – kui paketti on vaja fragmenteerida, siis pannakse fragmentidele püsti MF lipp, välja arvatud viimasele fragmendile. Sedasi saab teine pool fragmentidest uuesti suurema paketi kokku panna. Paketil, mis ei ole fragmenteeritud, on see lipp maas (nagu viimasel fragmendil).
- Fragmendi nihe (*fragment offset*) – fragmendi andmete alguskoht, mille alusel saab fragmenteeritud paketi uuesti kokku panna. Andmed jaotatakse loendamiseks kaheksabaidisteks plokkideks. Esimesel fragmendil on selle välja väärtus null, teisel fragmendil on selle väärtuseks esimese fragmendi andmete suurus jagatud kaheksaga pluss üks jne. Mittefragmenteeritud pakettidel on väärtuseks null. 13 bitti.
- Aega elada ehk TTL (*time to live*). Näitab, mitu hüpet (üheks **hüppeks** nimetatakse ühte marsruuteri läbimist) saab pakett veel teha, et sihtkohta kohale jõuda. Iga hüppega vähendatakse loendurit ühe võrra. Kui väärtus jõuab nullini, siis pakett visatakse minema. Sellega garanteeritakse, et ei teki olukorda, kus mingi pakett jääb lõputult ringlema (näiteks kui kusagil võrgus on tekkinud marsruutimistsükkel). Kui loendur jõuab nulli, siis saadetakse lähteaadressile "Time exceeded" tüüpi ICMP veateade. Algselt oli see

mõeldud nii, et loetakse puhvris oldud sekundeid, kuid seda on keerukas realiseerida. Tihti pannakse algselt TTL väärtuseks 64 või 128 (võimalik 0-255). 8 bitti.

- Protokoll. Määrab ülemise kihi protokolli. Tuntumatest näiteks TCP=6, UDP=17, ICMP=1, IGMP=2, IP=4, Ethernet=97, IPX=111. See on vajalik, et teada, mis protokolli abil tuleb paketis olevaid andmeid interpreteerida. Selline kapseldamine võib esmapilgul tunduda raiskamisena, kuid tegelikult võib see olla väga vajalik, näiteks VPN (*virtual private network*) jaoks (vaatleme hiljem pikemalt), IPv6 paketi saatmiseks üle IPv4 võrgu. Ametlik protokollide numbrite nimekiri on saadaval aadressil http://www.iana.org/assignments/protocol-numbers. 8 bitti.
- Päise kontrollsumma. Arvutatakse ainult IP päise põhjal. Kuna "aega elada" väli iga hüppega muutub, siis on vajalik peale igat hüpet see uuesti arvutada. 16 bitti.
- Lähteaadress. Paketi alglähetajahosti IP-aadress. 32 bitti.
- Sihtaadress. Paketi sihthosti IP-aadress. 32 bitti.
- Valikud. Võimaldab IP-l toetada erinevaid fakultatiivseid lisavõimalusi. Tihti ei kasutata. Kui valikubittide arv ei jagu 32-ga (4 baiti), siis lisatakse parajas koguses juurde täitebitte, et jaguks 32-ga. Täitebittideks on nullid. Näited võimalikest valikutest oleksid: turvalisus, veateated, ajatembeldus ja silumine jne. Valikute väli on varieeruva pikkusega, kuid jagub 32-ga. Seega pikkus on 0, 4, 8, 12, 16, 20, 24, 28, 32, 36 või 40 baiti.
- Andmed. Andmed, mis paketis sisalduvad. Paketi maksimaalne pikkus saab olla 65535 baiti (koos päisega).



Joonis 7.1: IPv4 paketi sisu.

Paketi suurus võib olla suurem kui vastaval sideliinil kasutatava teise kihi protokolli suurim edastatav andmehulk, mida nimetatakse MTU (*maximum transmission unit*). Näiteks Etherneti puhul on MTU 1500 baiti. Et paketti saaks siiski edastada väiksema MTU-ga sideliinide puhul, siis kasutatakse paketi fragmenteerimist, mis jagab paketi tükkideks ja saadab nad eraldi. Selleks kasutatakse väljasid "lipud" ja "fragmendi nihe". IP paketi enda maksimaalne suurus on 65 535 baiti. Seega, kui saadetakse pakett, mis on maksimumsuurusega, siis Etherneti puhul tuleks ta minimaalselt 44 fragmendiks jagada. Paketi kokkupanek fragmentidest toimub tavaliselt lõpphostis (võib ka marsruuteris/võrguväravas, kui võrk on suurema MTU-ga). Kasutades transpordiprotokollina TCP-d, jäetakse IP paketi fragmenteerimisel TCP päis ainult esimesele IP paketi fragmendile.

ICMP

IP protokoll on oma olemuselt ebausaldusväärne protokoll, sest väljasaadetud paketi edasise käekäigu kohta ei ole saatjal otseselt mingit kontrolli ning saatja ei tea, kas teine pool on kätte-

saadav (näiteks sihtpunkti ei eksisteerigi, tee peal ei osata antud paketti mitte kuhugi edasi suunata, sihtpunkt ei toeta vastavat protokolli vms.). IP protokollil on probleemolukordade jaoks laiendus ICMP (*Internet Control Message Protocol*) (RFC 792, 1122). ICMP protokolli kasutatakse IP võrgus lähtehosti teavitamiseks veaolukordadest (lisaks ka informeerimiseks). ICMP on üks tuumkomponentidest TCP/IP protokollistikus. ICMP on samuti ebausaldusväärne, sest ICMP pakettide kohalejõudmine ei ole samuti tagatud. ICMP pakettidega seotud vigade puhul ei teavitata toimuvatest vigadest mitte kedagi (vastasel korral võiks tulla lõputu veateadete edasi-tagasi saatmine).

ICMP protokolli teate struktuur on olenevalt teate tüübist erinev. Kõikidel teadetel on samasugused kolm esimest välja:

- Tüüp. Määrab teate tüübi. 8 bitti.
- Kood. Täiendav informatsioon vastavalt teate tüübile. 8 bitti.
- Kontrollsumma. 16 bitti.

ICMP puhul on olemas mitmeid teadete tüüpe. Nendest kasutatavamad on (sulgudes ICMP tüübi number):

- Kaja päring (*echo request*) (8), kaja vastus (*echo reply*) (0) – kasutatakse ühenduse olemasolu kontrollimiseks kahe otspunkti vahel (peab mõlemat pidi toimima). Üks pool saadab teisele kaja päringu ICMP teate. Teine pool vastab sellele kaja vastuse teatega. Niimoodi saab teada, kas ühendus on võrgukihi tasandil olemas. Samuti saab teada, kui kaua võtab edasi-tagasiedastus aega. Koodiväli peab olema null.
- Sihtpunkt kättesaamatu (*destination unreachable*) (3) – marsruuter ei saa mingil põhjusel datagrammi edasi saata. Näiteks on vastava võrgu liides maas (ei ole funktsioneeriv); paketti on vaja fragmenteerida, kuid on seatud IP protokolli lipp "ära fragmenteeri"; marsruuteri vigaselt häälestamisest tulenevad probleemid; sihtmasina võrguvärav, mis peaks paketi kohale toimetama, ei oma ARP-tabelis vastavale IP-aadressile vastavat MAC-aadressi ja ei saa ka ARP-päringule vastust (näiteks on masin välja lülitatud). Koodiväli annab täpsemalt teada, mis oli probleemiks. Näiteks: 0 – võrk oli kättesaamatu; 1 – host oli kättesaamatu; 2 – sihtpunkt ei toeta protokolli; 3 – port ei olnud kättesaadav; 4 – fragmenteerimine oli vajalik, kuid keelav lipp oli püsti pandud; jne.
- Eluaja ületamine (*time exceeded*) (11) – "aega elada" väli IP päises oli saanud nulliks, mille tõttu visati pakett ära (ICMP paketi puhul ei saadeta vastavat teadet).
- Marsruudi ümbersuunamine (*route redirection*) – antakse teada hostile, kui kohalikus võrgus on mingi muu parem marsruut sihtpunkti.
- Allika summutamine (*source quench*) – kasutatakse teavitamaks saatjat, et peataks ajutiselt edastamise, sest puhvrid on täis saanud ja pakette visatakse minema. Samuti võib selle teate saata sihthost, kui ta ei jõua nii kiiresti töödelda. Enam praktiliselt ei kasutata, sest tekitab niigi ummistunud võrku veel liiklust.

Ametlik nimekiri ICMP tüüpidest asub aadressil http://www.iana.org/assignments/icmp-parameters. Kaja päring on ainuke ICMP tüüp, mida otseselt initsieerib kasutaja. Seda tuntakse PING (*packet Internet groper*) nime all.

IP-aadress

IP-aadress on kokku 32-bitine number. Inimesele loetavamaks ja paremini käideldavamaks kujuks on IP-aadress, kus ta on jaotatud neljaks osaks (igas osas kaheksa bitti), mis eraldatakse omavahel punktidega ja kirjutatakse enamasti kümnendsüsteemi arvudega. Neid osasid nimetatakse **oktettideks** (koosneb kaheksast bitist). Iga okteti väärtuseks on 0-255 (2^8-1=255; -1, sest loendamist alustatakse nullist). Seega on IP-aadressi üldkuju järgmine: "A.B.C.D", kus A, B, C, $D \in \{0, 1,..., 255\}$.

Igas internetis peavad kõikide hostide IP-aadressid olema unikaalsed, Internetis samuti. Kuna IP-aadressi alusel peab olema võimalik saata paketti suvalisele aadressile internetis, kus võib olla väga palju hoste, siis on vaja adresseerimine loogiliselt üles ehitada.

IP-aadress jaotatakse kahte ossa: võrgu- ja hostiosa. Võrguosa bitid on IP-aadressis eespool olevad bitid ja hostiosa bitid on tagumised bitid. **Võrguaadressiks** (*network address*) nimetatakse IP-aadressi, kus hostiosa bitid on kõik nullid. Võrgu **leviedastusaadressiks** (*broadcast address*) (RFC 919) nimetatakse IP-aadressi, kus hostiosa bitid on kõik ühed. Võrgu kasutatavateks IP-aadressideks nimetatakse kõiki IP-aadresse, mis jäävad võrguaadressi ja leviedastusaadressi vahele. Võrguaadressil, kasutatavatetel aadressidel ja leviedastusaadressil on võrguosa bitid samasugused, kuid hostiosa bitid on erinevad. Iga leviedastusdomeen omab oma unikaalset võrguaadressi ehk siis igal leviedastusdomeenil on unikaalsed võrguosa bitid, mida ei ole mitte kusagil mujal Internetis. Ühe mingi leviedastusdomeeni siseselt on kõik hosti osa bitid samuti unikaalsed. Seega mitte ükski arvuti Internetis ei saa omada sama IP-aadressi. Edaspidi käsitleme IP-d Interneti kontekstis.

Võrgu- ja hosti aadressiosa piiritlemiseks kasutatakse võrgumaski (*network mask*). Võrgumask on samuti 32 bitti pikk, nagu IP-aadresski. Võrgumaskis on võrguosa bittide osas kõik ühed ja hosti osas kõik nullid. Seega võrgumask on binaarsel kujul 1...10...0. Võrgumask näitab, millised bitid IP-aadressis on võrguaadressi osa ja millised bitid on IP-aadressis suvalised, et vastav IP-aadress kuuluks vastavasse võrku.

Algselt oli võrguosa ainult esimene oktett. Seega sai eksisteerida ainult 256 võrku. Siis jagati IP-aadressiruum kolmeks erinevate suurustega võrkudeks, mida nimetati **aadressiklassideks**. Nendeks klasside nimetusteks on A, B ja C, kus A-klassi võrguaadress oli väga suurte võrkude jaoks, B-klass suurte võrkude jaoks ja C-klass väiksemate võrkude jaoks. Lisaks on kaks spetsiaalkasutusega klassi D ja E. D-klassi aadressid on multiedastus- (*multicast*) aadressid. E-klassi kuuluvad aadressid on reserveeritud, neid kasutatakse uurimis- ja arendustegevuseks. Multiedastust vaatleme pikemalt eraldi peatükis "Multiedastus" (lk. 86). IP-aadresse, mis üheselt määravad masina, nimetatakse **üksikedastusaadressideks** (*unicast address*). Klassidesse A, B ja C kuuluvaid aadresse kasutatakse üksikedastusaadressidena.

Klassi A aadresside puhul on binaarkujus kõige esimene bitt 0. Klassi B aadresside puhul on esimesed bitid "10". Klassi C aadresside puhul on esimesed bitid "110". Klassi D aadresside puhul on esimesed bitid "1110". Klassi E aadressid on kõik ülejäänud ehk siis nende esimesed bitid on "1111". Klassi A võrgumaskiks on "255.0.0.0", mis on kahendkujul "11111111.00000000.00000000.00000000". B-klassi võrgumaskiks on 255.255.0.0 ja C-klassi võrgumaskiks on 255.255.255.0.

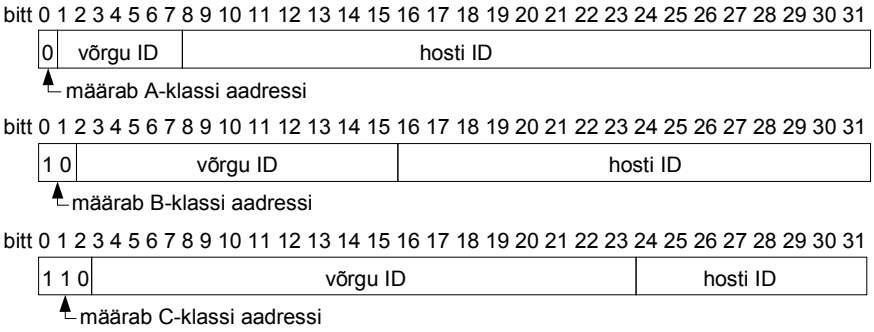
Tabel 7.1. Aadressiklasside võrdlused.

Klass	**Esimesed bitid**	**Mask**	**IP-aadresse võrgus**	**Võrkude arv**	**Esimene oktett**
A	0	/8	2^{24}=16 777 216	2^{7}=128	0-127
B	10	/16	2^{16}=65 536	2^{6+8}=16 384	128-191
C	110	/24	2^{8}=256	2^{5+16}=2 097 152	192-223
D	1110	-	-	-	224-239
E	1111	-	-	-	240-255

Siit võime välja arvutada erinevate klasside võrkude arvud. Näiteks võtame B-klassi. Esimeses oktetis on kaks esimest bitti määratud ja ülejäänud kuus määramata, lisaks teises oktetis on võrguosas veel 8 bitti, järelikult on olemas 2^{6+8} = 64·256=16384 B-klassi võrku. Sama tulemuse saame ka, kui teame esimese okteti algus- ja lõppväärtusi ning arvestame teist oktetti. Seega (191-128+1)·(255-0+1)=64·256=16384 (algus- ja lõpuokteti peame kaasa arvama). Igas B klassi võrgus saame IP-aadresse kokku arvutada, maski vaadates. Seal on hosti osas 16 bitti. Seega IP-aadresse on kokku 2^{16} = 65 536 tükki. Võib ka arvutada oktettide kaupa. Neljandas oktetis on 8

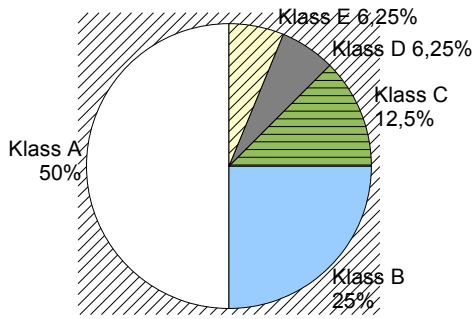
bitti ehk $2^8 = 256$. Kolmandas oktetis samuti 8 bitti. Seega $256 \cdot 256 = 65536$. Vaata ka tabelit 7.2. Joonisel 7.2 on toodud erinevate aadressiklasside bittide jaotus. IP-aadressiruumi jaotus klasside vahel on toodud diagrammil joonisel 7.3 (vaata ka tabelit 7.1).

Võrgumaski märkimiseks kasutatakse ka kaldkriipsu (*slash*) formaati, kus võrguaadressi järel asub kaldkriips, mille järel olev number näitab, mitu bitti on võrgu osas (inimesele mugav). Nt. A-klassi aadressimask on "/8", B-klassi aadressil "/16" ja C-klassi aadressil "/24". Näiteks kirjutades "199.203.12.42/24", on üheselt selge IP-aadress, mask, võrguaadress ja leviedastusaadress (vastavalt: "199.203.12.42", "255.255.255.0", "199.203.12.0", "199.203.12.255").



Joonis 7.2: Bittide jaotused A, B ja C aadressiklasside puhul.

Olukorda, kus IP-aadress oma klassi kuuluvuse järgi omab vastavat võrgumaski, nimetatakse klassidega adresseerimiseks (*classful addressing*). See tähendab, et klassidega adresseerimise juures määrab IP-aadress ära, milline on tema võrgumask (sellest lähtuvalt ka võrgu suuruse).



Joonis 7.3: IP-aadresside hulga jagunemine aadressiklasside vahel.

Aastaks 1992 hakkas Interneti väga kiire laienemise tõttu IPv4 aadressiruum A- ja B-klassi võrkude osas otsa saama. Järjest rohkem võeti kasutusele C-klassi aadresse, kuid see ajas Interneti magistraalmarsruuterite marsruutimistabelid liiga suureks. Lahendusena avaldati 1993. aastal ja hakati rakendama IP-aadresside jaotamisskeemina CIDR (*classless inter-domain routing*) (RFC 1519) meetodit. CIDR kaotas ära fikseeritud suurusega võrkude klassid, tuues sisse määratava suurusega võrgumaski, millega saab paindlikumalt eraldada IP-aadresse (näiteks IP-aadresse jagavatele organisatsioonidele) ja luua erineva suurusega võrke ning marsruuteritel agregeerida võrkusid, mis aitab vähendada marsruutimistabeli kirjete hulka (agregeeritud võrke nimetatakse supervõrkudeks (*supernet*)). Kuna CIDR kaotas ära aadressiklassid, muutus võrgumask väga oluliseks (enne oli klasside puhul aadressist tuletatav aadressiklass ja seega ka võr-

gumask). CIDR tehnika alusel võidakse asutusele jagada näiteks võrk 89.90.91.128/26 (ehk maskiga 255.255.255.192) (muutuva suurusega võrgumaski käsitleme lähemalt VLSM tehnika juures). Nii saadakse optimaalsem IP-aadresside kasutamine, sest enamus võrkudest ei kasutanud kogu vastava klassi võrgu IP-aadresse ära. Näiteks kui väiksemale asutusele anti C klassi võrk ja asutus vajas ainult 25 IP-aadressi, siis ligi 90% IP-aadressidest raisati. Marsruuterid said mitut järjestikust võrku agregeerida, tehes marsruutimistabelid väiksemaks. CIDR oli vajalik ka põhjusel, et fikseeritud suurusega võrgud hakkasid ammenduma. CIDR muutis ka Interneti aadresside jagamise skeemi rohkem hierarhilisemaks. Klassi B võrgud oleksid ammendunud umbes 1990. aastate keskel. CIDR puudutas ainult üksikedastusaadresse, kuid mitte D- ja E-klassi aadresse. Edaspidi lähtume vaikimisi CIDR-st. Samas on oluline ka aadressiklasse teada, sest mõned vanemad protokollid töötavad aadressiklasside alusel.

Igas võrgus olevat kahte spetsiaalkasutusega aadressi – võrguaadressi ja leviedastusaadressi – ei saa määrata ühelegi hostile IP-aadressiks. Näiteks IP-aadress 197.45.23.5/24 on kasutatav hosti IP-aadressina. Selle IP-aadressi võrgumaskiks on 255.255.255.0, võrguaadressiks on 197.45.23.0 ja leviedastusaadressiks on 197.45.23.255. IP-aadressi 123.231.246.95/27 puhul on viimane oktett binaarkujul "01011111". Hostiosa moodustab viimased viis bitti (32–27=5), seega on tegemist võrgu leviedastusaadressiga. Selle IP-aadressi võrguaadressi saame, kui muudame viimases oktetis hostiosa bitid nullideks, seega viimane oktett binaarkujul oleks "01000000", mis kümnendkujul on 64. Võrguaadressiks on seega 123.231.246.64. Tema võrgumask on 255.255.255.224 (viimane oktett on kahendkujul "11100000"). Võrgu 197.45.23.0/24 kasutatavad IP-aadressid on kujul, kus viimane oktett on 1, 2, ..., 254 (näiteks 197.45.23.67, 197.45.23.252 jne.). Võrgu 123.231.246.64/27 puhul on kasutatavad sellised IP-aadressid, mille viimane oktett jääb vahemikku 65-94. Võrgu 123.0.0.0/8 puhul on kasutatavad IP-aadressid, kus esimene oktett on 123 ja ülejäänud oktettide väärtused on suvalised, välja arvatud võrguaadress 123.0.0.0 ning leviedastusaadress 123.255.255.255. Näiteks 123.56.2.0.5, 123.1.2.3, 123.0.0.6, 123.4.55.255, 123.255.255.254.

Kuigi tänu CIDR kasutamisele suurenes aadressiruumi kasutusefektiivsus tunduvalt, on IP adresseerimine oma iseloomult suhteliselt raiskav, kuid samas on aadressiruumi jaotamine väga efektiivne. IP võrgus saavad võrgud olla kahe astme suurustega. Seega saavad võrkude suurused olla 4 (2^2), 8 (2^3), 16 (2^4), 32 (2^5), 64 (2^6), 128 (2^7), 256 (2^8), 512 (2^9), jne.

Reserveeritud IP-aadressid

IP-aadressidest on peale vahemike 224.0.0.0/4 (ehk kuni 239.255.255.255) (multiedastus) ja 240.0.0.0/4 (ehk kuni 255.255.255.254) (uurimis- ja arendustegevuseks) reserveeritud veel mõned IP-aadresside vahemikud, mida kasutatakse spetsiaalotstarbeks (lisaks võrkude võrgu- ja leviedastusaadressile) (RFC 3330). Nendeks vahemikeks on:

- 10.0.0.0/8, 172.16.0.0/12 (kuni 172.31.255.255), 192.168.0.0/16 – mõeldud privaatseks kasutamiseks (näiteks sisevõrgus NAT (*network address translation*) puhul). Need aadressid ei ole avalikus Internetis kasutatavad (paketid visatakse minema).
- 127.0.0.0/8 – kasutatakse tagasisidestusaadressina (*loopback address*). Näiteks aadress 127.0.0.1 on kohalik masin. Tagasisidestusaadressi kasutatakse testimiseks ja ka oma enda masina poole pöördumiseks. Sellesse vahemikku kuuluvat IP-aadressi ei saadeta võrku, vaid liiklus toimub ainult saatva arvuti piires. Standardne hostinimi 127.0.0.1 jaoks on "localhost".
- 0.0.0.0/8 – viitab lähtehostile lokaalses võrgus, 0.0.0.0/32 viitab iseendale.
- 14.0.0.0/8 – *Public Data Networks* aadressid. Määranguid: http://tools.ietf.org/html/rfc1700#page-181.
- 169.254.0.0/16 – aadressiplokki tuntakse kui lokaalse sideliini (*link local*) aadresse (RFC 3927). Automaatselt seadistatud aadress, mida kasutatakse näiteks juhul, kui host oli määratud dünaamiliselt IP-aadressi saama (ehk kasutatakse DHCP), kuid ei saanud serveriga (DHCP) ühendust.

- 128.0.0.0/16, 191.255.0.0/16, 192.0.0.0/24, 223.255.255.0/24 – IANA poolt reserveeritud IP-aadresside vahemikud tulevikus kasutamiseks.
- 192.0.2.0/24 – dokumentatsioonis ja näidisprogrammikoodis kasutamiseks. Tihti kasutatakse domeeninimena "example.com" või "example.net". Aadressivahemik on kasutatav ainult lokaalses võrgus.
- 192.88.99.0/24 – kasutatakse IPv6 suvaedastus- (*anycast*) aadressi tõlkimiseks IPv4-aadressiks IPv6 ja IPv4 vahel vahendavas marsruuteris (*relay router*).
- 198.18.0.0/15 – kasutatakse jõudlustestide tegemiseks võrgus.
- 255.255.255.255 – lokaalse võrgu leviedastusaadress.

Alamvõrk

Lihtne on mõista, et A- ja B-klassi võrgud on liiga suured ühe leviedastusdomeeni jaoks (tekitades liiga suure IP-aadresside kao). Enamasti on ka C-klassi võrk liiga suur ühe leviedastusdomeeni jaoks. Selle jaoks kasutatakse alamvõrgustamist (*subnetting*), kus üks suurem võrk jagatakse mitmeks väiksemaks alamvõrguks (*subnet*). Seega, näiteks kui omatakse B-klassi aadressi 135.222.0.0/16, siis see võrk on kahtlemata liiga suur. Võrk tuleks jagada väiksemateks osadeks, mille tulemusena saame ühe võrgu asemel teha mitu alamvõrku. Seega jagunevad vaadeldava IP-aadressi bitid kahe osa asemel nüüd kolme ossa: võrguosa, alamvõrgu osa ja hostiosa. Võrguaadress on teatavasti jagatud võrguosa ja hostiosa vahel. Alamvõrgustamisel võetakse alamvõrgu osasse jäävad bitid hosti osast. Seda nimetataks laenamiseks. Mida rohkem bitte laenatakse, seda rohkem alamvõrke saame teha (ehk $2^{<\text{laenatud bitte}>} - 2$) ja seda väiksemaks alamvõrgud muutuvad. Alamvõrgutehnika avaldati aastal 1985.

Alamvõrgustamise juures on samuti oluline tähele panna, et alamvõrgu bittide osas on ainult nulle või ainult ühtesid sisaldavad alamvõrgud mittekasutatavad. Sellest lähtuvalt on võimalik minimaalselt laenata kaks bitti ja maksimaalselt nii palju, et hosti ossa jääks alles minimaalselt kaks bitti.

Alamvõrke tehes tuleb täpselt teada maksimaalset vajatavate alamvõrkude arvu ja maksimaalset alamvõrgu suurust. Kui siit jääb bitte üle, siis need võib oma äranägemise järgi paigutada vastavalt alamvõrgu osasse või hosti osasse (või mõlemasse). Jagame vaatluse all oleva B klassi võrgu alamvõrkudeks ja oletame, et meil on vajadus 17 alamvõrgu järele ning võrgu suurus ei lähe kindlasti üle 100 hosti. Meil on kasutada 16 bitti (hosti osa bitid). Kui laename 2 bitti, siis saame 2^2-2 = 2 kasutatavat alamvõrku, mis ei ole piisav. Kui võtame 4 bitti, siis saame 2^4-2 = 14 kasutatavat alamvõrku, mis samuti ei ole piisav. Kui laename 5 bitti, siis saame 2^5-2 = 30 kasutatavat alamvõrku, mis on piisav 17 alamvõrgu jaoks (13 tükki jääb veel reservi ülegi). Hosti osasse jäi alles 16-5 = 11 bitti. Seega on iga alamvõrgu suuruseks 2^{11}-2 = 2046 hosti, mida on kõvasti rohkem, kui nõutud 100 kasutatavat IP-aadressi alamvõrgu kohta. Meile piisaks, kui hosti osasse alles jätta 7 bitti, millega saaks alamvõrgus olla 2^7 = 126 kasutatavat IP-aadressi. Seega on praeguse näite puhul 11-7 = 4 bitti meil üle jäänud. Need saame näiteks alles jätta hosti osasse (see on meie otsustada). Kui laenasime ainult vajaliku arvu bitte (viis tükki) ja jätsime ülejäänud hosti osasse, siis alamvõrgumaskiks jääb meil 11111111.11111111. 11111000.00000000 ehk 255.255.248.0 ehk /21. Toonitame veel, et kogu arvutus käib kahendsüsteemis, mille hiljem teisendame parema käideldavuse huvides kümnendsüsteemi. Alamvõrgud, mis said loodud, on toodud tabelis 7.2 (esimene ja viimane alamvõrk ei ole kasutatavad, kuid nad on välja toodud).

Olgu alamvõrkude tegemise juures ära öeldud, et kasutatakse ka esimest alamvõrku (kus alamvõrgu osas on kõik nullid), kuigi see on tehniliselt lubamatu. Seda toetavad mitmed uuemad marsruuterid. Seda nimetatakse *subnet zero*. Nullis alamvõrk on võetud kasutusele, sest tihti on IP-aadresse vähe ja olemasolevaid tuleks paremini ära kasutada. Samuti toetavad uuemad tarkvarad ka viimase alamvõrgu kasutamist, samas võib tekkida probleeme vanema tarkvara ja protokollidega.

Tabel 7.2. Alamvõrgustatud võrkude võrguaadressid,
kasutatavad IP-aadressid ja leviedastusaadress.

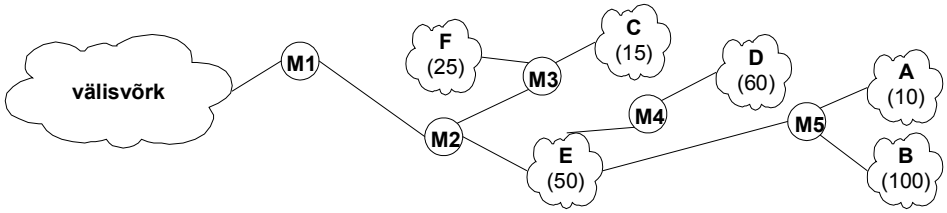
Nr	Alamvõrk	Kasutatavad aadressid	Leviedastus
0	135.222.0.0	135.222.0.1-135.222.7.254	135.222.7.255
1	135.222.8.0	135.222.8.1-135.222.15.254	135.222.15.255
2	135.222.16.0	135.222.16.1-135.222.23.254	135.222.23.255
3	135.222.24.0	135.222.24.1-135.222.31.254	135.222.31.255
4	135.222.32.0	135.222.32.1-135.222.39.254	135.222.39.255
...	...	...	...
30	135.222.240.0	135.222.240.1-135.222.247.254	135.222.247.255
31	135.222.248.0	135.222.248.1-135.222.255.254	135.222.255.255

Alamvõrgustamine toimub ainult antud võrgu siseselt. Väljaspoolt antud võrku paistab see ikka ühe suure võrguna. Kui väljastpoolt saadetakse pakett (alam)võrgu leviedastusaadressile, siis seda nimetatakse **suunatud leviedastuseks**. Tavaliselt on turvalisuse huvides (smurf-ründe ohu tõttu) suunatud leviedastus kinni keeratud.

VLSM

VLSM (*variable length subnet mask*) tehnika on sarnane CIDR-le. VLSM on avaldatud aastal 1987. VLSM lubab sama võrgu sees kasutada erineva suurusega võrgumaskidega alamvõrke, suurendades aadresside kasutamise efektiivsust. VLSM nimetatakse ka alamvõrkude alamvõrgustamiseks.

Paremaks arusaamiseks VLSM tehnikast kasutame näidet. Olgu asutus, millele on antud võrk 135.79.82.0/23. Asutuse võrk koos vajatavate IP-aadressidega alamvõrkudes on esitatud joonisel 7.4. Meie ülesandeks on VLSM tehnikat kasutades antud võrguaadress sobivalt ära jagada.

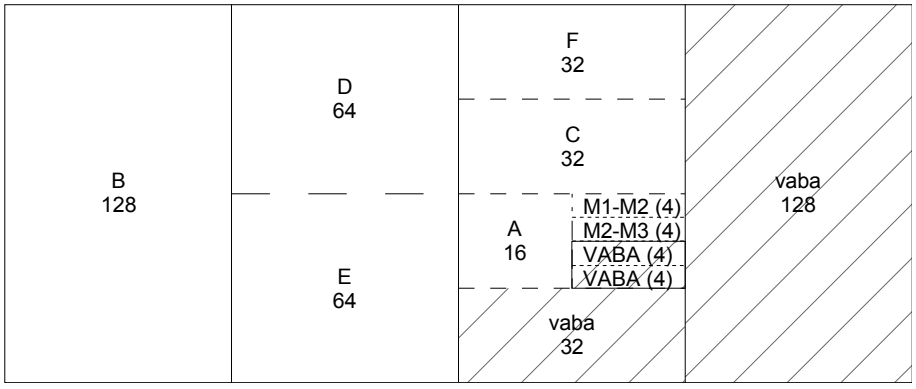


Joonis 7.4: Näidisvõrk, kus ringid on marsruuterid koos nimega, pilvekesed on võrgud koos nime ja vajatavate IP-aadressidega (marsruuteri IP on sisse arvatud). Marsruuterite vahelised jooned tähistavad punktist-punkti sideliine.

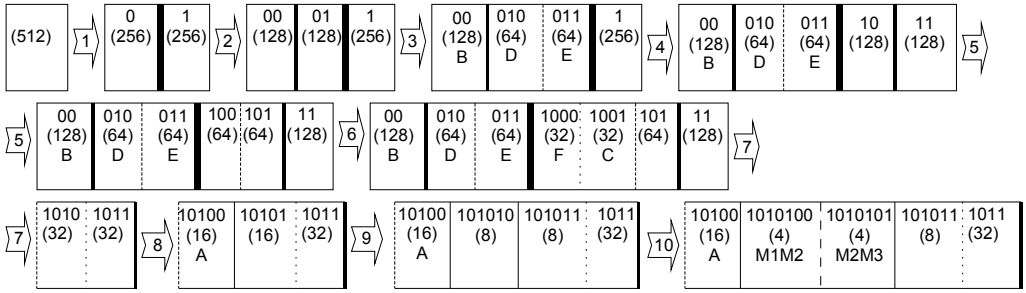
Võrgumaski järgi saame leida kogu selle võrgu suuruse, tehes tehte 32–23=9 (IP-aadressi pikkus on 32 bitti, võrguaadressi prefiksi pikkus on 23 bitti). Seega on meil hosti osas 9 bitti, mida saame kasutada. Üheksa bitti tähendab 2^9=512 erinevat IP-aadressi (koos mittekasutatavatega). Meil on joonise järgi kokku vaja 25+15+50+60+100+10=270 IP-aadressi. Lisaks on ka kaks punktist-punkti sideliini, seega on vajalik 260+4=264 IP-aadressi. Selle arvutuse tegime ülevaate saamiseks. Et kiiresti teada saada, kui palju IP-aadresse minimaalselt kulub võrkude peale, peab leidma iga võrgu kohta minimaalse naturaalarvu `n`, et 2^n-2 oleks suurem-võrdne IP-

aadresside hulgast. Leidnud vajalikud arvud, siis nad kokku liita. Seega minimaalne IP-aadresside vajadus näite puhul on 32+32+64+ 64+128+16+4+4=344.

Kui IP-aadresse oleks natuke vähem kui teoreetiliselt kasutatavaid IP-aadresse kokku, siis ei oleks vastav võrk piisavalt suur. Näiteks /24 maskiga ei oleks see võimalik, sest $2^8 = 256 < 264$. Edasi vaatame, kui suured alamvõrgud peame igale leviedastusdomeenile minimaalselt eraldama. Leiame väikseima täisarvu n väärtuse, mille puhul 2^n-2 on suurem-võrdne vajalike IP-aadresside hulgaga. Seega 10→16 (2^4); 25→32 (2^5); 15→32 (2^5)(16 ei saa, sest üks IP läks võrgu ja teine leviedastusaadressi tarvis); 60→64 (2^6); 50→64 (2^6); 100→128 (2^7); 2→4 (2^2); 2→4; (2^2). Järgmisena võtame kõige suurema alamvõrgu, milleks oli meie näite puhul võrk B. Jagame kogu ala niisuguse suurusega tükkideks. Seega saame võrgud 135.79.82.0/25; 135.79.82.128/25; 135.79.83.0/25; 135.79.83.128/25. Esimesse ettevõetavasse osasse võime panna kõige suurema võrgu B. Järgmisena oli meil suuruse järjekorras kaks 64 IP-aadressiga võrku. Nende jaoks saame ilusasti järgmise osa pooleks teha. Seega võrgu D võrguaadressiks saaks 135.79.82.128/26 ja alamvõrgu E aadressiks 135.79.82.192/26. Nüüd peame ära mahutama kaks 32 IP-aadressiga alamvõrku, need saame paigutada kolmandasse tükki. Seega oleksid alamvõrgu F võrguaadressiks 135.79.83.0/27 ja alamvõrgu C aadressiks 135.79.83.32/27. Edasi on vaja paigutada veel kaks 16 IP-aadressi suurust võrku. Selleks võtame esimese tükelduse kolmanda tüki alamtükeldamisel alles jäänud 32 IP-ga alamvõrgu kasutusele. Seega oleks alamvõrgu A võrguaadressiks 135.79.83.64/28. Edasi on vaja ära paigutada kaks nelja IP-aadressiga alamvõrku. Kuna viimasest tükeldamisest jäi ruumi üle, siis saame seda veel ära kasutada. Tükeldades jällegi väiksemateks osadeks. Seega oleks punktist-punkti sideliinide jaoks võrgud järgmised: 135.79.83.80/30, 135.79.83.84/30 ja 135.79.83.88/30. Nüüd oleme kõik alamvõrgud jaganud meile antud aadressiruumi, kasutades VLSM tehnikat. Alamvõrkudeks jaotamist VLSM tehnikaga võiks kujutada tordi lõikamisena, kus vaadeldavat tükki saab ainult võrdseteks tükkideks lõigata. Antud näidet on kujutatud joonisel 7.5. Joonisel 7.6 teeme võrgu jaotamise läbi ka samm-sammult.



Joonis 7.5: Eraldatud aadressiruumi jagamine järjest väiksemateks osadeks poolitamiste teel. Viirutatud alad on kasutamata jäänud aadressiruum. Seega saaks maksimaalselt mahutada veel ühe 128, 32 ja 8 hostiga võrgu. Samuti on neid võrke võimalik omakorda veel poolitada. Samas ei ole võimalik mahutada näiteks 130 hostiga võrku.



Joonis 7.6: Jagamised samm-sammult näidatuna. Esimesel real on vastava jagamise osa alamvõrgu bitid (alamvõrguprefiks), teisel real IP-aadresside hulk ja kolmandal real määratud võrgu tähis. Noolte sees on sammude numbrid.

Sammude kirjeldused joonisel 7.6:

1. Jagame ühe suure võrgu pooleks. Saame kaks alamvõrku.
2. Jagame esimese alamvõrgu uuesti pooleks. Siin näeme, et võrgu B jaoks on juba piisavalt väike võrk. Märgistame selle ära ja seda enam edasi ei tükelda.
3. Teeme alamvõrgu, mille esimesed bitid on "01" omakorda väiksemaks, saades 64 IP-aadressiga võrgud, mis on parajalt suured võrkude D ja E jaoks.
4. Esimesest jagamisest ülejäänud osa jagame samuti pooleks.
5. Kuna ükski järelejäänud võrk ei vajanud 128-IP-aadresilist võrku, siis jagame 128-IP-aadresilise alamvõrgu prefiksiga "10" omakorda pooleks. Tulemuseks saame kaks 64-IP-aadresilist võrku.
6. Kuna ükski järelejäänud võrk ei vajanud ka 64-IP-aadresilist võrku, siis jagame 64-IP-aadresilise alamvõrgu prefiksiga "100" veel omakorda pooleks. Tulemuseks saame kaks 32-IP-aadresilist võrku, mis on parajalt suured võrkude F ja C jaoks.
7. Võtame nüüd esimese vaba 64-IP-aadresilise tüki alamvõrgu prefiksiga "101" ja jagame ta kaheks 32 IP-aadressiga võrguks (lihtsuse mõttes on joonisel järgnevad sammud toodud eraldi, et ei peaks eelnevaid jagamisi kaasas kandma, et joonis liiga suureks ei läheks).
8. Jagame esimese 32-IP-aadresilise osa omakorda kaheks 16 IP-aadressiga võrguks, kus esimese osa määrame võrgule A.
9. Kuna meil oli vaja veel ka punktist-punkti sideliinide jaoks võrke, siis jagame teise 16-IP-aadresilise tüki alamvõrgu prefiksiga "10101" ning saame kaks 8 IP-aadressiga võrku.
10. Jagame 8-IP-aadresilise osa alamvõrgu prefiksiga "101010" omakorda kaheks 4 IP-aadressiga võrguks. Meil on nüüd üle jäänud 8, 32 ja 128 IP-aadressiga võrgud, mida saame kasutada kunagi tulevikus. Vajaduse korral saame neid omakorda tükeldada. Näiteks, kui on vaja nelja IP-aadressiga võrku, siis oleks mõtekas kasutusele võtta võrk alamvõrgu prefiksiga "101011". Kui on vaja näiteks 64 IP-aadressiga võrku, siis peaksime kindlasti kasutama võrku, mille alamvõrgu prefiks on "11".

VLSM kasutades saavad marsruuterid agregeerida mitu võrku kokku. Näiteks saab marsruuter M2 kuulutada marsruuterile M1, et tema kaudu on kättesaadavad supervõrk 135.79.82.0/24 (võrgud B, D, E).

IP-aadresside jaotamine

Kuna Internetis kasutatavad IP-aadressid peavad olema globaalselt unikaalsed, siis peab selle tagamiseks olema paindlik mehhanism. Selleks kasutatakse IP-aadresside jaotust CIDR meetodil (enne CIDR jaotati lihtsalt aadressiklassi võrke). IP-aadresside unikaalsus tagatakse hierarhi-

liselt jaotatud Interneti registri süsteemi (*Internet Registry System*) abil. Selle hierarhia tipus asub IANA. IANA-le alluvad regionaalsed Interneti registrid ehk RIR-id (*Regional Internet Registries*) ja neile omakorda lokaalsed Interneti registrid ehk LIR-id (*Local Internet Registries*) (RFC 2050). Regionaalsete Interneti registrite organisatsioonid ning nende hallatavad piirkonnad on järgmised:

- AfriNIC – haldab kogu Aafrika mandrit (koduleht: http://www.afrinic.net).
- APNIC (*Asian-Pacific Network Information Center*) – Vaikse ookeani Aasia ja Austraalia osa (v.a. Venemaa) ja Afganistan, Pakistan, India, Sri Lanka, Nepaal ja Kagu-Aasia riigid (koduleht: http://www.apnic.net).
- ARIN (*American Registry for Internet Numbers*) – Ameerika Ühendriigid, Kanada ja paljud Kariibi mere saareriigid (koduleht: http://www.arin.net).
- LACNIC – Lõuna- ja Kesk-Ameerika ja mitmed Kariibi mere saareriigid (koduleht: http://www.lacnic.net).
- RIPE NCC (*Reseaux IP Europeens Network Coordination Centre*) – Euroopa, Lähis-Ida, Venemaa ja Kesk-Aasia (koduleht: http://www.ripe.net).

Riikide jaotumist RIR-de vahel on võimalik vaadata aadressil http://www.arin.net/community/countries.html.

Lokaalsed Interneti registrid on näiteks internetiteenusepakkujad ehk ISP-id (*Internet service provider*), riigiasutused jms. RIPE-s registreeritud Eesti LIR-de nimekiri on saadaval aadressil http://www.ripe.net/membership/indices/EE.html.

IP-aadresside hierarhias määrab IANA mingi ploki IP-aadresse regionaalsetele Interneti registritele, kes omakorda jagavad neile antud aadressiplokist väiksemaid plokke lokaalsetele Interneti registritele. Viimased jagavad IP-aadresse juba konkreetsetele firmadele, teistele ISP-dele, riiklikele asutustele, eraisikutele jne. Näiteks kui kodukasutaja ühendub Internetti (näiteks ADSL ühendusega), siis ISP annab antud kodukasutajale kasutada mingi IP-aadressi. Asutus, millele anti mingi võrgu jagu IP-aadresse, võib neid ise edasi määrata.

Infot, kuidas IP-aadressid on esimese okteti alusel jagatud, leiab aadressilt http://www.iana.org/assignments/ipv4-address-space.

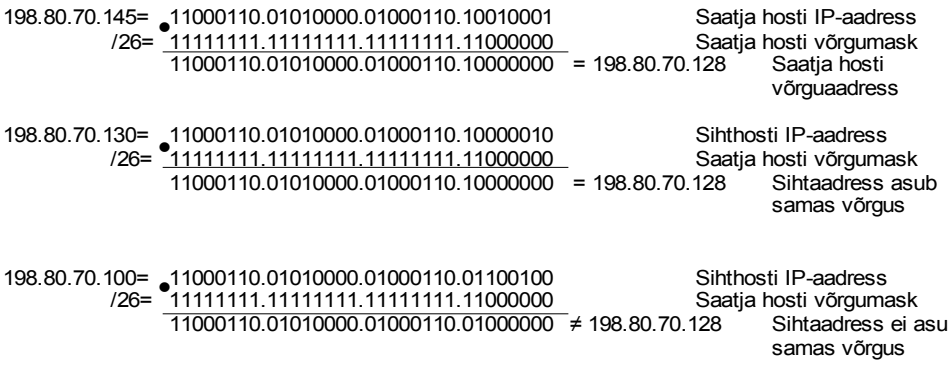
MAC ja IP-aadressi sidumine Ethernet kohtvõrgus

Saatja peab mingile teisele hostile andmete saatmiseks teadma sihtpunkti (kui host asub samas kohtvõrgus) või võrguvärava MAC-aadressi (kui host asub väljaspool kohtvõrku) ja sihtpunkti IP-aadressi. IP ja MAC-aadressi sidumisel on oluline teada, kas IP-aadress, kuhu me tahame paketti saata, asub saatjaga samas võrgus või mitte. Selle teadasaamiseks kasutatakse sihtkoha IP-aadressi ja hosti enda võrgumaski, millega kontrollitakse, kas sihtkoha IP-aadress asub samas võrgus. Esimesena rakendatakse loogilist korrutamist hosti enda IP-aadressi ja võrgumaskiga, saades oma võrguaadressi. Järgmisena korrutatakse sihtkoha IP-aadress võrgumaskiga ja saadakse mingi tulemus. Kui saadud tulemused on ekvivalentsed, siis asub sihtkoht samas võrgus ja kaadri MAC-aadressiks pannakse sihthosti MAC-aadress. Vastasel korral asuvad hostid erinevates võrkudes. Sellisel juhul saadetakse paketid üldjuhul läbi vaikevõrguvärava (*default gateway*), kus MAC-aadressiks pannakse võrguvärava oma. Kui masinas pole seadistatud vaikevõrguväravat, siis antakse veateade võrgu mittekättesaadavusest ("*Network unreachable*"). Vaikevõrguväravaks on tavaliselt marsruuter.

Vaatleme hostide käitumist joonisel 7.7 toodud näite põhjal. Oletame, et meil on host, mille IP-aadress on 198.80.70.145/26 ja ta soovib saata paketti järgnevatele aadressidele: 198.80.70.130 ja 198.80.70.100.

Hostid omavad tabelit, mis sisaldab lokaalsetele IP-aadressidele vastavaid MAC-aadresse (mittelokaalsetele IP-aadresside puhul kasutatakse võrguvärava MAC-aadressi). Seda tabelit nimetatakse **ARP**-tabeliks (*Address Resolution Protocol*). Kui arvuti A soovib saata paketti arvutile B (arvuti B asub samas leviedastusdomeenis), mille IP-aadress on 192.168.4.5, siis arvuti A otsib oma ARP-tabelist vastava IP-aadressiga kirjet. Kui see on olemas, siis pannakse

kaadri sihtaadressiks vastav MAC, mis vastas antud IP-aadressile. Juhul kui ei leitud ARP-tabelist vastavat IP-aadressi, siis peab teada saama MAC-aadressi. Selleks saadetakse kohtvõrku pakett, mida nimetatakse ARP-päringuks (*ARP request*), mille MAC-sihtaadressiks on leviedastusaadress (FF:FF:FF:FF:FF:FF) ja IP-aadressiks subjektiks olev IP-aadress, IP ja MAC lähteaadressid on tema enda omad. Hostid, mis saavad ARP-päringu paketi, annavad selle edasi OSI kolmandale kihile. Ainult vastava IP-aadressiga host saadab tagasi paketi "ARP vastus" (*ARP reply)*, mis sisaldab tema MAC ja IP-aadressi. Teised hostid viskavad ARP-päringu paketi minema, sest seal ei olnud nende IP-aadressi. Saanud "ARP vastus" paketi tagasi, saab paketi saatja panna MAC-aadressi saadetavasse kaadrisse. Juhul, kui mitte keegi ei vastanud "ARP vastus" paketiga, teavitatakse ülemist kihti tekkinud veast (host ei ole kättesaadav).



Joonis 7.7: IP-aadressi võrku kuulumise kontroll. Esimesena leitakse saatja enda võrguaadress. Järgmisena leitakse, kas sihtpunkti IP-aadress 198.80.70.130 asub samas võrgus (leiti, et asub). Viimasena leitakse, kas sihtpunkti IP-aadress 198.80.70.100 asub samas võrgus (leiti, et ei asu).

Host, mis saab temale saadetud paketi, kontrollib, kas antud paketi lähte-IP-aadress on olemas ARP-tabelis. Kui ei ole, siis lisatakse ARP-tabelisse paketi lähteaadressi MAC ja IP-aadress. Igal juhul värskendatakse kirje aega. ARP-tabeli kirjed, mida ei ole enam mingi fikseeritud aja jooksul kasutatud, eemaldatakse ARP-tabelist. ARP-tabel on tavaliselt dünaamiliselt muutuv ja seda hoitakse arvuti mälus. Üldiselt ei ole kasutajal vaja IP-aadresse siduda MAC-aadressidega (mõnikord võidakse seda teha turvakaalutlustel).

ARP-i vastandprotokoll on RARP (*Reverse ARP*), millega küsitakse MAC-aadressile vastavat IP-aadressi. Seda kasutati põhiliselt kettata masinate puhul oma IP-aadressi teadasaamiseks. Tänapäeval kasutatakse RARP-i harva, selle asemel on levinud palju komplektsem DHCP protokoll.

Marsruuterid ja marsruutimine

OSI teise kihi adresseerimine on füüsiline adresseerimine. Kolmas kiht on mõeldud loogiliseks adresseerimiseks läbi erinevate võrkude. Teekonna käigus võib pakett läbida mitmeid erinevaid võrke, kuni jõuab sihtpunkti. Võrguseade, mis erinevate võrkude vahel pakette suunab, on marsruuter. Pakettide suunamist nimetatakse marsruutimiseks. Kui pakett liigub leviedastusdomeenist väljapoole, siis käib see läbi marsruuteri (see marsruuter on sellele võrgule võrguväravaks). Marsruuter on masin, mis ühendab erinevaid võrke (võrgud ei saa olla kattuvad), olles ise oma erinevate liidestega erinevates võrkudes. Marsruuteri igal liidesel on vastavasse võrku kuuluv IP-aadress. Traditsiooniliselt pannakse marsruuteri liidese IP-aadressiks antud võrgu esimene kasutatav IP-aadress. See ei ole kohustus, vaid tava. Mõnikord pannakse ka viimane kasutatav IP-aadress. Samas võib panna ka suvalise antud võrgus kasutatava võrguaadressi.

Erinevalt MAC-aadressidest on IP-aadressid ja võrgud grupeeritavad. IP-võrke on võimalik hierarhiliselt grupeerida, selliselt ei ole vaja eraldi kõikide IP võrkude kohta arvet pidada. Kui IP-aadressid poleks organiseeritult grupeeritud, siis Internet ei saaks põhimõtteliselt toimida, sest võrguseadmed ei saa mahutada tohututes kogustes marsruutimisinfot ega saa olla kursis kõikide muudatustega, mis mujal kaugemal toimuvad. Marsruuterid on ka leviedastuse piirajaks. Marsruuter jagab võrgud leviedastusdomeenideks (*broadcast domain*).

Võrku paigaldatav marsruuter vajab (erinevalt kommutaatorist) eelnevat seadistamist, et saada ta töökorda. Selleks tuleb marsruuteri pordid töökorda seada: anda liidestele IP-aadressid, määrata marsruudid (millised võrgud, kus liideses asuvad) jne. Marsruutide informatsioon asub marsruutimistabelis. Me vaatleme praegu, mis moodi toimub marsruutimistabeli käsitsi seadistamine. Peatükis "Marsruutimisprotokollid" (lk. 92) vaatleme lähemalt marsruutimisprotokolle, nende plusse ja miinuseid võrreldes käsitsi seadistamisega. Marsruutimistabelite käsitsi seadistamine on väga tihti kasutatav, kui hallatav võrk ei ole suur, topoloogia on lihtsakoeline, topoloogia muutused on harvad ja pole palju dubleeritud ühendusi. Käsitsi seadistatud marsruute nimetatakse staatilisteks marsruutideks.

Marsruuter teeb paketi edastamise otsuseid marsruutimistabeli alusel. Marsruutimistabel sisaldab lihtsustatult järgmisi andmeid: sihtvõrk koos teda määrava võrgumaskiga ja naaber, kellele pakett edasi saata. Lisaks võib marsruutimistabeli kirje sisaldada ka muud informatsiooni (näiteks, kas liides on püsti), meetrikat (saab määrata, kuivõrd eelistatud on selle kirje marsruudi kasutamine vaadeldava paketi edastamiseks).

Kui marsruuter saab kaadri, siis ta kontrollib kontrollsummat ja vaatab, kas kaader on mõeldud temale. Kui on, siis annab ta kaadri edasi kolmandale kihile, vastasel korral viskab kaadri minema. Vajaduse korral värskendab marsruuter ARP-tabelit (Etherneti puhul). Kolmandas kihis tehakse jällegi vajalikud kontrollimised (näiteks kontrollsumma kontroll, vigane aadress jms.). Järgmisena vaadatakse, ega sihtaadress pole mingi tema enda mõne liidese IP-aadress. Kui on, siis antakse paketi andmed edasi marsruuteri neljandale kihile (tegemist võib olla näiteks marsruuteri veebiliidese kaudu seadistamisega). Kui pakett ei ole mõeldud marsruuterile endale, siis hakkab marsruuter marsruutimistabelit läbi käima, et leida võrk, kuhu antud IP-aadress kuulub kõige täpsemini. Kui leitud marsruutimistabeli kirje alusel on sihtvõrguks **vahetult ühendatud võrk** (marsruuteri mingi liides asub vastavas võrgus), siis saadetakse pakett otse sihthostile. Kui kirje järgi ei ole tegemist vahetult ühendatud võrguga, siis saadetakse pakett edasi vastava kirje juures märgitud järgmisele marsruuterile. Kui marsruutimistabel ei sisaldanud mitte ühtegi sobivat marsruutimiskirjet, siis visatakse pakett minema ja saadetakse paketi lähteaadressile vastavasisuline ICMP teade, et sihtvõrk ei olnud kättesaadav.

Tavaliselt asuvad asutuste ja kodude võrgud Interneti äärealadel, kus neil võivad olla mõned oma võrgud ja ülejäänud võrgud on kättesaadavad Interneti teenusepakkuja kaudu. Seetõttu oleks välisvõrku kuuluvate IP-aadresside puhul mõistlik kasutada mingit üldist reeglit, mis suunaks kõik ülejäänud paketid ISP-le. Sellist marsruuti nimetatakse **vaikemarsruudiks**, mida kasutatakse, kui mitte ükski marsruutimistabeli kirje ei rakendunud. Tabelikirje võrguaadressiks on 0.0.0.0 ja võrgumaskiks 0.0.0.0 (loogiliselt korrutades suvalist võrguaadressi võrgumaskiga 0.0.0.0 saadakse alati 0.0.0.0, mille tagajärjel leitakse antud võrk alati sobiv olevat).

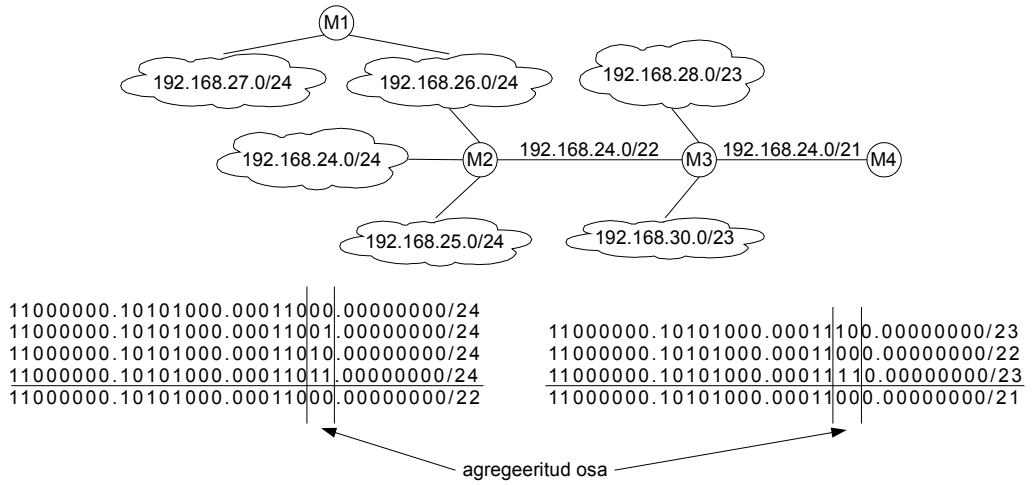
Kui marsruuter saadab paketi edasi, siis ta vähendab TTL ehk "aega elada" välja väärtust. Kui see saab võrdseks nulliga, siis visatakse pakett minema ja saadetakse lähteaadressile vastavasisuline ICMP veateade. Kui pakett saadetakse edasi ehk marsruuditakse, siis OSI teises kihis kapseldatakse pakett uude kaadrisse, mis oleneb järgmise hüppe naabri vahelisest kanalikihi protokollist. Etherneti puhul pannakse kaadri sihtaadressiks järgmise hüppe MAC-aadress ja lähteaadressiks marsruuteri enda MAC-aadress ning arvutatakse uus kontrollsumma, mille järel saadetakse kaader teele. Seega isegi sama kanalikihi protokolli kasutades muutuvad kaadri andmed iga hüppega. Paketi andmetest muutuvad iga hüppega väljad "aega elada" ja "kontrollsumma".

Ilma marsruute määramata saaks marsruuter edastada ainult neid pakette, mis on suunatud võrku, millesse ta ise ka kuulub mingi pordiga. Ülejäänud pakettide edastamiseks ei ole tal informatsiooni ning pakett visatakse minema (kui teatakse saatja võrku, siis saadetakse vastavasisuline ICMP teade).

Kuna marsruuterid peavad kogu kaadri vastu võtma, paketi lahti kapseldama, kasutama marsruutimistabelit ja asendama kaadri päise, siis on nad tunduvalt aeglasemad kommutaatoritest. Marsruuterid on kommutaatoritest aeglasemad ka seetõttu, et kommutaatorid töötavad üldiselt riistvaraliselt, kuid marsruuterid üldiselt tarkvaraliselt. Teise kihi seadmed on omakorda suurema viivitusega kui esimese kihi seadmed. Hubi, kommutaatori ja marsruuteri edastusoperatsioone võrreldes võiks öelda, et esimese kihi seadmed saadavad edasi kõik, teise kihi seadmed saadavad edasi kaadri, kui miski neid ei takista, ning kolmanda kihi seadmed saadavad edasi paketi ainult siis, kui nad peavad seda tegema.

Võrku planeerides peab arvestama, et sisuliselt on kolm IP-aadressi reserveeritud, sest lisaks võrgu- ja leviedastusaadressile vajab ka marsruuter vastavas võrgus olevat kasutatavat IP-aadressi.

Marsruuteril peab marsruutimistabelis olema kirje iga võrgu jaoks, sest marsruuter saadab paketi edasi ainult juhul, kui teab, kuhu seda saata. Võrkude planeerimisel on oluline ja väga suurte võrkude puhul ka praktiliselt mõõdapääsmatu marsruutimistabeli kirjeid kokku hoida, mida saab teha agregeerimisega. Seetõttu on tähtis juba võrgu disainimise käigus paigutada võrgud selliselt, et neid oleks võimalik võimalikult madalal tasemel agregeerida kokku suuremaks võrguks. Agregeeritud marsruute on ka kergem hallata ning nad tõstavad marsruuteri jõudlust (läbivaadatavaid kirjeid on vähem). Joonisel 7.8 on toodud näide võrkude agregeerimisest.



Joonis 7.8: Joonisel on näide võrkude agregeerimisest, kus mitme võrgu asemel saab marsruutimistabelisse panna ainult ühe võrgu. Esimesel juhul võetakse kokku võrgud 192.168.24.0/24, 192.168.25.0/24, 192.168.26.0/24 ja 192.168.27.0/24, kus marsruuteri M3 jaoks võib marsruutimistabelis olla marsruutimiskirje 192.168.24.0/22, mitte kõigi nelja võrgu kirjed eraldi. Marsruuteri M4 jaoks saab agregeerida võrgud kokku võrguks 192.168.24.0/21. Selliselt on marsruuteri M4 jaoks kuue marsruutimiskirje asemel vajalik ainult üks marsruutimiskirje. Joonise alumisel poolel on näidatud agregeerimine binaarsel kujul.

8. Transpordikiht

Transpordikihis tegeldakse andmete ettevalmistamisega üle võrgu saatmiseks ja loogilise ühenduse haldamisega. Transpordikiht võib pakkuda järgmist:

- Andmevoo teenust ja vookontrolli. Ülemine kiht saab kasutada andmevoogu lihtsalt andmete saatmiseks ja vastuvõtmiseks ega ei pea tegelema pakettide suuruste arvestamisega.
- Ühendusorienteeritud ühenduse loomist. Ülemine kiht saab kasutada loodud ühendust.
- Porte. Port on masinas vajalik kindla protsessi või teenusega suhtlemiseks (ühenduste eristamine).
- Ummistustega tegelemist. Transpordikihis tegeletakse ka potentsiaalsete ummistuste lahendamisega.
- Usaldusväärset ühendust. Kui mingid paketid lähevad kaduma (näiteks on kontrollsumma vale), siis saadetakse andmed uuesti.
- Järjekorra arvestamist. Võrgus võivad paketid jõuda sihtpunkti erinevas järjekorras kui nad välja saadeti või mõni pakett võib kaduma minna.

Transpordikihi andmeühikuks on segment. Erinevalt võrgukihi protokollist peavad kasutatavat transpordikihi protokolli toetama ainult ühenduse lõpphostid. Tänapäeval on kõige enam kasutatavateks transpordikihi protokollideks TCP (*Transmission Control Protocol*) ja UDP (*User Datagram Protocol*). Peale TCP ja UDP on olemas ka teisi transpordikihi protokolle, kuid need on nüüdseks kasutuselt praktiliselt kadunud või neid kasutatakse eriotstarbel või on nad veel alles väga uued.

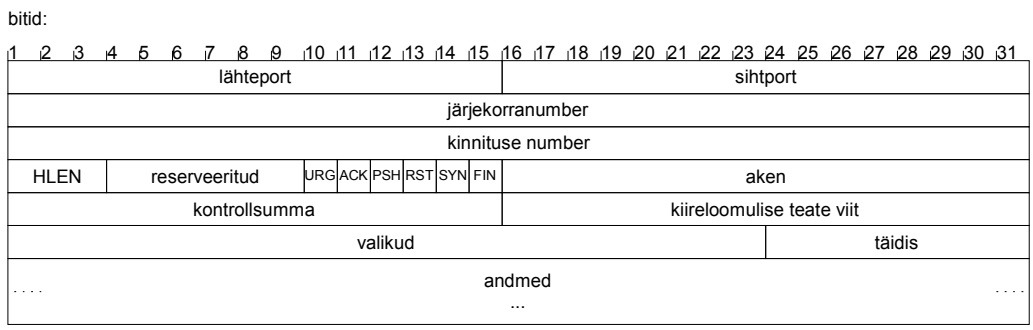
8.1 TCP

TCP (*Transmission Control Protocol*) (RFC 793, 2581, 2001) esialgne spetsifikatsioon avaldati IETF poolt 1981. aastal, võeti aluseks ARPAnet. TCP protokoll pakub ülemisele kihile kõiki transpordikihi funktsioone. TCP ühendused on mõeldud virtuaalse ja usaldusväärse täisdupleksühendusega ühenduse loomiseks andmevoo edastamisel kahe hosti vahel. TCP protokolli andmeühikuks on segment. TCP segmendi ülesehitus väljade kaupa on järgmine:

- Lähteport. Saatja-masina pordi number. 16 bitti.
- Sihtport. Vastuvõtja-masina pordi number. 16 bitti.
- Järjekorranumber. Määrab saadetava andmevoo esimese baidi (relatiivselt). Vajalik on, et vastuvõtja oskaks mitmes segmendis olevaid andmeid uuesti terviklikuks vooks kokku panna. 32 bitti.
- Kinnituse number. Määrab, mitmendast baidist (relatiivselt) alates järgmisena oodatakse andmeid. Kasutatakse, kui koodiväljal on püsti lipp ACK. 32 bitti.
- Päise pikkus. Näitab mitmest 32-bitisest sõnast päis koosneb. Väärtus on vahemikus 5-15. Vastavalt valikutele on segmendi päise pikkus 20-60 baiti (sarnane sama nimega IP protokolli väljaga). 4 bitti.
- Reserveeritud bitid tulevikuks. Peavad olema nullid. 4 bitti.
- Koodi bitid. Nende abil kontrollitakse ühendust. 8 bitti. Koodibitid on:
  - ◇ CWR (*congestion window reduced*) – lipu paneb püsti saatja pool, kui ta on saanud TCP segmendi, millel on püsti ECE lipp.
  - ◇ ECE (*ECN-echo* (*explicit congestion notification*)) – ühenduse loomisel pannakse lipp püsti, kui toetatakse ECN-i (asub IP päises).
  - ◇ URG (*urgent*) – tähendab, et tegemist on kiireloomulise teatega, et vastuvõtja peaks seda segmenti töötlema niipea kui võimalik. Kui see lipp on püsti, siis omab tähendust ka väli "kiireloomulise teate viit".

- ◇ ACK (*acknowledgment*) – märgib kas väli "kinnituse number" omab tähendust (ei oma tähendust ainult ühenduse loomise esimese segmendi puhul, teiste segmentide puhul on lipp püsti).
- ◇ PSH (*push*) – lipp pannakse püsti, kui ülemine kiht tahab, et andmed saadetaks koheselt (ei oodata segmendi andmetega täitumist jms.). Saaja host peab segmendi nii pea kui võimalik edastama programmile. Näiteks saates programmile katkestussignaali, võidakse kasutada seda lippu, et vältida järjekordasid.
- ◇ RST (*reset*) – ühenduse katkestamine. Selle käigus vabastatakse ressursid, mis kuulusid sellele ühendusele (näiteks puhvrites veel saatmata andmed, järjekorranumbrid jms.).
- ◇ SYN (*synchronize*) – kasutatakse ühenduse loomisel järjekorranumbrite sünkroniseerimiseks. Hiljem ühenduse käigus on SYN-lipp alati maas.
- ◇ FIN (*finish*) – saatja annab teada, et tema poolt pole rohkem andmeid saata.

- Akna suurus. Antakse teada, mitu baiti saatja peaks saatma, kuni ta jääb ootama kinnitust kohalejõudmise kohta. Maksimumsuuruseks on 65 535 (2^{16}-1) baiti. 16 bitti.
- Kontrollsumma. Arvutatakse TCP päise, pseudopäise ja andmete pealt. Pseudopäis on reaalselt IP päis. 16 bitti.
- Kiireloomulise teate viit (*urgent pointer*). See väli ütleb, kus lõpeb kiireloomulise teate osa segmendis ehk mitmes on selle viimane bait. Kasutatakse sageli juhtkäskude jaoks. 16 bitti.
- Valikud. Siin on mittekohustuslikud valikud. Tihti lepitakse ühenduse loomisel kokku valikutega segmendi maksimaalses suuruses ehk MSS (*maximum segment size*) valikus. Tänapäeval on akna suurus jäänud kitsaks, mistõttu on valikutesse juurde lisatud "akna suuruse skaleerimise" (*window scale*) valik, mis lubab suuremat akna suurust (lepitakse kokku ühenduse loomisel). Teised valikud on kättesaadavad aadressil http://www.iana.org/assignments/tcp-parameters. Muutuva suurusega 0, 32, 64, ... või 320 bitti.
- Andmed. See väli võib mõningatel juhtudel tühi olla (kui andmeid ei ole vaja saata).



Joonis 8.1: TCP segmendi väljad.

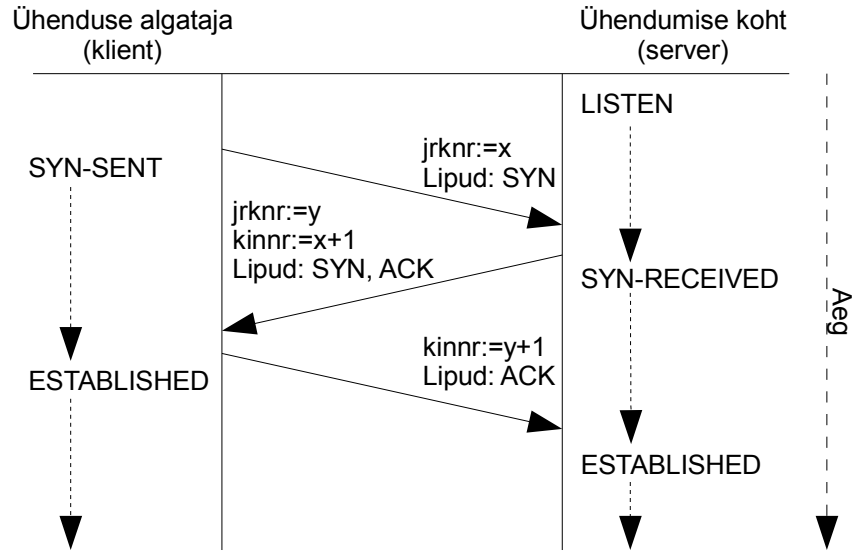
TCP puhul toimub ühendus kolmes etapis:

- Ühenduse loomine. Ühendus luuakse kahe hosti vahel ja lepitakse kokku vajalikes ühenduse parameetrites. Ühendust loomata ei ole võimalik andmeid vahetada. Ühendust saab luua ainult kahe osapoole vahel.
- Andmete transportimine. Saadetakse ja võetakse vastu andmeid hostide vahel.
- Ühenduse sulgemine. Ühendus võidakse sulgeda normaalselt või aegumisel. Peale ühenduse sulgemist vabastatakse hõivatud ressursid.

TCP ühenduse loomise protsessi kutsutakse **kolmeosaliseks käepigistuseks** (*three-way handshake*), mis toimub järgmiselt (joonis 8.2):

1. Klient ehk ühenduse algataja saadab segmendi, kuhu paneb oma algse järjekorranumbri `x` (väljal "järjekorranumber"). Lisaks pannakse koodivälja SYN lipp püsti. Arv `x` on juhuslik positiivne täisarv vahemikus $0...2^{32}-1$ (2^{32} on suurusjärgus 4 miljardit).
2. Server ehk ühenduse vastuvõtja saab kliendi saadetud segmendi, milles oleva järjekorranumbri `x` jätab ta omale meelde. Seejärel vastab ta esimesele osapoolele, pannes segmendi kinnitusvälja väärtuseks `x`+1, järjekorranumbri välja väärtuseks paneb juhusliku arvu `y` vahemikus $0...2^{32}-1$ (`y` on serveri andmete saatmise järjekorranumber). Koodiväljal pannakse püsti lipud SYN ja ACK.
3. Klient saab teiselt osapoolelt saadetu ja saadab serverile kinnituseks numbri `y`+1. Koodiväljal pannakse püsti lipp ACK. Sellest hetkest on ühendus loodud.

Enne ühenduse loomist ei ole võimalik andmeid vahetada. Ühenduse loomise käigus vahetavad lõpp-punktid omavahel järjekorranumbrite algväärtused (*initial sequence number*). Järjekorranumbrid annavad mõlemale osapoolele tõenduse, et vastaspool vastas õigele ühenduse päringule. Ühenduse algatajat võib vaadelda ka kliendina ja kohta, kuhu ühendus luuakse, serverina.

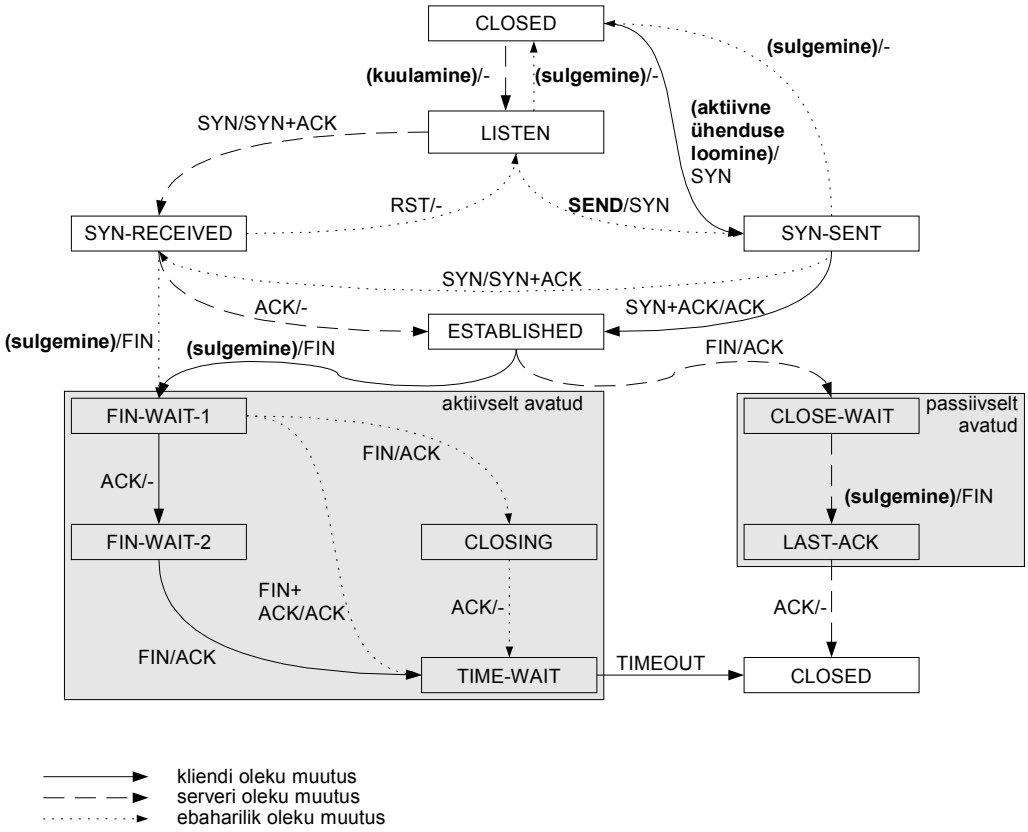


Joonis 8.2: Ühenduse loomise protsess graafiliselt. Äärtes on seisundid. Kui jõutakse ESTABLISHED seisundisse, siis on ühendus loodud ja võimalik andmeid vahetada.

TCP ühendus on täisdupleksühendus, mille tõttu on sulgemine neljaosaline, sest peab mõlemapoolsed ühendused sulgema. Olgu osapoolte A ja B vahel loodud ühendus (mõlema osapoole seisund on ESTABLISHED). Oletame, et A algatab TCP ühenduse sulgemise osapoolega B, siis aktiivne ühenduse sulgemise protsess oleks järgmine:

1. A saadab B-le segmendi, mille jooksev järjekorranumber on `x` ning püsti on pandud lipp FIN.
2. B saab A-lt segmendi, millel on FIN lipp püsti. B saadab A-le tagasi segmendi, millel on püsti lipp ACK ja kinnituse number on `x`+1.
3. Kui B otsustab samuti ühenduse sulgeda, siis saadab ta A-le segmendi, milles järjekorranumbriks on B jooksev järjekorranumber `y`. Kinnituse väljale pannakse `x`+1. Püsti pannakse lipud FIN ja ACK.
4. Sellele vastab A segmendiga, kus kinnituse numbriks on `y`+1 ja püsti on lipp ACK.

Kogu ühenduse olekumasinaskeem on toodud joonisel 8.3.



Joonis 8.3: TCP ühenduse erinevad seisundid ja üleminekud nende vahel. Noolejoontel olevad tekstid tähistavad enne kaldkriipsu saadud segmendi lippe või sulgudes olevana vastavat tegevust. Kaldjoone järel on lipud, mis seatakse püsti saadetud segmendis. ESTABLISHED seisundis toimub andmete vahetamine. Enne ESTABLISHED seisundit on ühenduse loomine ja pärast on ühenduse sulgemine.

Suurte andmemahtude edastamine

TCP hoolitseb selle eest, et kasutaja saaks saata korraga väga palju andmeid (näiteks rakendusprogramm edastab 700 MB suurust faili). TCP peab arvestama võrgukihi protokolli maksimaalse suurusega, mis IP puhul on 65535 baiti miinus TCP enda päise suurus. Selleks on vajalik andmete segmenteerimine ehk tükeldamine väiksemateks osadeks. Kõik segmendid varustatakse kasvavalt järjekorranumbritega, alustades ühenduse loomisel kokkulepitud järjekorranumbrist, mille abil vastuvõtja saab segmendid õiges järjekorras kokku panna (mõni segment võib vahel kaduma minna ja mõni segment võib kohale jõuda enne temast varem teele saadetud segmenti). Iga järgmise segmendi järjekorranumber kasvab vastavalt eelmise segmendi andmete hulgale baitides. Teisiti öeldes on järjekorra numbriks arv, mis näitab relatiivselt, mitmendast baidist vastava segmendi andmed algavad. Järjekorranumbri jõudmisel numbrini 2^{32}-1 (maksimaalne välja väärtus) on järgmine järjekorranumber 0. Järjekorranumbrid on erinevates ühenduse suundades sõltumatud.

Andmeedastuse mehhanismid

TCP tagab usaldusväärse ühenduskanali, kus saadetud andmed peavad jõudma vastuvõtjani. Kättesaamise kontrollimiseks peab vastuvõtja kinnitama, et ta on kätte saanud voo andmed mingi kohani (baidini, relatiivselt). Vastuvõtja pool kinnitab segmendi kättesaamist vastusega, pannes kinnituse numbri väljale järgmise oodatava osa andmete algusbaidi järjekorranumbri. See tähendab, et kui vastuvõtja sai kätte näiteks segmendi, mille viimane bait oli järjekorranumbriga 1478301238, siis vastuse segmendi kinnituse numbri väljale pannakse 1478301239. Kui saatja ei saa määratud aja jooksul (tavaliselt 3 sekundit) kinnitust, siis loeb ta segmendi kaduma läinuks ja saadab antud segmendi uuesti. Vajaduse korral jätkatakse taassaatmist seni, kuni ajalimiit täis saab, seejärel katkestatakse ühendus. Toonitame, et järjekorranumber on relatiivne, alustatakse juhuslikust arvust ning ühenduse käigus võib saata rohkem kui 2^{32} baiti.

Põhimõtteliselt võib saata andmeid lõpmatult, sest osapoolte ühenduste akende suurused koos puhvritega on Internetis veel üldjuhul suurusjärgu võrra väiksemad. Vastuvõtja poolt saadetav kinnituse segment ei pruugi sisaldada andmeid. Näiteks faili saates saadab faili vastuvõtja pool enamasti segmente, mis ei sisalda andmeid, sest tal ei ole midagi saata, kuid ta peab kinnitama andmete kättesaamist.

Edastamise protsess oleks kõige lihtsam, kui saatja saadab korraga ainult ühe segmendi, millele jääb vastuvõtjalt kinnitust ootama. Paraku on selline infovahetus liialt aeglane. Seetõttu on kasulik saata enamasti korraga enne kinnituse saamist mitu segmenti. Kui palju andmeid saadetakse enne kinnituse ootamist, on määratav dünaamiliselt akna suuruse (*window size*) välja muutes. Akna suurusega määratakse, mitu baiti saadetud andmeid võib olla kinnitamata. Saatnud akna jagu baite, millele ei ole kinnitust saadud, jäädakse ootama kinnitust. Kinnituse saabudes või ajalimiidi lõppedes saab uuesti andmeid saata (ajalimiidi lõppedes saadetakse vastavad segmendid uuesti). Oluline on tähele panna, et kinnitus saadetakse positiivsel juhul (andmed saadi kätte), negatiivsel juhul (mingil põhjusel ei saadud andmeid, näiteks sattus pakett ummistusse, visati ära) ei saadeta andmete saatjale teadet. Akna suurusega kontrollib TCP edastustempot. Mida suurem on aken, seda rohkem andmeid saab saatja korraga teele saata, kuid osapoolte vahel olev ühendus peab ka seda võimaldama.

Arusaadavalt ei tea osapooled, kui suur on nende vaheline läbilaskevõime, seetõttu alustatakse aeglaselt ja ühenduse käigus suurendatakse akent, kuni jõutakse optimaalse akna suuruseni. Alustatakse ühe segmendi suuruse aknaga, kus segment on maksimaalse suurusega, mille määras vastuvõtja pool ühenduse loomise faasis (käepigistusel). Kui vastuvõtjalt saadakse tagasi kinnitus, et kõik saadetud andmed on kätte saadud, siis suurendatakse akna suurust kaks korda. Akna suurendamist jätkatakse iga eduka kinnituse saamise puhul, kuni enam ei saada kogu saadetu kohalejõudmisele kinnitust või jõutakse vastuvõtja poolt teavitatud maksimaalse akna suuruseni. Selliselt saavutatakse ühenduse läbilaskevõime võimalik lagi. Niisugust ühenduse läbilaskevõime korrigeerimise meetodit kutsutakse **aeglaselt alustamise algoritmiks**. Kui saavutatakse arvatava ühenduse läbilaskevõime lagi, minnakse edaspidi ummistuste vältimise (*congestion avoidance*) režiimi, kus toimub akna suuruse muutmine (suurendamine või vähendamine) segmendi võrra. Seega toimub alguses akna suuruse muutmine eksponentsiaalselt kasvavas tempos (suurendades kaks korda akna suurust) ja lae saavutamisel muudetakse akna suurust edaspidi lineaarselt vastavalt kinnitustele suuremaks või väiksemaks (akna suurus võib kõikuda ühenduse käigus). Ummistuste korral võidakse vähendada akna suurust poole võrra. Aeglase alustamise algoritmi nimi on natuke petlik. Nimelt alustatakse küll väga aeglaselt, kuid maksimaalne kiirus saavutatakse suhteliselt kiiresti. Algoritmi puhul eeldatakse, et kinnitused jäävad saamata võrgu ummistuste tõttu, mitte selle tõttu, et näiteks kaader visati minema kontrollsumma mittesobivuse tõttu. Seetõttu ei pruugi antud algoritm olla heade jõudlusnäitajatega näiteks traadita ühenduste puhul. Kadunud pakettidest saadakse teada kas ajalimiidi täissaamise või dubleerivate kinnituste saamisega.

Igale segmendile vastamine ei ole otstarbekas, sest see suurendaks meta-andmete osakaalu koguliikluses, tõstes ühendusteede koormatust. TCP kasutab kinnituste saatmisel viivitusega

kinnituste saatmist. Nimelt saadetakse kinnitusi kumulatiivselt ehk ühe korraga, kogudes edastatavaid kinnitusi teatud aja jooksul (suurusjärgus 100-500 ms). Saadetavas kinnituse segmendis on kinnituse numbriks järgmise oodatava baidi relatiivne järjekorranumber, see tähendab, et kinnituse järjekorranumber on eelmisest kinnituse järjekorranumbrist suurem mitme segmendi andmete võrra.

Kui mingil ühenduse etapil oli akna suurus näiteks kaheksa segmendi suurune, kuid vastuvõtjani millegipärast ei jõua näiteks viies segment, siis vastuvõtja saadab tagasi kinnituse, mis viitab viiendas segmendis olevale esimesele baidile. See tähendab, et kui saatja ei saa viienda segmendi kohalejõudmise kinnitust, siis saadab ta segmendid uuesti (vahepeal võib juba uusi segmente teele panna, kui akna suurus lubab).

Võib juhtuda, et segmentide kohalejõudmise kinnitused ei jõua enne ajalimiidi täissaamist saatjani (näiteks ühendusteede ummistuste tõttu), siis loeb saatja segmendi(d) kaduma läinuks ja saadab andmed uuesti. Sellisel juhul võib vastuvõtja saada andmeid dubleeritult. Tänu järjekorranumbritele saab vastuvõtja pool kõrvale heita dubleeritult saadud andmed.

Mõnikord juhtub, et teele saadetakse mitu segmenti, kuid mingi keskmine segment ei jõua kohale. Jäädes ootama saatja poolt segmendi uuesti saatmist ning ületades kinnituse saamise ajalimiidi, tähendab see andmeedastamise kiiruse vähenemist. Seetõttu kasutatakse TCP puhul vajadusel ka võimalust, et küsitakse uuesti vastavaid segmente. Nõnda ei pea saatja ootama kinnituse saamise ajalimiidi lõppemist. Seda nimetatakse TCP puhul kiireks taassaatmiseks (*fast retransmit*).

Et vähendada asjatuid segmentide uuestisaatmisi, on TCP-s kasutusel meetod, mida nimetatakse kiireks taastumiseks (*fast recovery*). Selle meetodi puhul kasutatakse ühe võimalusena valikulise kinnituse teadet SACK (*selective acknowledge*), et kõiki neid segmente kinnitada, mis on tõesti kohale jõudnud. Sellega piiratakse uuesti saadetavate segmentide arvu. SACK on realiseeritud TCP valikute väljal ning selle kasutamise lubamises lepitakse kokku ühenduse loomise käigus (kõik hostid ei pruugi seda toetada).

Üks moodus, kuidas TCP võib vähendada sideliinide koormatust, kui kasutatakse tillukesi datagramme, on kasutada **Nagle algoritmi** (RFC 896). Selle mooduse puhul kogutakse andmeid puhvrisse ning saadetakse edasi alles siis, kui teiselt osapoolelt on saadud kinnitus või kui puhvrisse on andmeid kogunenud maksimaalse segmendi suuruse jagu, kuid ei ole ületatud akna suurust. Seega saadetakse ühes segmendis rohkem andmeid, millega vähendatakse ballasti osakaalu. Selliselt koormatakse sideliine vähem, aga koormatud ühenduste puhul on vastused suurema viivitusega. Segmente vahetatakse omavahel kordamööda. Nagle algoritmi on sobilik kasutada näiteks telnet ühenduste puhul, kuid ei ole sobilik suurte andmemahtude (näiteks FTP) või kiiremat reageerimist (näiteks X Window) nõudvate edastamiste korral. Telneti puhul vajutatakse mingit klahvi, mis saadab info serverisse. Server otsustab, mis tegevus sooritatakse (näiteks väljastatakse märk ekraanile või sooritatakse mingi muu operatsioon). Kui ühendus on aeglane, siis telneti abil midagi kirjutades ilmuvad tähed telneti aknasse viivitusega.

TCP peab tagama maksimaalse läbilaskevõime, mille saavutamiseks on vaja arvestada erandjuhtumitega. Toome näitena **rumala akna sündroomi** (*silly window syndrome*) (RFC 813). Oletame, et esimesel hostil on teisele hostile vaja saata palju andmeid. Olgu ühenduse akna suuruseks 8192 ja maksimaalseks segmendi suuruseks 1024 baiti. Normaalsel juhul saadetakse korraga teele kaheksa segmenti ja jäädakse kinnitust ootama, mille järel saadetakse uuesti kaheksa segmenti suurusega 1024 baiti. Oletame, et mingil põhjusel saadetakse andmeid, mis on vaja koheselt edastada. Selleks märgitakse segmendil PSH lipp ja saadetakse väike segment koheselt teele. Oletame, et selle segmendi pikkus oli 50 baiti. Edasi jätkatakse maksimaalse suurusega segmentidega. Selle tulemuseks on, et iga viimase ehk üheksanda segmendi suurus on 50 baiti (sest andmeid pannakse teele akna suuruse jagu). Selline asi vähendab jõudlust ja lõpeb andmete edastamise lõpetamisel. Üks lahendus on vähendada ajutiselt vastuvõtja poolt lubatud akna suurust (antud näite puhul vähendataks akna suurust 50 baidi võrra ehk 8142 baidile). Hiljem akna suurus taastatakse (antud näite puhul 8192 baidile). Saatja pool võib viivitada järg-

miste segmentide teele saatmisega, oodates ära kinnitused. Peale kinnituste saamist jätkatakse normaalset edastamist.

TCP korral võib tekkida probleeme järjekorranumbriga juhul, kui ühenduskiirus kahe otspunkti vahel on 1 Gbit/s või rohkem, mille käigus ei pruugi segmendi aegumise aeg täis saada (kuni 120 sekundit). 1 Gbit/s ühenduse maksimumkiirusel saab järjekorranumbrite ring täis ligikaudu 35 sekundiga. Selle tõttu on TCP protokollile lisatud juurde valik PAWS (*protect against wrapped sequences*), mis lisab segmentidele ajatempli, mille abil saab kindlaks teha järjekorra.

Tänu TCP vookontrollile ei pruugi saatva hosti rakendusprogrammi andmed jõuda samasuguste portsudena sihtpunkti, mida peab arvestama rakendusprogrammi programmeerimisel.

Pordid

Kuna ühes masinas on tavaliselt mitu erinevat programmi, mis peavad saama võrgu kaudu teiste hostidega ühendust pidada (ka samal ajal), siis on vaja ka mehhanismi, mille abil kindlaks teha, milline programm peaks saama võrgust saadud andmed. Selleks kasutatakse mõistet port. Port on TCP mõistes number, mis määrab üheselt, millisele programmile andmed edasi saata. Selleks hõivab iga programm, mis võrgu kaudu suhtleb, antud arvutis endale pordi(d). Pordi number (numbrid) seotakse mingi konkreetse protsessiga vastavas masinas. Seega peab ühenduse alustaja, mis soovib suhelda sihtmasina mingi programmiga, teadma programmi poolt hõivatud TCP pordi numbrit. Kuna ühenduse algataja peab teadma õiget pordi numbrit, siis on teenuste jaoks kokku lepitud vastavad pordid. Portide registreeringuid haldab IANA (*Internet Assigned Numbers Authority*). Näiteks protokollile HTTP (*Hypertext Transfer Protocol*) (kasutab veebiserver) on registreeritud port 80; FTP pordiks on 21 (pordil 20 liigutatakse andmeid); Telneti pordiks on 23; SSH pordiks on 22; DNS (*Domain Name System*) pordiks on 53 jne. Pordid ühest kuni 1023-ni on nn. "hästi teada pordinumbrid" (*well-known port numbers*). Ülejäänud pordid (1024 kuni 65 535) on vabalt kasutatavad programmide poolt. Porte 1024 kuni 49 152 nimetatakse registreeritud portideks, mis registreeritakse samasuguselt programmidele, mis pole üldlevinud ja mis üldjuhul paigaldatakse eraldi. IANA-s registreeritud pordid on saadaval aadressil http://www.iana.org/assignments/port-numbers. Porte 49 152 kuni 65 535 tuntakse dünaamiliste või privaatsete portidena. Mõeldud kasutamiseks dünaamiliste klientrakenduste puhul (enamasti ei kasutata, v.a. failijagamisprogrammid). Toonitame, et pordinumbrid on kokkuleppelised numbrid. Keegi ei keela programmil lasta avada mingit muud suvalist porti. Näiteks veebiserveri võib seadistada masinas hõivama pordi 81. Operatsioonisüsteemi tasemel ei pruugita lubada tavakasutaja õigustes üldiselt programmidel avada porte, mis on väiksemad kui 1024 (näiteks UNIX). UDP omab ka porte, mis on TCP portidest sõltumatud (masinas võib üks programm hõivata mingi TCP pordi numbri, mingi teine programm võib hõivata sama numbriga UDP pordi), pordi numbrid kirjutatakse kujul <pordi number>/<protokoll>, näiteks "80/tcp", "53/udp", "53/tcp". Pordid on TCP ja UDP sisesed asjad, seega on nad sisuliselt üksteisest eraldiseisvad. IP-aadresside märkimiseks koos pordi numbriga kasutatakse tavaliselt kirjaviisi, kus IP-aadress on pordinumbrist eraldatud kooloniga (näiteks "192.168.24.56:123").

Kui mingi programm soovib võrgu kaudu suhelda, siis avatakse talle mingi port, mis on suurem kui 1024. Näiteks võidakse operatsioonisüsteemi poolt anda veebibrauserile port 1753. See, port 1753, saab olema pordiks, kuhu saadetakse info tagasi (näiteks veebileht, pildid jms). See tähendab, et kui veebibrauser soovib suhelda veebiserveriga, siis ta paneb sihtpordiks 80. Lähtepordi annab tavaliselt operatsioonisüsteem. Antud juhul läheks siis lähtepordiks 1753. Veebiserver saadab tagasi vastuse, kus paneb sihtpordiks 1753 ja lähtepordiks 80 (vahetab ümber). Pordi number on sisuliselt masinasisene aadress. Analoogina tavakirja saatmisel märgitakse aadress (näiteks Tartu, Kuuse 2-1) ja saaja nimi (näiteks Aita Välja), mis oleks sarnane pordinumbriga. Nimega määrame ära, kes selle kirja saab (näiteks elab samal aadressil ka August Välja). See on selle korteri sisene asi, kes selle kirja saab (ehk korterisisene aadress). Võib öelda, et arvutis jooksvate protsesside vahelise ühenduse mingil ajahetkel määravad üheselt siht- ja lähtehosti ära IP-aadressid, siht- ja lähtehosti pordid ning kasutatav transpordikihi protokoll.

8.2 UDP

Teine põhiline transpordikihi protokoll Interneti protokollistikus on UDP (*User Datagram Protocol*) (RFC 768). UDP on lihtne ja vähese ballastiga protokoll, mis on ühenduseta ega taga, et saadetav info sihtpunkti jõuab. Seetõttu on UDP kiirem, sest osapoolte vahel ei looda ühendust ning segmendi päis on väiksem. Samuti ei ole UDP-s järjekorranumbreid, mille tõttu ei ole transpordikihis võimalik segmente õiges järjekorras kokku panna. Vajadusel võib vajatavad asjad eraldi ülemistes kihtides realiseerida. UDP segment koosneb järgmistest väljadest:

- Lähteport. Sama funktsioon TCP samanimelise väljaga, kuid UDP puhul ei ole vaja märkida, kui ei ole vaja vastust saata (märgitakse nulliks (port 0 on reserveeritud)). 16 bitti.
- Sihtport. Sama funktsioon TCP samanimelise väljaga (erinevalt lähtepordist on sihtport kohustuslik). 16 bitti.
- Pikkus. Päise ja andmete pikkus baitides. 16 bitti.
- Kontrollsumma. Mittekohustuslik (välja väärtus sel juhul null). Arvutatakse analoogselt TCP kontrollsumma väljale UDP päise, pseudopäise ja andmete pealt. 16 bitti.
- Andmed.

Portidel on UDP puhul sama funktsioon, nagu TCP puhul. UDP segmendi võib ka fragmenteerida, kuid see pole tihti soovitatav.

Erinevalt TCP-st on võimalik UDP puhul kasutada sihtaadressina ka multi- ja leviedastusaadresse. TCP puhul pidi looma kahe otspunkti vahelise ühenduse, kuid UDP-d kasutades pole see vajalik ning seetõttu on võimalik saata korraga mitmele vastuvõtjale (multi- ja leviedastuse kasutamine annab väga palju võimalusi andmete saatmiseks korraga mitmele osapoolele). Seda kasutatakse näiteks heli- ja videokonverentside puhul. Multiedastust vaatleme lähemalt peatükis "Multiedastus" (lk. 86).

8.3 Transpordikihi lõpetuseks

Transpordikiht on kõige ülemine OSI kiht, mis on seotud otseselt võrguga. Kõrgemad kihid on programmispetsiifilised ja ei ole nii tihedalt võrgutehnoloogiaga seotud. Transpordikihi protokolli valiku teevad programmid (vastavalt protokollile, mida kasutatakse). Näiteks TCP protokolli kasutavad FTP, SMTP, HTTP, DNS jne. UDP protokollid on näiteks DNS, SNMP, TFTP jne., samuti üldjuhul VoIP, reaalaja-video edastuse protokollid.

9. OSI seansi-, esitus- ja rakenduskiht

OSI ülemised kihid ei ole enam otseselt võrguga seotud ja nad on rohkem rakendusprogrammide spetsiifilised. Seetõttu vaatleme neid põgusamalt ja koos.

Seansikiht

Seansikihi ülesandeks on hallata loogilisi seansse hostide vahel. Siin luuakse, hallatakse, katkestatakse, jätkatakse jms. ühendusseansse hostide vahel. Tihti ei ole see kiht kasutusel ega vajalik.

Kasutatavamad seansikihi protokollid on:

- TLS (*Transport Layer Security*) ja SSL (*Secure Sockets Layer*) – kasutatakse krüpteeringuga turvalise suhtluskanali loomiseks. Kasutavad näiteks HTTPS (*Hypertext Transfer Protocol Secure*) (veebilehed. URL algab "https://"), SMTP (*Simple Mail Transfer Protocol*) (RFC 821) (tavaliselt pordil 25, kuid SMTP üle SSL-i seadistatakse tihti pordile 465).
- SSH (*Secure Shell*) – otspunktide vahel luuakse turvaline kanal, kasutades avaliku võtmega krüptosüsteemi. Sellele kanalile on võimalik ehitada mitmeid erinevaid protokolle. Näiteks SFTP, SCP.
- RPC (*Remote procedure call*) – võimaldab ühel arvutil teostada funktsioonide väljakutseid ka teistest arvutitest.
- NetBIOS (*Network Basic Input/Output System*) – võimaldab suhelda arvutitel üle kohaliku võrgu.
- PPTP (*Point-to-Point Tunneling Protocol*) – virtuaalsete privaatvõrkude ehk VPN (*virtual private network*) implementeerimise protokoll.

Esituskiht

Esituskihi ülesanneteks on pakkuda rakenduskihile andmete sõltumatust platvormist. Siia alla käib kodeerimine (mismoodi näiteks stringe, numbreid jne. esitatakse) (näiteks MIME), andmete pakkimine ja krüpteerimine, serialiseerimine (objektid, andmestruktuurid jne.) jms. Sageli kasutatakse XML (*Extensible Markup Language*) tehnoloogiat. Enamasti on esituskiht ja rakenduskiht koos, näiteks HTTP protokolli puhul.

Esituskihi protokollina võiks välja tuua XDR (*eXternal Data Representation*). XDR-i kasutab näiteks ONC RPC (*Open Network Computing Remote Procedure Call*), mis on Suni kaugprotseduuride realisatsioon.

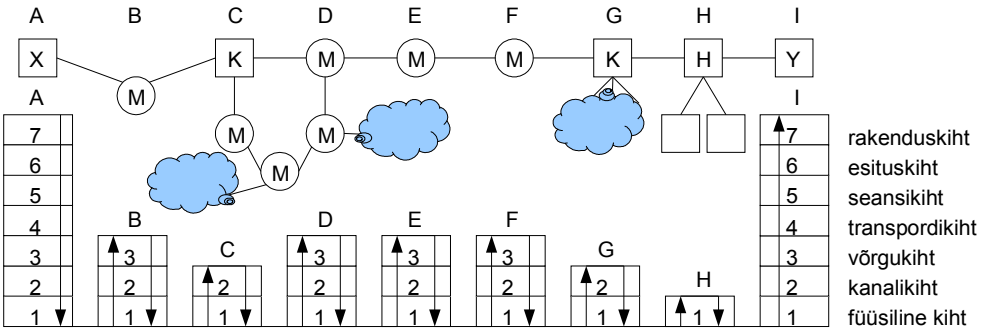
Rakenduskiht

Kõige ülemine OSI kiht on rakenduskiht, kuhu kuuluvad programmide eneste protokollid, mida on väga palju. Mõningaid tuntumaid rakenduskihi protokolle on HTTP (*Hypertext Transfer Protocol*) (veebilehed), SMTP (*Simple Mail Transfer Protocol*) (e-maili saatmine), FTP (*File Transfer Protocol*) (failide transport), Telnet (käsurea kasutamine üle võrgu), DNS (*Domain Name System*).

Kokkuvõte

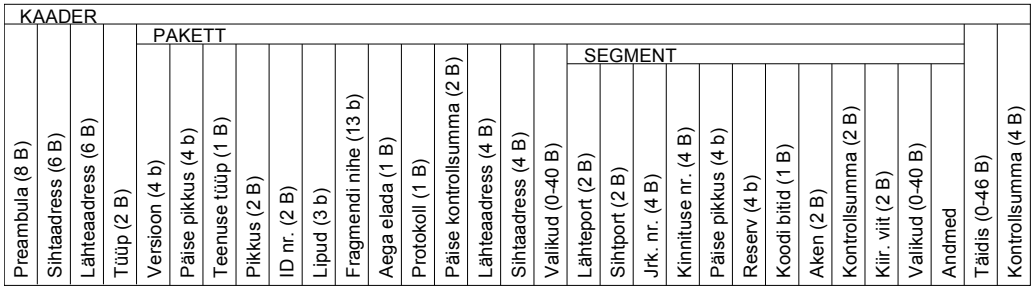
Kui mingi arvutis olev programm soovib alustada üle võrgu suhtlust mingi teise programmiga (ehk mingi teenusega), siis ta peab teadma teise programmi aadressi. Üldine aadress koosneb põhimõtteliselt kahest osast. Esimene on teise arvuti aadress võrgus (meie käsitlesime selle aadressina IP-aadressi) ning teine on teise arvuti sisene aadress (ehk pordi number). Programmi siseselt peaks olema selge, kas ja mismoodi teostatakse seanss ja andmete esitus osapoolte vahel (oleneb protokollist). Programmid kasutavad tihti kolmandate osapoolte teeke (näiteks SSH).

Ülejäänu eest hoolitsevad neli alumist kihti, millega programm, andes ette vajalikud parameetrid, enam ise ei pea tegelema. Programm saab kasutada transpordikihi protokolli, näiteks TCP või UDP (masinas on pakkuda ka muid transpordikihi protokolle). Kui kasutatakse TCP-d, siis hoolitsetakse selle eest, et andmed jõuaksid kohale soovitud porti, IP-aadressi järgi saadetakse andmed soovitud kohta. Ethernet hoolitseb selle eest, et andmed jõuaksid järgmisesse masinasse, teostades füüsilise adresseerimise ja andmete füüsilise saatmise üle sideliini. Joonisel 9.1 on ülevaatlik näide võrgusuhtluse andmete käitlemisest.

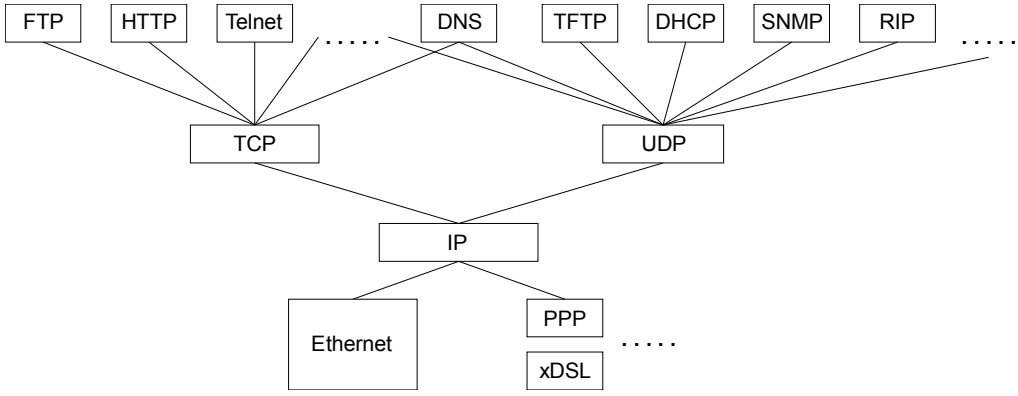


Joonis 9.1: Näidatud on andmetega tehtav töö. Andmete edastus toimub masinast A masinasse I. Nooled tähistavad andmete käitlemist OSI kihtide kaupa. Üleval on võrguseadmed ja nende vahel olevad ühendused. Masinas A liiguvad andmed OSI kihis ülevalt alla, kuni jõuavad füüsilise kihini, mille kaudu saadetakse andmed järgmise masinani, milleks on antud juhul marsruuter, mis võtab andmed lahti kuni kolmanda kihini, kust saab teada IP-aadressi ja teeb muud toimingud. Edasi suunatakse pakett kommutaatorile C, mis võtab andmed lahti ainult teise kihini ja suunab kanalikihi aadressi alusel edasi marsruuterile D. Marsruuterid marsruudivad paketi edasi kuni marsruuterini F, mis saadab andmed kohaliku võrgu kommutaatorile G, mis suunab need edasi porti, kus asub hub, mis saadab kaadri edasi kõikidesse portidesse, kaasaarvatud sihtmasinasse. Me eeldame, et marsruuterid ei sisalda muid funktsioone (nt. tulemüürid).

Joonisel 9.2 on kokku võetud seni vaadeldud protokollide poolt andmetele lisatavad väljad, mis läbi võrgu saadetakse (vaadeldes Ethernet + IP + TCP) (segmendi andmetes sisalduvad ka seansi-, esitus- ja rakenduskihi metaandmed) ning joonisel 9.3 on mõningatest protokollidest tehtud protokollipuu.



Joonis 9.2: Seni vaadeldud protokollide (transpordikihini) poolt lisatud ballast, mida kasutatakse saatmisel TCP/IP võrgus üle Etherneti.



Joonis 9.3: Näide protokollide omavahelisest seotusest protokollipuus.

10. Veel olulisi võrgutehnoloogia mõisteid

Nüüd oleme saanud esmase aluspõhja arvutivõrgu üldise toimimise kohta Ethernetist TCP/IP baasil. Järgnevalt vaatleme mõningaid samuti olulisi võrgutehnoloogia teemasid ja mõisteid.

Võrkude tüüpe

Internet on oma olemuselt võrkude võrk. Võrke, millest Internet koosneb, võib liigitada erinevateks tüüpideks. Liigitust ei saa siiski igakord täpselt teha. Võrkude tüüpidena eristatakse tinglikult (tegemist ei ole lõpliku nimekirjaga):

- LAN (*local area network*) ehk kohtvõrk. Üldiselt seob mõõdukalt suurel alal olevad hostid. See ala võib olla üks või mitu tuba, korrus, maja, hoonete kompleks. Mõõdukas suurus võib varieeruda mõnest meetrist mõne kilomeetrini. IEEE tõmbab tavaliselt piiri kuni 10 km raadiuses. Tüüpiliselt on kasutatavateks tehnoloogiateks Ethernet, IEEE 802.11, harvemini Token Ring, FDDI (*Fiber Distributed Data Interface*) ja ATM.
- WAN (*wide area network*) ehk laivõrk. Ühendab masinaid, mis asuvad geograafiliselt laial alal, ühendades linnu, riike jms. Alampiiriks loetakse mõnikord 8 km raadiusega ala. WAN on üldiselt LAN-ide ühendaja. Öeldakse ka, et WAN on LAN-de kogum. Sideliinidena võidakse kasutada ka näiteks satelliidiühendusi ja merealuseid kaableid jms. Tehnoloogiatena võidakse kasutada: DSL, Frame Relay, ATM, SONET/SDH jpt.
- MAN (*metropolitan area network*) ehk regionaalvõrk. Näiteks mingi suurem firma või asutus võib omada linnasisest või laiemat asutusesisest võrku, mis ühendab näiteks osakondasid. MAN on suurem kui LAN, kuid väiksem kui WAN. Tüüpilisteks tehnoloogiateks Gigabit Ethernet (valguskaabli baasil), ATM, FDDI jt.
- CAN (*campus area network*) ehk territoriaalvõrk. Mitu LAN võrku, mis on omavahel ühendatud ja asuvad piiratud alal. Näitena võiks välja tuua ülikoolilinnaku. CAN-i võib vaadelda ka kui MAN erivormi (CAN üldiselt väiksemal alal kui MAN).
- HAN (*home-area network*) ehk koduvõrk. Ühendab kasutaja kodus olevaid seadmeid (lisaks arvutitele ka muid võrgusuhtlust oskavaid välisseadmeid, nagu printer, televiisor, turvasüsteem, külmkapp).
- PAN (*personal area network*) ehk personaalvõrk. Väikesed inimeste kodude võrgud. Koduvõrkude eesmärgiks on näiteks printeri, failide, Interneti jms jagamine mitme arvuti vahel (lauaarvuti, laste arvuti, sülearvuti). Mõnel juhul mõeldakse ka ühte isikut ja tema arvutit, millega on ühendatud hiir, klaviatuur, printer ja muud lisaseadmed (ka traadita ühenduste puhul). Tegevusala umbes 10 meetri raadiuses. Protokollidest kasutatakse tavaliselt USB (*Universal Serial Bus*), FireWire, Bluetooth, Wi-Fi, IrDA (*Infrared Data Association*). Mõnikord eristatakse ka traadita personaalvõrku ehk WPAN (*wireless PAN*), mis hõlmab traadita ühendused.
- SAN (*storage-area network*) ehk salvestusvõrk. SAN on eriotstarbeline võrk, mida kasutatakse serverite ja salvestusseadmete ühendamiseks. Siin on nõudmisteks seatud kõrgjõudlus, mastabeeritavus ja tõrkekindlus. Protokollina kasutatakse enamasti SCSI ja muid seotud protokolle. SAN võib olla kasutusel suurtes asutustes.
- BN (*backbone network*) – kiire magistraalvõrk LAN-de ühendamiseks. Ulatus kuni mõni kilomeeter.
- GAN (*global area network*) ehk globaalvõrk. Globaalvõrguks nimetatakse WAN-ide kogumikku, mis katab terve maakera.

Laias laastus jagatakse Internet WAN ja LAN võrkudeks. Teised nimetatud on täpsemad määratlused. WAN ja LAN võrkudeks jagamine on tinglik, sest nende vahele ei saa tõmmata alati selget piiri ja see piir on viimasel ajal veelgi ähmastunud.

Klient-server ja partnerilt-partnerile ühendused

Arhitektuuriliselt jaotatakse ühendused hostide vahel klient-server ja partnerilt-partnerile (*peer-to-peer*) ühendusteks. Klient-server ühenduste puhul on olemas server, mis pakub mingisuguseid teenuseid ja kliendid on need, mis neid teenuseid kasutavad. Aktiivne pool on klient, mis esitab päringuid, mille peale server võib teha mingi toimingu ja vastata kliendile (ta ei ole kohustatud, näiteks võib nõuda server autentimist). Seega on server passiivne ja klient aktiivne osapool. Kuna klient alustab ühendust, siis peab klient teadma serveri aadressi. Näiteks kui kasutaja külastab veebibrauseriga mingit kodulehte, siis on kliendi pooleks kasutaja arvuti ja serveri pooleks kodulehe host.

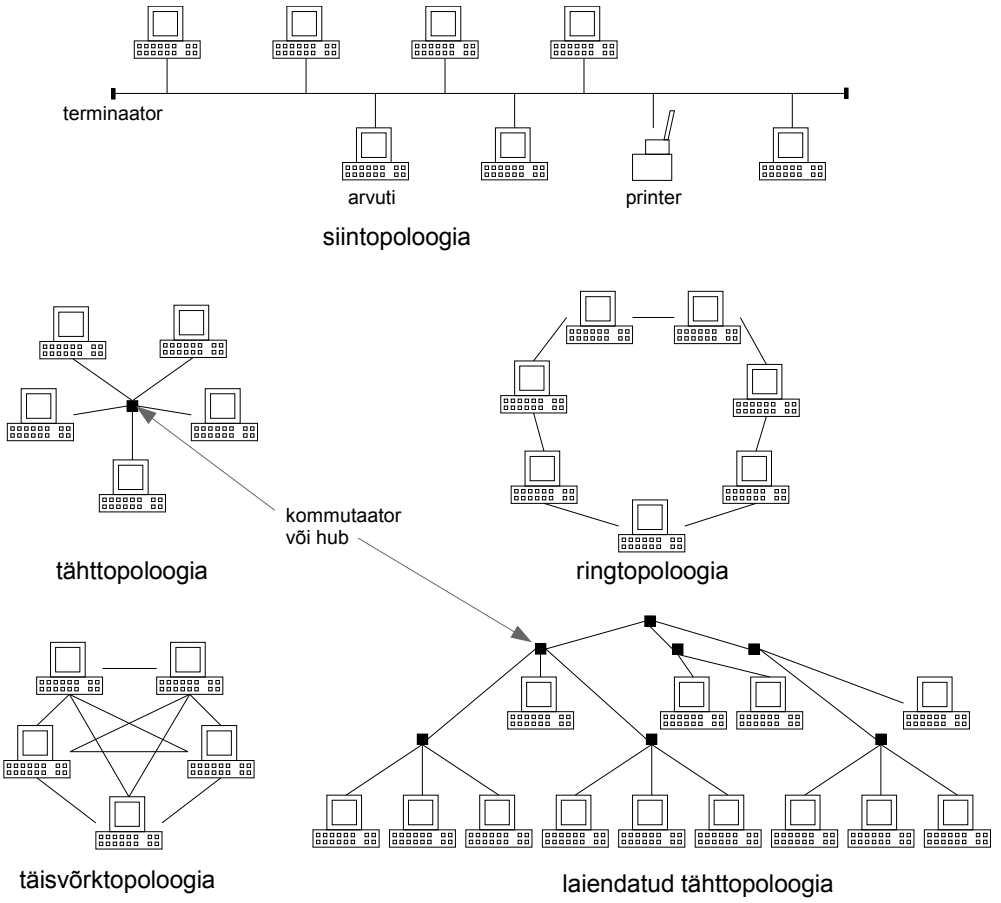
Partnerilt-partnerile ühenduse puhul on mõlemad osapooled võrdsed. See tähendab, et mõlemad osapooled saavad üksteisele teha päringuid ja neile vastata. Partnerilt-partnerile võrkude heaks näiteks on failivahetusprogrammid. Klient-server süsteemid on tavaliselt tsentraliseeritud süsteemid ja partnerilt-partnerile süsteemid tavaliselt hajusad.

Topoloogiad

Võrku on võimalik kujutada skeemina. Võrgutehnoloogias nimetatakse sellist skeemi topoloogiaks. Topoloogiat, mis näitab võrgus olevaid masinaid ning nende vahelisi füüsilisi ühendusi, nimetatakse **füüsiliseks topoloogiaks**. Topoloogiat, mis näitab, kuidas võrk on loogiliselt üles ehitatud, nimetatakse **loogiliseks topoloogiaks**. Loogiline ja füüsiline topoloogia tavaliselt kattuvad, kuid kui võrgus kasutatakse näiteks virtualiseerimist (näiteks virtuaalseid kohtvõrke), siis võivad füüsiline ja loogiline topoloogia üksteisest mingil määral erineda. Loogiline topoloogia ehitatakse füüsilise topoloogia peale ja seetõttu sõltub viimasest.

Võrgu disainimisel peaks eelnevalt valima sobiva võrgutopoloogia. Topoloogia peab arvestama võrgu eripäradega, näiteks võrgu suurusega, liikluse omapäraga, turvanõuetega, tõrkekindluse nõuetega, ehitamise maksumusega, olemasoleva infrastruktuuriga, võrgu laienemise perspektiividega jne. Järgnevalt toome mõningaid üldisi kohtvõrkude topoloogiaid (vt. ka joonist 10.1):

- Siintopoloogia (*bus topology*) – sõlmed ehk masinad on ühendatud ühe sideliini külge. Nõndamoodi on sideliini kogupikkus väike. Kasutati eelkõige koaksiaalkaablit. Tänapäeval praktiliselt ei kasutata.
- Tähttopoloogia – sõlmed on ühendatud läbi keskse võrguseadme, milleks on tavaliselt kommutaator, harvemini hub. Kõik ühendused läbivad keskset võrguseadet. Kohalikes võrkudes on see kõige enam kasutatavam ühendusviis. Iga sõlme juurde viiakse eraldi sideliin kesksest võrguseadmest.
- Ringtopoloogia – kõik sõlmed on ühendatud kahe teise arvutiga üheks suureks ringiks. Tänapäeval praktiliselt ei kasutata.
- Võrktopoloogia (*mesh topology*) – igal masinal on ühendus vähemalt kahe naabersõlmega ja kogu võrk on sidus. Võrktopoloogiana vaadeldakse ka võrku, kus on vähemalt kaks sõlme, mis on ühenduses kahte erinevat teed pidi.
- Täisvõrktopoloogia (*full mesh topology*) – iga sõlme juurest on kõigi teiste sõlmede juurde eraldi sideliin.
- Laiendatud tähttopoloogia (*extended star*) – võrgus on mitu tähttopoloogiat, mille kesksed sõlmed (võrguseadmed) on omakorda omavahel ühendatud.
- Puutopoloogia (ka hierarhiline topoloogia). Võrk on sidus, kuid puuduvad dubleeritud ühendused.



Joonis 10.1: Kohtvõrkude topoloogiate näiteid.

Ühendusviisid

Erinevad tehnoloogiad erinevad andmete edastamise viisi poolest. Üldiselt võib ühendusviisid jagada kaheks:

- Kanalkommutatsioon (*circuit switched*). Otspunktide vahel moodustatakse kommutatsiooni vältel füüsiline ühendus. Kommunikatsiooni ajal on kogu sideliin hõivatud. Näideteks on lauatelefon ja ISDN.
- Pakettkommutatsioon (*packet switched*). Korraga saadetakse sideliinis mingi kogus grupeeritud bitte (ehk kaader). Kui midagi saata ei ole, siis saavad teised masinad kanalit kasutada. Näiteks Ethernet, ATM, Frame Relay.

Pakettkommutatsiooni võib ühenduse alusel samuti jagada kaheks:

- Ühenduseta ühendus. Otspunktide vahel ei looda eelnevalt ühendust. Täisaadressi järgi suunatakse igas võrgusõlmes pakett edasi sihtpunkti poole. Näiteks IP on ühenduseta protokoll.
- Ühendusorienteeritud ühendus. Teekond on ette määratud. Näiteks Frame Relay.

Pakettkommutatsiooni võib kaheks jagada ka paketi suuruse varieeruvuse järgi:

- Muutuva suurusega. Näiteks Ethernet, Frame Relay.
- Fikseeritud suurusega. Näiteks ATM, SMDS.

ISP-de hierarhia

ISP-d ehk internetiteenusepakkujad jaotatakse tinglikult kolmeks tasemeks. Erinevate tasemete nimed on vastavalt: esimene tase (*tier-1*), teine tase (*tier-2*), kolmas tase (*tier-3*).

Esimese taseme internetiteenusepakkujad asuvad ISP-de hierarhia ülemisel tasemel ja nemad moodustavad globaalse Interneti magistraalvõrgu. Esimese taseme klientideks on teise taseme internetiteenusepakkujad, väga suured organisatsioonid ja firmad. Seetõttu peavad nende ühendused ja teenused olema töökindlad ja kiired (kasutades ka dubleeritust). Esimese taseme ISP-sid on Internetis ligikaudu kümme või natuke enam (oleneb ISP-de vahele piiri tõmbamisest). Mõningaid esimese taseme ISP-sid: Sprint Nextel (http://www.sprint.com), Verizon Business (http://www.verizonbusiness.com), AT&T (http://www.att.com), Level 3 (http://www.level3.com).

Teise taseme internetiteenusepakkujad on hierarhia teisel tasemel. Nad on ühendatud esimese taseme ISP-dega ja pakuvad teenuseid reeglina suurematele firmadele ja kolmanda taseme internetiteenusepakkujatele.

Kolmanda taseme internetiteenusepakkujad on hierarhia kõige alumisel tasemel. Nad on ühenduses teise taseme ISP-dega ja pakuvad teenuseid reeglina kodukasutajatele, väikestele ja keskmise suurusega firmadele ning organisatsioonidele.

11. Teisi olulisi protokolle ja IP laiendusi

11.1 DHCP

Seni oleme käsitlenud masinatele käsitsi mingi sobiva IP-aadressi määrangut. Tihti on olukordi, kus käsitsi kõigile masinatele IP-aadresside panemine on problemaatiline: kasutajad ei pruugi selleks vastavaid õigusi või teadmisi omada; kerged on tekkima IP-aadresside konfliktid, kus kaks või enam arvutit sätib omale samasuguse IP-aadressi; kasutajate jaoks on oluline mugavus ja lihtsus; mõnes kohas on kasutajad antud võrgus ainult ajutiselt (näiteks Wi-Fi võrgu puhul); kasutajaid võib summaarselt rohkem olla kui IP-aadresse, kuid kõik nad korraga ei ole võrgus; peale IP-aadressi on vaja masinale ka muud informatsiooni (nt. võrguvärav, DNS-server); võrguadministraatoritel oleks palju mittesisulist tööd. Need ja paljud teised probleemid võivad tekkida IP-aadresside käsitsi seadistamise korral.

Nende probleemide lahendamiseks on võimalik kasutada protokolli DHCP (*Dynamic Host Configuration Protocol*) (RFC 2131, 3396, 4388). DHCP töötab klient-server mudeli põhjal. DHCP klient on praktiliselt kõikidesse uutesse ja suurematesse operatsioonisüsteemidesse sisse ehitatud. DHCP on BOOTP edasiarendus ja asendaja. BOOTP andis ainult eeldefineeritud tabeli põhjal MAC-aadressile vastava IP-aadressi, võrgumaski, võrguvärava ja DNS-serveri aadressi. DHCP töötab BOOTP-ga samadel portidel (serveri port 67 ja kliendi port 68, transpordikihi protokollina kasutatakse UDP protokolli) ja teated on samuti samas formaadis. DHCP ülesandeks on jagada masinatele IP-aadresse ja muid vajalikke parameetreid. Seda protsessi nimetatakse liisimiseks, sest IP-aadress antakse küsivale masinale piiratud ajaks. Liisimisaja lõppedes pikendatakse liisimist, kui soovitakse IP-aadressi edasi kasutada. Vastasel juhul läheb IP-aadress taaskasutusse (masin on vaikselt lahkunud võrgust).

DHCP protokolli väljad on järgmised (lõpus on toodud välja suurus):

- op – teate tüübi kood, mis saab olla BOOTREQUEST (arvuliselt 1) (päringu puhul) BOOTREPLY(arvuliselt 2) (vastuse puhul). 1 bait.
- htype – füüsilise aadressi tüüp. Näiteks 10 Mbit/s Ethernet puhul "1". 1 bait.
- hlen – riistvaralise aadressi pikkus baitides. Etherneti puhul on väärtuseks 6 (MAC-aadress on kuue baidine). 1 bait.
- hops – DHCP klient paneb väärtuseks nulli. DHCP kasutab seda edastuse märgistamiseks paketi saatmisel väljapoole leviedastusdomeeni, kus igat marsruuterit läbides suurendatakse selle väärtust ühe võrra, kuni see enne DHCP-serverisse jõudmist mingi marsruuteri puhul ületab seadistatud maksimumväärtust (näiteks 2-4), mille peale visatakse see ära. Selliselt piiratakse DHCP paketi leviku kaugust. 1 bait.
- xid – juhuslik arv, mille määrab DHCP klient. Kasutatakse kliendi ja serveri vaheliste teadete seostamiseks. 4 baiti.
- secs – klient märgib, mitu sekundit tagasi ta alustas aadressi saamist või uuendamist (alguses on null). 2 baiti.
- flags – lipud. Kui klient ei ole ilma TCP/IP-d seadistamata võimeline vastu võtma mitteleviedastusvastuseid, siis pannakse esimene lipp püsti ja server saadab vastused leviedastusaadressile (mitte üksikedastusaadressile). Ülejäänud lipud on reserveeeritud tulevikulaienduste jaoks ja peavad olema nullid. 2 baiti.
- ciaddr – kliendi IP-aadress, mis täidetud DHCPREQUEST, DHCPINFORM ja DHCPRELEASE teates, teiste puhul null. 4 baiti.
- yiaddr – DHCP-serveri poolt pakutud IP-aadress kliendile. 4 baiti.
- siaddr – DHCP-serveri IP-aadress leviedastusdomeenist välja saatmisel. 4 baiti.
- giaddr – DHCP klient sätib välja väärtuse nulliks. Mõeldud leviedastusdomeenist välja saatva marsruuteri IP-aadressi jaoks (kellele DHCP-server vastuse tagasi saadab). Kui

see väärtus on null, siis see tähendab DHCP-serverile, et klient asub samas leviedastusdomeenis. Kui väärtus ei olnud null, siis paneb DHCP-server IP sihtaadressiks antud välja väärtuse. 4 baiti.

- chaddr – DHCP kliendi füüsiline aadress (Etherneti puhul MAC-aadress). 16 baiti.
- sname – DHCP-serveri hostinimi (null-termineeritud sõne kujul). 64 baiti.
- file – laadefaili nimi (nulltermineeritud sõne kujul). 128 baiti.
- options – mittekohustuslike parameetrite väljad. Näiteks saab klient öelda soovitava IP-aadressi (klient võib olla selle IP-aadressiga varasemalt konfigureeritud, näiteks enne taaskäivitust). 312 baiti.

Väljasid "hops", "siaddr" ja "giaddr" kasutatakse olukorras, kus tahetakse kasutada keskset DHCP-serverit, mis teenindab mitut võrku. Nende väljade abil on võimalik seadistada marsruutereid vastavat paketti vahendama, saates ta DHCP-serverisse üksikedastusena. Sellisel juhul peab võrgu marsruuter olema eraldi seadistatud DHCP releena.

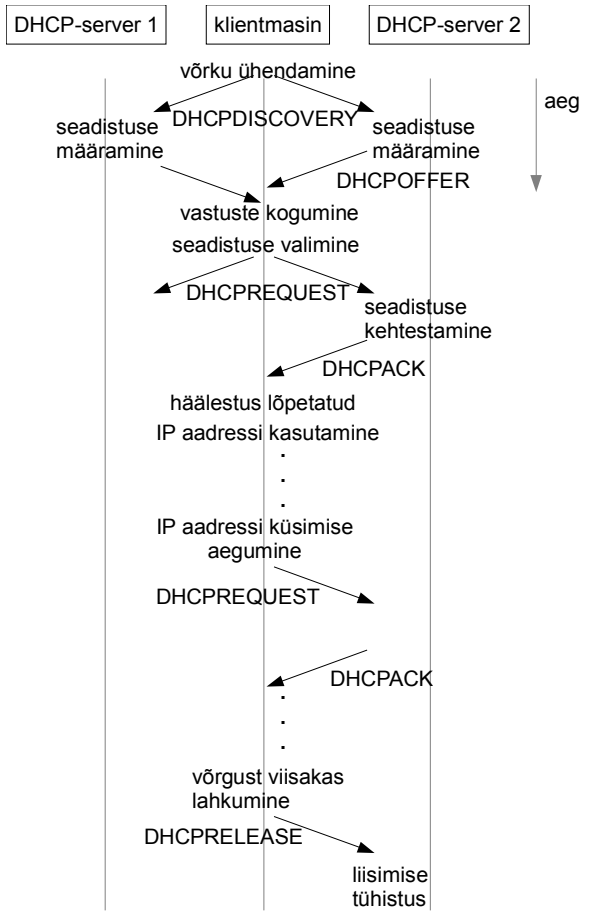
Kui näiteks pannakse käima arvuti, millel on seadistatud IP-aadressi saamine dünaamiliselt ehk DHCP abil (ta on DHCP klient), siis protsess on järgmine (vt. ka joonis 11.1):

1. DHCP klient saadab oma leviedastusdomeeni leviedastuskaadri, mida kutsutakse DHCPDISCOVER. IP ja MAC-aadressiks on leviedastusaadress. Lähte-IP-aadressiks on kõik nullid.
2. DHCP-server saab kätte DHCPDISCOVER kaadri (teised hostid viskavad selle minema) ja DHCP-server saadab selle tagasi kliendi kanalikihi üksikedastusaadressil. See teade sisaldab IP-aadressi ja valikute väljal teisi parameetreid (näiteks võrgumask, DNS-serveri aadress ja võrguvärava IP-aadress jne). DHCP-server proovib tavaliselt pingida (oleneb serveri implementatsioonist) vastavat IP-aadressi, et olla kindel aadressi unikaalsuses. Seda teadet nimetatakse DHCPOFFER.
3. DHCP klient saab kätte DHCP-serveri saadetud DHCPOFFER teate. DHCP klient vaatab, kas talle need parameetrid sobivad ja siis saadab DHCP klient teate leviedastusaadressil, millega ta taotleb endale pakutud parameetreid (võrgus võib olla veel teisigi DHCP-servereid, mis teaksid siis, milline pakkumine vastu võeti, tavaliselt on selleks esimene). Seda kaadrit nimetatakse DHCPREQUEST. Klient võib kaks esimest etappi vahele jätta, kui ta teab DHCP-serveri aadressi ja kui eelnevalt ei ole saanud DHCPNAK teadet.
4. DHCP-server saab DHCPREQUEST kätte. Kui kõik on korras, siis DHCP-server on välja liisinud pakutud IP-aadressi koos muude parameetritega. Selleks ta saadab kliendile kinnituse. Antud teadet nimetatakse DHCPACK. Kui esines mingeid vigu, siis saadetakse DHCPNAK teade.
5. Kui DHCP klient saab DHCPACK kaadri, siis klient seadistab oma võrguparameetrid vastavalt DHCP-serveri poolt pakutud parameetritega ning võib hakata neid koheselt kasutama.
6. Kui klient soovib lahkuda, siis võib ta enne liisimisaja täissaamist vabastada IP-aadressi. Selleks saadab ta DHCPRELEASE teate. Sellega vabastatakse IP-aadress, mille DHCP-server saab uuesti ringlusesse.

DHCP-server saadab vea korral kliendile DHCPNAK teate (näiteks IP liisimisaeg sai läbi, DHCP kliendi IP-aadress oli vigane, sest nihutati uude alamvõrku jms). Kui klient avastab, et talle liisitud IP-aadress on juba kasutusel, siis ta saadab serverile DHCPDECLINE teate. Kui kliendil on olemas IP-aadress (nt. käsitsi seadistatud), kuid soovib muid parameetreid, siis klient saadab DHCPINFORM teate.

DHCP-serveril on IP-aadresside jagamiseks seadistusega ette määratud IP-aadresside varu (*pool*), kust ta liisib IP-aadresse. Kui IP-aadress vabaneb (näiteks klient lahkus ja edastas DHCP-serverile DHCPRELEASE teate või liisitud IP-aadressi liisimisaeg sai täis), siis lisatakse ta tagasi IP-aadresside varusse, kust seda on võimalik järgmisele kliendile välja liisida.

Kui DHCP kliendil saab liisinguaeg täis, siis ta küsib omale liisimise pikendamist. Liisimisaja vaikeväärtus võib varieeruda umbes mõnest tunnist mõne päevani (oleneb implementatsioonist ja DHCP-serveri seadistusest). DHCP-serverit saab seadistada nii, et ta seoks mingi kindla MAC-aadressi alati mingi kindla IP-aadressiga (nagu BOOTP puhul kasutati). Selliselt saab fikseeritud masinale määrata alati sama IP-aadressi. Selline sidumine on kasulik, kui koduvõrgus on vajalik sülearvutile kindel IP-aadress, kuid samas satutakse ka teistesse võrkudesse (nt. Wi-Fi võrk). Nii ei ole vaja pidevalt seadistust muuta staatiliselt ja dünaamiliselt seadistatuks. Hallatavuse huvides võib kasutada keskset seadistuse määramist DHCP-serveris, millega võrguadministraator saab muuta keskselt masinate IP aadresse ja nende muid võrguseadistusi (näiteks, kui kolitakse või muudetakse võrgutopoloogiat). Samuti ei vajata füüsilist kohalolekut ja klientmasina töötamist.



Joonis 11.1: Tavapärane IP-aadressi liisimise protsess DHCP-serverilt. Joonisel on toodud näide, kus võrgus on kaks DHCP-serverit. Klient liisib IP-aadressi DHCP-server 2 käest.

Serverite, marsruuterite ja muude selliste seadmete puhul, mille puhul on oluline staatiline IP-aadress, pole üldjuhul otstarbekas kasutada DHCP abi, vaid teha seadistused käsitsi. DHCP on mõeldud eelkõige arvutitele, mille puhul võib aadress olla dünaamiliselt määratud.

Olenevalt serverist võib DHCP-ga edastatavaid parameetreid olla mitmeid. Näiteks võib seadistada DHCP-serverit andma ka muid põhiparameetreid: võrguväravat, võrgumaski, DNS-serveri IP-aadressi, lisaks ka näiteks WINS server(id), domeeni nime jne. Registreeritud parameetrid jm. registreerimiste info on kättesaadav aadressilt http://www.iana.org/assignments /bootp-dhcp-parameters.

Erinevalt DHCP kliendist vajab DHCP-server eelnevalt seadistamist. DHCP-serverile tuleb ette anda IP-aadresside vahemik, mida ta liisib (kodudes kasutatavatel marsruuteritel on tihti IP-aadresside varu fikseeritud näiteks võrgu 192.168.0.0/24 või 192.168.1.0/24 aadresside kasutamisele). Lisaks tuleb seadistada ka muud parameetrid (DNS-serverid, võrguväravad jms.).

DHCP-serveri poolt saab IP-aadressi määramine toimuda kolmel viisil:

- Dünaamiline määramine fikseeritud ajaks, mille möödudes peab liisimisaega pikendama. Dünaamiline määramine on kõige kasutatavam.
- Käsitsi määramine. DHCP-serveris on seadistatud IP-aadress mingile kindlale MAC-aadressile. Selle tulemusena antakse iga kord vastavale masinale sama IP-aadress. Kasutatav eelnevalt vaadeldud sülearvuti puhul koduvõrgus.
- Automaatne määramine. IP-aadress antakse alatiseks.

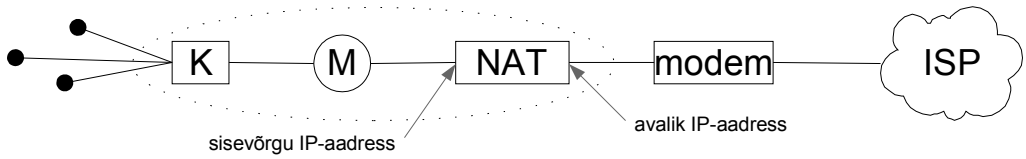
11.2 NAT

Tavapärane NAT (*Network Address Translation*) (RFC 3022) mõiste, mida tavaliselt tuntakse, koosneb detailsemalt NAT ja PAT (*port address translation*) osast, kus esimene on IP-aadressi tõlkija ja teine on pordi tõlkija. NAT ja PAT ühiselt vaatlemiseks kasutatakse ka terminit NAPT (*network address port translation*). Mõningad NAT-id võimaldavad ka eraldi võrguaadresside üks-ühele tõlkimist ja mitme IP-aadressi kasutamist väliste IP-aadressidena ning seda ka koos PAT funktsiooniga. Järgnevalt vaatleme NAT-i funktsionaalsust ühe välise IP-aadressiga ja porditõlkimisega. Antud peatükis käsitleme NAT-i marsruuteri lisafunktsionaalsusena, mitte eraldi seadmena.

IP-aadressiruum on ammendumas, mistõttu proovitakse IP-aadresse võimalikult optimaalselt kasutada ja soovijatele eraldada. Eelnevalt oleme vaadelnud VLSM ja CIDR tehnikaid, IP-aadresside kokkuhoiu suurendamiseks, kasutades efektiivsemalt IP-aadressiruumi. Järgnevalt vaatleme NAT-mise lahendust, mis on laialt rakendatav meetod IP-aadresside kokkuhoiuks.

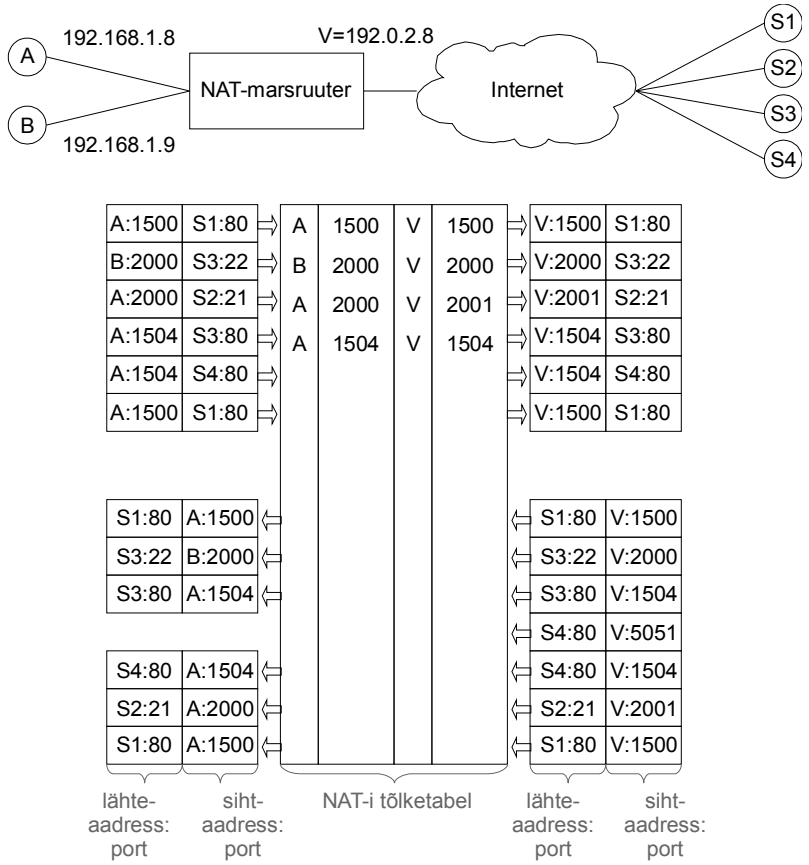
VLSM tehnikat kasutades proovisime vähendada IP-aadresside raiskamist. Paraku on tihti IP-aadresside vajadus tunduvalt suurem kui eraldatud IP-aadreside hulk. Näiteks eraldatakse kodukasutajatele ja väikestele kontoritele ainult üks IP-aadress, kuid kodus ja kontoris on enamasti rohkem kui üks arvuti, mis soovib Internetiga ühenduda ja seda samaaegselt. Siin ei saa VLSM enam aidata.

Eelnevalt vaatlesime jaotises "IP-aadress" (lk. 44), et IP-aadressiruumist on reserveeritud kolm erineva suurusega IP võrku privaatseks kasutamiseks (10.0.0.0/8, 172.16.0.0/12 ja 192.168.0.0/16). Neid aadresse saab kasutada sisevõrgus. OSI transpordikihi juures vaatlesime porte, mis on sisuliselt masinasisesed aadressid. Porte on võimalik ära kasutada nii, et nende abil oleks võimalik Internetiga ühendada ühte IP-aadressi, kasutades samaaegselt mitut masinat.



Joonis 11.2: Joonisel on erinevad funktsioonid lahku löödud. Tavaliselt on kodukasutuses kommutaator, marsruuter ja NAT-funktsionaalsus ühes seadmes, tihti ka näiteks ADSL-modem. Tavaliselt on selles masinas olemas ka DHCP-server, mis teenindab sisevõrku.

NAT-i tegeval seadmel on kaks poolt millest üks ühendatakse välisvõrguga ja omab välisvõrgu IP-aadressi (näiteks 123.45.67.89), teine pool ühendatakse sisevõrguga (vt. ka joonist 11.2). Sisevõrgus kasutatakse privaatseks kasutamiseks mõeldud aadressiruumi, kus NAT-marsruuter omab samuti vastavasse võrku kuuluvat IP-aadressi. NAT toimib tõlgina välisvõrgu ja sisevõrgu vahel, kasutades tõlkimiseks porte (TCP ja UDP omasid).



Joonis 11.3: Joonisel on näide, kus kaks sisevõrgus olevat masinat A (IP-aadressiga 192.168.1.8) ja B (IP-aadressiga 192.168.1.9) on NAT-tud võrgus, mille avalik IP-aadress on 192.0.2.8. Nad suhtlevad välisvõrgus olevate serveritega S1, S2, S3, S4. Tähistame joonisel IP-aadresse lühiduse mõttes tähtedega (nt. "A" tähistab IP-aadressi "192.168.1.8", "V" IP-aadressi "192.0.2.8", serverite IP-aadresse ei ole üldsegi välja toonud ja tähistame neid lihtsalt vastavalt S1, S2, S3 ja S4.). Pildi vasakpoolses osas on paketid, mida sisevõrgumasinad A ja B välja saadavad (joonisel kokku kuus paketti), mille tulemusena lisatakse NAT-marsruuterisse vastav kirje (joonise keskel), mille veerud on järgmised: sisevõrgumasina IP-aadress, sisevõrgumasina pordinumber, avaliku võrguliidese IP-aadress ja avaliku võrguliidese pordinumber. Lihtsuse mõttes on vastuspaketid joonise alumises osas. Ühel juhul on pakett pordile 5051, mille kohta NAT-i tõlketabelis ei ole sobivat kirjet, mille tulemusena visatakse ta minema. Kolmanda paketi saatmisel oli port 2000 juba masina B poolt hõivatud, mille tõttu võetakse kasutusele mingi muu vaba port.

Lihtsa NAT-i korral saab ühendust luua ainult sisevõrgust. Ühendus luuakse, kui NAT sisemisest poolest saadetakse pakett välise poole IP-aadressile (suvaline IP-aadress Internetis, näiteks 97.53.186.42). NAT asendab väljuva paketi lähteaadressi (näiteks 192.168.1.25) avaliku IP-aadressiga (seega asendatakse 192.168.1.25 aadressiga 192.0.2.8). NAT peab arvet välise IP-

aadressiga seotud portide kasutamise kohta. Uue väljuva ühenduse paketi puhul võtab NAT esimese vaba välise liidese pordinumbri (nt. 1357) ja asendab segmendi lähtepordi numbri (nt. 2345). Sihtaadress ja sihtport jäävad samaks. NAT lisab oma tõlketabelisse tehtud tõlke ehk asenduse (192.168.1.25:2345 asendatakse 192.0.2.8:1357). Kui NAT-i välisliides saab paketi, mis sisaldab välisliideseks tõlget (nt. 192.0.2.0:1357), siis NAT asendab paketi sihtaadressi ja pordi sisevõrgu aadressiga, mis sisaldus vastava kirje juures (seega 192.0.2.8:1357 asendatakse 192.168.1.25:2345) ja saadetakse sisevõrku.

Kui sama sisevõrgumasin saadab sama lähteaadressi ja -pordiga paketi, siis kasutatakse olemasolevat NAT-i tõlkekirjet (olenevalt NAT-i realiseeritusest ei pruugi sihtaadress ja -port olla olulised, et nende üle eraldi arvet pidada). NAT tõlketabel on iga transpordikihi protokolli (näiteks TCP, UDP) kohta eraldi.

Iga NAT tõlketabeli kirje puhul peetakse arvet tõlkekirje viimase kasutamise kohta (teoreetiliselt on tõlkeid võimalik teha kuni 65536 tükki, kuid tavaliselt NAT-i tegevad seadmed võimaldavad kuni mõnituhat tõlget/ühendust). Kui tõlget ei ole kasutatud määratud aja jooksul (oleneb implementatsioonist ja seadistusest), siis loetakse kirje aegunuks ja kustutatakse tõlketabelist. Kui välishost peaks vastama väga pika viivitusega, siis ei pruugi vastus enam jõuda sisevõrgu masinani. Kui NAT tõlketabel on täis saanud, siis ei ole võimalik luua mitte ühtegi uut ühendust, kuni mõni eksisteeriv ühendus aegub. Kasutajate jaoks paistavad ühendused välisvõrku mitte töötavat. NAT tõlketabel võib täis saada, kui sisevõrgus kasutatakse programme, mis teevad väga palju ühendusi (nt. failivahetusprogrammid).

NAT-i abil saavad ainult ühte avalikku IP-aadressi omades mitu arvutit samaaegselt Internetti kasutada. NAT-il on mitmeid plusse ja ka miinuseid:

+ Hoiab kokku suures koguses IP-aadresse. Saab väiksema arvu IP-aadresside taha panna rohkem masinaid. Sageli on kasutada ainult üks IP-aadress.
+ Kui toimub Interneti aadressi vahetamine (näiteks Interneti teenusepakkuja vahetamine), siis NAT-tud võrgu puhul ei ole vaja hakata muutma sisevõrgu arvutite IP-seadeid. Sisevõrgu jaoks jääb kõik samaks. Muutus toimub ainult NAT-marsruuteris, mis hakkab kasutama tõlkimiseks uut avalikku IP-aadressi.
+ Lisab võrguturvalisust. Sisevõrgu aadressid ja topoloogia ei ole väljapoole nähtavad. Ilma lisaseadistusteta ei ole väljastpoolt võimalik sisevõrgu arvutite poole ühendust luua.
− Piirab ühenduste arvu, mida saab teha sisevõrgust välisvõrku.
− Lõhub TCP mudeli, kus ühenduse olekuid peavad teadma ainult otspunktid.
− Kaob otspunktidevaheline jälitatavus (raskendatud suhtlevate masinate kindlakstegemine).
− Mõned programmid ning protokollid ei saa NAT-tud võrguväliste hostidega tööd teostada, sest NAT peidab lõpp-punktide aadressid.
− Suurendab viivitust, sest lisatööd tehakse IP ja vajadusel pordi asendamisel (transpordi kihis). Esimese paketi saatmine on edasistest aeglasem (kui NAT-i tabelis pole vastavat kirjet).

NAT-marsruuterit nimetatakse ka loomulikuks tulemüüriks, sest NAT suunab edasi ainult need paketid, millele tal on tõlketabelis olemas kirje. Teised paketid visatakse ära. Samas on tihti vajadus panna sisevõrgus püsti mingi server, kuid see ainuke IP-aadress on juba NAT-marsruuteri poolt kasutusel. Enamus NAT-marsruuteritel on võimalus, mida nimetatakse virtuaalseks serveriks või pordi edastuseks, mille puhul määratakse NAT-marsruuteri välimine port staatiliselt edasi suunama mingile fikseeritud sisevõrgu IP-aadressile. Kui sellisel juhul saab NAT-i väline liides paketi, mille TCP pordi number on 80, siis see suunatakse edasi fikseeritud sisevõrgu IP-aadressile. Seda võib vaadata ka kui staatilist NAT tabeli tõlget, mis ei aegu.

Vaikimisi visatakse ära kõik paketid, millele NAT-i tõlketabelis kirjet ei leita. Samas on paljude NAT-marsruuterite puhul võimalik need paketid suunata edasi võrguadministraatori poolt

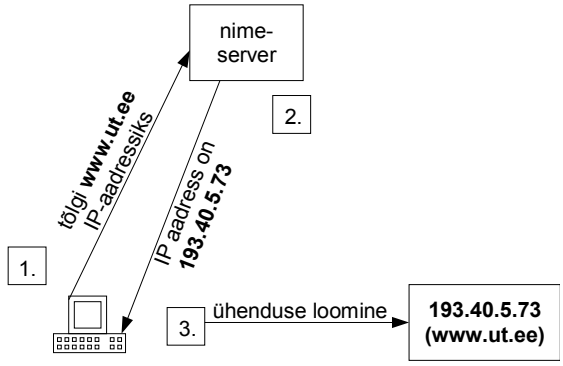
määratud sisevõrgumasinale. Seda masinat nimetatakse demilitariseeritud tsooniks, ka *DMZ (demilitarized zone)*. Täpsem nimetus oleks demilitariseeritud host. Demilitariseeritud hostis asuvad tavaliselt avalikud serverid (veebi-, e-maili-, FTP jm serverid). Üks võimalus NAT-st tingitud probleemide lahendamiseks on kasutada UPnP-d, mida vaatleme lähemalt peatükis "UPnP ja NAT" (lk. 149).

11.3 DNS

Seni oleme vaadelnud võrku IP-aadressi alusel. Paraku ei jää inimesele numbrid hästi meelde, kuigi need võivad olla hästi grupeeritud. Palju paremini jäävad meelde märksõnad, mis midagi ka tähendavad või on tuletatavad. Seega oleks hea, kui saaks numbrijada asemel kasutada palju rohkemütlevamaid nimesid. Selle probleemi lahenduseks on tänapäeval kasutusel protokoll DNS (*Domain Name System*) (RFC 1035), mis avaldati esmakordselt 1980. aastate keskel. DNS-i peamine eesmärk on seada IP-aadress domeeninimega vastavusse. Tänapäeval on DNS üks Interneti võtmeprotokollidest.

Tänu sellele, et inimesed kasutavad serverite poole pöördumiseks domeeninimesid ja domeeninimi tõlgitakse DNS-serverite poolt vajalikuks IP-aadressiks, on kasutajatele märkamatult võimalik muuta serveri asukohta, näiteks kolitakse teise ISP-i teenust kasutama või hoopis teise riiki. DNS teeb nimed sõltumatuks IP-aadressidest.

DNS-i kasutame näiteks siis, kui külastame veebibrauseriga mingit veebilehekülge, näiteks "http://www.ut.ee". Sel juhul on "www.ut.ee" vaja tõlkida IP-aadressiks ("http://" määrab kasutatava protokolli ja sellega ka vaikimisi pordi, millega vastava IP-aadressiga hosti poole pöördutakse). Selleks küsib arvuti DNS-serveri käest "www.ut.ee"-le vastavat IP-aadressi. Alles peale seda, kui domeeninimele "www.ut.ee" on saadud vastuseks IP-aadress, saab pöörduda soovitud serveri poole (ehk www.ut.ee poole). Vaata ka joonist 11.4.



Joonis 11.4: DNS-i põhiline kasutusjuhtum. Klient soovib pöörduda hosti "www.ut.ee" poole, milleks laseb tõlkida domeeninime IP-aadressiks ja siis saab pöörduda soovitud serveri poole.

DNS koosneb kolmest komponendist:

- Nimeruum ja ressursikirjed. Hierarhilisele nimeruumile on määratud mingid ressursikirjed. Päringuga nimeruumile saadakse tagasi soovitud tüüpi informatsioon vastava nimeruumi kirje kohta.
- Nimeserverid. Internetis olevad serverid, mis hoiavad hajutatult nimeruumi andmeid ja vastavad DNS klientide päringutele.
- Nimelahendajad (*name resolver*). Hostid, mis küsivad informatsiooni nimeserveritest. Näiteks tavalised arvutid, mida igapäevaselt oma laual kasutame, on nimelahendajad. Nimelahendaja on klient-server mudelis kliendi pool ja nimeserver serveri pool.

Nimeruum on Interneti mõistes sisuliselt üle maailma hajutatud andmebaas. See andmebaas sisaldab erinevaid ressursikirjeid (*resource records*). Nimeruum on jagatud väiksemateks osadeks, mida nimetatakse tsoonideks, mis asuvad erinevates nimeserverites (ühes nimeserveris võib asuda ka mitu tsooni).

Ressursikirje parameetrid on järgmised:

- Nimi (*name*). Sõlm, millele vastav ressursikirje kuulub. Kuni 255 baiti.
- Tüüp (*type*). Määratleb ressursikirje tüübi. 2 baiti.
- Klass (*class*). Näitab, mis protokollistikule antud ressursikirje käib. Tähistatakse kahe ladina tähega (kaks baiti). Mõningaid näiteid:
  - ◇ "IN" on Interneti süsteemi protokollid (tavapäraselt kasutatav).
  - ◇ "CH" on Chaos süsteemide protokollid.
- Aega elada (TTL (*time to live*)). Näitab, mitu sekundit on ressursikirje eluaeg. Kasutatakse põhiliselt puhverdatud ressursikirjete puhul. Kui see on null, siis ressursikirjet ei lubata puhverdada. 4 baiti.
- RDLENGTH. Näitab RDATA välja pikkust baitides. 2 baiti.
- RDATA. Ressursikirje põhiinfo, mis sõltub ressursikirje tüübist ja klassist.

Ressursikirje põhilisi tüüpe "IN" klassis:

- A (*address*) – hosti IPv4-aadress. RDATA väli sisaldab hosti IPv4-aadressi (32 bitti).
- AAAA – hosti IPv6-aadress.
- PTR (*pointer*) – Näitab IP-aadressile vastava arvuti nime (pöördteisenduse jaoks) või viitab muule kohale informatsiooni kohta.
- CNAME (*canonical name*) – kanooniline nimi, millega seatakse ühele domeeninimele vastavusse mingi teine domeeninimi, millega mitu domeeninime automaatselt viitavad ühele IP-aadressile. Kanoonilisi nimesid nimetatakse ka aliasteks. Näiteks on domeeninimi "neti.neti.ee" aliaseks domeeninimele "www.neti.ee". RDATA väli sisaldab domeeninime.
- MX (*mail exchange*) – domeeni e-posti vastuvõtja. RDATA sisaldab e-mailivahetushosti nime.
- NS (*name server*) – domeeni pädev nimeserver. RDATA väli sisaldab antud domeeni pädevat nimeserveri hosti nime.
- SOA (*start of authority*) – viitab andmete allikale ehk vastava tsooni primaarsele nimeserverile.
- TXT (*text*) – mingi tekst.
- SRV (*service*) – teenuse asukoha kirje.
- HINFO (*host information*) – hosti info (operatsioonisüsteemide tähistused on aadressil http://www.iana.org/assignments/operating-system-names).
- RP (*responsible person*) – vastutav isik.

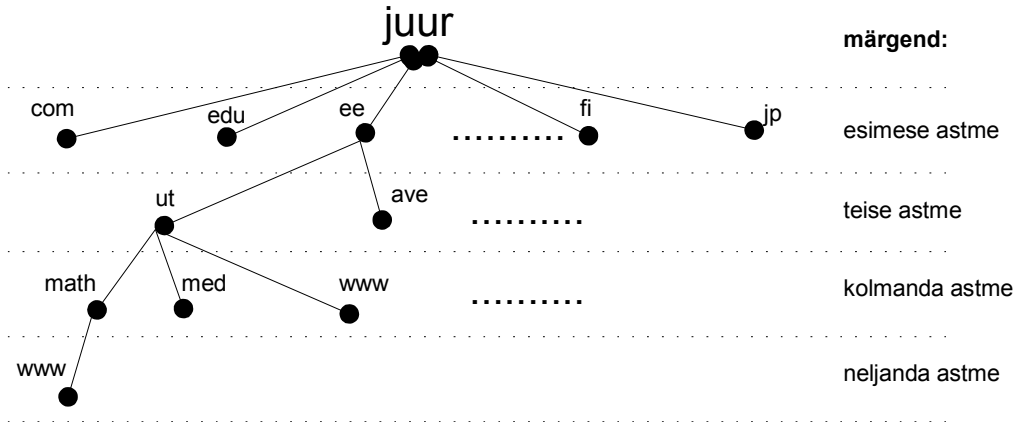
Ametlik nimekiri ressursikirjete klasside ja tüüpide registreeringutest on aadressil http://www.iana.org/assignments/dns-parameters.

Domeeninimeruum on puukujuline hierarhiline süsteem, kus tippe tähistatakse märgenditega (*label*). Joonisel 11.5 on kujutatud üks osa domeeninimeruumist. Iga märgend saab koosneda maksimaalselt 63 tähemärgist. Domeeninime enda pikkus saab kokku olla maksimaalselt 255 märki. Märgend võib koosneda tähtedest, numbritest ja sidekriipsust (sidekriips ei saa olla esimene märk). Internetis on ka laiendusi mitte ladina tähtedele, kuid need ei ole suures osas toetatud.

Iga tipu kohta on defineeritud ressursikirje. DNS põhiline kasutus on domeeninimele vastava IP-aadressi andmine, kuid peale selle võib ressursikirje sisaldada ka muud informatsiooni.

Domeeninimi kirjutatakse kokku järjekorras, kus erineva astme märgendid eraldatakse punktiga ja esimese astme märgend on kõige viimane ning kõige madalam domeeninime märgend on esimene. Madalama astme domeeninimi sisaldab ülemise astme märgendeid alates juurest. Seega joonise peal on vähemalt teise astme domeenideks "ut.ee", "math.ut.ee", "med.ut.ee",

"www.ut.ee", "www.math.ut.ee", "ave.ee", kus esimene ja viimane on teise astme domeeninimed, viies on neljanda astme domeeninimi ja ülejäänud on kolmanda astme domeeninimed. Domeeninime lõpus asub punkt, mille võib ka ära jätta.



Joonis 11.5: DNS domeeninimi on hierarhiline süsteem.

Kõige ülemise astme domeeni nimetatakse esimese taseme domeeniks. Esimese taseme domeenid jagunevad kahte ossa: riigi ja tegevusala järgi. Eesti puhul on esimese taseme domeeniks "ee", Soomel "fi", Etioopial "et", Taanil "dk" jne. Märgitakse lühendiga ccTLD (*country code top level domain*). Täisnimekiri riikide esimese taseme domeenidest asub aadressil http://www.iana.org/root-whois/index.html. Tegevusala järgi on esimese taseme domeenideks näiteks kommertsettevõtetel ".com" (tihti ka mitte ainult kommertsfirmad), haridusasutustel ".edu", militaarasutustel ".mil", valitsusasutustel ".gov", muuseumitel ".museum", üksikisikutel ".name" jne. Märgitakse lühendiga gTLD (*generic top level domain*). Täisnimekiri tegevusala esimese taseme domeenidest asub aadressil http://www.icann.org/registries/listing.html (vaata ka http://www.iana.org/gtld/gtld.htm). Eraldi on veel spetsiaalkasutusega ".arpa", mille registreerimised on aadressil http://www.iana.org/arpa-dom. Näiteks "in-addr.arpa" kasutatakse pöördteisenduste jaoks (IP-aadressile vastava domeeninime saamiseks).

Alates teise taseme domeeninimest registreeritakse domeeninimesid soovijatele Internetis ehk iga täisdomeen on vähemalt teise astme domeen, kuid võib olla ka madalama astme domeen (kolmanda, neljanda, ...). Teise taseme domeeninimed on näiteks "naide.ee", "ave.ee", "ut.ee". Kolmanda taseme domeeninimi on seega mittekohustuslik. Kolmanda astme domeeninime puhul määrab asutus (mille nimele teise astme domeeninimi registreeriti) selle, missugus(t)ele IP-aadressi(de)le see viitab. Kõige sagedamini pannakse kolmanda astme domeeninime kolmandaks märgendiks "www", mis viitab veebiserveri IP-aadressile, "ftp" viitab FTP serveri IP-aadressile, "mail" viitab SMTP, POP3, IMAP serverite IP-aadressile jne. (see ei ole kohustuslik). Asutus võib omakorda veel neljanda astme domeeninimesid teha, näiteks on osakondadel kolmanda astme domeeninimi, mida nad saavad omakorda alamastmeteks teha (näiteks eraldi veebiserveri, e-mailiserverite jm. jaoks). Domeeninimed ei ole tõstutundlikud ("www.ave.ee" on sama, mis "WwW.Ave.Ee").

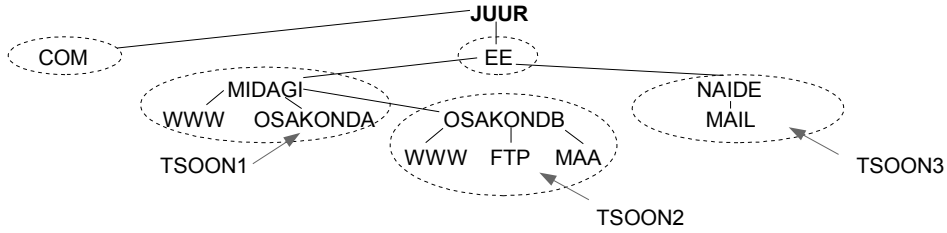
Domeeninimede juurorganisatsiooniks on ICANN (*Internet Corporation for Assigned Names and Numbers*), kes volitab esimese astme domeeninimede registripidajaid. Eesti puhul on esimese taseme domeen "ee" registreeritud Eesti esindajale, kelleks on akadeemik Endel Lippmaa. ICANN poolt volitatud esimese taseme domeenide registraatorite nimekiri asub aadressil: http://www.internic.net/regist.html.

Näiteks "ut.ee" on Tartu Ülikoolile registreeritud domeeninimi. Asutus, mis omab mingit domeeni, võib selle jagada omakorda alamdomeenideks. Näiteks "www.ut.ee" viitab veebiserveri IP-aadressile. Omakorda võivad ka mingite osakondade jaoks olla omad alamdomeenid, näiteks TÜ Matemaatika-informaatikateaduskonna domeen on "math.ut.ee". Sellel omakorda võib olla alamdomeene. Näiteks "www.math.ut.ee" viitab eelpool mainitud teaduskonna veebiserverile. Näiteks domeeninimi "raud.ut.ee" viitab IP-aadressidele 193.40.5.117 ja 193.40.15.176.

Osad esimese taseme domeeninimed on spetsiaalkasutusega, milledeks on näidetes kasutamiseks ".test" ".example", ".invalid". Ülemtaseme domeen ".localhost" kasutatakse viitamaks tagasisidestusaadressile (127.0.0.1).

DNS jagunemine serverite vahel

DNS andmebaas on jagatud sektsioonideks, mida nimetatakse tsoonideks (*zone*). Selliselt jagatakse hästi suur DNS andmebaas väiksemateks osadeks, mis on eraldi administreeritavad. Tsooni suurusele ei ole seatud piiranguid. Tsoon ei pea sisaldama kõiki oma domeeni alamdomeene, vaid need võivad olla eraldi tsoonis, mida nimetatakse alamtsooniks. Üks domeen võib olla jagatud alamdomeenideks, sest domeen võib olla väga suur. Samuti on vähem liiklust domeenide replikeerimisel klientide päringute puhul. Joonisel 11.6 on näide domeeninimede jaotumisest erinevate tsoonide vahel.



Joonis 11.6: Näidis tsoonidest. Juureks on ICANN.

Omavahel mitte alam- ja ülemdomeeni suhtes olevad domeenid on üksteisest sõltumatud. Joonisel näiteks domeenid "www.midagi.ee", "www.osakondb.midagi.ee", "ftp.osakondb.midagi.ee" ja "mail.naide.ee" on kõik üksteisest sõltumatud domeenid.

Nimeserverid

Nimeserverid jaotatakse oma funktsiooni poolest mitmesse erinevasse tüüpi.

Juurnimeserver (*root name server)*. Need on nimeruumi juurikas asuvad teistest nimeserveritest natuke erinevad nimeserverid, mis on DNS selgrooks. Nemad annavad viite vastavat domeeninime haldavale nimeserverile.

Tsooni andmeid võivad serveerida mitu replitseeritud nimeserverit, milledest üks on primaarne ja teised, millesse replitseeritakse, on sekundaarsed nimeserverid antud tsooni jaoks.

Primaarne nimeserver hoiab algset tsoonifaili koopiat. Muudatused antud tsoonis tehakse ainult primaarses nimeserveris. Iga tsoon omab oma primaarset nimeserverit.

Sekundaarsed nimeserverid saavad tsoonifailide koopiad muust nimeserverist. Nad ei saa teha mingeid muudatusi. Sekundaarsete nimeserverite eesmärgiks tsoonis on dubleerida primaarset nimeserverit, juhuks kui primaarse nimeserveriga peaks midagi juhtuma (DNS disainimisel on pandud rõhku dubleeritusele). Teisalt aitab sekundaarsete nimeserverite kasutuselevõtt vähendada võrguliiklust ja primaarse nimeserveri koormust. Sekundaarsete nimeserverite paigutamisel on mõistlik jälgida, et võimalikult paljud päringud jääksid lokaalseks, samuti ka haju-

tatuks, et ühendusprobleemide korral vähemalt mõni nimeserver oleks kättesaadav.

Nimeserver, kust sekundaarsed nimeserverid saavad tsoonifaili, nimetatakse **ülemnimeserveriks** (*master name server*). Sekundaarsed nimeserverid on sel juhul seadistatud ülemnimeserverit kasutama (IP-aadressi järgi). Ülemnimeserver ise võib olla primaarne, kuid võib ka olla sekundaarne. Kui ülemnimeserver on primaarne, siis sekundaarne nimeserver saab oma tsoonifailid otse allikast. Kui ülemnimeserver on ise sekundaarne antud tsooni jaoks, siis see tähendab, et saadakse uuendusi läbi ühe või rohkema vahendaja. Selle tõttu võtab primaarsest nimeserverist kaugemal olevatesse nimeserveritesse muudatuste jõudmine rohkem aega. Ainuke liiklus primaarse ja sekundaarsete nimeserverite vahel on primaarse nimeserveri saadetava tsooniinformatsiooni koopia transportimine sekundaarsetesse nimeserveritesse.

Nimeserver saab hoida mitut tsooni. Nimeserver saab olla mõne tsooni jaoks primaarne ja samaaegselt mingi teise tsooni jaoks sekundaarne.

Kui DNS-server ei serveeri ühtegi tsooni (primaarselt või sekundaarselt), siis teda tuntakse **puhvermälu nimeserverina** (*caching-only*). Selline server ainult vahendab päringuid ja hoiab alles saadud ressursikirjed (järgmine kord sama asja küsides saab neid ära kasutada ega pea uuesti pöörduma teiste nimeserverite poole). Vahendatud ressursikirjeid säilitatakse parameetri "kaua elada" poolt määratud sekundite jooksul. Üldjuhul puhverdavad ka teised nimeserverid päringuvastuseid, kui need ei ole nende endi tsooniinfost pärit.

Asutuse sisevõrgus võib olla mitu DNS-serverit, kuid tihti ei taheta neid kõiki hakata eraldi turvama. Seetõttu on võimalik kasutada spetsiaalset vahendajat, mida nimetatakse edastajaks (*forwarder*). Edastaja vahendab kõiki väljapoole võrku saadetavaid päringuid, seega on DNS-serverite jaoks tegemist sisuliselt võrguväravaga (*gateway*). Selleks tuleb võrgusisesed nimeserverid seadistada edastajat kasutama. Edastaja puhverdab ka erinevate nimeserverite päringuid, seega võidakse kasutada vähem välisühendust, mis on oluline, kui välisühendus on aeglane. DNS-servereid, mis kasutavad edastajaid nimetatakse **orinimeserveriteks** (*slave name server*).

Ressursikirjeid, mida hoitakse DNS primaarses või sekundaarses serveris, nimetatakse **pädevateks andmeteks**. Kirjed, mis on vahendamise teel saadud (puhvermälus), nimetatakse **mittepädevateks andmeteks**. Seega primaarsete ja sekundaarsete serverite tsoonikoopiates hoitavaid andmeid nimetatakse pädevateks andmeteks. Teiste nimeserverite puhul on vastavad ressursikirjed puhverdatud ja neid andmeid nimetatakse **mittepädevateks**.

DNS puhul on vajalik, et iga nimeserver peab teadma vähemalt ühe juurnimeserveri aadressi. Nimelahendaja peab omakorda teadma vähemalt ühe nimeserveri aadressi, soovitavalt mitut.

Päringud

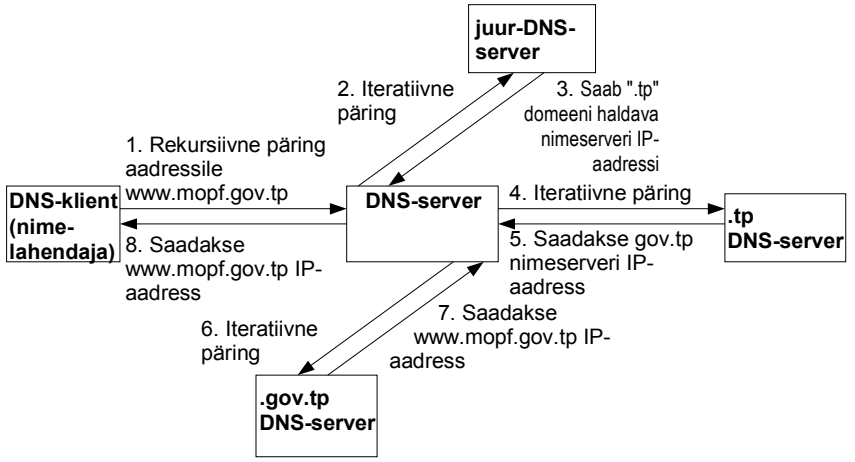
DNS-serverile tehtavaid päringuid on kahte liiki: **rekursiivne** ja **iteratiivne**. Iteratiivse päringu puhul küsitakse nimeserverilt täpseimat informatsiooni, mida nimeserver teab oma tsooniinfo ja puhvermälu põhjal. Kui nimeserver ei tea küsitule täpset vastust (näiteks IP-aadressi), siis annab ta täpseima viite nimeserverile, mida ta teab. Pärija teeb järgmisena päringu viidatud nimeserverile. Protsess toimub seni, kuni jõutakse nimeserverini, mis omab soovitud informatsiooni või saadakse negatiivne vastus (näiteks, kui tehti päring mitteeksisteeriva domeeninime kohta). Iteratiivne päring tehakse sihtdomeenile lähimale domeeniserverile (nimeruumi puu seisukohalt), mida teatakse. Kui ei teata isegi ülemise taseme domeeni, siis alustatakse iteratiivse päringuga juurserverile, mille IP-aadressi nimeserverid eelnevalt teavad. Iteratiivset päringut kasutavad nimeserverid teiste nimeserverite poole pöördumisel.

Rekursiivse päringu puhul peab DNS-server tagastama küsitud info küsitud domeeni kohta (või teatama ebaõnnestumisest). DNS-server peab vajaduse korral tegema vajalikud pöördumised teiste nimeserverite poole. Tavaliselt teevad rekursiivseid päringuid nimelahendajad nimeserveritele ja nimeserverid edastajatele. Vaata joonisel 11.7 toodud näidet domeeninime lahendamisest.

Nimeserveri kirjetele määratakse nende kehtivusaeg, mille järel muutuvad kopeeritud ja puh-

verdatud kirjed aegunuks. Sekundaarne nimeserver värskendab oma tsooni andmeid ülemnimeserveri käest. Aegunud mittepädevad andmed eemaldatakse puhvermälust. Iga päring, millele nimeserver annab vastuse, sisaldab ka kehtivusaega. Kehtivusajaga arvestab ka nimelahendaja.

Üldjuhul on operatsioonisüsteemides säilinud enne DNS-i kasutusel olnud hostifailid, mis võivad sisaldada nimelahendusi. Windowsi puhul asub failis "%windir%\System32\Drivers\etc\hosts". UNIX-il baseeruvate puhul failis "/etc/hosts". Hostifail on esimene koht, mida vaadatakse, seejärel pöördutakse kohaliku puhvermälu ja edasi nimeserveri poole.



Joonis 11.7: Klient tahab tõlkida domeeninime "www.mopf.gov.tp" IP-aadressiks. Klient teeb rekursiivse päringu, nimeserver teeb iteratiivseid päringuid, kuni jõutakse nimi täielikult ära lahendada. Joonisel domeeni gov.tp nimeserver teadis IP-aadressi.

Toome ühe näite joonisel 11.7. Soovigu kasutaja vaadata veebilehte, mis asub aadressil "http://www.mopf.gov.tp". Oletame, et kohalik nimeserver ei tea isegi domeeni "tp" (Ida-Timorile kuuluv esimese taseme domeeninimi), seega pöördub ta juurnimeserveri poole, sealt saab ta viite "tp" nimeregistripidaja juurde. Kui ta teab kohe soovitud domeeninime, siis annab ta kohe vastuse, kui ei, siis saab ta sealt "gov.tp" DNS-serveri viida ja sealt võidakse edasi suunata "mopf.gov.tp" DNS-serveri poole, kust ta saab lõpuks kätte soovitud IP-aadressi. Joonisel oleva näite kohaselt teab "gov.tp" nimeserver infot soovitud domeeninime kohta ("gov.tp" nimeserver võib olla domeeninimele "www.mopf.gov.tp" pädevaks nimeserveriks või on tegu puhverdatud andmetega mingist eelnevast päringust).

DNS on rakenduskihi protokoll. Ta töötab nii TCP kui ka UDP transpordikihi protokollidel, kus mõlemate puhul on pordiks 53. Päringud DNS-serverile esitatakse eelistatult UDP protokolli kasutades. Kui päringu vastuses on andmeid rohkem kui 512 baiti, siis antakse sellest vastuses teada vastava lipuga ja päringut korratakse, kasutades TCP protokolli. Primaarse ja sekundaarse nimeserveri vahel toimub tsooni andmete kopeerimine üle TCP protokolli.

Tavalised tööarvutid on reeglina nimelahendajad. Programmid (näiteks veebibrauser, SSH klient) kasutavad tavaliselt operatsioonisüsteemi pakutavaid nimelahendusvahendeid.

Kõikide DNS-teadete formaat on samasugune (päringud, vastused, veateated, serverite vahelised ressursikirjete ülekanded), mis jaguneb järgmisteks sektsioonideks:

- Päis (HEADER).
- Küsimus (QUESTION). Küsimus nimeserverile.
- Vastus (ANSWER). Ressursikirjed vastuseks päringule nimeserverilt.
- Pädevus (AUTHORITY). Ressursikirjed viitavad pädevale nimeserverile.
- Lisa (ADDITIONAL). Ressursikirjed sisaldavad lisainformatsiooni.

Päise sektsioon on alati kohustuslik. Päise formaat on päringul ja vastusel samasugune. Päise väljad on järgmised:

- ID. Päringu identifikaator, mille genereerib päringu tegija ja päringule vastaja kopeerib vastusesse päringu identifikaatori. 16 bitti.
- Erinevad juhtbitid:
  - QR. Kui lipp on maas, siis on tegemist päringuga, kui püsti, siis vastusega.
  - Operatsiooni kood ehk Opcode. Päringu tüüp, mis on vastuses ka päringuga samasugune. 4 bitti. Erinevad päringud välja väärtuste puhul:
    - 0: standardpäring (QUERY).
    - 1: pööratud päring (IQUERY).
    - 2: seisundipäring (STATUS).
    - 4: teadustamispäring (NOTIFY).
    - 5: värskendamispäring (UPDATE).
  - AA (*Authorative Answer*). Kui lipp püsti, siis on tegemist pädeva vastusega, vastasel korral mittepädeva vastusega.
  - TC (*Truncated Response*). Lipp püsti, kui vastus oli pikem kui 512 baiti ja seda kärbiti. Kui klient tahab kogu päringu vastust, siis peab ta tegema päringu üle TCP. Kui lipp maas, siis vastus ei olnud kärbitud ja ei olnud suurem kui 512 baiti. Lipp on vajalik UDP puhul.
  - RD (*Recursion Desired*). Kui lipp on püsti, siis on tegu rekursiivse päringuga, vastasel korral iteratiivsega. Vastusesse kopeeritakse lipu väärtus.
  - RA (*Recursion Available*). Omab tähtsust vastuses. Kui lipp on püsti, siis nimeserver toetab rekursiivset päringut, vastasel juhul mitte.
  - DO (*DNSSEC OK* (*DNS Security Extensions okey*)). Lipu paneb päringutel püsti DNSSEC toetusega nimelahendaja. Kui nimeserver saab päringu, millel on DO lipp maas, siis eemaldatakse kõik DNSSEC ressursikirjed (välja arvatud juhul, kui konkreetselt küsitakse vastavat ressursikirjet). Järgmised kaks lippu on ka seotud DNSSEC DNS laiendusega (DNSSEC vaatleme jaotises "DNSSEC" (lk. 84)).
  - CD (*checking disabled*). Kasutatakse DNSSEC toetusega nimelahendaja ja nimeserveri vahel, kui nimelahendaja soovib blokeerida signatuuri valideerimist (nimeserver kopeerib vastuses lipu väärtuse).
  - AD (*authentic data*). Lipp pannakse püsti, kui kõik vastuse ja pädevuse ressursikirjed on autentsed (DNSSEC korral).
  - Tulemuse kood ehk Rcode (*result code*). Vastuse tulemuskood. 4 bitti. Koodid on järgmised:
    - 0: Viga ei esinenud (Noerror).
    - 1: Formaadiviga. Server ei osanud päringut käsitleda (FormErr).
    - 2: Server ei saa vastata (ServFail).
    - 3: Domeeninime ei eksisteeri. Seda vastust võivad väljastada ainult pädevad nimeserverid (NXDomain).
    - 4: Päringu tüüpi ei toetata (NotImplemented).
    - 5: Nimeserver keeldub vastamast, näiteks turvakaalutlustel (Query Refused).
- QDCOUNT. Näitab, mitu ressursikirjet on QUESTION sektsioonis.
- ANCOUNT. Näitab, mitu ressursikirjet on ANSWER sektsioonis.
- NSCOUNT. Näitab, mitu ressursikirjet on AUTHORITY sektsioonis.
- ARCOUNT. Näitab, mitu ressursikirjet on ADDITIONAL sektsioonis.

Küsimuse (QUESTION) sektsioon koosneb kolmest väljast:

- QNAME. Sisaldab domeeninime.
- QTYPE. Määratakse, millist tüüpi ressursikirjet soovitakse vastuseks (näiteks A, NS, MX).

- QCLASS. Määrab päringu klassi (Internetis "IN").

Tavaliselt koosneb päring ainult ühest küsimussektsioonist (siis QDCOUNT=1). Teiste kolme sektsiooni formaat on sama:

- NAME. Domeeni nimi, sama, mis QNAME küsimussektsioonis.
- TYPE. Kirje tüüp.
- CLASS. Kirje klass.
- TTL. Ressursikirje säilivusaeg (puhvermälus hoidmise aeg).
- RDLENGTH. RDATA sektsiooni pikkus.
- RDATA. Transporditav ressursikirje.

DNS väljatöötamisel on arvestatud, et domeeninimede muutused pole sagedased. Kui domeeninime on vaja vahetada, siis peab arvestama, et domeeninimede puhvermälud säilivad "aega elada" väärtuse jagu. Kui peaks tekkima vajadus muuta domeeninime informatsiooni, näiteks domeeninimele pannakse vastavusse teine IP-aadress, siis on kasulik vähendada enne muutmist vastava ressursikirje "aega elada" väärtust, millega tagatakse, et aegunud informatsioon on võrgus lühemat aega. Peale IP-aadressi vahetust võib "aega elada" väärtuse jälle suuremaks panna.

Pöördteisendus

DNS puhul on pöördteisenduste tarvis kasutusel spetsiaalne domeen "in-addr.arpa". Pöördteisenduse kirjed esitatakse kujul, kus IP-aadress on pööratud oktettide kujul "in-addr.arpa" alamdomeenina. Näiteks IP-aadress "193.40.5.73" on kirjutatud kujul "73.5.40.193.in-addr.arpa". Pöördteisenduse tulemusena tagastatakse PTR tüüpi ressursikirje, mis sisaldab DNS-nime.

DNSSEC

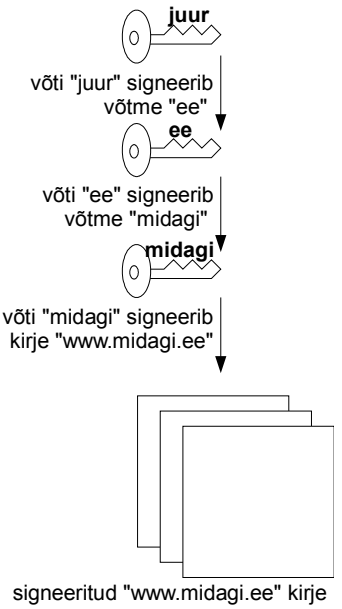
DNS-serverile tehtavad päringud ja vastused ei kasuta tõestatavat autentimist, mille tõttu on võimalik kasutada vahendusrünnet ja suunata kasutaja õige IP-aadressi asemel muule IP-aadressile (tõestatavaks autentimiseks nimetatakse autentimist, kui osapool omab mingit tõestust oma identiteedi kohta (üldjuhul implementeeritud krüptograafiliste vahendite abil)). Peale vahendusründe on DNS-serverites hoitavale informatsioonile ohuks serverite kompromiteerimised, mille käigus muudetakse DNS-nimeserveri tsoonifaile sissemurdjale sobivalt (näiteks suunates libapangalehele, mille abil on võimalik teostada vahendusrünnet ja varastada saadud paroolidega kontodelt raha). Võimaldamaks usaldusväärset nimeteenust, on välja töötatud DNS laiendus, mida tuntakse DNSSEC (*DNS Security Extensions*) nime all (RFC 3225, 3833, 4034- 4035).

DNSSEC on DNS-i laiendus, mis võimaldab domeeni operaatoritel nende domeenis olevad andmed krüptograafiliselt signeerida. Klient, saades päringule vastuse, kontrollib signatuuri järgi vastuse autentsust, mis võimaldab tõestada saadud ressursikirjete autentsust. DNSSEC toetusega DNS-nimeserverile tehtud päringu vastuse usalduse pidepunktiks on juurtsooni võti. Seda võtit kasutatakse esimese taseme domeenide signeerimiseks. Iga esimese taseme domeen kasutab oma võtit endast järgu võrra madalamate domeenide signeerimiseks jne. Selliselt käitudes tekib usaldusahel juurserverist lõppdomeenini. Kui mingi taseme domeeni haldav nimeserver ei toeta DNSSEC laiendust, siis alates sellest kohast ei ole informatsioon krüptograafiliselt kaitstud (ehk ei ole tõestatavalt autenditav), kuigi domeeni alamdomeene haldavad nimeserverid võivad jällegi toetada DNSSEC laiendust. Tähtis on katkematu usaldusahel.

Kui nimelahendaja sai päringu vastuse, mille kogu informatsioon saadi DNSSEC laiendust kasutades, siis on tulemus kaitstud vahendusründe ja serverite kompromiteerimisrünnakute eest eeldusel, et ründaja ei pääse ligi mingi haru signeerimise võtmele. Mõlemal juhul ei ole tänu signeerimisele võimalik andmeid muuta, sest klient avastab signatuuri kontrollides ebakõla. DNSSEC ei ole siiski kaitstud teenusetõkestusrünnete vastu.

DNSSEC kasutuselevõtuga on ka mitmeid probleeme, milletõttu on ka DNSSEC spetsifikatsiooni ümber tehtud, kuid siiski esineb jätkuvalt probleeme, mille tõttu DNSSEC ei ole globaalselt kasutusele võetud. Üheks esilekerkivaks probleemiks on ühilduvuse probleem. Ei ole teada,

kas vastav domeen on DNSSEC toetusega või mitte. Siiski on võimalik DNSSEC laiendust kasutades lahendada ka ühilduvuse usaldatavuse probleem. Nimelt saab domeen avalikuks teha kõik tsoonid, millele ta on signeerijaks. Järelikult peavad vastavad tsoonid toetama tõestatavat autentimist ehk DNSSEC laiendust. Selliselt on võimalik kontrollida, et vahepeal ei ole libanimeserver, mis väidab, et ei toeta DNSSEC laiendust. Täielikult DNSSEC-le üle minnes suureneks informatsiooni hulk, mis asub DNS ressursikirjetes ning kasutatav ribalaius mitu korda (olenedes suuresti, missugust informatsiooni pakkuda). See võib osutuda probleemiks, sest DNS ressursikirjed peavad olema kiiresti kättesaadavad (seega hoitakse mälus). Näiteks VeriSign, mis haldab ".com" domeeni, peab oma tsoonis hoidma 40 miljonit kirjet. Kui lisada juurde DNSSEC-i puhul vajalikud signatuurid, siis tekiks sellise koguse informatsiooni mahutamiseks probleem mälumahuga. DNSSEC-i probleemide leevendamiseks on lisatud juurde mitmeid võimalusi ja tehtud kitsendusi, näiteks, et päringu tegemisel saaks rohkem kirjeid küsida ja kirjeid saaks turvalisemalt hajutada. Teine tõsisem probleem DNSSEC-i kasutuselevõtul on lõppkasutajate tarkvara ühilduvuse saavutamine. Lõppkasutajaid on väga palju ja neile on vaja pakkuda DNSSEC-i kasutavat tarkvara ja ka selgeks teha, miks on see neile kasulik. Praegu on DNSSEC ühilduvusega tarkvara väga vähe ja väike osa organisatsioonidest on DNSSEC toetusega. Üks peamisi põhjusi on juurserverite DNSSEC toetuse puudumine.



Joonis 11.8: Näide domeeninimede andmete signeerimise usaldusahelast.

DNS-ründed pole tänapäeval väga sagedased, võrreldes teiste rünnetega, sest niisuguse rünnaku ettevalmistamine vajab rohkesti tööd ja planeerimist (tehakse ka DNS-vastaseid ründeid). Internetis on rohkem teisi rünnakuteks sobivaid subjekte, mis on kergemad ja vähem töömahukad. Vaatamata sellele on DNS puhul potentsiaalne võimalus suure kahjuga rünnakuteks. Seetõttu on oluline turvalisema keskkonna nimel kasutusele võtta DNSSEC laiendus.

DNSSEC kohta täiendava informatsiooni leidmiseks vaata http://www.dnssec.net.

Domeeninimi ja virtuaalserverid

Domeeninimesid on võimalik kasutada ka IP-aadresside kokkuhoiuks. Nimelt paljudel väi-

kestel asutustel, firmadel jne. on vajadus oma väikese veebilehe järele, mille jaoks on registreeritud ka domeeninimi. Enamasti on väiksema firma veebilehte mõtekas hoida vastavat teenust pakkuva firma serveris. Veebilehti majutav firma majutab mitme asutuse veebilehti ja neile viitavaid domeeninimesid. Veeb töötab, kasutades HTTP protokolli, kus versioonis 1.1, mida valdav enamus veebibrauseritest kasutavad, on kohustuslikuks väljaks päringus "host" parameeter, millega antakse teada küsitava hosti nimi. Seda informatsiooni on võimalik ära kasutada, et panna ühele IP-aadressile vastama mitu domeeninime ja anda serverile võimalus kuvada domeeninimele vastav koduleheküIg. Näiteks võivad serveris olla korraga "www.naide.ee" ja "www.muunaide.ee" kodulehed. Päringu "host" väärtuseks on päringu esitaja kohustatud panema vastava hosti (näiteks rida: "Host: www.naide.ee").

DDNS

Hostil, millel ei ole staatilist IP-aadressi, kuid mida oleks vaja viidata domeeninimega, on võimalik kasutada DDNS (*Dynamic DNS*) teenust (RFC 2136, 2137). DDNS on DNS laiendus, mis võimaldab DNS tsooni andmeid muuta ka üle võrgu.

DDNS üheks kasutatavamaks rakenduseks on dünaamilise IP-aadressiga kodukasutajatele staatilise domeeninime pakkumine (internetiteenusepakkujad jagavad IP-aadresse dünaamiliselt, et hoida kokku IP-aadresse). Tänu DDNS-le on võimalik inimesel pöörduda oma kodus oleva arvuti poole, näiteks kaugtöölaua kasutamiseks või testimiseks mõeldud veebiserveri jaoks.

DDNS teenusepakkujad pakuvad mingi hulga domeeninimesid, millele saab ise lisada soovitava alamtaseme märgendi. Näiteks võib DDNS teenusepakkuja pakkuda domeeninimesid "naide.com", "muunaide.nu". Valides sobiva domeeninime, saab vabalt valida alamdomeeninime. Näiteks võib kasutaja valida oma domeeninimeks "minuarvuti.naide.com" (mõned DDNS teenusepakkujad võivad ka teise taseme domeeninime pakkuda, kuid need on reeglina tasulised). Domeeninime saamiseks on vajalik DDNS teenusepakkuja juures teha konto, kus seadistatakse ka vajalikud parameetrid. Peale DDNS teenusepakkuja juures konto loomist ja domeeninime registreerimist on võimalik seadistada oma koduarvuti või marsruuter nende samade parameetritega (teenusepakkuja, kasutajatunnus, parool, domeeninimi). Seejärel ühendub arvuti või marsruuter DDNS teenusepakkujaga ja teatab talle oma IP-aadressi, mille järel tehakse DNS-nimeruumi vastavas domeeninimes uus alamdomeen või värskendatakse andmeid.

DDNS toetavad paljud kodudes kasutatavad marsruuterid. Neis on tavaliselt ette antud ka üks või mitu DDNS teenusepakkujat (tavaliselt tasuta DDNS teenusepakkujad, kuid on ka tasulisi).

Kuna DDNS teenuse kaudu seadistatud domeeninimede puhul võib IP-aadress olla muutlik, siis on DDNS abil seadistatud domeeninimed lühikese "aega elada" väärtusega. Samuti on vajalik DDNS teenust kasutaval hostil mingi aja tagant anda endast elumärki ja vajadusel värskendada andmeid. Vastasel korral võib DDNS teenusepakkuja arvata, et teenuse kasutamine on lõpetatud ning vabastab ressursid (lastes vabaks registreeritud domeeninime ja kustutades loodud konto).

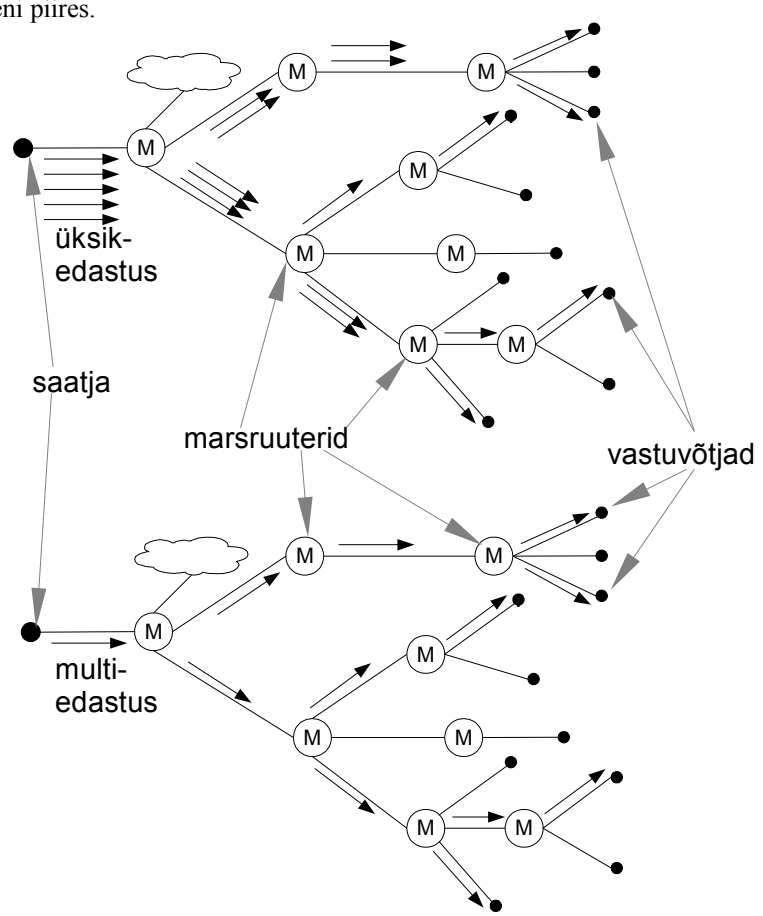
11.4 Multiedastus

Mõnikord on vaja infot edastada samaaegselt mitmele masinale. Muidugi võib saata kõigile eraldi sama andmevoo, kuid see oleks dubleerimine ja tihti on piiravaks teguriks sideliinide ribalaiused. Leviedastuse kasutamine ei ole igakord võimalik (näiteks sihtmasinad asuvad erinevates võrkudes) ega soovitav (tahetakse edastada ainult valitud masinatele). Selleks otstarbeks ongi kasutatav multiedastus.

Multiedastus eksisteerib nii kanalikihis (nt. Etherneti puhul) kui võrgukihis. Tavapäraselt mõistetakse multiedastust võrgukihis olevana.

Kanalikihi multiedastuse realiseerimine on lihtsam kui võrgukihi multiedastuse realiseerimine. Kanalikihi puhul on võrguliideses (arvuti puhul võrgukaardis) võimalik määrata, milliseid

multiedastuskanaleid "kuulatakse", registreerides vastavad multiedastusaadressid. Osad Etherneti võrgukaardid võivad multiedastust ka mitte toetada. Sellisel juhul antakse kaadri andmed edasi võrgukihile. Samamoodi käitutakse ka juhul, kui võrgukaardi maksimaalne multiedastusaadresside registreerimiste hulk on ületatud. Multiedastusaadress seotakse üldjuhul mingi protokolliga. Multiedastuse jaoks eraldati Etherneti puhul aadressiruum, kus MAC-aadressi esimese okteti viimane bitt on "1". Kanalikihi multiedastus saab levida ainult leviedastusdomeeni piires.



Joonis 11.9: Pildil on toodud juhtum, kus kuuele hostile korraga on vaja infot edastada (näiteks videokonverentsi puhul). Ülemisel juhul on kasutatud üksikedastust, alumisel juhul multiedastust. Multiedastust kasutades väheneb saatja ja tema lähedaste võrguseadmete sideliinide koormus märgatavalt.

Võrgukihi multiedastus on märgatavalt komplekssem kanalikihi multiedastusest. Järgnevalt vaatleme võrgukihi multiedastust IP kontekstis. Joonisel 11.9 on toodud näide IP multiedastuse efektiivsusest võrreldes üksikedastusega, kui andmeid on vaja edastada korraga mitmele hostile. Selliselt andmete edastades üle marsruuterite vastuvõtjateni tekib nn. multisaate jaotuspuu.

Multiedastus kogu Interneti ulatuses ei ole kasutatav, vaid piiratud Interneti osades. Multiedastuse toimimiseks peavad kõik teekonnal olevad marsruuterid toetama multiedastust. Paraku kõik marsruuterid ei toeta multiedastust või see on ära keelatud, nt. marsruuterite koormatuse tõttu, turvalisuse kaalutlustel (mitmete teenusetõkestusrünnete vältimiseks).

Multiedastuse saab jagada üks-mitmele ja mitu-mitmele multiedastuseks. Üks-mitmele multiedastuse puhul on üks saatja ja mitu vastuvõtjat. Mitu-mitmele puhul on osapooled üldjuhul

võrdsed ja saavad üksteisele saata ja teistelt vastu võtta. Rohkem on kasutatav ühe allikaga ehk ühesuunaline ehk üks-mitmele multiedastus. Mitmesuunalist multiedastust on märksa keerulisem realiseerida (nt. efektiivsuse ja marsruuterite koormatuse probleemid, kaotsi läinud paketid, sünkroniseerimine jm.). Lihtsuse mõttes vaatleme vaikimisi üks-mitmele multiedastust.

Üks-mitmele multiedastuse rakendusteks võivad olla: televisiooni- ja raadiosaadete edastamine; video- ja häälepõhiste konverentside, loengute jms. reaalajas edastamised; suunatud meedia, milleks võivad olla uudiste lühiuudised, ilmateate uuendused; failide hajusülekanne ja vahemälusalvestus, nt. replitseerimise eesmärgil; teadaanded (kellaaeg, erinevad uuenduste teadaanded jms.); monitoorimine ja tulemuste teavitamine.

Mitu-mitmele multiedastuse rakendusteks võivad olla: heli ja videopõhised konverentsid; ressursside sünkroniseerimine (nt. hajusandmebaasid); paralleelarvutusprogrammid; koostööprogrammid (nt. sama faili redigeerimine); kiirsuhtlusprogrammid; mitme mängijaga mängud; jm.

IP-multiedastuse ülesanne on edastada andmeid määratud hostidele ühest allikast. Neid määratud hoste ja saatjat nimetatakse multiedastusgrupiks. Multiedastusgrupp võib sisaldada ühte või enamat hosti. Multiedastusgrupp identifitseeritakse mingi multiedastuse IP-aadressiga. Üks host võib korraga kuuluda samaaegselt mitmesse multiedastusgruppi. Baasmultiedastuse tarvis on vaja kolme komponenti:

- Sihtaadresside hulka, mis kuuluvad konkreetsesse gruppi.
- Mehhanismi, mismoodi on võimalik ühineda grupiga ja lahkuda grupist.
- Marsruutimisprotokolli, mis võimaldab marsruuteritel piirata grupi liiklust väikseima rajaga puuks allikast kõikide grupi liikmeteni.

Multisaate jaotuspuud jaotatakse kaheks liigiks: lähtepuu (*source tree*) ja jagatud puu (*shared tree*). Jagatud puu multisaate korral kasutatakse ühte ühist juurt, milleks on võrgust valitud marsruuter. Sõltuvalt multisaate marsruutimisprotokollist nimetatakse antud marsruuterit kohtumispunktiks ehk RP (*Rendezvous Point*) või tuumaks (*Core*).

Lähtepuu põhise multiedastusgrupi loomiseks on liikmeid ühendavatel marsruuteritel vaja teada optimaalseid teekondasid teiste samasse multiedastusgruppi kuuluvate hostideni. Koostatavat teekonnakaarti nimetatakse multiedastuse levikupuuks. Teekonnakaart on puu kujuline, et vältida tsükleid ja moodustada optimaalne teekond kõigi multiedastusgrupi liikmeteni. Selliselt moodustub lühima tee puu ehk SPT (*shortest path tree*). Multiedastusgrupi levikupuu juureks on multiedastusgrupi liikluse allikas ja lehtedeks multiedastusgrupi liiklust vastuvõtvad liikmed. Marsruuterite jaoks erineb multiedastuspaketi saatmine üksikedastuspaketi saatmisest selle poolest, et ta võib olla sunnitud saatma paketi edasi mitte ainult ühte liidesele, vaid mitmele (marsruuterisse sisenes üks pakett ja saadeti edasi mitme teise liidese kaudu). Multiedastuse levikupuu on marsruutide "kaart", mismoodi multiedastuspakett edastatakse ühte või mitmesse harusse. Multiedastuse teekonnakaardi loomise eest hoolitsevad multiedastusmarsruuterid automaatselt, kasutades IGMP protokolli, multiedastuse marsruutimise protokolle ja multiedastuse marsruutimise algoritme. IGMP protokolli kasutatakse multiedastusgruppi kuuluvuse haldamiseks. Multiedastuse marsruutimisprotokolle kasutatakse multiedastusmarsruuterite vahel, et jagada topoloogia informatsiooni võrgus teiste multiedastuse marsruuteritega. Koos multiedastuse marsruutimisalgoritmidega arvutab üks või enam multiedastuse marsruuterit optimaalse multiedastuse levikupuu, et jõutaks kõikide vajalike sihtmarsruuteriteni. Kui ainult üks marsruuter teostab arvutused, siis see tsentraliseeritud marsruuter levitab vajalikku informatsiooni, loomaks marsruutimistabeli kirjed igasse teise marsruuterisse. Peale multiedastusgrupi marsruuteritabelite korda saamist on levikupuu kasutatav.

Multiedastuse jaoks on vajalikud erinevad protokollid: lokaalse võrgu sees kasutatakse multiedastuse juhtimiseks IGMP protokolli; marsruutimisdomeeni sees kasutatakse PIM, MOSPF ja DVMRP protokolle; marsruutimisdomeenide vahel kasutatakse multiedastuse juhtimiseks MBGP (*Multiprotocol Extensions for BGP*) (RFC 2858) protokolli. Multiedastamiseks kasutatakse transpordikihi protokollina UDP-d. TCP protokolli ei ole võimalik multiedastuseks kasutada, sest TCP eeldab kahte otspunkti, mis loovad omavahel ühenduse, mille kaudu hakatakse

andmeid vahetama.

IP multiedastuse protokollid klassifitseeritakse hõredat (*sparse*) ja tihedat (*dense*) laadi protokollideks. Hõredus ja tihedus viitavad multiedastusgrupi liikmete arvule võrgus sõltuvalt kogu hostide arvust võrgus. Hõreda laadi protokollid on tehniliselt komplekssemad ja nõuavad rohkem kavandamist.

Tiheda laadi multiedastamisel eeldatakse, et multiedastusgrupi lõpp-punktid paiknevad võrgus tihedalt (enamus võrgusolijatest kuulub multiedastusgruppi). Tiheda laadi multiedastuspaketiga ujutatakse esmalt üle kogu võrk (allikas saadab kõikidele vahetult ühendatud marsruuteritele ja nemad igale teisele vahetult ühendatud masinale). Multiedastuse marsruutimisprotokollid ja algoritmid on disainitud nii, et kindlustada pakettide jõudmine kõikide võrgus olevate marsruuteriteni. Kui ükski host ega muu allavoolu ühendatud marsruuter ei soovi vastavasse multiedastusgruppi kuuluda, siis saadab marsruuter puu pügamise teate (*prune*) allikapoolsele marsruuterile, millega lõpetatakse talle vastava multiedastuse saatmine. Kasutatavamad tihedas laadis töötavad multiedastuse marsruutimisprotokollid on DVMRP (*Distance Vector Multicast Routing Protocol*) (RFC 1075), MOSPF (*Multicast OSPF*) (RFC 1584), PIM-DM (*Protocol Independent Multicast - Dense Mode*) (RFC 3973).

Hõreda laadi multiedastuse puhul eeldatakse, et multiedastusgruppi kuulujad asetsevad võrgus hõredalt, mistõttu võrgu üleujutamine multiedastusega ei ole ratsionaalne. Hõredas laadis edastab marsruuter multiedastuse pakette ainult siis, kui naaber on eelnevalt avaldanud soovi vastava multiedastusgrupi pakettide vastuvõtmiseks. Selleks ehitatakse puu ainult nendeni, mis on avaldanud soovi olla vastava multiedastusgrupi liige. Ühinemiseks kasutatakse liitumisteadet (*join*) ja lahkumiseks pügamisteadet (*prune*). Hõreda laadi protokollideks on PIM-SM (*Protocol Independent Multicast - Sparse-Mode*) (RFC 4601) ja CBT (*Core Based Trees*) (RFC 2189, 2201) (CBT on katsetamisjärgus). Multiedastuse levikupuu konstrueerimiseks kasutatakse tavaliselt PIM-SM protokolli.

Üksikedastuse puhul suunatakse pakett järjest lähemale sihtpunktile. Multiedastuse puhul saadetakse paketid allikast eemale. Iga marsruutimisprotokoll proovib vältida marsruutimistsükleid. Multiedastuse puhul on see eriti tähtis, sest pakette ka paljundatakse. Seepärast leiab iga multiedastuse marsruutimisprotokoll iga grupi jaoks vastuvoolu liidese ja pärivoolu liidesed. Vastuvoolu liides on lähim allikale ja pärivoolu liidesed on lähimad grupi teistele liikmetele. Kui vastuvoolu liides ja pärivoolu liidesed on identifitseeritud, edastatakse vastava grupi pakette ainult vastuvoolust saadud liidesest, mis saadetakse edasi pärivoolu liidestesse. Pärivooluliidestest saadud paketid visatakse minema. Selliselt välditakse võimalikke tsükleid. Tegemaks kindlaks vastuvoolu ja pärivoolu liidesed, leiab multiedastuse marsruutimisprotokoll lühima tee allikasse (üksikedastuse puhul leiti lühim tee sihtkohta). Enamasti leitakse lühimad teed üksikedastuse marsruutimisprotokolli abil. Multiedastusprotokolle, mis kasutavad marsruutimistabelit, leidmaks lühimat teed allikasse, nimetatakse protokollist sõltumatuteks. Seevastu spetsiaalselt multiedastuse marsruutimiseks on olemas marsruutimisprotokoll DVMRP (*Distance Vector Multicast Routing Protocol*) (RFC 1075).

IGMP

Multiedastusgrupi haldamiseks peavad olema teada gruppi kuulujad. Selleks kasutatakse IGMP (*Internet Group Management Protocol*) protokolli. IGMP on ka TCP/IP protokollistiku komponent. IGMP pakub kolme põhilist funktsiooni:

- Hostid kasutavad IGMP-d, et teavitada vahetult ühendatud marsruuterit, et nad soovivad ühineda multiedastusgrupiga või lahkuda multiedastusgrupist.
- Marsruuterid kasutavad IGMP-d, et pärida multiedastusgrupi liikmete kohta vahetult ühendatud võrkudest, kui on saabunud uue multiedastusgrupi liiklus ning multiedastusgrupist ühe või enama vahetult ühendatud liikme lahkumisel.
- Kui mitu marsruuterit on vahetult ühendatud sama võrguga ja nad saavad sama multiedastusgrupi liiklust samast allikast, siis nad kasutavad IGMP-d, otsustamaks, milline

marsruuter hakkab edastama liiklust antud võrku.

IGMP teated kapseldatakse IP paketti, kus IP protokolli "protokoll" välja väärtuseks on 2. IGMP protokollist on olemas kolm versiooni:

- IGMPv1 (RFC 1112). Hostid saavad ühineda multiedastusgrupiga. Puudub multiedastusgrupist lahkumise teade. Lahkumine toimub aegumisega.
- IGMPv2 (RFC 2236). Lisatud grupist lahkumise teated.
- IGMPv3 (RFC 3376). Versioon 3 lisas allika järgi filtreerimise toetuse, mis võimaldab saada pakette ainult määratud lähteaadressidelt toetamaks SSM-i (*Source-Specific Multicast*) või kõikidelt välja arvatud määratud lähteaadressidelt vastavat multisaadet. Versioon 3 on disainitud olema koostalitusvõimeline versioonidega 1 ja 2.

IGMPv3 puhul peavad olema realiseeritud järgnevat tüüpi teated, et funktsioneerida ja olla ühilduv eelmiste versioonidega (sulgudes number tähistab tüübi numbrit): liikmelisuse päring (*membership query*) (0x11), IGMPv3 liikmelisuse raport (*membership report*) (0x22), IGMPv2 liikmelisuse raport (*membership report*) (0x16), IGMPv1 liikmelisuse raport (*membership report*) (0x12) ja IGMPv2 grupist lahkumine (*leave group*) (0x1723).

Vaatleme põgusalt liikmelisuse päringu teate väljasid:

- Tüüp. Määrab teate tüübi, mille väärtus antud juhul on 0x11. 8 bitti.
- Maksimaalne vastuse aeg kümnendiksekundites. 8 bitti.
- Kontrollsumma. 16 bitti.
- Grupi aadress. Kui tegemist multiedastusgrupi spetsiifilise teatega, siis vastava grupi IP-aadress. Üldise päringu korral nullid. 32 bitti.
- Reserveeritud. 4 bitti.
- S (*suppress router-side processing*). Ignoreeritakse normaalseid taimerite uuendusi, kui lipp on püsti. 1 bitt.
- QRV (*querier's robustness variable*). Määrab taimerite ja uuesti proovimiste arvu. 3 bitti.
- QQIC (*querier's query interval code*). Päringu intervall. 8 bitti.
- Lähteaadresside hulk järgmises väljas. 16 bitti.
- Lähteaadressid. IP üksikedastusaadresside loend. Varieeruva suurusega (32-biti kordne).

IPv6 jaoks kasutatakse IGMP sarnasteks ülesanneteks protokolli MLD (*Multicast Listener Discovery*) (RFC 2710, 3590, 3810, 4604), mis kasutab ICMPv6 teadete tüüpe (mitte enam IGMP omasid).

Multiedastusaadressiruumi jaotus

IP multiedastuse jaoks on eraldatud võrk 224.0.0.0/4 (ehk vahemik 224.0.0.0 kuni 239.255.255.255) (RFC 3171). Multiedastusaadresside jaotus jagatakse kolme kategooriasse: staatiline, skoobist sõltuv ja dünaamiline.

Multiedastusaadresse jagatakse otstarbe järgi:

- Lokaalse sideliini multiedastusaadressid – vahemik 224.0.0.0/24. Selliste multiedastusaadressidega pakette marsruuter edasi ei saada (nende "aega elada" väärtus on ka 1). Mõningaid registreeringuid:
  - ◇ 224.0.0.1 – kõikidele hostidele antud kohtvõrgus.
  - ◇ 224.0.0.2 – kõikidele marsruuteritele antud kohtvõrgus. Näiteks grupist lahkumise IGMP teade saadetakse antud multiedastusaadressile.
  - ◇ 224.0.0.5, 224.0.0.6 – kõikidele marsruutimisprotokolli OSPF kasutavatele marsruuteritele (224.0.0.6 on täpsemalt OSPF DR ja BDR marsruuteritele suunatud).
  - ◇ 224.0.0.9 – kõikidele marsruutimisprotokolli RIPv2 kasutavatele marsruuteritele.
- Võrkudevahelise juhtimisinfo plokk (*internetwork control block*) – vahemik 224.0.1.0/24. Aadressid, mis on seotud protokolliga ja edastatakse Internetis. Näiteks 224.0.1.1 on registreeritud SNTP (*Simple Network Time Protocol*) (RFC 4330) protokollile.

- Juhuvõrgu (*ad-hoc*) aadressid – vahemikud 225.0.0.0 kuni 231.255.255.255 ja 224.0.2.0 kuni 224.0.255.0.
- Reserveeritud – vahemik 234.0.0.0 kuni 238.255.255.255.
- GLOP plokk – vahemik 233.0.0.0/8 eraldatud autonoomsetele süsteemidele. Teine ja kolmas oktett sisaldavad autonoomse süsteemi ehk AS numbrit. (RFC 3180)
- Administratiivse skoobiga multiedastusaadressid – vahemik 239.0.0.0/8 (RFC 2365).

Multiedastusaadresside registreerimised asuvad aadressil http://www.iana.org/assignments/multicast-addresses.

IP multiedastuse ja Etherneti puhul tehakse IP-aadressi sidumist MAC-aadressiga. Sidumiseks kasutatakse reserveeritud MAC-aadresside vahemikku 01:00:5E:00:00:00 kuni 01:00:5E:7F:FF:FF ehk on eraldatud 23 bitti sidumaks IP multiedastusaadressi ja Etherneti multiedastusaadressi. Kuna IP multiedastusaadresside jaoks on kasutusel 28 viimast bitti, siis on MAC-aadressis viis bitti vähem kui IP-aadressis. Seetõttu ei ole IP-aadressi ja MAC-aadressi sidumine üks-ühele, vaid IP multiedastusaadressiosast lõigatakse ära esimesed viis bitti. Näiteks RIPv2 kasutatav multiedastusaadress 224.0.0.9 tõlgitakse MAC-aadressiks 01:00:5E:00:00:09. IP-aadresside 224.1.2.3 ja 225.1.2.3 puhul kasutakse sama MAC-aadressi. Osa võrguliideseid võimaldavad jooksvalt registreerida kuulatavaid multiedastusaadresse (kui vastavasse multiedastusgruppi ei kuuluta, siis ei koormata arvuti keskprotsessorit).

12. Marsruutimisprotokollid

Marsruutimise juures vaatlesime staatilist marsruutimist, kus võrguadministraator pidi käsitsi täitma marsruutimistabelit. Selline lähenemine on igati mõistlik, kui hallatav võrk on väike, lihtsakoeline ja stabiilne. Suuremad võrgud võivad olla keerulisemad, kasutada tõrkekindluse ning läbilaskevõime suurendamiseks dubleeritud ühendusi jne. Samuti muutub suurtes võrkudes töömahukamaks ning keerukamaks kõikide marsruuterite marsruutimistabelite uuendamine, kui võrgu topoloogias toimub muutusi (võrkusid tõstetakse ümber, juhtuvad tehnilised rikked, muudetakse ühendusi jne.). Seetõttu võib osutuda vajalikuks kasutada vahendeid, mis reageeriksid topoloogia muudatustele automaatselt, uuendaksid marsruutimistabeleid uuenenud infoga ja pakuksid koormuse jaotamist.

Võrguseadmete vahel automaatselt informatsiooni vahetamiseks on välja töötatud mitmeid protokolle, mida nimetatakse marsruutimisprotokollideks. Marsruutimisprotokollide ülesanneteks on hoida marsruutimistabelite sisu dünaamiliselt ajakohane. Seetõttu nimetatakse marsruutimisprotokollide vahendusel teada saadud marsruute dünaamilisteks marsruutideks. Käsitsi seatud marsruute nimetatakse staatilisteks.

Marsruutimisprotokoll on sisuliselt lihtsustatult marsruuterites töötav programm, mis hoiab kättesaadavate võrkude kohta informatsiooni, mida vahetab teiste marsruuteritega ning arvutab teadaoleva info põhjal parimad marsruudid erinevatesse võrkudesse, mis lisatakse marsruutimistabelisse.

Võrku nimetatakse konvergentseks, kui kõikidel marsruuteritel on samasugune ettekujutus võrkude kaugustest (vastavalt nende perspektiivile). Staatilist marsruutimist kasutades on võrk mittekonvergentne, kui näiteks on seadistatud vigane marsruut või mingi sideliin on maha läinud. Marsruutimisprotokolle kasutades on mingi sideliini maha minekul mittekonvergentsus lühiajaline (kuni jõutakse informatsiooni levitada).

Marsruutimisprotokollide peamisteks ülesanneteks on määrata kõige odavamad ehk parimad teed teistesse võrkudesse ja olla kursis võrgutopoloogiaga. Selle info põhjal muudetakse marsruutimistabelit. Informatsiooni saamiseks vahetavad erinevad marsruuterid omavahel informatsiooni kasutades marsruutimisprotokolle.

Internet on väga suur võrkude võrk, mille tõttu on ka marsruutimise info haldamine kompleksne ülesanne. Seetõttu on vajalik jagada ülesanne väiksemateks hierarhilisteks osadeks. Võrkude kogumikku, mida kontrollib üks üksus, nimetatakse **autonoomseks süsteemiks** ehk AS (*autonomous system*). AS on teisisõnu üks iseseisev üksus (näiteks mingi ülikooli võrk, ISP võrk). Autonoomset süsteemi nimetatakse vahel ka marsruutimisdomeeniks või administratiivseks domeeniks. AS-de eesmärk on jagada Internet väiksemateks hallatavateks võrgukogumikeks. AS-e identifitseeritakse numbritega. Numbrite registreerimised regionaalsetele Interneti registraatoritele asuvad aadressil http://www.iana.org/assignments/as-numbers. ARIN-i piirkonna AS numbrite registreeringute nimekiri asub aadressil ftp://ftp.arin.net/info/asn.txt.

Marsruutimisprotokollid jaotatakse IGP (*interior gateway protocol*) ja EGP (*exterior routing protocol*) protokollideks. IGP marsruutimisprotokollid on autonoomse süsteemi siseseks marsruutimisinfo vahetamiseks ja EGP on erinevate autonoomsete süsteemide vahel marsruutimisinfo vahetamiseks.

IGP protokolle:

- RIPv1 (*Routing Information Protocol*) (RFC 1058).
- RIPv2 (*RIP version 2*) (RFC 2453).
- OSPF (*Open Shortest Path First*) (RFC 2328).
- IGRP (*Interior Gateway Routing Protocol*). Cisco firmaomane.
- EIGRP (*Enhanced IGRP*). Cisco firmaomane, IGRP edasiarendus.

- IS-IS (*Intermediate system to intermediate system*) (ISO 10589). Töötati välja algselt OSI protokollistiku marsruutimisprotokolliks, hiljem kohandati IP võrkude jaoks. Sarnane OSPF-le. Kasutatakse vähe, peamiselt väga suurte ISP-de poolt.

EGP protokollid:

- EGP (*Exterior Gateway Protocol*). Vananenud ja enam ei kasutata.
- BGP (*Border Gateway Protocol*) (RFC 4271). Tänapäeval põhiliselt kasutatav (versioon 4).

IGP protokollidest toetavad VLSM tehnikat RIPv2, OSPF, EIGRP ja IS-IS (samuti ka staatiline marsruutimine). VLSM tehnikat ei toeta IGRP ja RIPv1. Mitte VLSM tehnikat toetavate marsruutimisprotokollide puhul ei saa kasutada marsruutide agregeerimist.

Oluline on teha vahet **marsruutimise** ja **marsruuditavate protokollide** vahel. Marsruutimisprotokollid muudavad dünaamiliselt marsruutimistabeleid ja käivad kaasas muutustega võrgutopoloogias, et marsruutimistabel oleks ajakohane ja efektiivne. Marsruutimistabelit kasutatakse marsruuditavate protokollide marsruutimiseks. Meie vaatleme marsruuditava protokollina IP protokolli.

Võrgus võib olla erinevate võrkude vahel mitu erinevat teed ehk marsruuti. Need marsruudid võivad olla erinevate kaugustega, läbilaskevõimetega ning muude parameetritega. Seega on kasulik, et mingil viisil ja määral levitataks informatsiooni marsruudi omaduste kohta. Marsruutimisel on kasulik eelistada paremaid marsruute, seetõttu on vaja erinevad marsruudid panna paremusjärjekorda. Selleks leitakse igale marsruudile kindel maksumuse väärtus, mida nimetatakse **meetrikaks**. Meetrika arvutatakse olenevalt marsruutimisprotokollist erinevate parameetrite alusel (näiteks mitu marsruuterit tuleb läbida sihtpunktini, missugune on sideliini ribalaius, vigade esinemissagedus jne). Meetrikat, mis koosneb mitmest parameetrist, nimetatakse **komposiitmeetrikaks**. Üldiselt on meetrika väärtuseks mingi täisarv, kus väiksem arv viitab paremale meetrikale ehk paremale marsruudile ehk väiksemale maksumusele. Marsruutimistabelisse lisatakse parim marsruut. Olenevalt protokollist võib olla lisatud ka mitu parimat marsruuti, siis on võimalik kasutada ka tasakaalustatud ühendusi. Osad marsruutimisprotokollid toetavad mitme ebavõrdse marsruudi kasutamist (muudetava koefitsendi piires) ehk mittetasakaalustatud marsruutide kasutamist.

Võrreldes staatilise marsruutimisega, on dünaamilise marsruutimise positiivseteks külgedeks:

- Võrguadministraatoril on vähem tööd.
- Muudatused võrgus kajastuvad marsruutimistabelites automaatselt, ilma võrguadministraatori vahelesegamiseta.
- Veaohtlikkus seadistamisel on väiksem.
- Võrgu suurenemisega kaasaskäimine tekitab vähem probleeme ja tööd.

Dünaamiliste marsruutimisprotokollide negatiivseteks külgedeks võib pidada:

- Kasutatakse marsruuteri ressursse (mälu, protsessorit, andmeedastust).
- Probleemide lahendamine võib vajada paremaid teadmisi marsruutimisprotokolli kohta.
- Turvalisuse riskid.

Marsruutimisprotokollid jagatakse oma marsruutimisalgoritmi poolest distantsvektor- (*distance vector*) ja sideliini-oleku- (*link-state*) marsruutimisprotokollideks.

IGP marsruutimisprotokollidest on klassidega protokollid RIPv1, IGRP. Klassideta on RIPv2, EIGRP, OSPF, IS-IS.

12.1 Distantsvektor-marsruutimisprotokollid

Distantsvektorprotokollid baseeruvad Bellman-Fordi algoritmil. Distantsvektorprotokollid näevad võrku läbi naabrite perspektiivi. Iga marsruuter, mis on vahetult ühendatud mingite võrkudega, teab nende võrkude olemasolust. Nendes võrkudes on potentsiaalselt ka teised marsruuterid, millele ta kuulutab välja võrke, mida ta teab (näiteks levi- või multiedastusega) (tavaliselt seadistatakse marsruutimisprotokollile, milliseid vahetult ühendatud võrke kuulatakse ja millis-

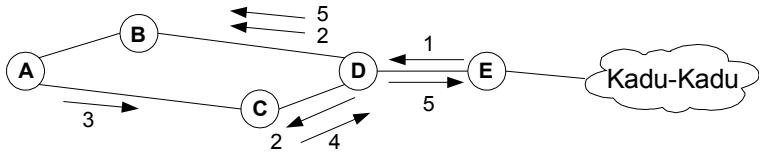
tesse vahetult ühendatud võrkudesse kuulutatakse). Kui vastavates kohtvõrkudes oli marsruutereid, siis nimetatakse neid **naabermarsruuteriteks**. Marsruuterid kuulutavad oma naabermarsruuteritele endaga vahetult ühendatud võrke ja teisi marsruute, mida on kuskilt mujalt õpitud. Selliselt levib informatsioon võrgus kõikide marsruuteriteni (vastava piirkonna või autonoomse süsteemi piires). Lisaks teadmisele võrkude kättesaadavuse kohta sisaldab informatsioon ka teadmisi marsruutide meetrika kohta. Kui marsruuter saab oma naabri käest informatsiooni mingite marsruutide kohta, siis ta teeb arvutused igasse võrku parima marsruudi leidmiseks (mingi teise naabri kaudu võis antud võrku olla parem marsruut). Marsruutimisinfo juures märgitakse, millise naabri käest on ta õpitud. Marsruutimisinfo saatmine toimub perioodiliselt fikseeritud aja tagant.

Nii teavad marsruuterid võrgust seda, missuguse naabri kaudu millised võrgud on kättesaadavad ja millise meetrikaga. Kui mingi aja jooksul ei saada vahetult ühendatud marsruuterilt uut uuendust, siis eemaldatakse selle marsruuteri kaudu õpitud marsruudid.

Distantsvektorprotokolli kirjeldatakse ka kui kuulujuttude levitamist, kus naaber räägib oma naabritele, mida on oma naabritelt kuulnud ja see levib kumulatiivselt. Kuulujuttude sisuks on võrgud ja nende maksumused (ehk meetrikad) rääkija kaudu. Rääkijatena võime ette kujutada turumutte, kellel on fikseeritud pikkusega mälu teistelt kuuldu kohta (neile peab perioodiliselt üle rääkima, et neil ikka kõik meeles püsiks, samas silmsidega naabreid (ehk vahetult ühendatuid) ei unustata).

Mingi sideliini funktsioneerimise lõppemisel võtab võrgu konvergentsuse saavutamine aega, mis sõltub marsruutimisdomeeni suurusest, marsruutimisuuenduste saatmiste aegadest (saatmised ei ole sünkroonis) ja tihedusest.

Distantsvektorprotokollidel on ka mõningaid nõrkusi, mis võivad probleeme tekitada. Näiteks kui võrgus, kus on olemas mitu marsruuti, kaob ära ühendus mingi teise marsruuteriga ja selle informatsiooni levitamine võtab aega, siis on võimalik, et tekivad marsruutimistsüklid. Joonisel 12.1 on näide marsruutimistsükli tekkimisest.



Joonis 12.1: Joonisel olevad numbrid tähistavad tegevuste järjekorda (alljärgnevalt viidatud tekstis sulgudes) ja ringid erinevaid marsruutereid. Algselt on kogu marsruutimisdomeen konvergentne. Marsruuteri A jaoks on parima meetrikaga marsruut võrku "Kadu-Kadu" läbi marsruuteri B. Kui marsruuter E kaotab ühenduse võrguga "Kadu-Kadu" (võrk "Kadu-Kadu" ei ole muud moodi ühendatud), annab ta sellest marsruutimisprotokolli kaudu teada marsruuterile D (tegevus 1). Nüüd sai D teada, et võrk "Kadu-Kadu" pole enam kättesaadav, kuid teised marsruuterid peale D ja E veel ei tea sellest (nad jätkavad pakettide saatmist marsruuter E suunas, mille tulemusena marsruuter D ei oma marsruuti vastavasse võrku ja saadab tagasi vastavasisulise ICMP teate, et võrk ei ole kättesaadav). Järgmiste marsruutimisuuenduste saatmisega saadab marsruuter D uuendused marsruuteritele B ja C, kes saavad ka nüüd teada võrgu "Kadu-Kadu" kadumisest (tegevus 2). Nüüd teavad toimunud topoloogia muutusest kõik marsruuterid peale marsruuteri A. Nüüd saadab marsruuter A välja oma marsruutimistabeli uuendused (tegevus 3). Marsruuter C saab teada, et võrk "Kadu-Kadu" on kättesaadav marsruuteri A kaudu. Marsruuter C saadab informatsiooni edasi marsruuterile D (tegevus 4) ja D omakorda marsruuteritele B ja E (tegevus 5), mis saavad nüüd teada, kus asub võrk "Kadu-Kadu" (mis on aga ebaõige informatsioon). Nüüd on tekkinud tsükkel, kus võrku Kadu-Kadu adresseeritud andmepakett saadetakse ringiratast, kuni ta IP päise "aega elada" väärtus saab nulliks ning ta lõpuks kõrvaldatakse, saates lähetajale "eluaja ületamise" ICMP teate. Enne paketi kõrvaldamist koormatakse asjatult võrku.

Erinevate probleemide vältimiseks on kasutusele võetud mitmesuguseid tehnikaid:

- Meetrikate maksimaalväärtused. Marsruutide meetrikate juures fikseeritakse maksimaalsed väärtused, millest võrgu meetrika ehk maksumus ei saa suurem olla. Kui juhtub, et meetrika jõuab marsruutimisuuenduste tõttu üle maksimumväärtusest, siis seda võrku eiratakse (ta on lõpmatult kaugel) ja paketid, mis saadetakse sinna võrku, visatakse minema.
- Vaatevälja jagamine (*split horizon*). Põhimõte on mitte anda marsruutimisuuendustega edasi võrke naabrile, kellelt vastav info oli saadud ehk õpitud (ei ole mõtet hakata õpetama midagi kellelegi, mida ise oled õppinud temalt endalt).
- Marsruudi mürgitamine. Kui mingi marsruuteri liides läheb maha, siis saadab ta naabritele marsruudi, mis annab teada, et võrk ei ole kättesaadav. Selliselt välditakse teiste ebaõigete marsruutide tekkimist domeenis.
- Ajendatud uuenduste saatmised. Kui toimub topoloogia muutus (marsruuteri enda liides või mingi muu marsruuter teavitab topoloogia muutumisest), siis ootamata ära järgmist marsruutimistabelite uuendamise aega, saadetakse kohe uuendused naabritele. Selliselt saab kogu domeen kiiremini uuesti konvergentseks ja välditakse võimalike probleemide tekkimist.
- Maashoidmise taimerid (*holddown timers*). Põhimõte on takistada vastava marsruutimistabeli kirje uuendamist mingi aja jooksul, et lasta topoloogia muutusel ennast võrgus propageerida (ehk lasta muutusel levida). Kui algselt kättesaadav võrk uue uuendusega enam ei ole kättesaadav, siis märgitakse vastav võrk mittekättesaadavaks ja käivitatakse limiidiga taimer (RIP puhul vaikimisi 180 sekundit). Kui enne aja täissaamist saadakse naabrilt uuendus, kus vastav võrk on kättesaadav, siis tehakse võrk uuesti kättesaadavaks ja eemaldatakse ajamõõdik. Kui enne ajalimiidi täissaamist saadakse mingilt naabermarsruuterilt vastavasse võrku marsruut, mille meetrika ei ole parem algsest, siis seda marsruuti ignoreeritakse, aga kui see on parema meetrikaga, siis asendatakse marsruut ja võrk saab jälle kättesaadavaks (uue naabri kaudu). Selliselt lastakse informatsioonil levida võrgus ja ei lasta võimalikust ebaõigest informatsioonist tekitada probleeme. Maas hoidmise taimerid aitavad hoida kokku topoloogia muudatustega seotud arvutuste sagedust, kui mingi sideliini või võrgu kättesaadavus on katkendlik.

Distantsvektor-marsruutimisprotokollideks on RIP, RIPv2, IGRP, BGP. BGP ja EGP liigitatakse ka eraldi radavektorprotokollideks. Järgnevalt vaatleme neid protokolle lähemalt, kusjuures peatume natuke pikemalt RIPv2 juures, sest RIP ja IGRP ei toeta VLSM ja CIDR tehnikaid. Eraldi vaatleme ka BGP protokolli.

RIP

Kõige lihtsam ja kõige enam toetatud marsruutimisprotokoll on RIP (*Routing Information Protocol*). Marsruutimisprotokollist RIP on olemas kaks erinevat versiooni RIPv1 (RFC 1058) ja RIPv2 (RFC 1723). RIPv2 on RIPv1 edasiarendatud ja täiendatud versioon, mille põhitäiendused on VLSM, CIDR, autentimise toetus, lisainformatsiooni andmine marsruutimisinfoga (näiteks kas marsruut on õpitud RIP-i abil või mingil muul moel), järgmise hüppe IP-aadress (enne võeti lähteaadressist) ja võrgu võrgumask. RIPv1 saatis uuendusi välja leviedastusaadressil 255.255.255.255, kuid RIPv2 saadab uuendusi välja multiedastusaadressil 224.0.0.9. RIP IPv6 versiooni nimetatakse RIPng (*RIP next generation*) (RFC 2080). Järgnevalt keskendume RIP osas peamiselt RIPv2-le.

RIP on mõeldud kuni keskmise suurusega võrkude jaoks. Mõlemate puhul on meetrikaks hüpete arv lõpp-punkti (ehk mitu marsruuterit peab läbima, et jõuda sihtpunkti). Kui hüpete loendur on suurem kui 15, siis pakett visatakse minema. Vaikimisi toimub marsruutimistabelite uuendamine iga 30 sekundi tagant. Uuendused saadetakse vaikimisi välja kõigist liidestest (vastavalt RIP-i seadistusele), isegi kui seal ei ole RIP protokolli kasutavaid masinaid (näiteks kohalikku lõppkasutajate võrku). Sellisel juhul on tegemist raisatud ribalaiusega ja turvariskiga

(mõnikord on marsruutimistabelite saatmine mõningatele hostidele, mis kasutavad saadud informatsiooni, siiski vajalik, kuid see pole tavajuhtum). Soovitatav on marsruuter seadistada sellistesse portidesse marsruutimisuuendusi mitte saatma.

RIP kasutatab marsruutimisprobleemide vastu mitmeid eelnevalt vaadeldud tehnikaid. Meetrika maksimaalväärtuseks on 15 hüpet. See tähendab, et kui `A` naabermarsruuter `B` raporteerib, et mingi võrk asub 15 hüppe kaugusel ja `A` ei oma sinna võrku muud marsruuti, siis antud võrk jääb ikkagi marsruuterile `A` kättesaamatuks, sest nüüd oleks võrk juba 16 hüppe kaugusel. RIP kasutab **mürgitamist**, s.t., et paneb meetrikaks 16 (rohkem, kui lubatud). RIP puhul on naabritele marsruutingute saatmise ajaks vaikimisi 30 sekundit, marsruutide aegumise ajaks 180 sekundit (kirje märgitakse mittekasutatavaks) ja kui peale seda ikka ei tule uuendust, siis eemaldatakse vastav marsruut (tavaliselt 240 või 300 sekundi vanuselt). Kui peaks toimuma topoloogia muutus, saadetakse koheselt uuenenud marsruutimistabelid naabritele. Vaatamata erinevatele meetoditele võtab konvergentsi saavutamine märgatavalt aega.

RIP puhul on võimalik kasutada ka sideliini koormuse tasakaalustamist RIP jaoks võrdsete marsruutide osas, kuigi nad võivad olla erinevate läbilaskevõimetega. Tasakaalustamiseks kasutatakse ringiratast meetodit (kasutatakse võrdselt kõiki marsruute järjekorra alusel tsükliliselt).

RIP marsruutimisprotokolli on väga hea kasutada väikses ja homogeenses võrgus, kus ta on kõige efektiivsem. RIP kasutab transpordikihi protokolli UDP ja pordi numbrit 520.

RIP protokolli paketi ülesehitus:

- Käsk (*command*). Päringuteate puhul 1, vastuse puhul 2. Päringut kasutab näiteks uus marsruuter, siis kui ta ilmub ja soovib marsruutimisinfot. 1 bait.
- Versiooninumber. RIPv1 puhul 1, RIPv2 puhul 2. 1 bait.
- Kasutamata. 2 baiti.
- Marsruudi kirjed. Saab olla kuni 25 kirjet. Kui kasutatakse autentimist, siis esimene kirje on autentimiseks.
  - ◇ AFI (*address family identifier*). Näitab, missuguse adresseerimisega on tegemist. IP puhul on 2. Autentimise puhul väärtuseks FFFF. Täieliku marsruutimistabeli sisu päringu puhul 0 (uus marsruuter küsib marsruutimisinfot).
  - ◇ Marsruudi märgend (RIPv1 puhul oli kasutamata). Kasutatakse väljaspoolt RIP-i õpitud marsruutide märgistamiseks. Siin võidakse märkida AS number (RIP ise ei kasuta). 2 baiti.
  - ◇ IP-aadress. Sihtvõrgu aadress. 4 baiti.
  - ◇ Võrgumask (RIPv1 puhul oli kasutamata). 4 baiti.
  - ◇ Järgmise hüppe IP-aadress. (RIPv1 puhul oli kasutamata). Näitab paremat järgmise hüppe aadressi. Kui väärtuseks on null, siis näitab järgmise hüppena kuulutajat ennast (kui võrk temaga vahetult ühendatud). 4 baiti.
  - ◇ Meetrika. Hüpete loendur vahemikus 1 kuni 16 (1-15 tavakasutuseks, 16 kasutatakse mürgitamiseks või päringu puhul). 4 baiti.

Päringuga küsitakse andmeid mingi võrgu kohta (IP-aadressi väljal), millele vastatakse päritud võrku kuuluvate võrkudega. Päringu puhul aadressile "0.0.0.0" saadetakse kogu marsruutimistabeli sisu. Päringu puhul on meetrikaks 16, positiivse vastuse puhul 1-15 ja negatiivse vastuse (ei olnud sellist võrku) puhul on samuti 16.

IGRP

IGRP on Cisco firmaomane marsruutimisprotokoll, mis baseerub aadressiklassidel (loodud 1980-ndate alguses). IGRP komposiitmeetrika pannakse kokku sideliini ribalaiusest, viivitusest, koormusest, töökindlusest ja MTU-st. Vaata meetrika arvutamise valemit 12.1. Marsruutimistabeleid saadetakse vaikimisi välja iga 90 sekundi tagant. Üldiselt enam ei kasutata (uuem Cisco marsruuterite tarkvara ei sisalda enam IGRP toetust).

EIGRP

EIGRP on teistest erinev marsruutimisprotokoll, mis on hübriid distantsvektor- ja sideliini-olek-marsruutimisprotokollidest. Ta kuulub pigem distantsvektor-marsruutimisprotokollide alla, kuid kasutab ka sideliini-oleku-marsruutimisprotokollide omadusi. EIGRP on edasiarendus protokollist IGRP, olles ka tagasiühilduv. EIGRP-ga (ka IGRP-ga) on võimalik kasutada ebavõrdsete sideliinide tasakaalustamist (saab kasutada mingisse võrku korraga mitut mittevõrdset marsruuti, et kasutada mitut sideliini). Marsruutide leidmiseks kasutatakse DUAL (*Diffused Update Algorithm*) algoritmi. Marsruutimisuuendusi saadetakse välja topoloogia muutumisel multiedastusaadressil 224.0.0.10. EIGRP maksimaalne hüpete arv võrkude vahel on 224 (IGRP puhul 255).

$$meetrika = \left(K1 \cdot ribalaius + \frac{K2 \cdot ribalaius}{256 - koormatus} + K3 \cdot viivitus \right) \cdot \frac{K5}{töökindlus + K4}$$

Valem 12.1: IGRP meetrika arvutamise valem. Valemis K1, K2, K3, K4 ja K5 on konfigureeritavad väärtused, mis vaikimisi on järgmised: K1=K3=1 ja K2=K4=K5=0. Kui K5 on null, siis K5/(töökindlus+K4) osa valemist ei kasutata. Kuna vaikimisi on K2, K4 ja K5 nullid, siis mõjutavad meetrikat ribalaius ja viivitus, seega meetrika = ribalaius + viivitus. Ribalaiuse väärtus saadakse, kui leitakse marsruudi väikseim ribalaius ühikuna Kbit/s ja jagatakse 10 000 000 antud arvuga.

12.2 Sideliini-oleku-marsruutimisprotokollid

Erinevalt distantsvektorprotokollidest vahetavad sideliini-oleku-marsruutimisprotokollid omavahel informatsiooni konkreetsete sideliinide kohta. Selle informatsiooni baasil ehitatakse üles topoloogia andmebaas, millest omakorda arvutatakse igasse võrku lühimad teed ning need lisatakse marsruutimistabelisse. Sideliini-oleku-marsruutimisprotokollid teavad oma domeenis olevaid marsruutereid ja seda, kuidas nad on omavahel ühendatud.

Sideliini-oleku-marsruutimisprotokollideks on OSPF (*Open Shortest Path First*) ja IS-IS (*Intermediate system to intermediate system*). Kuna OSPF on palju enam kasutusel, siis käsitleme sideliini-oleku-marsruutimisprotokolle OSPF põhjal. IS-IS on suuresti sarnane OSPF-le.

OSPF

OSPF marsruutimisdomeeni nimetuseks on **ala** (*area*). OSPF alas haldab iga marsruuter oma topoloogia andmebaasi, mis sisaldab alas olevaid marsruutereid ja nende vaheliste ühenduste infot (vastava marsruuteri enda perspektiivis). Selle andmebaasi põhjal leitakse parimad marsruudid kõikidesse võrkudesse ja tehakse vastavad muudatused marsruutimistabelisse. Parimate marsruutide leidmiseks vaadeldakse topoloogia tabelit graafina, millele rakendatakse Dijkstra algoritmi, et leida igasse võrku parima meetrikaga tee. OSPF kasutab meetrikana maksumust. Mida väiksem maksumus, seda parem meetrika. Tavaliselt on maksumus vaikimisi seotud sideliini läbilaskevõimega, kuid seda on võimalik ka administraatori poolt ümber määrata. Leidnud lühima tee võrkudesse, lisatakse need marsruudid marsruutimistabelisse.

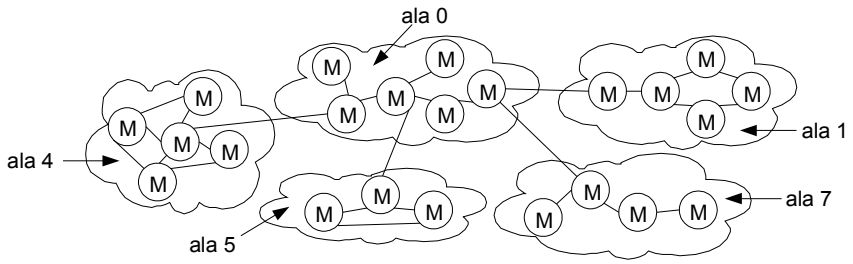
OSPF marsruuterid peavad arvet naabrite kohta. Selleks avastavad OSPF marsruuterid uusi OSPF naabreid ja loovad naabritega naabrussuhteid. Naabrite avastamiseks ja naabrussuhte hoidmiseks kasutatakse Hello protokolli, mis saadab perioodiliselt iga liidese kaudu välja Hello-pakette (aadressile 224.0.0.5). Hiljem jätkatakse perioodiliselt Hello-pakettide saatmist, millega kontrollitakse naabri korrasolekut.

Kui OSPF marsruuteri sideliiniga toimub mingi muudatus, saadetakse kõikidesse liidestesse, kus on olemas OPSF naaber, multiedastusaadressile 224.0.0.5 välja LSA (*link-state advertise-*

ment) teated LS Update paketis (ühes paketis võib olla ka mitu LSA-d), mis annab teistele OSPF marsruuteritele teada topoloogia muutusest vastava võrguliidese kohta (näiteks mingi sideliin on maha või püsti läinud, meetrika muutunud). Liidestesse, kus marsruuteri jaoks ei ole mitte ühtegi naabrit, topoloogia muudatustest informatsiooni ei saadeta. Naabrid saadavad omakorda LSA-d edasi oma naabritele vastavas OSPF alas (tagasi ei saadeta), et kõik vastava ala marsruuterid saaksid info topoloogia muutusest. Peale LSA-de saatmisi uuendatakse oma topoloogia-andmebaas ja leitakse uuesti parimad marsruudid kõikidesse võrkudesse ning värskendatakse marsruutimistabelit.

Erinevalt distantsvektorprotokollidest saadetakse ainult uuendus vastava võrguliidese muudatuse kohta (mitte kogu marsruutimistabel) ja see saadetakse kõikidele marsruuteritele (mitte ainult naabritele). Distantsvektorprotokollide puhul oli vajalik leida uuenenud info põhjal parimad marsruudid ja siis alles saadi edasi saata naabritele marsruutimistabelid uuenened infoga, kuid see võtab aega. OSPF-marsruuterid saadavad enne uuenduse edasi, alles seejärel hakkavad leidma parimaid marsruute, mida marsruutimistabelisse panna.

OSPF puhul on oluline mõiste "ala". Iga marsruuter kuulub mingisse alasse. Ala seespoolsed muudatused on nähtamatud väljapoole. Seega, kui toimub mingi sideliini maha minek, siis ei ujutata kõiki marsruutereid üle LSA-dega, vaid ainult oma alasse kuuluvad marsruuterid. Alasid on laias laastus kahte liiki (numbri alusel). Esimest liiki on "0" (null) ala ja teised on muud alad, millede ala identifikaator on nullist suurem. Nullala peab OSPF marsruutimisdomeenis olemas olema ja vajadusel ühendatakse temaga teised alad. Teised alad saavad omavahel ühenduda vaid läbi nullala. Väiksemas võrgus, kus ei ole vaja aladeks jaotust, peavad kõik marsruuterid kuuluma nullalasse. Kui võrk on suurem ja tekib vajadus alasid tükeldada, siis kehtivad reeglid, et kui on ainult üks ala, siis see ala on nullala. Kui on rohkem kui üks ala, siis peavad kõik teised alad olema ühendatud nullalaga. Nullalasid võib OSPF domeenis olla ainult üks. Lisavõimalusena on võimalik kasutada ka virtuaalset ühendamist läbi teiste alade nullalaga. Selliselt on võimalik teha OSPF võrgud märksa mastabeeritavamateks. Kui kasutatakse mitut ala, siis nullala nimetakse ka **magistraalalaks** (*backbone area*), sest ta ühendab erinevaid alasid omavahel. Selliselt tehakse alasid väiksemaks, mille tõttu on ka liiklust vähem (LSA-sid vahetatakse ainult oma ala piires). Joonisel 12.2 on toodud näide võrgust, mis on jagatud aladeks.



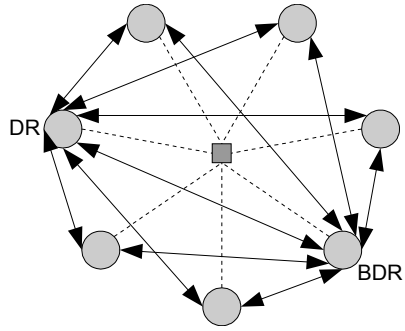
Joonis 12.2: OSPF aladeks jaotamine. Ala 0 ühendab kõiki teisi alasid, täites magistraali ülesandeid.

OSPF marsruuteritel peab olema unikaalne marsruuteri identifikaator. Marsruuteri identifikaator võib olla lahendatud sõltuvalt realisatsioonist, kasutades näiteks kõige väiksemat (või suurimat) marsruuteri IP-aadressi. Soovitatav on seadistada OSPF marsruuteri jaoks tagasisidestusliides, millega vähendatakse probleeme, kui väikseima (või suurima) IP-aadressiga liides või kõik liidesed peaksid maha langema.

Võrgus võib esineda olukord, kus ühte leviedastusdomeeni kuuluvad paljud OSPF marsruuterid (näiteks marsruuterite suurem sõlmpunkt). Sellisel juhul looksid kõik marsruuterid kõigiga naabritevahelise otseühenduse, mis tähendab, et kokku oleks neid $n \cdot (n-1)/2$, kus n on leviedastusdomeenis olevate marsruuterite arv. Seega, kui sõlmpunktis on kaheksa marsruuterit, siis

tuleks luua 28 naabritevahelist ühendust. Seetõttu jaotatakse naabrussidemed võrgutüüpide järgi nelja erinevasse klassi: *broadcast multiaccess* (näiteks Ethernet puhul); *nonbroadcast multiaccess* (näiteks Frame Relay, X.25 ja SMDS puhul); punktist-punkti (*point-to-point*) (näiteks PPP, HDLC puhul); *point-to-multipoint* (käsitsi seadistatav). Nendest kahe esimese puhul on olemas eelpool mainitud probleem, kahe viimase puhul seda ei esine. Kahe esimese korral valitakse välja üks marsruuter, mis hakkab kõikide naabriks (loob naabriühendused kõikide teistega). Seda marsruuterit nimetatakse **määratud marsruuteriks** ehk DR (*designated router*). DR marsruuteri kaudu hakatakse levitama leviedastusdomeenis LSA-sid. DR marsruuter leitakse valimiste teel (arvestatakse marsruuteri liidese prioriteeti ning nende võrdsusel marsruuteri identifikaatorit). Kuna DR marsruuteri maha langemisel ei ole enam edasisaatjat kohe olemas, siis valitakse paralleelselt välja ka tagavara DR marsruuter, mida nimetatakse tagavara määratud marsruuteriks ehk BDR (*backup designated router*). DR ja BDR marsruuteriga võrgus kasutatakse multiedastusaadressi 224.0.0.5 (ehk kõikidele OSPF marsruuteritele) DR marsruuteri poolt kõikidele marsruuteritele antud võrgus. DR ja BDR marsruuteritele antakse teada topoloogia muutustest LSA-dega multiedastusaadressil 224.0.0.6 (ehk OSPF DR ja BDR marsruuteritele). Vaata ka joonisel 12.3 olevat näidet.

Hiljem lisandunud OSPF marsruuterite puhul ei korraldata uusi valimisi, kuigi ta võib olla prioriteetsem (näiteks endine DR). DR maha langemisel võtab DR koha üle BDR ja valitakse uus BDR. BDR maha kukkumisel valitakse uus BDR ja mõlemate maha langemisel valitakse mõlemad.



Joonis 12.3: Ringid tähistavad OSPF marsruutereid ja keskel olev nelinurk kommutaatorit (või hubi). Punktiirjooned tähistavad füüsilisi ühendusi. Nooltega jooned tähistavad marsruuterite vahelisi suhtlemisi.

OSPF teadete päis on järgmine:

- Versioon. OSPFv2 puhul väärtuseks 2, OSPFv3 puhul väärtuseks 3. 1 bait.
- Tüüp. Määratakse, mis liiki teatega on tegemist, sellest oleneb paketi andmete osa. Paketi tüüpe (number sulgudes on identifikaator): Hello (1), DBD (*Database Description*) (2), LS (*Link State*) Request (3), LS Update (4), LS Acknowledgment (5). 1 bait.
- Paketi pikkus. Paketi pikkus baitides koos päisega. 2 baiti.
- Marsruuteri identifikaator. Lähtemarsruuteri identifikaator. 4 baiti.
- Ala identifikaator. Näitab, mis alast on pakett pärit. 4 baiti.
- Kontrollsumma. Kasutatakse standardset IP kontrollsummat kogu paketi kohta, välja arvatud autentimise väli. 2 baiti.
- Autentimise tüüp. 2 baiti.
- Autentimine. 8 baiti.
- Teatele spetsiifilised väljad, mis määrati väljaga "tüüp".

Hello-tüüpi pakette kasutatakse naabruse avastamiseks ja naabrussuhte ülesseadmiseks, DR ja BDR marsruuterite välja selgitamiseks. Hello-paketile spetsiifilised väljad:

- Võrgumask. 4 baiti.

- Hello intervall. Mitme sekundi tagant saadetakse Hello pakette. 2 baiti.
- Valikud. 1 bait.
- Marsruuteri prioriteet. Kasutatakse DR ja BDR marsruuteri valimistel. 1 bait.
- Marsruuteri surnuks tunnistamise intervall. Kui märgitud aja jooksul ei ole saadud Hello-paketti, siis eemaldatakse vastav naaber sideliini olekute andmebaasist ja saadetakse teade sideliini oleku muutuse kohta kõikidesse liidestesse, kus teatakse olevat OSPF marsruuter. 4 baiti.
- Määratud marsruuter ehk DR. Määratud marsruuteri identifikaator (kui on olemas). 4 baiti.
- Tagavara määratud marsruuter ehk BDR. Tagavara määratud marsruuteri identifikaator (kui on olemas). 4 baiti.
- Naabrite loend. Loend sisaldab naabermarsruuterite identifikaatoreid. Iga elemendi kohta 4 baiti.

Võrreldes distantsvektorprotokollidega on sideliini-oleku-protokollid palju vähem vastuvõtlikumad marsruutimisinfo probleemidele. Antud juhul saavutatakse konvergents kiiremini. Sideliini-oleku-protokollid kasutavad vähem ribalaiust, sest nad saadavad topoloogia muutumisel vastava muutuse kohta uuenduse. Kuid sideliini-oleku-protokollid vajavad palju rohkem mälu ja võimsamat protsessorit (sõltub suuresti võrgu suurusest, seadistusest ja stabiilsusest). Suurte võrkude puhul on mõttekas jagada OSPF-domeen aladeks, mis vähendab vajatavat mälumahtu ja protsessori koormatust (ühes suures võrgus on ka sagedamini muudatusi). OSPF käitamisel on vajalikud põhjalikumad teadmised kui distantsvektorprotokollide korral (OSPF-i kasutatakse tavaliselt suuremates võrkudes).

OSPF-st on IPv4 puhul kasutusel versioon 2 ja IPv6 puhul kasutusel versioon 3. OSPF töötab otse IP protokollil, kus protokolli välja väärtuseks on OSPF pakettide puhul 89.

12.3 Veel marsruutimisprotokollidest

Mõnikord on vajalik kasutada mitut erinevat marsruutimisprotokolli (osad marsruuterid ei toeta näiteks valdavalt kasutusel olevat marsruutimisprotokolli). Ühtlasi võib esineda vajadus kombineerida dünaamilisi marsruutimisi staatiliste marsruutidega (näiteks võrkudesse, kus ei ole marsruuteritel marsruutimisprotokollide toetust; vastavad marsruudid on tagavaraks jms.). Mõnel juhul võib lugeda mõne marsruutimisprotokolli infot usaldatavamaks kui mingi teise marsruutimisprotokolli infot (näiteks OSPF kaudu õpitu on eelistatum kui RIPv2 abil õpitu). Selleks on olemas mõiste administratiivne kaal (*administrative distance*). **Administratiivne kaal** määrab erinevate marsruutimisprotokollide ja staatiliste marsruutide prioriteedid marsruutimistabelisse lisamisel (igale protokollile on määratud oma administratiivne kaal). Kui juhtub, et mitu marsruutimisprotokolli raporteerivad marsruudist mingisse võrku, siis marsruutimistabelisse pannakse parema administratiivse kaaluga marsruutimisprotokolli poolt õpitud marsruut.

Marsruutimisprotokolle ei ole mõtet kasutada (raiskavad ainult ressursse), kui marsruuter on ühendatud ainult ühe liidese kaudu mittevahetult ühendatud võrkudega. Sellist marsruuterit nimetatakse ka **tupikmarsruuteriks**. Tupikmarsruuteris on vaja määrata ainult staatiliselt üks vaikemarsruut. RIP marsruutimisprotokolli tasuks kasutada siis, kui dubleeritud sideliinide osas ei ole suuri erinevusi.

BGP

BGP (*Border Gateway Protocol*) (RFC 4271) on EGP protokoll. Tänapäeval on valdavalt kasutusel BGP versioon 4. BGP4 on disainitud pakkuma tsüklivaba domeenide vahelist marsruutimist autonoomsete süsteemide ehk AS (*autonomous system*) vahel. Marsruutimiseks autonoomsete süsteemide vahel peab igas ühes neis olema vähemalt üks marsruuter, mis suhtleb BGP protokolli alusel. Kõik kättesaadavad sihtkohad on identifitseeritud AS numbriga. IP-aadressivahemikud on seotud sihtkoha AS-ga ja täisteega sihtkoha AS-i kuulutavast BGP marsruu-

terist. Täistee on võrkude (ehk AS-de) loend, mida peab teel sihtpunkti läbima. Radasid kirjeldab atribuutide nimekiri (nt. mis tüüpi teenuseid saab vastu võtta ja piirangud raja kasutamisel). Marsruudi arvutamisel ja raja valimisel toimub otsustamine lühima tee järgi sihtpunkti. Tee on seda lühem, mida vähem ta läbib autonoomseid süsteeme. BGP on distantsvektor-marsruutimisprotokoll, mis erinevalt IGP distantsvektorprotokollidest identifitseerib raja elemendid (näiteks RIP teadis, kui kaugel subjekt asub). Seetõttu ei kannata BGP tsüklite ja aeglase konvergentsi probleemi tõttu. Erinevalt IGP-dest ei pea BGP marsruuterid olema vahetult ühendutud, seetõttu nimetatakse osapooli **partneriteks** (*peer*), mitte naabriteks.

BGP marsruuterid edastavad omavahel kolme liiki teateid: avamise (*open*), uuenduse (*update*) ja teavituse/ülalhoide (*notification*/*keepalive*) teateid. Transpordikihi protokollina kasutatakse TCP protokolli usaldusväärse ühenduse saamiseks (port on 179), mille poolest erinetakse teistest marsruutimisprotokollidest. Kõigi kolme liiki teadete puhul on esimesed kolm välja samasugused.

BGP avamise teateid kasutatakse partnerite vahelise suhte loomiseks ja häälestamiseks. Peale suhte loomist vahetatakse partnerite vahel ülalhoideteateid. BGP avamise teate sisu:

- Marker. Sisaldab autentimise informatsiooni või on kõik bitid ühed. 16 baiti.
- Pikkus. BGP paketi suurus baitides (koos päisega). Väärtus on vahemikus 19 kuni 4096. 2 baiti.
- Tüüp. Määrab teate tüübi. Väärtused on järgmised: 1 – avamise teade; 2 – uuenduse teade; 3 – teavituse teade; 4 – ülalhoide teade. 1 bait.
- Versioon. BGP4 puhul väärtuseks 4. 1 bait.
- Autonoomse süsteemi number. 2 baiti.
- Kehtimise aeg. Kui vastava aja jooksul ei saada BGP teadet, siis loetakse partner maha langenuks. 2 baiti.
- BGP identifikaator. Igal BGP marsruuteril on unikaalne identifikaator, mida ta kasutab teistega suhtlemisel. 4 baiti.
- Mittekohustuslike parameetrite pikkus. 1 bait.
- Parameetri tüüp. Võimaldab mittekohustuslike funktsioonide lisamisi. 1 bait.
- Parameetri pikkus. 1 bait.
- Parameetri väärtus. Muutuva suurusega.
- Täiendavad parameetrid.

Peale BGP partnerite omavahelise suhte loomise vahetavad nad marsruutimistabeleid BGP uuenduste teadetega. Hiljem kajastatakse teadetes topoloogia muutusi. Samuti korratakse perioodiliselt olemasolevaid marsruute. BGP uuenduse teate sisu on järgmine:

- Eelnevalt vaadeldud väljad marker, pikkus ja tüüp.
- Eemaldatud marsruutide pikkus (*unfeasible routes length*). Näitab järgmise välja pikkust baitides. 2 baiti.
- Eemaldatud marsruudid (*withdrawn routes*). Sisaldab listi IP-võrkudest, mis ei ole enam kättesaadavad vastavas autonoomses süsteemis. Muutuva suurusega.
- Kogu raja atribuutide pikkus. 2 baiti.
- Raja atribuudid. Kirjeldab vaadeldavat marsruuti. See info lisatakse ka vastava marsruudi juurde marsruutimistabelis. Muutuva suurusega. Raja atribuutide alamväljad:
  - ◇ Lipud (O, T, P, EL). 4 baiti.
    - O (*optional*). Näitab, kas tegemist mittekohustusliku atribuudiga.
    - T (*transitive*). Näitab, kas fakultatiivne atribuut on transitiivne (sihiline).
    - P (*partial*). Näitab, kas raja atribuutide list on lõplik või järgneb veel.
    - EL (*extended length*). Kui lipp püsti, siis atribuudi pikkus on 2 baiti.
  - ◇ Reserveeritud. 4 baiti.
  - ◇ Atribuudi tüübi kood. 1 bait.
  - ◇ Atribuudi pikkus. 1-2 baiti.
  - ◇ Atribuudi väärtus. Muutuva suurusega.

- NLRI (*network layer reachability information*). Loend IP-võrkudest, mis on kättesaadavad vastava BGP marsruuteri kaudu. See info lisatakse marsruutimistabelisse. Raja atribuutide väli sisaldab vastava marsruudi täiendavat infot. Muutuva suurusega.

Atribuutidest on kohustuslikud:

- ORIGIN. Marsruudi päritolu. 0=IGP; 1=EGP; 2=muud moodi.
- AS_PATH. Autonoomsete süsteemide radade segmentide järjestus. Iga segment sisaldab raja tüübi, pikkuse ja väärtuse andmeid.
- NEXT_HOP. Teekonna järgmise hüppe moodustava piirimarsruuteri IP-aadress.

Teavituste eesmärk on teada anda veasituatsioonist. Teavituse sisu on järgmine:

- Eelnevalt vaadeldud väljad: marker, pikkus ja tüüp.
- Veakood. 1 bait.
- Vea alamkood. 1 bait.
- Andmed. Muutuva suurusega.

Ülalhoidepakett sisaldab ainult kohustuslikke väljasid. Ülalhoidepaketti kasutatakse BGP marsruuterite töökorras olemise kontrollimiseks. Neid saadetakse perioodiliselt fikseeritud aja tagant.

Administraatori poolt on võimalik piirata BGP marsruutide vastuvõtmist ja saatmist. Näiteks soovitakse jõuga kasutusele võtta halvema meetrikaga marsruut. Marsruudil, mille kättesaadavus on katkendlik, tehakse meetrika tehislikult pikemaks, mis suurendab tõenäosust, et valitakse mingi teine marsruut. Kui AS siseselt on mitu piirimarsruuterit, siis on nende vahel vaja vahetada informatsiooni. Selleks on võimalik kasutada ka IGP protokolle, kuid parem on kasutada ka sisemiselt BGP-d. Sisemiselt kasutatavat BGP-d kutsutakse IBGP-ks (*interior BGP*) ja väliste piirimarsruuteritega suhtlemist EBGP-ks (*exterior BGP*) (tegemist ei ole erinevate protokollidega).

13. IPv6

Praegu põhiliselt kasutusel olev loogilise adresseerimise ehk võrgukihi protokoll IPv4 on avaldatud 1980. aastate alguses. See oli algselt mõeldud USA kaitseministeeriumi tarvis ning ei olnud disainitud ülemaailmselt kasutatavaks protokolliks. Seda arvestades on IPv4 suutnud koos täiendustega üsna hästi vastu pidada. Siiski on mitmeid probleeme ja kitsaskohti, mis tingivad vajaduse uue loogilise adresseerimise protokolli järele. Tulevikuperspektiivis tuleb IPv4 asendada uue protokolliga. Peamisteks IPv4 puudusteks on muutunud aadressiruumi kitsaksjäämine ja Interneti magistraalmarsruuterite marsruutimistabelite kasvamise ja hallatavuse probleemid. Välja on töötatud uus protokoll IPv6 (*Internet protocol version 6*) (RFC 2460, 2780), mis eelduste kohaselt peaks kunagi tulevikus vahetama välja praegu kasutusel oleva IPv4, mida ta juba tasapisi teeb. IPv6 ei too endaga kaasa fundamentaalseid muudatusi. Kehtivad edasi CIDR ja VLSM põhimõtted.

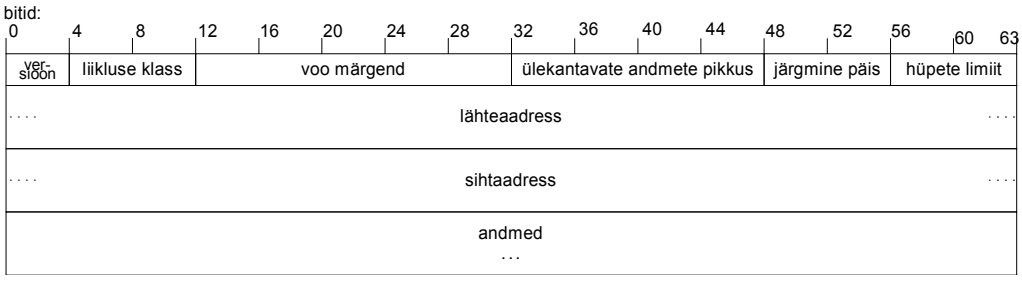
IPv6 paketi sisu on järgmine (vt. ka joonist 13.1):

- Versioon. Näitab IP protokolli versiooni. IPv6 puhul alati väärtuseks "0110", mis on kümnendsüsteemis 6. 4 bitti.
- Liikluse klass (*traffic class*). Sama tähendusega, mis IPv4 diferentseeritud teenuste väli. 8 bitti.
- Voo märgend (*flow label*) (RFC 1809). See on mõeldud kiiresti edastatava voo teostamiseks, mis on oluline eelkõige reaalajasüsteemide puhul. Ülesseatud ühendus kahe otspunkti vahel märgitakse märgendiga, mille alusel on võimalik vältida marsruutimistabelite läbivaatamisi ja muid tegevusi normaalse IP paketiga. Kiirust lisab asjaolu, et seda on riistvaraliselt võimalik lihtsalt teostada. Lisaks on võimalik voole ribalaiust reserveerida. Kasutatav QoS teenuse realiseerimiseks, näiteks RSVP (*Resource ReSerVation Protocol*) protokolliga. Voo märgendid võivad olla erinevate sideliinide peal erinevad ja teoreetiliselt saab korraga ühel sideliinil neid olla natuke üle miljoni (2^{20}=1048576). Kui voo märgendit ei toetata või ei ole see vajalik, siis on väärtuseks null. 20 bitti.
- Ülekantavate andmete pikkus (*payload length*). Näitab paketi andmete osa pikkust. Andmed algavad peale päise lõppu. Andmete osas on ka laienduspäised. 16 bitti.
- Järgmine päis. Sisu on sama IPv4 välja "protokoll" sisuga, kuhu on lisatud juurde IPv6 laienduste päiste identifikaatorid. Kui IPv6 pakett ei sisalda laienduspäiseid, siis väli "järgmine päis" viitab järgmisele protokollile. IPv6 paketis, kus sisaldub laienduspäiseid, viidatakse vastava laienduspäise identifikaatori tüübile, mis järgneb päisele. Laienduspäis viitab vajadusel omakorda järgmisele laienduspäisele. Kõige viimasena viidatakse transpordikihi protokolli identifikaatorile. Laienduspäiste järjekord on oluline.
- Hüpete limiit. Põhimõtteliselt sama otstarve, mis IPv4 puhul väljal "aega elada". 8 bitti.
- Lähteaadress. 128 bitti.
- Sihtaadress. 128 bitti.
- Andmed. Andmete esimeses osas asuvad üks või mitu päiselaiendust (kui neid esineb).

Võrreldes IPv4-ga, on paketi sisus peamisteks erinevusteks:

- Päis on IPv6 puhul fikseeritud pikkusega (40 baiti). Selle tõttu kaob ära päise pikkuse väli.
- Aadresside pikkused on suuremad. IPv4 puhul 32 bitti, IPv6 puhul 128 bitti.
- Juurde on lisatud liikluse klassi väli diferentseeritud teenuste asemel (varem oli teenuse tüübi väli).
- Lisatud on voo märgend. Mõeldud reaalaja-süsteemide jaoks.
- Kogupikkuse väli on asendatud ülekantavate andmete pikkuse väljaga. Sellega saab IPv6 puhul olla ülekantavate andmete hulk natuke suurem kui IPv4 puhul.
- Väli "protokoll" on asendatud väljaga "järgmine päis". IPv6 puhul võidakse lisaks protokollile viidata ka laienduspäisele.

- Väli "aega elada" on asendatud väljaga "hüpete limiit" (semantika ei ole muutunud, kuid see on rohkem sisule vastav termin).
- Eemaldatud on fragmenteerimise jaoks vajalikud väljad (identifitseerimisnumber, fragmendi nihe ja lipud). Fragmenteerimine on nüüd võimalik ainult laiendusvälja kasutades. IPv6 puhul proovitakse vältida fragmenteerimist, kasutades pigem raja MTU leidmist (*path MTU discovery)* (RFC 1981). Kui üks fragment läheb kaduma, siis on vaja kogu pakett uuesti saata.
- Eemaldatud on päise kontrollsumma, kuna see ei õigustanud ennast enam, sest tänapäeval on sideliinid kvaliteetsemad. Kanali- ning transpordikihis on IP päis ja andmed juba kontrollsummaga kaitstud (mõned harvad erandid välja arvatud). Päise kontrollsumma eemaldamine võimaldab kiiremat töötlust ja marsruutimist.



Joonis 13.1: IPv6 paketi struktuur.

Kuna IPv6 päis on suurem kui IPv4 päiseväli, siis on vajalik kohandada TCP ja UDP protokollid IPv6 jaoks (TCP ja UDP kontrollsumma välja tõttu). IPv6 UDP puhul on kohustuslikuks muudetud kontrollsumma väli.

IPv6 puhul on minimaalseks MTU väärtuseks IPv6 marsruuterites 1280 baiti, soovituslik on 1500 baiti. IPv4 puhul oli soovituslik 576 baiti ja ametlik miinimumväärtus 68 baiti. MTU leidmiseks kasutatakse ICMPv6 teadet "pakett liiga suur". Fragmenteerimise toetuse jätmine laienduspäisesse vabastab fragmenteerimise kohustusest paketi vahendajad (tavaliselt marsruuterid).

IPv6 aadressiruumi suuruseks on 128 bitti, mis teeb erinevate aadresside hulgaks 2^{128} = 340 282 366 920 938 463 463 374 607 431 768 211 456. Kuna aadress on muutunud neli korda pikemaks, siis pole otstarbekas kasutada IPv4-aadresside üleskirjutusviisi. IPv6 puhul on kasutusele võetud uus notatsioon. IPv6-aadressi puhul kirjutatakse arvud kuueteistkümnendsüsteemis, mis rühmitatakse nelja kaupa ja eraldatakse kooloniga (seega kokku 8 grupeeringut (128/(4·4)=8)) (arve kümnendsüsteemis esitades oleks arvude vahemik 0-65535; samuti on kuueteistkümnendsüsteemi mugavam kasutada, teades teisendamise reegleid). Näiteks on IPv6-aadress "2001:0bb8:0000:0000:0209:3dff:0000:e8c0". Aadressi lühendamiseks on lubatud ära jätta nulle rühmitatute algusest kuni mitte nullini. Kui järjest on üks või mitu rühmitust, mis sisaldavad ainult nulle, siis võib selle rühma asendada kahe kooloniga, kuid seda ainult ühes kohas (teistes rühmitustes peab vähemalt ühe nulli alles jätma). Topeltkoolonit tasub kasutada selles kohas, kus on kõige rohkem järjest ainult nulle sisaldavaid rühmitusi. Seega oleks viimane aadress kirjutatav lühemalt "2001:bb8::209:3dff:0:e8c0". IPv6-aadress "0000:0000: 0000:0000:0000:0000:0000:1" on kirjutatav kujul "::1" (RFC 4291).

IPv6 puhul on võrgumaskil sama tähendus kui IPv4 puhul, kuid lihtsalt kaldjoone formaadis ehk CIDR notatsioonis on võimalikud teoreetilised väärtused /0 kuni /128 (IPv4 puhul on /0 kuni /32). Kui võrgu suurus on näiteks /80, siis võrgumask alternatiivkujul oleks "ffff:ffff:ffff:ffff:ffff::", /78 puhul "ffff:ffff:ffff:ffff:fffc::".

URI-is kirjutatakse IPv6-aadress kantsulgude sisse. Kantsulge kasutatakse pordi numbri pärast. Näiteks olgu aadress "2001:bb8::209:3dff:0:e8c0" ning veebiserver, mis töötab pordil

8080. Seega URL selle aadressini oleks: "http://[2001:bb8::209:3dff:0:e8c0]:8080". URI "http://[::1]/kataloog/index.php" on kohaliku masina veebiserveri aadress, mis töötab pordil 80 ja kust päritakse kataloogist "kataloog" faili "index.php".

IPv6 laienduspäised

Tavaliselt laienduspäiseid ei kasutata, sest need vajavad lisatöötlust ja vähendavad marsruutimise jõudlust. Erinevad laienduspäised on erineva struktuuriga, kuid esimene väli on "järgmine päis". Päise tüüpide identifikaatorite registreerimised asuvad aadressil http://www.iana.org/assignments/protocol-numbers. Kui vastuvõtja ei tunne mingit laienduspäist, siis saadab ta tagasi vastava ICMPv6 veateate (parameetri probleem). Laienduspäiste puhul on oluline nende järjestus. Järjestus on järgmine (sulgudes olev number näitab päise identifikaatorit "järgmine päis" väljas): IPv6 päis; hüpe-hüppelt (*hop-by-hop*) laienduspäis (0), sihtpunkti valikute laienduspäis (60); marsruutimise laienduspäis (43); fragmenteerimise laienduspäis (44); AH (*Authentication Header* – IPsec protokolli pakett) laienduspäis (51); ESP (*Encapsulating Security Payload* – IPsec protokolli pakett) laienduspäis (50); sihtpunkti valikute laienduspäis (60); ülemise kihi protokoll. Sihtpunkti laienduspäis on järjestuses kaks korda. Vaatleme mõnda laienduspäist lähemalt.

Hüpe-hüppelt laienduspäis võimaldab kaasa panna fakultatiivset informatsiooni, mida iga võrgu edastaja (tavaliselt marsruuter) peab vaatama. Hüpe-hüppelt laienduspäise väljad: järgmine päis (1 bait); laienduspäise pikkus (1 bait); valikud (muutuva suurusega).

Sihtpunkti laienduspäist kasutatakse lisaparameetrite edastamiseks sihtpunktile. Sihtpunkti laienduspäise väljad on järgmised: järgmine päis (1 bait); laienduspäise pikkus (1 bait); valikud (muutuva suurusega).

Marsruutimise laienduspäist kasutatakse marsruudi määramiseks paketile, andes ette edastajate nimekirja ühest või mitmest IP-aadressist, mida pakett peab läbima. Marsruutimise laienduspäise väljad on järgmised: järgmine päis (1 bait); laienduspäise pikkus (1 bait); marsruutimise tüüp (1 bait); mitu lüli veel (1 bait); marsruutimistüübile omased andmed (muutuva suurusega). Kui marsruutimise tüüp on null, siis on marsruutimistüübile omasteks väljadeks reserveeritud väli (4 baiti) ja üks või rohkem IPv6-aadressi järjestikku.

Fragmenteerimise laienduspäis pakub võimalust jagada pakett väiksemateks osadeks. Erinevalt IPv4-st toimub fragmenteerimine vajadusel ainult saatvas masinas. Fragmenteerimise laienduspäise väljad: järgmine päis (1 bait); reserveeritud (8 bitti); fragmendi nihe (13 bitti); reserveeritud (2 bitti); lipp "veel fragmente" (1 bitt), identifikaator (32 bitti).

Lisaks on IPv6 puhul olemas eraldi järgmise päise identifikaator "ei ole järgmist päist", mille järel enam edasi ei vaadata ja kui pakett saadetakse edasi, siis muutumatul kujul.

IPv6 aadressiruum

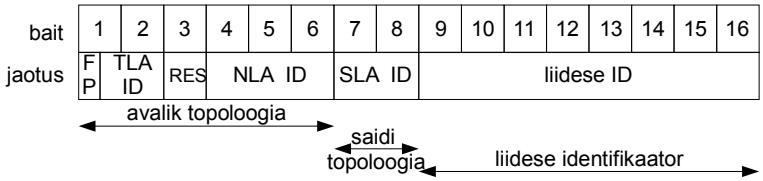
IPv6 (RFC 3513, 2374) aadressid jagatakse kolme kategooriasse:

- Üksikedastus. Sarnane tähendus kui IPv4 puhul.
- Multiedastus. Sarnane tähendus kui IPv4 puhul.
- Suvaedastus (*anycast*). Suvaedastus on multiedastuse variatsioon, kus suvaedastus tähistab lähimat hosti (arvutused teeb marsruuter, kasutades marsruutimisprotokolli). Lähim võib tähendada marsruuteri jaoks ka vähem hõivatud sideliini kasutamist (seega võimalik teostada hõivatuse jagamist). Suvaedastuse aadress on tavaline üksikedastusaadress, mille muudavad suvaedastuseks marsruuterid. Suvaedastuseks sobivad olekuta teenused (kliendi ja serveri vahel ei looda seanssi ja ei ole oluline, millise serveri poole järgmine päring saadetakse). Näiteks võidakse kasutada suvaedastust pöördumiseks nimeserveri poole või replitseeritud andmebaasist andmete saamiseks.

IPv6 puhul saab lähteaadressiks olla samuti ainult üksikedastusaadress või kui oma IP-aadressi ei teata, siis spetsiaalne IPv6 aadress ::0 (näiteks kui küsitakse IP-aadressi).

IPv6-aadress jaotatakse kuude ossa, milleks on (alates teisest väljast käib globaalselt unikaalsete IP-aadresside kohta) (vaata ka joonist 13.2):

- Formaadi prefiks (*format prefix*). 3 bitti. Erinevad prefiksid omavad järgmisi tähistusi:
  - ◇ 000 – spetsiaalkasutuse jaoks.
  - ◇ 001 – globaalselt unikaalsete üksikedastusaadresside jaoks (IP-aadress, mida arvutid Internetis omavad).
  - ◇ 111 – multiedastusaadresside ja kohaliku võrguasukoha (*local-site*) jaoks.
- Ülemtaseme agregeerimise identifikaator ehk TLA ID (*top level aggregation identifier*). TLA ID identifitseerib magistraalühenduste teenusepakkuja.
- RES. Reserveeritud tulevikus kasutamiseks. 8 bitti.
- Järgmise taseme agregeerimise identifikaator ehk NLA ID (*next level aggregation identifier*). 24 bitti.
- Võrguasukoha taseme agregeerimise identifikaator ehk SLA (*site level aggregation identifier*). Kui asutus vahetab teenusepakkujat, siis alates SLA ID-st jäävad IP-aadressid samaks (muutub SLA ID-st ettepoole jääv osa). 16 bitti.
- Liidese identifikaator.



Joonis 13.2: IPv6 aadressi bittide jaotus baitide kaupa.

RFC 3587 muutis aadressiformaati, kus viimased 64 bitti on ikka liidese identifikaator, SLA ID nimetati ümber alamvõrgu identifikaatoriks (*subnet ID*) (kui on vajalik alamvõrgustamine, siis tavaliselt jagatakse välja 16 bitise alamvõrgu osaga) ning esimesed bitid globaalseks marsruutimise prefiksiks (*global routing prefix*).

IPv6 puhul on mitmeid spetsiaalkasutusega aadresse või aadressivahemikke, milledest mõningad olulisemad on järgmised:

- ::1/128 – tagasisidestusaadress (*loopback address*) (IPv4 analoog on 127.0.0.1).
- ::/128 – spetsifitseerimata aadress. Tohib kasutada ainult lähteaadressina, kui ei teata oma aadressi ja alles saadakse omale IP-aadressi. Taolisi pakette ei saadeta marsruuterite poolt edasi.
- 0000::/8 – reserveeritud.
- 0200::/7 – reserveeritud NSAP (*Network Service Access Point*) jaoks eraldamiseks (prefiks: "0000 001").
- 0400::/7 – reserveeritud IPX jaoks eraldamiseks (prefiks: "0000 010").
- ::/96 – IPv4 jaoks, sõlmedele, mis toetavad IPv6. Viimased 32 bitti on IPv4-aadressi bitid. Enam ei kasutata.
- ::FFFF:0:0/96 – IPv4 sõlm ei toeta IPv6. Viimased 32 bitti on IPv4-aadressi bitid.

Olulisemad globaalselt unikaalsete üksikedastusaadresside spetsiaalkasutusega aadressiplokid:

- 2001:db8::/32 – kasutatakse dokumenteerimiseks ja näidistes (RFC 3849).
- 2002::/16 – kasutatakse 6to4 adresseerimisel.
- fe80::/10 – lokaalse sideliini aadressid (*link-local*). IPv4 analoogiks on 169.254.0.0/16 aadressiruum. Marsruuterid ei saada neid pakette edasi (prefiks: "1111 1110 10").
- fec0::/10 – lokaalse võrgukoha aadressid (*site-local*). Analoogne IPv4 privaatsete IP-aadresside võrkudega (192.168.0.0/16, 10.0.0.0/8, 172.16.0.0/12) (Interneti marsruuterid

viskavad need ära). Lokaalse võrgukoha aadresside prefiksi järgsed 38 bitti on nullid, kuni SLA osani (sise- ja välisvõrku saadetavate pakettide puhul on alates SLA osast hosti IP-aadressi bitid samad) (prefiks: "111 1110 11").

- fc00::/7 – ULA (*Unique Local IPv6 Unicast Addresses*) (RFC 4193) aadressid on marsruuditavad ainult võrgu ja temaga ühendatute sees. ULA asendab lokaalse võrgukoha aadressid (RFC 3879). ULA omadusteks on väga suure tõenäosusega globaalne unikaalsus; hästi teada prefiks, mille alusel on lihtne võrgukoha äärtes välja filtreerida pakette (selliselt ei satu võrgukoha liiklus väljapoole); võimaldab kinniselt ühendada erinevaid võrke, ilma IP-aadresside konflikti ohuta (näiteks VPN-iga); sõltumatu ISP-st ehk internetiteenusepakkujatest ega vajagi ühendust Internetiga; marsruutide või DNS kirjete lekkimiste korral ei ole konflikte teiste aadressidega.
- ff00::/8 – multiedastusaadressid (prefiks: "1111 1111").

Üldine aadressiruumi jaotus on kättesaadav aadressilt http://www.iana.org/assignments/ipv6-address-space. IPv6 aadressideruumi jaotus RIR-de vahel on kättesaadav aadressilt http://www.iana.org/assignments/ipv6-unicast-address-assignments.

IPv6 aadressiruum on võrreldes IPv4 aadressiruumiga muutunud veelgi hierarhilisemaks. IP-aadressid on juba oma formaadi poolest hierarhilised (tänu TLA ID, NLA ID ja SLA ID osadele). Agregeeritava marsruutimise hierarhia tõttu saavad eelkõige magistraalmarsruuterite marsruutimistabelid olla väiksemad (võrreldes IPv4 magistraalmarsruuteritega). Selliselt saavad SLA marsruuterid kasutada vaikemarsruudiks NLA marsruutereid. NLA marsruuterid omakorda saavad kasutada vaikemarsruudiks TLA marsruutereid. TLA marsruuterite marsruutimistabelid peavad sisaldama kirjeid ainult TLA ja tema all olevate NLA marsruuterite kohta. NLA marsruuterite marsruutimistabelid omakorda peavad sisaldama kirjeid NLA ja tema all olevate SLA marsruuterite kohta.

IPv6 ja turvalisus

IPv4 oli algselt disainitud väheste turvalisuse meetmetega. Hiljem on IPv4-le jooksvalt lisatud juurde mitmeid turvalisuse lisandusi. IPv6 disainimisel on juba algselt arvestatud turvalisusega (RFC 2401).

IPv6 puhul on juba algselt lisatud IPsec toetus autentsuse ja konfidentsiaalsuse vajaduste tarvis. IPsec-i käsitleme pikemalt peatükis "IPsec" (lk. 146).

IPv4 puhul on ohtlik ühendada haavatavat arvutit (puudub tulemüür ja viimased turvauuendused) Internetiga. Arvuti nakatub mõne minuti jooksul mõnda turvaauku kasutava ussiga. Nakatunud arvuti hakkab ise samuti juhuslikele IP-aadressidele viirusi levitama. Eriti haavatavad on Windows-operatsioonisüsteemiga masinad. IPv6 aadressiruum on väga suur, mille tõttu muutub haavatavate hostide otsimine Internetis eesmärgiga neid nakatada viirustega sisuliselt mõttetuks, sest tõenäosus neid leida on liialt väike.

Erinevused IPv6 ja IPv4 vahel

Põhilisteks täiendusteks IPv6 puhul võrreldes IPv4-ga võiks lugeda:

- Aadressiruumi laiendamine ja aadressi suuremaks muutmine. Aadressiruumi kitsikus oli IPv4 põhiline kitsaskoht, mis tingis vajaduse uue IP protokolli järele.
- Paketi päise lihtsustumine. Kaotati valikute väli, mis ei leidnud tihti kasutamist. Valikute välja kaotamisega ei pea võrguseadmed enam tegema pikkuse kontrolli, see kiirendab paketi käsitlemist. Samuti kaotati paketi fragmenteerimiseks vajalikud väljad.
- Täiendatud valikute võimalusi ja laiendusi.
- Voo märgendi võimekus on mõeldud eelkõige reaalaja suhtluseks ja voogudeks, samuti ka diferentseeritud teenuste tarbeks.
- Täiendatud on autentimist ja privaatsust.

- IPv6 ei sisalda võrgu leviedastusaadressi. Selle asemel kasutatakse multiedastusaadressi "ff02::1"
- IPv6 on optimeeritud 64-bitise töötluse jaoks (IPv4 oli 32-bitise töötluse jaoks optimeeritud).
- Mitmeid laiendusi on modifitseeritud ja lisatud uusi. Üks laiendus on tohutusuurte pakettide toetamiseks (tuntakse kui *jumbogram*), mille suurus on kuni 4 GB (RFC 2675). IPv4 puhul oli paketi suuruse maksimumiks 64 KB. Nii suurte pakettide kasutamise võimaldamiseks on vajalikud modifikatsioonid transpordikihi protokollides (TCP, UDP). Selliste pakettide kasutamine võimaldab jõudluse kasvu väga kiiretes ja suure MTU-ga võrkudes (näiteks Etherneti puhul on katseid võimaldada suuremaid kaadreid).

ICMPv6

ICMPv6 (RFC 4443) on IPv6 sisemine osa, nagu IPv4 puhulgi ning omab samasugust rolli (teadete formaat ja funktsioonid). Vaatamata sellele ei ole ICMPv4 ja ICMPv6 omavahel ühilduvad. ICMPv4 ja ICMPv6 olulisemate erinevustena võiks välja tuua:

- Eemaldatud on mitmeid funktsioone, mida IPv6 enam ei vaja. ICMPv6 ei sisalda võrreldes ICMPv4-ga enam järgmisi teadetetüüpe: allika summutamine (*source quench*), ajatempli päring (*timestamp request*), ajatempli vastus (*timestamp reply*), informatsiooni päring (*information request*), informatsiooni vastus (*information reply*), alamvõrgumaski päring (*subnet mask request*), alamvõrgumaski vastus (*subnet mask reply*).
- Lisatud on mitmed uued teated.
- Mõned väljad on IPv6 jaoks suuremad.
- ICMPv6 puhul on piiratud ka saatmise sagedus. Näiteks kui mingi viga (näiteks sihtpunkt kättesaamatu) kordub väga sageli (näiteks sekundis sada korda), siis ei olda kohustatud vastama igale vea tekkimise juhule. Sellega lisatakse turvalisust teenusetõkestusrünnete vastu.

ICMPv6 paketi üldine sisu:

- Tüüp. Määrab ICMPv6 teate tüübi. 1 bait.
- Kood. Määrab ICMPv6 teate tüübi spetsiifilise koodi. 1 bait.
- Kontrollsumma. Erinevalt IPv4 ICMP kontrollsummast arvutatakse ICMPv6 kontrollsumma ka üle pseudopäise (nagu TCP ja UDP puhul). 2 baiti.
- Protokollispetsiifiline. Olenevalt teatest võib antud väli olla ka kasutamata. Kasutatakse näiteks teates "parameetri probleem", kus viidatakse kohale, kus probleem esines. Kaja päringu ja kaja vastuse teated on natuke erinevad. Nimelt neljanda välja kaks esimest baiti on päringu identifikaator ja kaks ülejäänud baiti on järjekorranumber. 4 baiti.
- Andmed. Vajadusel kopeeritakse algse paketi andmed enamasti kuni 1280 baidini (näiteks teadete "sihtpunkt kättesaamatu", "pakett liiga suur", "eluaja ületamine" puhul). Lähetaja saab nii lihtsamalt identifitseerida, milline pakett ei jõudnud sihtpunkti (näiteks loeb sealt välja algse pordi). Kaja päringu teate andmeväljale pannakse null või rohkem baiti suvalisi andmeid, mis kaja vastuse teatesse kopeeritakse (vajalik pingimise paketi suuruse määramiseks).

ICMPv4 ja ICMPv6 tüüpide numbrid ei lange kokku. ICMPv6 tüüpide registreerimised on aadressil http://www.iana.org/assignments/icmpv6-parameters. ICMP veateadete tüübi väärtused on 0 kuni 127. Informeerivad teated algavad tüübi numbrist 128. Uued ICMPv6 teadete tüübid võib jagada kolme osasse: MTU avastamine, multiedastuse teated ja naabrite avastamise teated. Multiedastuse ICMPv6 teateid vaatleme alampeatükis "IPv6 multiedastus" (lk. 111).

"Pakett liiga suur" (2) on uus ICMP teate tüüp. Kasutatakse suhtlevate hostide vahelise raja MTU ületamise probleemist teatamiseks (pakett oli liiga suur).

IPv6 puhul on lisandunud naabrite avastamise protokoll (RFC 2461), mille ülesanneteks on:

- Võimaldada masinatel leida marsruuter.
- Võimaldada hostidel kindlaks teha üksteise kanalikihi aadresse.

- Võimaldada hostidel avastada teisi hoste.
- Võimaldada hostidel korras hoida kättesaadavuse informatsiooni.
- Hankida hostidele radade staatuse infot aktiivsete naabriteni.

Naabrite avastamise protokoll kasutab ICMPv6 teateid, milledeks on (sulgudes olev number näitab tüübi numbrit):

- Marsruuteri leidmise palve (*router solicitation*) (133). Saadetakse hostide poolt, et saada vastuseks marsruuteri kuulutuse teadet.
- Marsruuteri kuulutus (*router advertisement*) (134). Saadetakse marsruuteri poolt kohtvõrku. Sisaldab vajalikku infot võrgu kohta, peamiselt võrgu prefiks automaatse seadistamise jaoks (IP-aadressi 64 esimest bitti). Marsruuteri kuulutus saadetakse võrku perioodiliselt ja vastusena marsruuteri leidmise palve teatele.
- Naabri leidmise palve (*neighbor solicitation*) (135). Saadetakse hostide poolt saamaks teada naabri kanalikihi aadressi. Leiab kasutamist ka duplikaataadressi ja vahemälus oleva kanalikihi aadressi kontrolliks.
- Naabri kuulutus (*neighbor advertisement*) (136). Saadetakse vastusena naabri leidmise palve teatele, sisaldades küsitud infot.
- Ümbersuunamine (*redirect*) (137). Ainuke naabrite avastamise ICMP teadetest, mis oli olemas ka ICMPv4 protokollis. Marsruuter annab teada hosti, kui leidub parem esimese hüppe sõlm kui tema ise.

ICMPv6 protokollis on säilinud (võrreldes ICMPv4-ga) mitmeid teadete tüüpe, millest mõningad kasutatavamad on järgmised (sulgudes ICMPv6 tüübi identifikaator): sihtpunkt kättesaamatu (1); eluaja ületamine (3); parameetri probleem (4); kaja päring (128); kaja vastus (129).

IP-aadressi saamine ja naabrite avastamise protokoll

IPv6 võrgus on peale staatilise IP-aadressi seadistuse võimalik saada automaatselt kahte moodi konfiguratsioon: **olekuta automaatseadistus** (RFC 4861, 4862) ja **olekuga seadistus**.

Olekuta (ka olekuvaba) automaatseadistuse puhul määratakse masina käivitamise käigus kanalikihi aadress (Etherneti puhul MAC-aadress), mille järel saadetakse võrku ICMPv6 "marsruuteri leidmise palve" teade. Kui marsruuterit ei ole võrgus, siis seadistatakse endale lokaalse sideliini IP-aadress (kanalikihi aadressi baasil), mis jääb ka ainukeseks masina IP-aadressiks. Kui võrgus leidub marsruuter, siis vastab ta koheselt ICMPv6 teatega "marsruuteri kuulutus" (*router advertisement*). Sellega saab masin omale IP-aadressi prefiksi, mille kombineerib oma võrguliidese aadressiga. Marsruuter saadab ka perioodiliselt võrku ICMPv6 teateid "marsruuteri kuulutus".

Teatega "marsruuteri kuulutus" antakse peamise infona teada võrgu prefiks (64 esimest bitti IP-aadressist). Host ise lisab ülejäänud 64 bitti. Etherneti puhul võetakse tavaliselt need 64 viimast bitti MAC-aadressist, millele pannakse keskele "FF:FE" (MAC-aadress on ainult 48 bitti). Näiteks olgu MAC-aadress "00-16-D4-57-E2-88", siis IPv6-aadressi teine pool oleks "16:D4FF:FE57:E288". Kuna MAC-aadressid on unikaalsed, siis on hosti IP-aadressi teine pool igas võrgus samasugune. Nõnda ei vajata konfigureerimist ega DHCP-serverit. Paraku vähendab MAC-i järgi IPv6 aadressi saamine privaatsust (masinat on võimalik identifitseerida olenemata sellest, kus võrgus ollakse). Seetõttu kasutatakse ka IPv6 viimase poole bittide saamiseks juhuslikku genereerimist. Kui võrgus on mitu marsruuterit ja nad saadavad marsruuteri kuulutusi, millel on erinevad võrguaadressid, siis host loob igaühe jaoks eraldi IPv6 aadressi (hostil võib olla korraga mitu IP aadressi). Marsruuteri leidmise palvet võidakse teostada kiiruse huvides ka paralleelselt lokaalse sideliini aadressi unikaalsuse kontrollimisega.

Marsruuteri leidmise palve (*router solicitation*) ICMPv6 teate väljad (paketi lähteaadressiks lokaalse sideliini aadress või ::0, sihtaadressiks multiedastuse aadress kõikidele marsruuteritele):

- ICMPv6 standardalgus (tüüp=133, kood=0, kontrollsumma). 4 baiti.
- Reserveeritud. 4 baiti.

- Valikud. Sisaldab lähtemasina kanalikihi aadressi (kui teatakse). Tulevikus võib lisanduda muid valikuid. Muutuva pikkusega.

Marsruuteri kuulutuse (*router advertisement*) ICMPv6 teate väljad (paketi lähteaadressiks pannakse marsruuteri enda lokaalse sideliini aadress, sihtaadressiks pannakse multiedastusaadress "kõikidele sõlmedele" või lokaalse sideliini aadress päringu vastuse puhul ning kui marsruuteri leidmise palve ICMPv6 teates sisaldus lähteaadress):

- ICMPv6 standardalgus (tüüp=134, kood=0, kontrollsumma). 4 baiti.
- Maksimaalne hüpete limiit. Määrab IPv6 päise hüpete limiidi maksimaalse väärtuse. 16 bitti.
- M (*managed address configuration*) lipp. Kui lipp on püsti, siis näidatakse, et aadressid on kättesaadavad DHCPv6 kaudu.
- O (*other configuration*). Kui lipp on püsti, siis on muu seadistus saadaval DHCPv6 kaudu. Kui "M" lipp on püsti, siis "O" lipp muutub tähtsusetuks.
- Reserveeritud. 6 bitti.
- Marsruuteri eluiga. Näitab, mitu sekundit on kehtiv marsruuter vaikevõrguväravana töötanud. Kui marsruuter ei ole vaikevõrguväravana kasutatav, siis välja väärtuseks on null. Kehtivusaeg kuni 18,4 tundi. 16 bitti.
- Kättesaadavuse aeg. Alates viimasest kättesaadavuse kinnitusest möödunud aeg millisekundites, mille möödudes loetakse naaber mittekättesaadavaks. Välja väärtus null tähistab seda, et marsruuter on jätnud välja väärtuse määramata. Välja väärtust kasutab naabri mittekättesaadavuse avastamise algoritm. 32 bitti.
- Taassaatmise aeg. Aeg millisekundites, mille möödudes "naabri leidmise palve" ICMPv6 teade saadetakse uuesti. Välja väärtus null tähistab, et marsruuter on jätnud välja väärtuse määramata. Seda välja kasutavad aadressi lahendamise ja naabri mittekättesaadavuse avastamise algoritmid. 32 bitti.
- Valikud. Võimalikud valikud: marsruuteri kuulutuse saatja kanalikihi aadress; MTU väärtus varieeruva MTU suurusega sideliinide puhul; võrgu prefiks(id).

Olekuga seadistuse puhul kasutatakse IP-aadressi määramiseks võrgus spetsiaalset serverit, milleks on tavaliselt DHCPv6 server, mis jagab soovijatele IP-aadresse ja muud vajalikku informatsiooni sarnaselt IPv4 DHCP-serveriga.

Olekuta automaatseadistuse positiivseid külgi:

+ Ei vajata serverit.
+ Hosti seadistamine ei ole vajalik.
+ Minimaalne või üldse ebavajalik marsruuteri seadistamine.

Olekuta automaatseadistuse negatiivseid külgi:

- Kesised seadistuse parameetrid. Klient ei saa muid tavaliselt vajalikke seadistusparameetreid, näiteks nimeserverite IP-aadressid.
- Vähene turvalisus. Ei ole võimalik autentida seadistust küsivat klienti.

Olekuga seadistuse positiivsed külgi:

+ Võimalik anda küsivale süsteemile lisaparameetreid.
+ Lihtne autentimise võimalus, millega on võimalik otsustada, milline süsteem saab seadistuse andmed.

Olekuga seadistuse negatiivseid külgi:

- Vajalik on serveri olemasolu (võib asuda ka marsruuteris).
- Vajalik administraatori olemasolu.

Mõlema automaatse seadistuse puhul kasutatakse duplikaataadressi avastamise algoritmi, mis tagab aadressi unikaalsuse. Kui leitakse, et aadress on unikaalne, siis seatakse see liidese IP-aadressiks. Olekuta automaatseadistust ja olekuga seadistust on võimalik kasutada ka koos. Alguses võidakse IP-aadress saada olekuta automaatseadistuse abil ning seejärel lisaseadistusparameetrid olekuga seadistuse abil.

Naabrite avastamise protokolli alla kuulub ka sisevõrgu IP-aadressile vastava kanalikihi aadressi leidmine. IPv6 puhul kasutatakse selleks ICMPv6 "naabri leidmise palve" ja "naabri kuulutus" teateid (IPv4 puhul kasutati ARP-päringut ja vastust). IPv6 puhul on kanalikihi aadressi saamine kanalikihi protokollist sõltumatu (IPv4 puhul töötas ARP-protokoll ainult Ethernet ja FDDI võrgus). Näiteks on võimalik kasutada Etherneti asemel ka ATM võrku. Kanalikihi aadressi saamise mehhanism töötab IPv6 võrgus sarnaselt IPv4 ARP-protokollile. Kui vajatakse kohaliku võrgu IP-aadressile vastavat kanalikihi aadressi, siis saadetakse välja "naabri leidmise palve" ICMPv6 teade, millele siht-host vastab "naabri kuulutus" ICMPv6 teatega.

IPv6 naabrite kohta hoitakse kättesaadavuse infot. Kättesaadavuse olekud:

- Poolik (*incomplete*) – kanalikihi aadress on kindlaks määramata, sest aadressi lahendamise protsess on käimas.
- Kättesaadav (*reachable*) – naaber teatakse olevat kättesaadav vähem kui kümme sekundit tagasi.
- Seismajäänud (*stale*) – ei ole kindel, kas naaber on kättesaadav, kuid pakette saadetakse vastavale hostile. Kättesaadavust ei kontrollita.
- Aegumise viivitus (*delay*) – ei ole kindel, kas naaber on kättesaadav, hiljuti on saadetud vastavale masinale liiklust. Selle asemel, et koheselt naabrit sondeerida, viivitatakse mõnda aega sondeerimisega, et ülemise kihi protokollidele anda võimalus hankida kättesaadavuse kinnitus.
- Sondeeritav (*probe*) – ei ole kindel, kas naaber on kättesaadav. Naabrile saadetakse üksikedastusena "naabri leidmise palve" ICMPv6 teade, kontrollimaks kättesaadavust.

Naabri leidmise palve väljad on järgmised:

- ICMPv6 standardalgus (tüüp=135, kood=0, kontrollsumma). 4 baiti.
- Reserveeritud. 4 baiti.
- Sihtaadress (ei saa olla multiedastusaadress). 16 baiti.
- Valikud. Sisaldab lähtesõlme kanalikihi aadressi. Tulevikus võib täieneda. Muutuva suurusega.

Naabri kuulutuse ICMPv6 teate väljad on järgmised:

- ICMPv6 standardalgus (tüüp=136, kood=0, kontrollsumma). 4 baiti.
- R (*router*) lipp. Kui lipp püsti, siis saatja on marsruuter.
- S (*solicited*) lipp. Kui lipp püsti, siis tegemist naabri leidmise palve teatele vastusega. Kasutatakse naabri mittekättesaadavuse avastamise protsessis kinnitamaks kättesaadavust.
- O (*override*) lipp. Kui lipp püsti, siis asendatakse olemasolev vahemälu kirje ja uuendatakse kanalikihi aadressi.
- Reserveeritud. 29 bitti.
- Sihtaadress. Kui tegemist päringu vastusega, siis sisaldab naabri leidmise palve teate sihtaadressi. Vastasel korral kanalikihi aadressi muutusi. 128 bitti.
- Valikud. Sisaldab saatja kanalikihi aadressi.

IPv6 multiedastus

IPv6 puhul on multiedastus laialdasemalt kasutusele võetud. Protseduurid ühinemaks multiedastusgruppidega on samasugused nagu IPv4 puhul, kuid erinevalt IPv4 multiedastusest ei kasutata enam IGMP teateid. IGMP-le ekvivalentsed kolme gruppi kuuluvad teated on lisatud ICMPv6 teadeteks.

Multiedastuse jaoks on eraldatud prefiks ff00::/8 (RFC 4291). Multiedastusaadressi formaat on järgmine:

- Multiedastusaadressiploki prefiks ("1111 1111"). 8 bitti.
- Lipud. 4 bitti.
- Skoop. Multiedastused jaotatakse erinevatesse skoopidesse leviku osas. Skoobi abil teab marsruuter, kas edastada paketti või mitte. 4 bitti.

- Grupi identifikaator. 112 bitti.

Multiedastusaadresse (RFC 2375, 3306) jaotatakse:

- Lokaalsete sõlmede skoop:
  - ◇ FF01::1 – kõikide sõlmede aadress.
  - ◇ FF01::2 – kõikide marsruuterite aadress.
- Lokaalse sideliini skoop:
  - ◇ FF02::1 – kõikide sõlmede aadress.
  - ◇ FF02::2 – kõikide marsruuterite aadress.
  - ◇ FF02::5, FF02::6 – kõik OSPF marsruuterid (vastavalt kõik ja määratud OSPF marsruuterid).
  - ◇ FF02::1:2, FF02::1:3, FF02::1:4 – vastavalt kõik DHCP agendid, serverid ja releed.
- Lokaalse saidi skoop:
  - ◇ FF05::2 – kõikide marsruuterite aadress.
  - ◇ FF05::1:1000-FF05::1:13FF – *Service Location Protocol* (RFC 2165) jaoks (võrguteenuse avastamine ja valik, millega ei pea vajatava teenuse hosti nime teadma).

Multiedastuse ICMPv6 teated on järgmised (sulgudes olev number näitab ICMPv6 tüübinumbrit):

- Grupi liikmelisuse päring (*Group Membership Query*) (130). Marsruuterid saadavad antud teateid, et määrata, millised kohalikud masinad on vastava grupi liikmeks.
- Grupi liikmelisuse raport (*Group Membership Report*) (131). Saadavad masinad näitamaks vastavasse gruppi kuulumist. Antud teade saadetakse grupi liikmelisuse päringu vastusena.
- Grupi liikmelisuse eemaldamine (*Group Membership Reduction*) (132). Saadavad masinad, et lõpetada vastava multiedastuse grupi liikmelisust.

Multiedastuse haldamise ICMPv6 teate väljad on järgmised:

- Tüüp, kood, kontrollsumma – kolm esimest välja on kõikidel ICMPv6 teadetel samasugused. Kood on multiedastuse haldamise ICMPv6 teadetel seatud nulliks.
- Maksimaalne reaktsiooni aeg. Kasutatakse grupi liikmelisuse päringus näitamaks aega, kui kaua võib teatega viivitada. Grupi liikmelisuse raporti ja eemaldamise teate puhul seatud nulliks. 2 baiti.
- Kasutamata. Seatakse nullid. 2 baiti.
- Multiedastuse aadress.

IPv6 marsruutimisprotokollid

IPv6 jaoks on vaja ka uusi või kohandatud marsruutimisprotokolle (ainuüksi aadressi pikkuse tõttu). Suurimad muudatused võrreldes IPv4 marsruutimisprotokollidega on vajalikud EGP marsruutimisprotokollide osas (BGP4 jaoks), kuna IPv6 sisaldab võrreldes IPv4-ga mitmeid erinevaid aspekte. Seetõttu on välja töötatud uus marsruutimisprotokoll IDRP (*Interdomain Routing Protocol*), mis baseerub mitmetel BGP põhimõtetel. Siiski on arendatud ka IPv6-ga ühilduvaid BGP versioone (RFC 2858). IDRP eelisteks BGPv4 ees võiks lugeda:

- IDRP kasutab hierarhilist marsruutimise võimalust, mida pakub IPv6 aadressiruum (BGPv4 identifitseerib kõik vahepealsed autonoomsed süsteemid).
- IDRP kasutab teadete edastamiseks datagramme (BGPv4 kasutab TCP-d), mistõttu IDRP on väiksema ballastiga.
- IDRP võimaldab opereerida erinevate aadressitüüpidega (BGP4 ainult 32-bitiste IP-aadressidega).
- IDRP kasutab muutuva suurusega välja aadressidomeenide jaoks (BGPv4 fikseeritud pikkusega).

IGP marsruutimisprotokollide (nt. RIP, OSPF) osas on muudatused väiksemad ja seetõttu on täiendatud olemasolevaid marsruutimisprotokolle, et lisada IPv6 toetus.

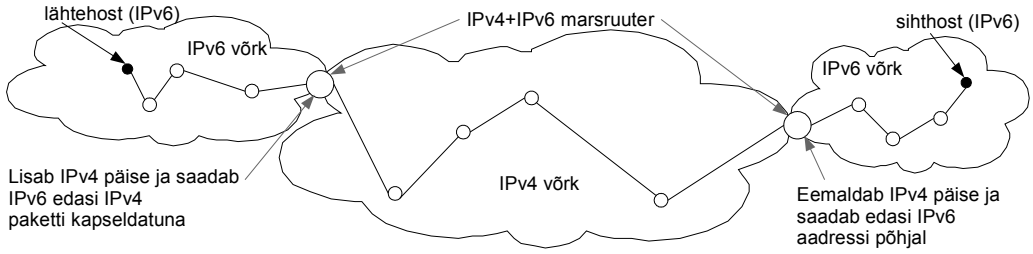
IPv6 kasutuselevõtt

Reeglina on uue protokolli asendamine seotud teatavate raskuste ja maksumusega, eriti nii olulise protokolli puhul nagu IP, mis on Interneti alusprotokolliks. Seetõttu ei saa üleminek toimuda üleöö. Üleminek IPv6 protokollile saab toimuda ja toimub järkjärgult, võttes IPv4 kõrval kasutusele IPv6 protokolli. Nad peavad teatud aja koos eksisteerima, kuni enamus Internetist on IPv6 põhine ja IPv6 saab primaarseks.

Tänapäeval peab arvestama, et Internet on väga oluline, selle häired on kallid ja tihti ka vastuvõetamatud, sest asutuste ja inimeste töö on väga suures sõltuvuses Internetist. Üleminek peab toimuma võimalikult odavalt, lihtsalt ja väheste eeltingimustega.

Osad asutused on oma võrgud viinud IPv6 toetavaks (tavaliselt paralleelselt IPv4-ga), paraku ei ole kõik võrgud IPv6 toetusega. Selliselt tekivad IPv6 saarekesed. Saamaks saarte vahel suhelda, kasutatakse tunneldamist. Nimelt paigutavad saarte äärmised IPv6 marsruuterid IPv6 paketi IPv4 paketi sisse ning nad saadavad IPv6 paketi selliselt ainult üle IPv4 toetava võrgu, kus teisel pool võetakse IPv4 paketist IPv6 pakett välja ja saadetakse edasi IPv6 paketina. Näide tunneldatud ühendusest on joonisel 13.3.

Tunnelite ülesseadmiseks on olemas põhiliselt kaks moodust: käsitsi või 6to4 automaatne tunneldamine. 6to4-ga saab host või marsruuter luua IPv6 aadresside vahemiku IPv4 aadressidest. 6to4 aadresside esimene plokk on "2002". Selliselt on 6to4 mõistvatel süsteemidel võimalik tunneldada IPv6 pakette õigesse kohta üle IPv4. 6to4 ei vaja seadistamist. 6to4-l on ka omad probleemid, nimelt ei saa 6to4 kasutada koos NAT-ga. Samuti on 6to4 sõltuvuses avalikest lüüsidest (*gateway*), mis teeb side aeglasemaks ja vähem töökindlaks, võrreldes teiste võimalustega. IPv6 tunneldamiseks on võimalus kasutada ka "*tunnel broker*" (RFC 3053) teenust. *Tunnel broker* puhul on IPv4 protokollis protokolli välja väärtuseks 41.



Joonis 13.3: IPv6 tunneli kasutamine üle IPv4 võrgu.

Kui IPv6 ja IPv4 toetusega host küsib DNS-serverilt domeeninime, siis nad küsivad nii "A" kui ka "AAAA" kirjet. Kui ei ole "AAAA" kirjet või ei saada ühendust, siis ühendutakse IPv4-aadressi kaudu. Seetõttu võib IPv6 kasutaval hostil esineda pikemaid viivitusi ühenduste loomisel.

IETF töötab edasi mobiilsuse (*mobility*) (RFC 3775, 4877) ja multikodu (*multihoming*) (RFC 3582, 3776, 4218) laienduste kallal IPv6-le. Mobiilsuse laiendus võimaldab liikuda ühest võrgust teise, hoides sama IP-aadressi. Selliselt on võimalik näiteks VoIP kõnet alustada kodus, jätkata traadita ühenduse teenusega ja lõpetada kõne töö juures (kõne kestel liigutakse mitme võrgu vahel). Multikodu laiendus võimaldab olla ühendatud samal ajal rohkem kui ühe ISP-ga. Kui ühe ISP-ga midagi juhtub, siis ühendused suunatakse automaatselt teise ISP-i alla.

14. Kommutaatorite laiendusi

OSI kanalikihi juures vaatlesime kommutaatorite põhifunktsioone, milledeks on kaadrite edastamine porti, kuhu nad on suunatud, ja teha väiksemaks või kaotada kollisioonidomeenid. Suuremates asutustes on suuremad nõudmised tõrkekindluse, kiiruse, hallatavuse jms. osas, mistõttu ainult põhifunktsioonidega kommutaatorid ei rahulda kõiki vajadusi. Järgnevalt vaatleme mõningaid levinumaid kommutaatorite puhul kasutatavaid võimalusi. Enamasti on laiendatud funktsioonidega kommutaatorid kallimad.

14.1 STP

Asutuse kohtvõrkude laienemisel kasvab kohtvõrkude tõrkekindluse olulisus (remondi käigus lõhutakse sideliin, liides võib üles öelda, kommutaator rivist välja minna, kommutaatorid ühendatakse omavahel tsükliliselt jne.). Võrgu rikked võivad halvata asutuse töö (arvutitega ei ole võimalik sooritada tavapäraseid tööoperatsioone, veebiserver ei ole kättesaadav jne.). Sellises olukorras tekib vajadus kasutada dubleeritud ühendusi, selleks et vähendada võrguprobleemide esilekerkimisega seotud kahju. Marsruuterite puhul otseselt sellist probleemi ei ole, sest nad saavad paketi saatmiseks kasutada mitut teed. Tavaliste kommutaatorite puhul ei olnud võimalik dubleeritud ühendusi luua, sest tekiksid tsüklid, mistõttu näiteks leviedastusaadressiga kaader jääks igavesti tsükliliselt liikuma ning võiks veel omakorda paljuneda. Et kommutaatorite vahel saaks dubleeritud ühendusi kasutada, on välja töötatud protokoll STP (*Spanning-Tree Protocol*) (IEEE 802.1D).

STP põhineb toesepuu (*spanning-tree*) algoritmil. STP koostab tsüklilisest graafist (tippudeks kommutaatorid, lehtedeks üldjuhul STP-protokolli mittekasutavad masinad ja kaarteks nende vahelised ühendused) puu. Puuks teisendades jäetakse mingite kriteeriumite alusel osad sideliinid, mis tekitaksid tsükli, tagavaraks. Tagavaraks olevaid sideliine ei kasutata edastuseks. Nad võetakse kasutusele, kui mingi töötav sideliin langeb maha ning üks nendest tagavarasideliinidest on kõige parem asendaja. Võrgus, milles pole dubleeritud ühendusi, seega ka tsükleid, pole üldjuhul mõtet STP-protokolli kasutada, kuna STP-protokoll kasutab kommutaatori ressursse (liideste ribalaiust ja protsessoriaega). Mõnikord võidakse STP-d siiski kasutada, et minimeerida apsakate poolt tehtud kahju, näiteks kaablite valesti ühendamisel, mille tulemusena tekiks tsükkel.

Igal STP kommutaatoril on 8 baidiline BID (*bridge identification*) number, mille kaks esimest baiti on prioriteedi number ja ülejäänud 6 baiti kommutaatori enda MAC-aadress. Prioriteedi number on administraatori poolt muudetav.

STP kommutaatorid kasutavad omavaheliseks suhtluseks kaadrit, mida nimetakse BPDU-ks (*bridge protocol data unit*). BPDU sisaldab järgmist informatsiooni: juurkommutaatori BID; kui kaugel on juurkommutaator; mis on saatja BID; port, mille kaudu BPDU välja saadeti.

STP võimelised kommutaatorid valivad esimese operatsioonina (nt. kommutaatori käivitumisel) endi seast välja juurkommutaatori (*root bridge*). Kommutaator, mis sisse lülitatakse, arvab automaatselt, et ta on juurkommutaator ja saadab välja BPDU, kus juurkommutaatori BID on kommutaatori enda BID ja juurkommutaatori kauguseks on null. Juurkommutaatoriks saab kommutaator, millel on kõige väiksem BID-i väärtus (kui prioriteedid on võrdsed, siis määrab juurkommutaatoriks saamise MAC-aadress, mis teatavasti on unikaalne). Kui kommutaator saab BPDU, mille BID on väiksem temale teada olevast, siis ta asendab omal juurkommutaatori BID väärtuse. Iga kommutaator, mis saadab BPDU, paneb juurkommutaatori väärtuseks väikseima BID-i, mida ta teab. Vaikimisi saadetakse BPDU-sid välja iga kahe sekundi tagant.

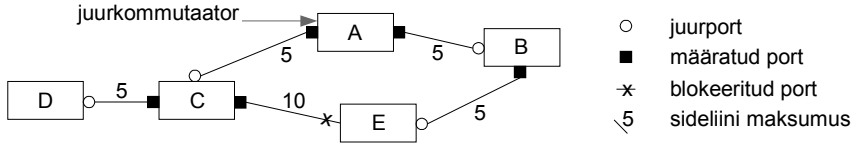
Järgmise etapina pärast juurkommutaatori valimist, hakkavad kõik mitte-juurkommutaatorid leidma oma kõikide portide kaudu teekonna maksumust juurkommutaatorini (kaugus võib põhineda näiteks juurkommutaatorini ühendavate sideliinide ribalaiustele määratud maksumuste

summal), kasutades vastavat BPDU kaadrite saatmist ja BPDU-de kuulamist. Kui ollakse leidnud kõik teed, mis viivad juurkommutaatorini, siis valitakse välja kõige väiksema kaugusega port ja ülejäänud pordid blokeeritakse STP poolt (neid nimetatakse blokeeritud portideks). Seda väljavalitud porti nimetatakse **juurpordiks** (*root port*). Ülejäänud porte, mis ei ühendunud juurkommutaatoriga, nimetatakse **määratud portideks** (*designated port*).

STP kommutaatori portidel on viis erinevat seisundit, mida läbitakse järgnevalt:

1. Mittefunktsioneeriv seisund. Port ei ole ühenduses mingi muu seadmega või on administratiivselt keelatud.
2. Blokeeriv seisund (*blocking*). Võetakse vastu ainult BPDU kaadreid info kogumiseks. Kestus 20 sekundit.
3. Kuulamise seisund (*listening*). Siin kuulatakse, kas on olemas tee juurkommutaatorini. Kui port oli juurkommutaatoripoolne, kuid tee oli kehvem parimast teadaolevast teest juurkommutaatorini, siis läheb port tagasi blokeerivasse seisundisse. Tegeletakse ainult BPDU-dega. Kestus 15 sekundit.
4. Õppimise seisund (*learning*). Hakatakse õppima saabuva liikluse pealt MAC-aadresse, kuigi kaadreid edasi ei saadeta. Samuti töödeldakse BPDU-sid. Kestus 15 sekundit.
5. Edastav seisund (*forwarding*). See on pordi normaalne seisund, kus õpitakse MAC-aadresse ja saadetakse edasi kaadreid.

Vaikimisi on aegumised arvestatud nii, et juurkommutaatorist väljub ahel, mis koosneb seitsmest kommutaatorist (väärtusi saab tavaliselt vajadusel ümber seadistada). Igast seisundist on võimalik, et port läheb mittefunktsioneerivasse seisundisse, kuid mittefunktsioneerivast seisundist edastavasse seisundisse minekuks peab läbima kõik vahepealsed etapid. Võrguadministraatoril on soovitatav määrata käsitsi juurkommutaatoriks sobiv kommutaator, seadistades vastava kommutaatori prioriteedinumbri väikseimaks. Olenedes olukorrast ja topoloogiast võib see anda suurema jõudluse. BPDU-sid saadetakse lühikeste intervallidega, et juhul, kui mingi aktiivne rada või kommutaator peaks rikki minema, saaks koostada uue toesepuu. Joonisel 14.1 on näide kommuteeritud võrgust, kus kommutaatoritel kasutatakse STP-protokolli ja mis on määranud portidele sobivad rollid.



Joonis 14.1: STP poolt tehtud puu-kujuline topoloogia. Juurkommutaatoriks on kommutaator A. Kommutaatorist E lähim port juurkommutaatorini on läbi kommutaatori B, mistõttu E blokeerib kommutaatorisse C viiva pordi, kuna see on kehvema meetrikaga.

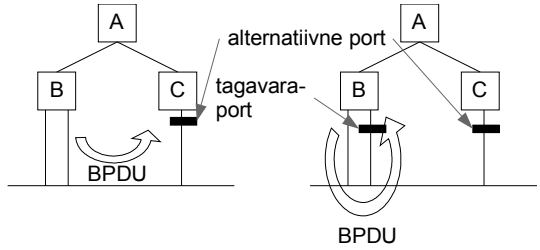
Kui topoloogias on muudatusi liiga tihti või see on lihtsalt ebastabiilne (näiteks mingi port käib ise maha ja püsti tehniliste probleemide tõttu), siis see võib võrguliikluse mingiks ajaks katkestada (vaikimisi maksimaalselt kokku 50 sekundiks (20+15+15)). Algselt ei olnud 30-50 sekundiline maasoleku aeg enamasti probleemiks, kuid tänapäeval paljusid see enam ei rahulda. Seetõttu on tehtud edasiarendusi ja laiendusi. Laiemalt kasutatavad on RSTP (*Rapid Spanning Tree Protocol*) ja MSTP (*Multiple Spanning Tree Protocol*).

RSTP

RSTP (IEEE 802.1w) on STP edasiarendus. Võrreldes STP-ga on RSTP oluliselt kiirem topoloogiamuudatustega kaasaskäimisel. Selleks on vähendatud portide seisundeid, lisatud uusi pordi tüüpe, võetud kasutusele uus BPDU formaat (STP ignoreerib RSTP BPDU-sid), erinev on konvergentsi saavutamise mehhanism jne.

Mittefunktsioneeriv, blokeeriv ja kuulamise seisund nimetati ümber kõrvalejäetud (*discarding*) seisundiks. Algsete pordi rollide (juur-, määratud ja blokeeritud port) puhul jagati blokeeritud pordi roll kaheks (vt. ka joonist 14.2):

- Alternatiivne port – võtab üle juurpordi rolli, kui algne juurport maha kukub.
- Tagavaraport – võtab üle määratud pordi rolli, kui algne määratud port maha kukub.



Joonis 14.2: RSTP alternatiivse ja tagavarapordi rollide jaotus.

Pordid jaotatakse omaduste järgi veel sideliini tüübi ja ääretüübi alusel. Sideliini tüübid seadistatakse automaatselt dupleksi alusel: täisdupleksside liin on punktist-punkti (*point-to-point*) ja poolduplekssideliin on jagatud tüüpi. Pordi ääre-tüüp sõltub sellest, kas antud pordist saadakse BPDU-sid või mitte. Kui pordist saadakse BPDU, siis on tegemist mitte-ääretüübiga, vastasel korral, kui ei saada BPDU-sid, on tegemist ääretüübiga.

14.2 Virtuaalsed kohtvõrgud ehk VLAN-id

Suureks kasvanud asutuse kohtvõrgu puhul võib mingist asjaolust tekkida vajadus jagada see väiksemateks kohtvõrkudeks. Suure kohtvõrgu puhul on palju leviedastust, multiedastust jms., mis jõuab kõigi kohtvõrgus olijateni ja koormab neid asjatult. Üks võimalus oleks jagada suur kohtvõrk asukoha alusel mitmeks väiksemaks kohtvõrguks. Paraku aga on tihti vaja ühendada erinevates kohtades (ruumides, korrustel või majades) masinad ühte kohtvõrku (näiteks osakondade, funktsioonide vmt. kaupa). Samas ei ole eriti mõistlik hakata mitut füüsiliselt kohalikku võrku ehitama (kallis, pole paindlik). Samuti oleks kulukas ja problemaatiline hoida eraldi asetsevate osakondade osasid erinevates väikestes kohtvõrkudes (arvestades seejuures turvaseadeid, vajalikke tunneleid jm. seadistamise vajadusi).

Probleemi ületamiseks on kommutaatoritele välja töötatud lahendus, millega on võimalik üks suur kohtvõrk jagada mitmeks virtuaalseks kohtvõrguks. See tähendab, et kõik kohtvõrgu masinad grupeeritakse mingi tunnuse alusel ühte või mitmesse sobivasse virtuaalsesse kohtvõrku. Virtuaalne kohtvõrk on kasutajate jaoks samasugune nagu tavaline kohtvõrk. Erinevatel virtuaalsetel kohtvõrkudel on erinevad võrguaadressid, erinevatesse virtuaalkohtvõrkudesse kuuluvad hostid ei saa omavahel suhelda. Erinevatesse virutaalkohtvõrkudesse kuuluvad hostid saavad suhelda ainult läbi marsruuteri isegi siis, kui nad on ühendatud sama kommutaatori kõrvuti olevatesse portidesse.

Masinate grupeerimiseks VLAN-de vahel on kasutatavad järgmised moodused:

- Kommutaatori pordi alusel. Iga port on märgitud mingisse VLAN-i. Masin kuulub sellesse VLAN-i, mille porti ta on ühendatud. Algselt on kõik pordid ühes VLAN-s.
- MAC-aadressi alusel. Eelnevalt määratakse võrguadministraatori poolt MAC-aadressid erinevatesse VLAN-desse kuuluvateks. Masin saab ühenduda suvalise kommutaatori porti ja ta on alati samas VLAN-s (võib kasutajale olla mugavam ja paindlikum kui pordipõhine grupeerimine). Probleemiks on väike jõudluse kadu. Võrguadministraatori jaoks on haldamine töömahukam ja veaohtlikum, lisaks turvaprobleemid (MAC-aadressi on võimalik muuta). Kasutatakse harva.

- Alamvõrgu järgi. Kommutaator grupeerib VLAN-desse võrgukihi aadressi alusel. Võrreldes MAC-aadressi alusel grupeerimisega on veel aeglasem. Praktiliselt ei kasutata.
- Protokolli alusel, näiteks: IP, IPX, LAT jne. järgi. Enam praktiliselt ei kasutata.

Edaspidi käsitleme portide alusel virtuaalseteks kohtvõrkudeks jaotamist. Seda viisi kasutatakse ka kõige enam. Pordi alusel kohtvõrkudesse jaotamisel peab võrguadministraator iga pordi määrama soovitavasse VLAN-i. VLAN-id ise identifitseeritakse naturaalarvudega. Kommutaator haldab sisuliselt iga VLAN-i kohta eraldi MAC tabelit. Seega, kui VLAN 1 kuuluv arvuti paneb sihtaadressiks VLAN 2 oleva MAC-aadressi, siis see kaader saadetakse laiali kõikidele VLAN 1 kuulujatele (sest VLAN 1 MAC tabelis seda aadressi ei olnud), kuid VLAN 2 arvutini antud kaader ei jõua.

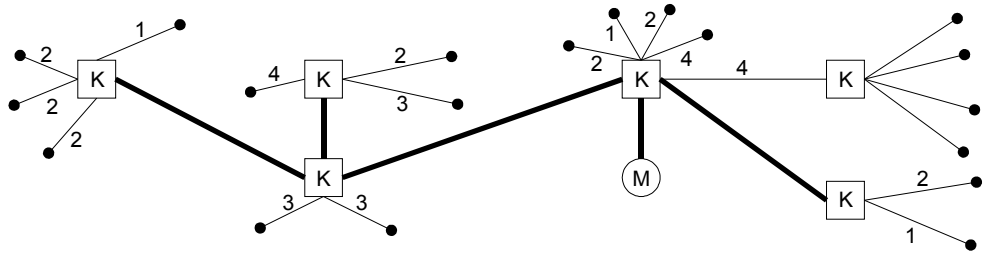
Virtuaalne kohtvõrk võib koosneda ka mitmest kommutaatorist, sest kommutaatori jaoks pole oluline, kas porti on ühendatud kommutaator, arvuti või mingi muu masin. Seega võivad kommutaatoritele ehitatud virtuaalsed kohtvõrgud olla suured, paljudest kommutaatoritest koosnevad. Oluline on seadistada kommutaatoreid ühendavad pordid samadesse VLAN-desse.

Standardsel juhul peaks kommutaatorite vahel olema iga VLAN-i kohta ühendatud eraldi kaabel. Samuti on VLAN-de ühendamiseks kasutataval marsruuteril vaja iga VLAN-i kohta eraldi porti. Aga, kui võrk on suur ja seal on palju virtuaalseid kohtvõrkusid, tekib probleem, et raisatakse liialt kommutaatorite ja marsruuteri(te) porte. Ühendused võivad olla ka kulukad (nt. majadevahelised fiiberoptikaga ühendused). Selle probleemi jaoks on võimalik kasutada nn. *trunk* sideliini, mille kaudu saab edastada erinevatesse VLAN-idesse kuuluvaid kaadreid. *Trunk* sideliini kasutamisel on vajalik kaadritega kaasa anda VLAN-i kuuluvus. Iga kaader, mis *trunk* sideliini läbib, märgistatakse tema VLAN-i identifikaatoriga, mille järgi vastuvõttev kommutaator teab, kas ja kuhu kaadrit edasi suunata ehk millisesse VLAN-i antud kaader kuulub. *Trunk* sideliin tuleb kommutaatorites seadistada (mõlemas kommutaatoris seadistada soovitav port *trunk*-laadi), kuid ei ole vaja seadistada ülekantavaid VLAN-e. *Trunk* sideliinile pole tähtis, milliseid VLAN-i kaadreid ta üle kannab (vajadusel võib turvakaalutlustel kasutada eraldi filtreid, et lubada edastada ainult soovitud VLAN-de kaadreid).

Kaadrite märgistamise protokollina on kasutusel järgmised protokollid: IEEE 802.1Q, ISL (*Inter-Switch Link*) (Cisco firmaomane), DISL (*Dynamic Inter-Switch Link Protocol*) (Cisco firmaomane). Eelpool nimetatud protokolle nimetatakse *trunking* protokollideks. Kaadri märgistamine ei ole kommutaatoreid kuigi palju koormav lisaoperatsioon. Kui teha üks suur kohtvõrk, siis kommutaatorite MAC-aadresside tabelid muutuvad suurteks. VLAN-de puhul ei pruugi kõik virtuaalsed kohtvõrgud olla seotud kõikide kommutaatoritega. Näiteks võib olla, et mingi virtuaalkohtvõrk on esindatud ainult osades kommutaatorites. Näiteks kaheksast kommutaatorist kolmes on vastav VLAN esindatud. Sellisel juhul ülejäänud viis kommutaatorit ei pea vaadeldava grupi hostide MAC-aadressidega tegelema (kaadreid edastama) ega nende MAC-aadresse tabelis hoidma.

Tavaliselt ühendatakse virtuaalsed kohtvõrgud marsruuteriga samuti üle *trunk* sideliini. Võimalik on ka arvutini viia *trunk* sideliin, kuid arvuti võrgukaardil ja tema draiveritel peab olema vastav toetus (võidakse kasutada näiteks serverite ja monitoorimise jaoks).

Virtuaalsete kohtvõrkude kasutuselevõtt võib tublisti lihtsustada ja paindlikumaks muuta võrgu laienemist, muutmist, administreerimist, monitoorimist, turvamist jms. Vaatamata sellele võib suuremates võrkudes, mis on ehitatud üles virtuaalsetele kohtvõrkudele, olla haldamise probleeme. Võrguadministraator peab iga kommutaatorit eraldi konfigureerima, kui näiteks kasutaja liigub ühest kohast teise, või kui tuleb juurde uus kasutaja. Selle lihtsustamiseks on loodud ka tarkvara, mis võimaldab virtuaalseid kohtvõrke tsentraalselt hallata ning anda võrguadministraatorile ülevaade. Näiteks Cisco on välja töötanud oma kommutaatoritele protokolli VTP (*VLAN Trunking Protocol*), mis aitab hulga tööd automatiseerida. Joonisel 14.3 on toodud näide kohtvõrgust, mis on jagatud virtuaalseteks kohtvõrkudeks.



Joonis 14.3: Näide virtuaalseteks kohtvõrkudeks jagatud võrgust. Joonisel on neli VLAN-i, *trunk* sideliinid on tähistatud jämedate joontega ja tavalised Ethernet ühendused peenema joonega. Joonisel on ka üks kommutaator, mis ei toeta virtuaalseid kohtvõrke. Kõik hostid vastava kommutaatori taga kuuluvad VLAN-i number 4.

14.3 Kommutaatorite muid lisavõimalusi

Võrgu administreerimisel, liikluse analüüsil ja muudel puhkudel võib võrguadministraatoril tekkida vajadus uurida võrguliiklust. Hube kasutades oli see kerge, kuid kommutaatorite puhul ei saadeta kõiki kaadreid kõikidesse portidesse. Selle tõttu saab arvutist jälgida ainult leviedastust, multiedastust ja üksikuid muid kaadreid (saadeti MAC-aadressile, mida kommutaatori MAC-tabel ei sisaldanud). Selliste ja võibolla veel mingite muude vajaduste rahuldamiseks on võimalik osadel kommutaatoritel määrata port, kuhu saata kogu liikluse koopia (lisaks sihtpordile saadetakse kaader ka määratud porti). Sellist funktsiooni toetava kommutaatori vaadeldavat porti nimetatakse peegelpordiks (nimetatakse ka pordi peegeldus, SPAN (*switch port analyzer*) port, *spanning port*, *link mode port* ja *roving analysis* port). Peegelpordiga võib ühendada arvuti, mis võtab kogu läbiva liikluse vastu ja analüüsib või logib seda. Peegelpordi puhul on tihti pordist väljuv liiklus keelatud. Peegelporti kasutatakse ka IDS (*intrusion-detection system*) jaoks (IDS on tarkvara, mis püüab avastada võimalikke ründeid).

Osasid kommutaatoreid nimetatakse ka kolmanda kihi kommutaatoriteks. Kolmanda kihi kommutaatorid on sarnased marsruuteritele, kuid peamiseks erinevuseks on asjaolu, et marsruuteril on marsruutimine teostatud tarkvaraliselt, kuid kolmanda kihi kommutaatorite puhul riistvaraliselt, kasutades ASIC-kiipe (*application-specific integrated circuit*). Tänu riistvaralisele teostusele on kolmanda kihi kommutaatorid üldiselt marsruuteritest kiiremad.

15. Traadita ühendused

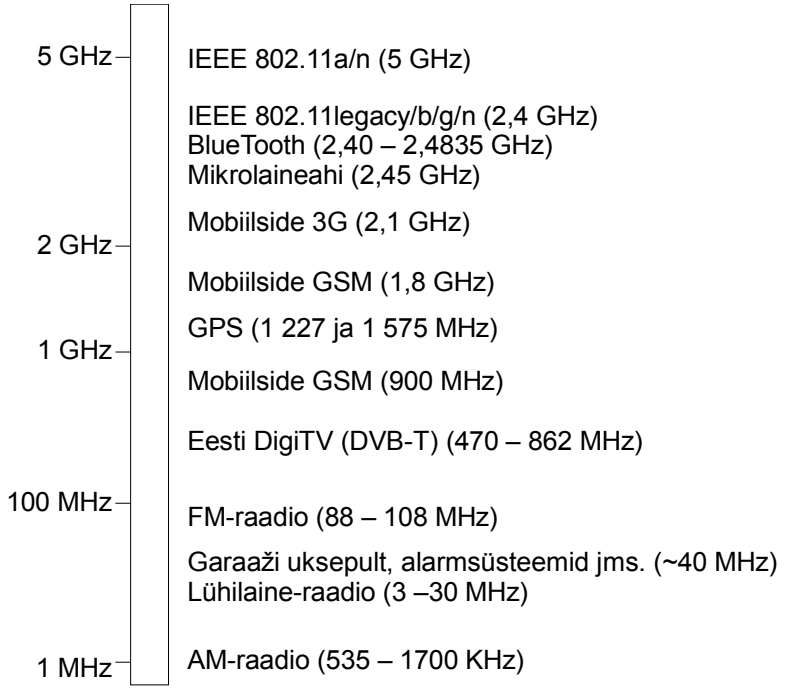
Traadita ühendused pakuvad võimalust vähendada vajalikke kaableid ja olla mobiilsemad (käib töö isegi traadivabalt elektrienergia edastamiseks), kuid selle eest tuleb teatavat hinda maksta. Traadita ühendused vajavad paljuski teistsugust lähenemist, võrreldes traatühendustega. Peamiseks probleemiks on asjaolu, et traadita ühendused on paratamatult jagatud meediumi tehnoloogiad ja nad on rohkem kaitsetud müra suhtes. Müradena võiks koduses keskkonnas välja tuua näiteks mikrolaineahjud, fluorestsentslambid, traadita telefonid, elektrimootorid ja teised juhtmevabad ühendused. Vaata ka joonist 15.1.

Traadita ühenduste jaoks on välja töötatud mitmeid lahendusi, milledest järgnevalt on esile toodud mõningaid kasutatavamaid:

- IEEE 802.11 – tuntakse ka Wi-Fi (*Wireless Fidelity*) nime all, mis ei ole päris täpne vaste. Kasutatakse sagedasti sülearvutite ühendamiseks kohtvõrku tööl, kodus, kohvikutes, linnas ja mujal. Nimetatakse mõnikord ka traadita Ethernetiks.
- WiMAX (*Worldwide Interoperability for Microwave Access*) (IEEE 802.16) – erinevalt IEEE 802.11-st on WiMAX mõeldud pikemate vahemaade ühendamiseks (maksimaalselt 50 km). Maksimaalne ribalaius on 70 Mbit/s. Kasutatavad sagedusalad on 10-66 GHz ja 2-11 GHz.
- Bluetooth (IEEE 802.15) – kasutatakse peamiselt pisiseadmete (nt. mobiiltelefonid, digitaalkaamerad, traadita hiired, traadita klaviatuurid, mobiilide peakomplektid) ühendamiseks omavahel ning arvutiga. Võrreldes IEEE 802.11-ga on väiksem energiatarve ja lihtsam elektroonika, kuid väiksem läbilaskevõime ja maksimaalne ühenduskaugus. Sagedusala on 2,40 GHz - 2,4835 GHz (USA ja Euroopa). Bluetooth'st on mitu erinevat versiooni: 1.2 (ühenduskiirus 1 Mbit/s); 2.0 + EDR (*Enhanced Data Rate*) (ühenduskiirus 3 Mbit/s); 3.0 + HS (*High Speed*) (24 Mbit/s). Bluetooth'l on kolm erinevat klassi:
  - ◇ 1. klass. Maksimaalne lubatud võimsus: 100 mW ja tööraadius ~100 meetrit.
  - ◇ 2. klass. Maksimaalne lubatud võimsus: 2,5 mW ja tööraadius ~10 meetrit.
  - ◇ 3. klass. Maksimaalne lubatud võimsus: 1 mW ja tööraadius ~1 meetrit.
- GSM (*Global System for Mobile communications*) – kasutatakse peamiselt mobiiltelefonides. Andmeedastusena kasutatakse GSM-il põhiliselt GPRS (*General Packet Radio Service*) (ribalaius kuni 107 Kbit/s) ja EDGE (*Enhanced Data rate for GSM Evolution*) (ribalaius kuni 384 Kbit/s) protokolle.
- IrDA (*Infrared Data Association*) – kasutatakse pisiseadmete puhul (mobiilid, sülearvutid jms.). Vahemaa kuni meeter, ribalaius kuni 16 Mbit/s. Seadmete vahel peab olema otsenähtavus. Kasutatakse infrapunakiirguse sagedusriba. IrDA asemel on tänapäeval enamasti kasutusele võetud Bluetooth.

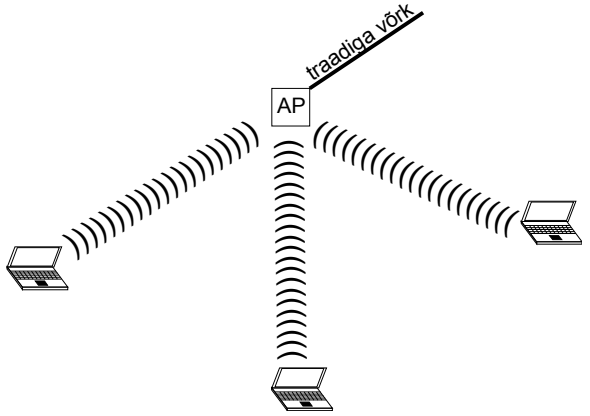
IEEE 802.11

IEEE 802.11 töötab kahes laadis – infrastruktuuri ja juhuvõrgu (*ad-hoc*) laadis. Infrastruktuuri laadi nimetatakse ka BSS (*basic service set*) ja juhuvõrgu laadi IBSS (*independent BSS*). Infrastruktuuri laadis on olemas ligipääsupunktid ja kliendid. Ligipääsupunkte nimetatakse ka AP-deks (*access point*) ja baasjaamadeks. Kliendid suhtlevad ainult ligipääsupunktidega, et suhelda omavahel ja kohtvõrguväliste hostidega. Joonisel 15.2 on toodud näide infrastruktuurilaadis töötavast võrgust. Juhuvõrgu puhul ei ole olemas ligipääsupunkte ja masinad töötavad partnerilt-partnerile (*peer-to-peer*) laadis, see tähendab, et kõik seadmed suhtlevad otse naabermasinatega. Tavalises IEEE 802.11 juhuvõrgus ei ole marsruutimist, mistõttu on võimalik suhelda ainult teiste levialas olevate masinatega. Joonisel 15.3 on toodud näide juhuvõrgulaadis töötavast võrgust. Tavaliselt töötavad IEEE 802.11 võrgud infrastruktuuri laadis, seetõttu vaatleme edaspidi vaikimisi peamiselt infrastruktuurilaadi.

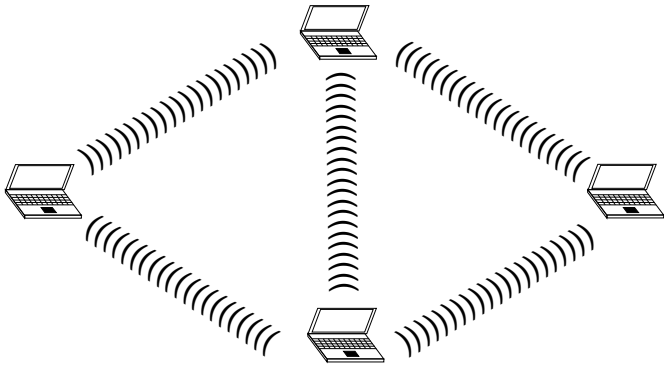


Joonis 15.1: Mõningaid Eestis kasutatavaid raadiolainete sagedusvahemikke.

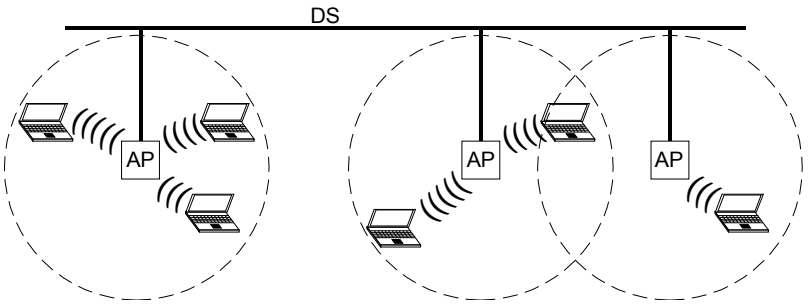
Kui kaks või enam BSS-i on omavahel ühendatud, kasutades DS-i (*distribution system*), siis on võimalik laiendada traadita kohtvõrku, mida nimetatakse ESS-ks (*extended service set*). AP-de ühendamiseks omavahel kasutatakse tavaliselt traatühendust. Kõik baasjaamad ei pruugi toetada DS-i. Joonisel 15.4 on toodud näide ESS võrgust. ESS puhul klient suhtleb ühe baasjaamaga. Kliendi ühest AP levialast teise minekut nimetatakse rändluseks (*roaming*).



Joonis 15.2: Näide infrastruktuuri laadis töötavast kohtvõrgust. Kõik masinad suhtlevad otse ainult AP-ga.



Joonis 15.3: Näide juhuvõrgu laadis ühendatud arvutitest. Joonisel kõige vasakpoolsem ja parempoolsem on üksteise suhtes levialast väljas.

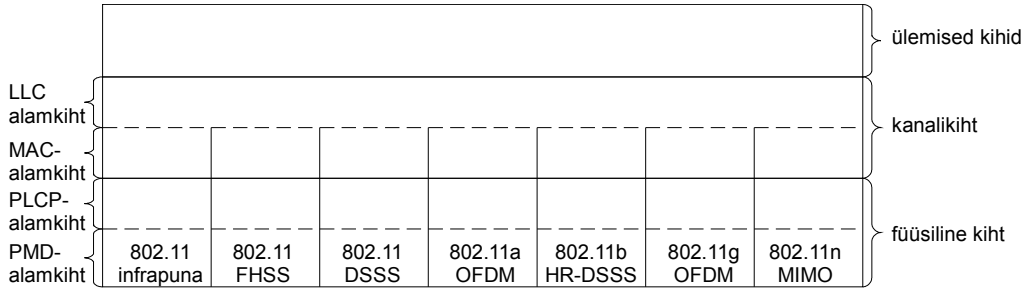


Joonis 15.4: Näide ESS laadis töötavast kohtvõrgust. ESS-ga on võrku võimalik laiendada. Erinevad AP-d ühendatakse tavaliselt omavahel traatühendusega. Kliendid saavad liikuda ühest AP levialast teise võrguga ühendust kaotamata.

IEEE 802.11 kasutab 2,4 ja 5 GHz sagedusalasid, mis on enamikes riikides vabalt kasutatavad sagedusalad (vt. ka tabelit 15.2). Näiteks GSM-mobiilides kasutatavad sagedusalad (900 MHz ja 1800 Mhz) ei ole vabalt kasutatavad, vaid on riigi poolt litsentseeritavad. IEEE 802.11-st on olemas mitu varianti: IEEE 802.11, IEEE 802.11a, IEEE 802.11b, IEEE 802.11g ja IEEE 802.11n. Esimese IEEE 802.11 nimetus oli IEEE 802.11, kuid see on segadust tekitav. Seetõttu kasutatakse ka täpsemaks määratlemiseks sufiksit "legacy". IEEE 802.11a ei ole ühilduv IEEE 802.11b-ga. IEEE 802.11g on ühilduv IEEE 802.11b-ga. IEEE 802.11n kasutab MIMO't (*multiple-input and multiple-output*) ehk mitut antenni, võimaldades suuremat andmeedastuskiirust ja distantsi. Vaata tabelit 15.1 erinevate IEEE 802.11 variantide andmete kohta. Joonisel 15.5 on toodud erinevate IEEE 802.11 versioonide kihtideks jagunemine.

Tabel 15.1. IEEE 802.11 erinevate versioonide andmed.

protokoll	avaldamisaasta	sagedusala	ribalaiused (Mbit/s)	umbkaudne leviulatus väljas (siseruumis) (m)
legacy	1997	2,4 GHz	1; 2	~100 (~20)
a	1999	5 GHz	6; 9; 12; 18; 24; 36; 48; 54	~120 (~35)
b	1999	2,4 GHz	5,5; 11	~140 (~40)
g	2003	2,4 GHz	6; 9; 12; 18; 24; 36; 48; 54	~140 (~40)
n	2009	2,4 ja 5 GHz	20 MHz ribalaiuse korral: 7,2; 14,4; 21,7; 28,9; 43,3; 57.8; 65; 72,2. 40 MHz ribalaiuse korral: 15; 30; 45; 60; 90; 120; 135; 150.	~250 (~70)



Joonis 15.5: IEEE 802.11 kihid ja erinevate versioonide kasutatavad tehnikad. Joonisel kasutatud akronüümide lahtikirjutused: PLCP (*physical layer convergence procedure*), PMD (*physical medium dependent*), FHSS (*frequency-hopping spread spectrum*), DSSS (*direct-sequence spread spectrum*), OFDM (*orthogonal frequency-division multiplexing*), HR-DSSS (*high rate-DSSS*), MIMO (*multiple-input and multiple-output*).

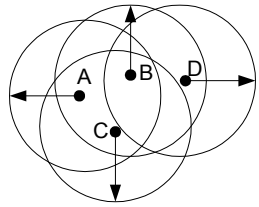
IEEE 802.11 puhul kasutatakse põhimõtteliselt jagatud meediumit. Aja jaotamiseks meediumis kasutatakse kahte laadi operatsioone: DCF (*Distributed Coordination Function*) ja PCF (*Point Coordination Function*). PCF-i kasutab baasjaam, et kontrollida tegevust omal alal. DCF-i puhul ei kasutata tsentraalset kontrolli (nagu ka Etherneti puhul), kuid traadita ühendustega ei ole võimalik kasutada Ethernetis kasutatavat jagatud meediumi kasutamise protokolli CSMA/CD. Põhjuseks on asjaolu, et erinevalt pooldupleks-ühendustega Ethernet võrgust ei pruugi IEEE 802.11 puhul raadiosignaal jõuda kõikidelt võrgus olijatelt kõikide teisteni. Meediumile ligipääsukontrollina kasutatakse CSMA/CA (*carrier sense multiple access with collision avoidance*) protokolli.

Vaatleme CSMA/CA tööpõhimõtet joonisel 15.6 oleva näite abil, kus on ka nn. peidetud jaama probleem. Olgu A levialas B ja C (C võib asuda ka B levialas, kuid see pole siin oluline). Olgu D masina B levialas, kuid mitte masina A levialas. Oletame, et masin A soovib alustada andmete saatmist masinale B. Selleks saadab masin A masinale B päringu, et lubataks alustada andmeedastust, saates selleks RTS kaadri. Kui B saab edukalt RTS (*request to send*) kaadri kätte, siis ta otsustab, kas lubada andmete edastust. Kui B lubab masinalt A edastust, siis saadab B masinale A tagasi CTS (*clear to send*) kaadri. Seejärel A saadab oma andmekaadri ja käivitab

ACK ajaloenduri. Kui masin B saab masina A lähetatud andmekaadri edukalt kätte, siis B saadab lõpetuseks tagasi ACK kaadri. Kui masina A ACK ajaloendur aegub, ilma et oleks saanud masinalt B ACK kaadrit, siis kogu protsess algab otsast peale (arvatavasti oli võrgus kollisioon). Kuna masin C on meil A levialas, siis saab ka masin C RTS kaadri ja teab, et ei saa kanalit kasutada. Kui masin B saadab CTS kaadri, siis saab ka masin D teada, et meedium on hõivatud. Masinad C ja D saavad ise andmeid saata siis, kui edastus on kindlasti lõppenud või kui seda signaliseerib ACK kaader. Vaata andmevahetuseks vajalikke toiminguid joonisel 15.7 toodud skeemilt ja meediumi hõivatust kõigi nelja masina perspektiivis jooniselt 15.8.

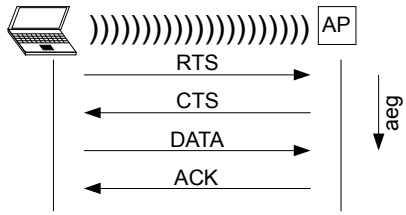
Tabel 15.2: Erinevates riikides võivad olla kasutatusel erinevad kanalid Wi-Fi sagedusalas. ETSI (*European Telecommunications Standards Institute*) on Euroopas olev standardiseerimisorganisatsioon.

kanali number	sageduse kese (MHz)	Põhja-Ameerika ja Austraalia	ETSI	Prantsusmaa	Jaapan	Iisrael
1	2412	✓	✓		✓	
2	2417	✓	✓		✓	
3	2422	✓	✓		✓	✓
4	2427	✓	✓		✓	✓
5	2432	✓	✓		✓	✓
6	2437	✓	✓		✓	✓
7	2442	✓	✓		✓	✓
8	2447	✓	✓		✓	✓
9	2452	✓	✓		✓	
10	2457	✓	✓	✓	✓	
11	2462	✓	✓	✓	✓	
12	2467		✓	✓	✓	
13	2472		✓	✓	✓	
14	2484				✓	

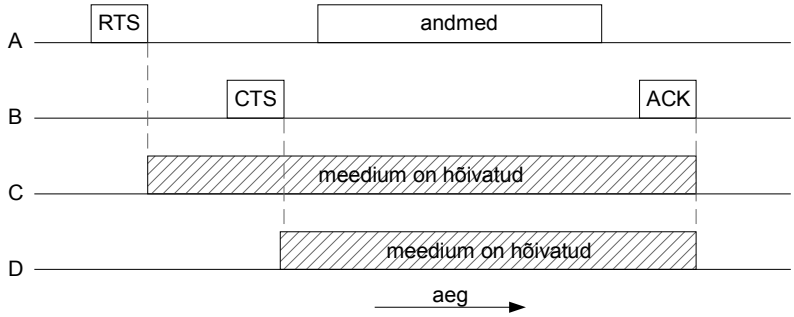


Joonis 15.6: Erinevate osapoolte signaali levimine. Joonisel on näha, et A ja D on mõlemad B levialas, kuid A ja D ei ole üksteise levialas. Kui A ja D soovivad samal ajal suhelda B-ga, siis ei saa nad kumbki aru tekkinud kollisioonist.

DCF-i kasutatakse nii infrastruktuuriga kui ka juhuvõrgu laadis, PCF-i ainult viimases. Kõik implementatsioonid peavad toetama DCF-i. PCF on fakultatiivne. DCF ja PCF võivad koos töötada ka sama BSS piires. Selleks kasutatakse nelja erineva pikkusega kaadrite vahelisi ajapilusid, milledeks on lühimast alustades: SIFS (*short interframe space*), PIFS (*PCF interframe space*), DIFS (*DCF interframe space*) ja EIFS (*extended interframe space*). Vaata detailsemat infot jooniselt 15.9.



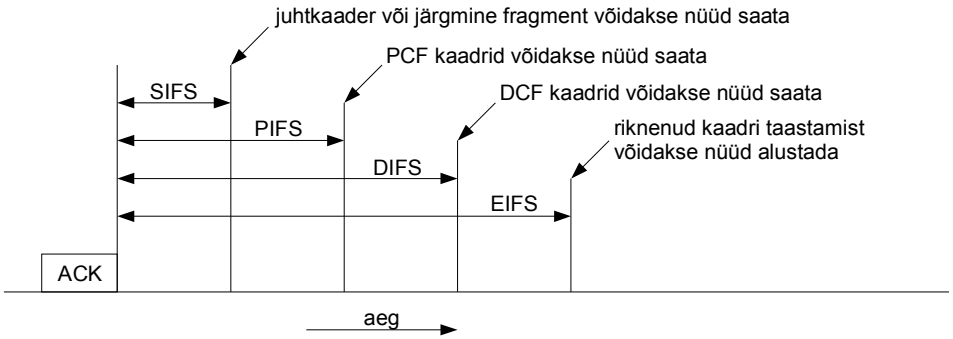
Joonis 15.7: Andmevahetus Wi-Fi võrgus. Eelnevalt peab vastuvõtja andma loa edastamiseks, sest kui joonisel 15.6 A ja D soovivad üheaegselt suhelda B-ga, siis B ei saaks kummagi edastatut kätte, samas A ja B ei tea teineteise olemasolust.



Joonis 15.8: Andmete saatmine, kasutades CSMA/CA protokolli, võttes aluseks masinate paiknemise joonisel 15.6. Kui masin A soovib saata andmeid, siis ta küsib B käest saatmiseks luba. Kui B saab RTS kaadri kätte, siis ta vastab CTS kaadriga. RTS kaader jõudis ka masinani C, mistõttu ta nüüd teab, et meedium on hõivatud kuni aegumiseni või ACK kaadrini. Kui B vastab CTS kaadriga, siis saab ka D teada meediumi hõivatusest. Antud juhul lõpetab meediumi hõivatuse ACK kaader.

Võrreldes traadiga pooldupleks-ühendustega, on traadita võrgus palju rohkem võimalusi kollisioonide tekkimiseks. Kollisioonid võivad tekkida ka andmekaadri saatmisel. Seega, mida suuremad on kaadrid, seda suurem on tõenäosus, et kaader saab vigastada. Seetõttu on parem andmeid edastada pigem mitme väiksema kaadriga. Sellepärast lubab IEEE 802.11 tükeldada kaadreid väiksemateks osadeks. Tükeldatud kaadrid on nummerdatud ja igale kaadri saatmisele oodatakse kinnitust (uut kaadrit ei saadeta, kui eelmise kohta ei ole saadud kinnitust). Sellega suureneb tavaliselt läbilaskevõime, sest ei ole nii suures koguses andmete taassaatmist.

PCF-laadis töötadesl küsitleb baasjaam masinaid, kas nad soovivad andmeid edastada. Selliselt on edastamise järjekord baasjaama poolt kontrollitav. Nii välditakse kollisioonide tekkimise ohtu. Standardi järgi on küsitlemise mehhanism spetsifitseeritud, kuid jäetud lahtiseks küsitlemiste sagedus, järjekord jms. Põhiline meetod on saata välja märgutule (*beacon*) kaadreid (10-100 korda sekundis). Märgutulekaader sisaldab informatsiooni, milleks on võrgu nimi ehk SSID (*service set identifier*), ajatemplid, märgutule intervall, võimalused, toetatud kiirused, parameetrid jm. Märgutule kaadritega saab uus masin teada võrgust. Kui masin ühineb võrguga, siis saadakse ka mingi osa ribalaiusest antud BSS-is.



Joonis 15.9: IEEE 802.11 puhul kasutatavate kaadrite vahelised minimaalsed ajapilud ja nende kasutus.

Võrgu identifitseerib SSID, ESS puhul on kõikidel baasjaamadel sama SSID. Baasjaama identifitseerib BSSID (baasjaama IEEE 802.11 liidese MAC-aadress). Kui klient on ühinenud mingi Wi-Fi võrguga ja antud võrgus on mitu sama SSID-iga juurdepääsupunkti, siis klient otsustab ise, millisega ta nendest ühendub (ESS alal olles). Korraga saab olla ühendatud ainult ühe juurdepääsupunktiga. IEEE 802.11 jätab ühenduseks valitava juurdepääsupunkti valiku kliendile, kuid tavaliselt ühendutakse ligipääsupunktiga, millelt saadakse kõige tugevam signaal (vanema IEEE 802.11 puhul ei pruugi toimida). Klient võib soovi korral ligipääsupunkti vahetada (näiteks liiguti sülearvutiga majas ringi).

IEEE 802.11 kasutab kolme sorti kaadreid, milleks on andmete, juhtimise ja halduse kaadrid. Andmete kaadri väljad on järgmised:

- Kaadri juhtimine. Sisaldab ühteteist alamvälja. 16 bitti.
  - ◇ Versioon. Määrab protokolli versiooni. Praeguse 802.11 puhul 0. 2 bitti.
  - ◇ Tüüp. Määrab kaadri tüübi, milleks võivad olla (sulgudes välja väärtus) haldus- (0), juht- (1) ja andmekaader (2). Viimane välja väärtus (3) on reserveeritud tulevikuks. 2 bitti.
  - ◇ Alamtüüp. Määrab kaadri tüübi alamtüübi, näiteks juhtkaadri puhul põhiliselt kasutatavad kaadrid on RTS (*request to send*), CTS (*clear to send*) ja ACK (*acknowledgement*). 4 bitti.
  - ◇ To DS (*to distribution system*) lipp. Lipp on püsti, kui kaader on suunatud AP-le.
  - ◇ From DS (*from distribution system*). Lipp on püsti, kui kaader tuleb AP-lt.
  - ◇ MF (*more fragments*). Kui lipp on püsti, siis järgneb veel fragmente.
  - ◇ Taassaatmine (*retry*). Kui lipp on püsti, siis on tegemist kaadri taassaatmisega (eelnevalt oli kaader või tema kinnitus saadetud ebaõnnestunult).
  - ◇ Võimsuse haldus (*power management*). Kasutab baasjaam, et panna vastuvõtja magavasse olekusse või sellest äratada. IEEE 802.11 puhul on arvestatud, et võrgus olevad seadmed töötavad akude peal, mistõttu on vaja energiat kokku hoida. Baasjaam kogub magamise ajal kaadreid puhvrisse.
  - ◇ Veel andmeid (*more data*). Kui lipp on püsti, siis saatjal on veel kaadreid vastuvõtja jaoks.
  - ◇ WEP (*Wired Equivalent Privacy*). Kui lipp on püsti, siis kaadri keha on krüpteeritud, kasutades WEP-i.
  - ◇ O (*order*) lipp. Kui lipp on püsti, siis seda kaadrit peab käitlema järjekorras.
- Kestus või AID (*association identity*) (olenevalt kaadri tüübist). Määrab, kui kauaks kaader ja tema kinnitus hõivavad kanali. Väli on AID võimsus-säästu küsitlemiskaadrite korral, millega päritakse baasjaamalt puhverdatud kaadrite olemasolu kohta. 2 baiti.

- Aadress-1. Alati vastuvõtja MAC-aadress. 6 baiti. Aadressiväljade tähendused olenevad "to DS" ja "from DS" lippudest. Vaata aadressiväljade tähendusi tabelist 15.4.
- Aadress-2. Alati saatja MAC-aadress. 6 baiti.
- Aadress-3. MAC-aadress, mis oleneb lippudest "to DS" ja "from DS". 6 baiti.
- Järjekorranumber. Kui kaadrit on vaja tükeldada väiksemateks osadeks, siis siin märgitakse järjekorranumber, et vastuvõtja saaks uuesti algse kaadri kokku panna. 12 esimest bitti määravad kaadri ja neli bitti kaadri fragmendi järjekorranumbri. 16 bitti.
- Aadress-4. MAC-aadress, mis oleneb lippudest "to DS" ja "from DS". 6 baiti.
- Andmed (tavaliselt IP pakett). Varieeruva suurusega, kuni 2312 baiti.
- Kontrollsumma (CRC-32). 4 baiti.

Tabel 15.3: Aadressiväljade sõltuvus lippudest "to DS" ja "from DS". Tähised: DA (*destination address*) – sihtmasina MAC-aadress (ehk sihtaadress); SA (*source address*) – saatva masina MAC-aadress (ehk lähteaadress). BSSID – ligipääsupunkti MAC-aadress; RA (*receiver address*) – vahetu vastuvõtja MAC-aadress; TA (*transmitter address*) – vahetu kaadri edastaja MAC-aadress. IEEE 802.11 puhul on lisa-aadressid vajalikud, et pakkuda võimalust suunata pakette kohtvõrgu erinevate rakkude vahel.

To DS	From DS	Aadress-1	Aadress-2	Aadress-3	Aadress-4
0	0	DA	SA	BSSID	-
0	1	DA	BSSID	SA	-
1	0	BSSID	SA	DA	-
1	1	RA	TA	DA	SA

IEEE 802.11 puhul kuuluvad füüsilise kihi juurde PLCP (*Physical Layer Convergence Procedure*) kaadri väljad. Toome välja IEEE 802.11b kaadrite väljad:

- Sünkronisatsioon. Sisaldab vaheldumisi bitte 0 ja 1. Vastuvõttev osapool sünkroniseerib ennast nende abil saatjaga.
- Kaadri alguse eraldaja. Tähistab kaadri algust, mis sisaldab bitte 1111001110100000.
- Signaal. Tähistab kaadri andmete kiiruse väärtust korda 100 Kbit/s. Näiteks väärtus 10 tähendab 1 Mbit/s, väärtus 20 aga 2 Mbit/s. PLCP kaader saadetakse alati 1 Mbit/s kiirusel.
- Teenus (*service*). Reserveeritud tulevikus kasutamiseks, mis sisaldab praegu ainult nullbitte.
- Pikkus. Näitab, mitu mikrosekundit võtab aega PPDU (*PLCP protocol data unit*) (ehk kanalikihist saadud kaader) sisu edastamine. Vastuvõtja saab selle abil teada, millal kaader lõpeb.
- Kontrollsumma (CRC-16).
- PPDU sisu. Sisaldab kanalikihi kaadrit.

Standard IEEE802.11 määrab üheksa teenust, mida peab toetatama. Need jagunevad viieks levikuteenuseks (*distribution*) ja neljaks jaamateenuseks. Levikuteenused on ala liikmelisuse haldamiseks ja koostööks teiste jaamadega väljaspool ala. Neid teenuseid pakuvad baasjaamad, mis haldavad masinate levialasse tulemist ja lahkumist. Levikuteenus pakub järgnevat:

- Ühendamine. Kasutavad masinad, et ühenduda baasjaamaga. Kasutatakse siis, kui mobiilne masin liigub vastava baasjaama alale. Siin identifitseeritakse ja määratakse toetatavus (ühenduskiirused PCF küsitlemise jaoks, energiahalduse nõudmised jm.). Baasjaam võib nõustuda või keelduda ühendumisest. Baasjaam võib nõuda lisaks ka autentimist.

- Lahtiühendamine. Kui mobiilsed jaamad soovivad lahtiühenduda viisakal moel, siis kasutavad nad seda teenust. Masinad võivad lahtiühenduda, sest näiteks mingi muu baasjaam saab eelistatumaks praegusest. Samuti võib baasjaam ennast lahti ühendada (näiteks kui baasjaam mingil põhjusel sulgetakse).
- Taasühendamine. Uuesti ühinetakse eelmise baasjaamaga, mida varasemalt kasutati (kasutatav, kui liigutakse sama kohtvõrgu ühe baasjaama alalt teisele, üldjuhul andmekaota).
- Levik. Teenus määrab, kuidas marsruutida kaader baasjaamani. Kui ollakse ühendatud baasjaamaga, siis saadetakse otse, vastasel korral kasutatakse DS-i baasjaamade vahel.
- Kombineerimine. Kui kaader on vaja saata läbi mitte 802.11 võrgu erineva adresseerimisskeemiga või kaadri formaadiga, siis antud teenus konverteerib kaadri vajalikku lähtevõrgu formaati.

Jaamateenused on seotud tegevusega konkreetse ala sees. Nendeks teenusteks on:

- Autentimine. Baasjaam võib nõuda ka autentimist. Kui autentimist nõutakse, siis enne seda ei ole võimalik andmeid saata ega vastu võtta.
- Autentimise tühistus. Kui autenditud masin ei soovi enam võrku kasutada, siis ta tühistab oma autentimise võrgust lahkudes.
- Privaatsus. Teenus hoolitseb krüpteerimise ja dekrüpteerimise eest.
- Andmete transport. Kontrollitakse ja korrigeeritakse tekkinud vead andmete ülekandmisel.

Turvalisus

Traadita ühenduse puhul on suurem oht võrguliikluse pealtkuulamiseks ja ressursi kasutamiseks ilma volituseta (pääsetakse ligi kohtvõrgu teistele masinatele ja kasutatakse näiteks tasuta Internetti), sest tegemist on jagatud meediumiga ja kõik, kes on piisavalt lähedal traadita ühendusega kohtvõrgule, võivad liiklust pealt kuulata ja ressursse kasutada. Pealtkuulamise ja volituseta ressursikasutamise vältimiseks on vaja traadita ühenduste puhul rakendada vajalikke meetmeid, eelkõige krüpteerimist. Kasutatavad protokollid pakuvad konfidentsiaalsust, andmete terviklikkust ja ligipääsu kontrolli.

Wi-Fi puhul on olemas kolm põhilist krüpteerimise protokolli, milleks on WEP (*Wired Equivalent Privacy*), WPA (*Wi-Fi Protected Access*) ja WPA2. WPA realiseerib osa IEEE 802.11i turvastandardist ja täielikult on IEEE 802.11i realiseeritud WPA2-na. WPA väljalaske tingisid WEP-i ebaturvalisuse probleemid.

WEP protokoll on muutunud ebaturvaliseks. Selle lahtimurdmine on minutite küsimus ja Internetis leidub selleks otstarbeks tehtud programme. Seetõttu on soovitatav mitte kasutada WEP-i. WEP puhul on olemas ka mitmeid modifikatsioone, kuid need on samuti ebaturvalised ja ei pruugi olla toetatavad.

WPA puhul on parandatud mitmeid turvalisuse nõrkusi:

- WPA kasutab paremini RC4 (*Rivest Cipher 4*) krüpteerimist suurendades IV (*initialization vector*) väärtusi ja parandab taasesitusründe vastast kaitset.
- WPA kasutab iga paketi korral erinevat salajast võtit.
- WPA ei kasutata krüpteerimiseks originaalvõtit otse, vaid see tuletatakse.
- WPA-sse ehitati sisse turvaline võtmehaldus (käsitsi võtmete vahetus on töömahukas, mille tõttu teostatakse võtmete vahetust harva).
- WPA kasutab tõhusamat terviklikkuse kontrolli.
- WPA lubab paroolis kasutada nii tähti ja numbreid, kuid WEP võtmeks on kuueteistkümnendsüsteemi arv, mis on inimestele ebamugav kasutada.

WPA2 kasutab krüpteerimiseks AES (*Advanced Encryption Standard*) plokkšifrit. Võimaluse korral on privaatseks kasutamiseks mõeldud traadita kohtvõrgus soovitatav kasutada WPA või WPA2 protokolli.

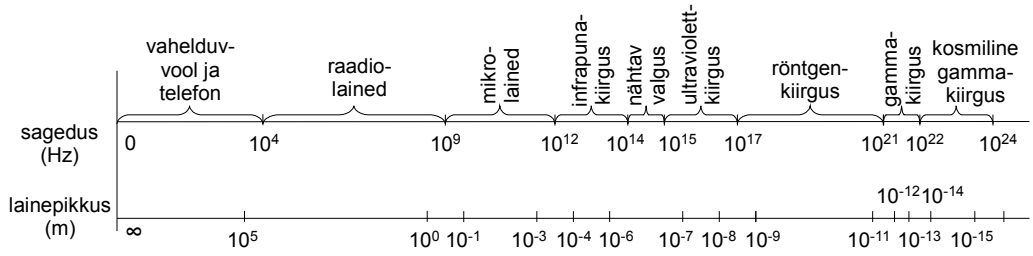
Võrdlus traatühendustega

Lõpetuseks mõningaid traadita edastuse positiivseid ja negatiivseid külgi:

+ Võimaldab mugavalt ühenduda võrguga ilma füüsiliselt midagi tegemata ja vabamalt arvutit liigutada.
+ Võimaldab kasutajal mugavalt rändlust kasutada.
+ Kasutatav meedium ei vaja hooldust ning ei maksa midagi.
+ Seadmete maksumus on vähenemas ja võrreldav traatühendusega seadmete hinnaga.
– Aeglasemad ühendused.
– Võimalik mõju inimese tervisele, eriti ajule.
– Kohtvõrgu tööd on võimalik tahtlikult või tahtmatult häirida (kasutades sama sagedusala, nt. mikrolaineahjud ja spetsiaalsed häirijad).
– Madalam töökindlus.
– Turvalisuse riskid.
– Ühilduvusprobleemid. Kõik seadmed ei pruugi omavahel ühendust saada.

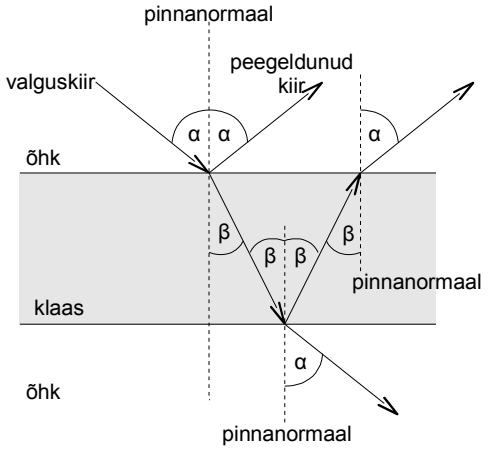
16. Fiiberoptika

Fiiberoptikas kasutatakse signaalide edastamiseks valgusimpulsse. Valgus on elektromagnetlainete teatav sagedusvahemik. Inimsilm näeb valgusena vahemikku 700-400 nm (nanomeetrit) (vt. ka joonist 16.1). Kõige kiiremini liiguvad elektromagnetlained vaakumis, aeglasemalt muudes keskkondades. Materjali **murdumisnäitajaks** nimetatakse arvu, mis saadakse, jagades valguse kiiruse vaakumis valguse kiirusega vaadeldavas materjalis. Mida suurem on murdumisnäitaja, seda suurem on vastava materjali optiline tihedus. Fiiberoptikas kasutatakse lainepikkusi 850 nm, 1310 nm ja 1550 nm, sest nad on fiiberoptikas väikseima optilise takistusega.



Joonis 16.1: Elektromagnetlainete spekter.

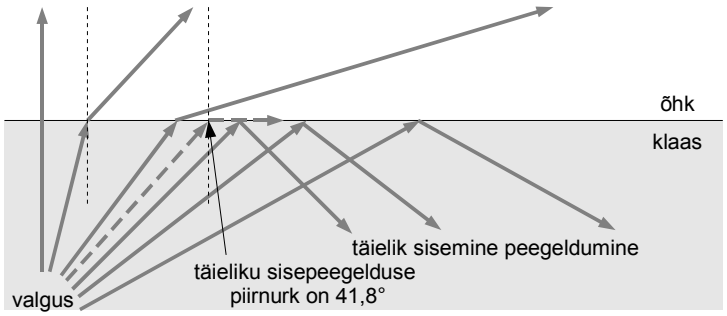
Valguse liikudes ühest keskkonnast teise peegeldub osa valgusest keskkonna pinnalt ning osa siseneb uude keskkonda, kus ta murdub vastavalt keskkondade suhtelisele murdumisnäitajale (ehk murdumisnäitajate erinevusele). Kiir, mis läheb suurema murdumisnäitajaga keskkonda, murdub pinnanormaali poole (pinnanormaal on pinnaga risti). Väiksema murdumisnäitajaga materjali sisse liikuv valgus murdub pinnanormaalist eemale. Murdumisnurk sõltub valguse kokkupuutenurgast materjaliga ja materjalide murdumisnäitajate erinevusest. Tabelis 16.1 on toodud mõningate materjalide murdumisnäitajaid. Mida suurem on valguse langemisnurk pinnanormaali suhtes, seda suurem hulk valgust peegeldub. Alates teatud nurgast peegeldub kogu valgushulk. Seda nurka nimetatakse täieliku peegelduse piirnurgaks. Näiteks klaasi ja õhu korral on selleks nurgaks 41,8º (vt. joonist 16.3). Seda asjaolu kasutataksegi ära fiiberoptikas. Joonisel 16.2 on toodud näide valguskiirest, mis liigub õhust klaasi ja sealt edasi jälle õhku.



Joonis 16.2: Valguse peegeldumine ja murdumine erinevatesse keskkondadesse sisenemisel.

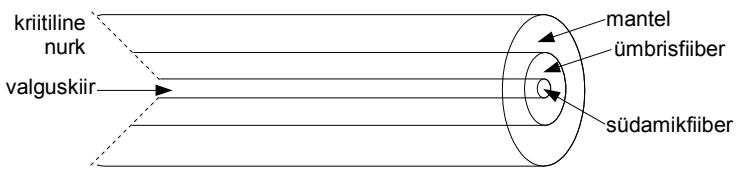
Tabel 16.1: Mõningate materjalide murdumisnäitajad.

materjal	murdumisnäitaja
vaakum	1
õhk	1,0003
jää	1,31
vesi	1,3330
klaas	1,523
NaCl	1,544
teemant	2,419
räni	4,01



Joonis 16.3: Vastavalt keskkondadele on olemas täieliku sisepeegelduse piirnurk, mille juures valgus peegeldub ja ei sisene uude keskkonda. Joonisel on näide klaasi ja õhuga.

Fiiberoptiline kaabel koosneb kolmest põhiosast: südamik (*core*, *nucleus*), mille ümber on ümbrisfiiber (*cladding*, *wrapping*) ning selle ümber omakorda kattemantel (lisaks võib olla ka tugevduskihte jms.). Fiiberoptilise kaabli südamiku kaudu edastatakse valgussignaale. Valguse allikaks on laser või valgusdiood ehk LED (*light-emitting diode*). Signaliseerimine toimub valguskiirt sisse ja välja lülitades. Fiibri südamiku ümber olev ümbrisfiiber toimib peeglina, et valguskiir ei murraks südamikust välja (kaabel ei pea olema absoluutselt sirge). Fiiberkaabli südamik on suurema murdumisnäitajaga kui ümbrisfiiber ja läbiva valguskiire nurk peab olema suurem kui kahe fiibri murdumisnäitajatest tuleneva täieliku sisepeegelduse piirnurk.



Joonis 16.4: Optilise fiibri kaabli sisu lihtsustatult (mantli all võib olla näiteks tugevduskiht).

Fiiberkaabli puhul on üldjuhul tegemist simpleks-edastusega (ühes otsas saatja, teises vastuvõtja). Mõlemat pidi ühendus nõuab kahte eraldi suundadega fiiberkaablit või eri suundade jaoks erinevate lainepikkuste kasutamist.

Fiiberoptilised kaablid jagatakse kahte laadi: ainumood- (*single-mode*) ja multimood- (*multimode*) fiiberkaabliteks. Nende erinevusteks on:

- Südamiku läbimõõt. Ainumoodkaabli südamik on peenem (8,3 µm kuni 10 µm) kui multimoodkaablil (50 µm või 62,5 µm).
- Ainumoodkaablis edastatakse korraga ühte signaali. Multimoodkaablis siseneb valguskiir erinevate nurkade all (mille tõttu võtab edasi liikumine aega erinev ajahulk).
- Hajuvus. Multimoodkaablil on üldiselt suurem hajuvus.
- Maksimaalne kaabli pikkus. Multimoodi puhul kuni 2 km, ainumoodi puhul 100 km või enamgi (oleneb ka kaabli kvaliteedist).
- Valguse allikas multimoodkaabli puhul on LED, ainumoodkaabli puhul tavaliselt laser (LED on odavam kui laser).

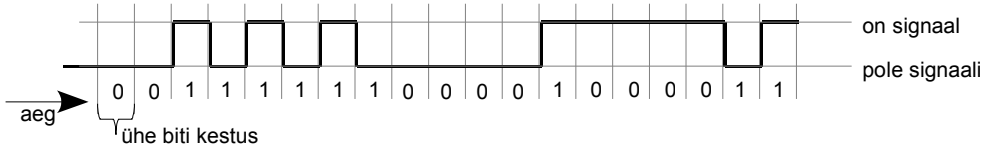
Fiiberkaablit on võimalik jätkata, kokkukeevituse teel. Fiibrite keevitamine on väga suurt täpsust nõudev töö ja seda tehakse spetsiaalsete seadmetega. Fiiberkaabli otsad tehakse samuti spetsiaalsete vahenditega (otsad ei tohi olla narmastatud).

Fiibril baseeruvate ühenduste tegemine on kallim kui traatühenduste tegemine, kuid seevastu on nende kaudu võimalik saavutada eeldatavasti suuremaid andmeedastuskiiruseid, nad on immuunsed elektromagnetilisele mürale ning võimaldavad katta pikemaid vahemaid.

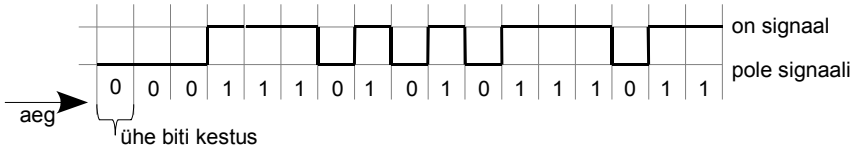
Ohutustehnilise aspektina on oluline, et ühendatud fiiberkaabli otsa ei tohi otse vaadata, sest valguskiir võib kahjustada silma. Fiiberkaabli otsi tuleb kaitsta tolmu ja mustuse eest. Selleks kasutatakse tavaliselt kummilutte. Määrdunud otsi puhastatakse spetsiaalsete vahenditega.

16.1 Ethernet üle fiibri

Ethernet-tehnoloogiat on laiendatud ka fiiberoptika kasutamiseks. Fiibripõhiselt suudetakse saavutada suuremat ribalaiust kui elektrijuhtidel põhinevate kaablitega. Vaata võrdlusandmeid tabelis 16.2. Kohtvõrkude puhul kasutatakse valguskaablit eelkõige pikemate magistraalühenduste jaoks (näiteks majade ja korruste vahel).



Joonis 16.5: NRZI signaliseerimine. All on edastatavad bitid.



Joonis 16.6: NRZ signaliseerimine. All on edastatavad bitid.

Sarnaselt 100BASE-T'le on ka 100BASE-FX korral kodeerimine kaheetapiline. Mõlemal on esimene etapp 4B/5B kodeerimine, kuid 100BASE-FX puhul kasutatakse järgmisena NRZI (*Non-return-to-zero, inverted*) signaliseerimist. NRZI puhul toimub sisendbiti "1" puhul signaali üleminek, biti "0" puhul mitte. Tänu eelnevale 4B/5B kodeerimisele ei teki liialt palju jär-

jestikulisi nullbitte. NRZI kodeerimist kasutab ka näiteks USB. NRZI kodeerimise näide on toodud joonisel 16.5.

Tabel 16.2: Erinevad Etherneti tüübid, kasutatavad kaablid ja nende maksimaalsed pikkused.

Etherneti tüüp	Kaabli tüüp	Maksimaalne pikkus
10BASE-5	jäme koksiaal (*thicknet coaxial*) (50 Ω)	500 m
10BASE-2	peenike koksiaal (*thinnet coaxial*) (50 Ω)	185 m
10BASE36	koksiaalkaabel	3,6 km
10BASE-T	Cat 3 või parem	100 m
10BASE-FP	multimoodfiiber	1 km
10BASE-FL	multimoodfiiber	2 km
10BASE-FB	multimoodfiiber	2 km
100BASE-T4	Cat 3 või parem	100 m
100BASE-T2	Cat 3 või parem	100 m
100BASE-TX	Cat 5 või parem	100 m
100BASE-FX	multimoodfiiber	2 km
1000BASE-T	Cat 5e või parem	100 m
1000BASE-TX	Cat 6	100 m
1000BASE-SX	multimoodfiiber	550 m
1000BASE-LX	ainumoodfiiber	2 km
10GBASE-CX4	Twin-axial	100 m
10GBASE-T	Cat 6a, Cat 7	100 m
10GBASE-SR	multimoodfiiber	300 m
10GBASE-LR	ainumoodfiiber	10 km või enam
10GBASE-ER	ainumoodfiiber	40 km
10GBASE-SW	multimoodfiiber	300 m
10GBASE-LW	ainumoodfiiber	10 km
10GBASE-EW	ainumoodfiiber	40 km
10GBASE-LX4	multimoodfiiber	300 m
10GBASE-LX4	ainumoodfiiber	10 km

Gigabitise Etherneti ühenduse puhul üle fiibri kasutatakse esimesel etapil 8B/10B kodeerimist, mis on analoogne 4B/5B kodeerimisele. Sellele järgneb NRZ signaliseerimine. 10 Gbit/s Etherneti ühenduse puhul üle fiibri on sisse toodud rohkem muudatusi, võrreldes varasemate Etherneti versioonidega. Seejuures ei ole muutunud kaadri ülesehitus. Kasutatav on ainult täisdupleksühendus, mistõttu puudub vajadus CSMA/CD järele (gigabitise Etherneti puhul oli standardis olemas, kuid reaalselt ei ole implementeeritud). Sisse on toodud väikeseid muudatusi SONET/SDH tehnoloogiatega koostalitluseks. 10Gbit/s fiibripõhise Etherneti puhul kasutatakse esimesel etapil 8B/10B (10GBASE-LX4 puhul) või 64B/66B kodeeringut. 64B/66B väljund sisaldab tunduvalt vähem ballasti kui 8B/10B. Teisel etapil kasutatakse NRZ signaliseerimist.

Arendamisel on 100 Gbit/s ja 40 Gbit/s Ethernet, kus fiiberoptika puhul on suund erineva lainepikkusega signaalide paralleelselt kasutamiseks. 10GBASE-LX4 puhul juba kasutatakse nelja erineva lainepikkusega kanalit, kasutades WDM multipleksimist.

17. WAN tehnoloogiaid

Senini oleme vaadelnud peamiselt LAN ehk kohtvõrgu tehnoloogiaid. Järgnevalt heidame pilgu ka WAN tehnoloogiatele. WAN-i peamiseks ülesandeks on ühendada omavahel LAN võrke üle pikkade vahemaade. Tüüpiliselt on LAN (või ka näiteks kodus üks arvuti) ühendatud internetiteenusepakkujaga ehk ISP-ga (*Internet service provider*). ISP-ga ühenduses olijat nimetatakse abonendiks (*subscriber*). Sideseadmeid, mis paiknevad abonendi juures, nimetatakse kliendiseadmeteks ehk CPE (*customer premises equipment*). Kliendiseadmeteks võivad olla näiteks telefon, DSL modem, kaabelmodem, telefonikeskjaam jms. Kliendiseade ühendub teenusepakkuja seadme külge, mida nimetatakse keskjaamaks (*central office*). Kliendiseadme ja keskjaama vahelist sideliini nimetatakse abonendisiiniks (*local loop*), aga ka "viimaseks miiliks" (*last mile*). Kohta, kus lõpeb ISP vastutusala ja algab abonendi vastutusala, nimetatakse demarkatsioonipunktiks ehk sidumispunktiks (*demarcation point*).

Kohtvõrkudes kasutatakse OSI esimese ja teise kihi tehnoloogiaid vähe, peamiselt Ethernet ja IEEE 802.11, kuid WAN ühenduste puhul on tehnoloogiaid ja protokolle märksa rohkem. Mõnedeks põhjusteks on WAN-i edastusmeediumite rohkus (nt. valguskaablid, telefoniliinid, sateliitühendused, traadita ühendused jne.), odavuse huvides olemasolevate võimaluste (nt. telefoniliinide) kasutamine, spetsialiseeritumad nõudmised (nii asutuse enda kui ka klientide poolt) ja kaablite vedamise kallidus (pikad vahemaad, kaevetööd jne.).

Mõningaid WAN tehnoloogiaid ja protokolle: DSL (*Digital Subscriber Line*), POTS (*plain old telephone service*) (tuntud lauatelefonina), sissehelistamine (dial-up), GSM (*Global System for Mobile communications*), SONET (*Synchronous Optical Networking*), SDH (*Synchronous Digital Hierarchy*), ISDN (*Integrated Services Digital Network*), PPP (*Point-to-Point Protocol*), PPPoE (*PPP over Ethernet*), Frame Relay, ATM (*Asynchronous Transfer Mode*), PPPoA (*PPP over ATM*) MPLS (*Multi Protocol Label Switching*). Järgnevalt vaatleme mõningaid WAN-i tehnoloogiaid ja protokolle lähemalt.

17.1 DSL

Eeskätt kodumajapidamiste ühendumiseks internetiteenusepakkujatega on tänapäeval üheks kõige kasutatavamaks lahenduseks DSL (*Digital Subscriber Line*) tehnoloogiad. DSL tehnoloogiad on välja töötatud väga laialdaselt kasutusel olevate vaskkaablipõhiste telefoniliinide kasutamiseks. Telefoniühenduse puhul oli kasutusel väike osa madalsagedusalast, kuid kõrgemate sageduste ala, mida DSL võiks kasutada, oli kasutamata. See teeb DSL tehnoloogiate kasutamise odavaks ja populaarseks, eeskätt kodukasutajate hulgas. Vaata DSL-i kohta ka DSL konsortsiumi "DSL Forum" veebilehte aadressil http://www.dslforum.org.

Võrgust kasutajani suunduvat edastust nimetatakse **allavooluks** (*downstream*). Kasutajast võrku suunduvat **ülesvooluks** (*upstream*). DSL tehnoloogiad jagatakse sümmeetrilisteks ja asümmeetrilisteks. Sümmeetriliste tehnoloogiate puhul on allavoolu ja ülesvoolu ribalaiused samasugused, asümmeetriliste puhul erinevad. DSL tehnoloogiate peresse kuulub mitu tehnoloogiat: ADSL (*Asymmetric Digital Subscriber Line*), HDSL (*High bit (data) rate Digital Subscriber Line*), IDSL (*ISDN (Integrated Services Digital Network) Digital Subscriber Line*), RADSL (*Rate-Adaptive Digital Subscriber Line*), SDSL (*Single line/Symmetric Digital Subscriber Line*), VDSL (*very high speed DSL*). Vaata tabelist 17.1 lisainformatsiooni erinevate DSL tehnoloogiate kohta.

Kõige enam kasutatavad on ADSL tehnoloogiad. ADSL standardi tähis on "ITU-T G.992.1" (tuntakse ka nimetuse "G.dmt" all). ADSL on asümmeetriline DSL, sest ta erinevate suundade ribalaiused on erinevad. Allavoolu nimetusena kasutatakse tihti sõna allalaadimiskiirus ja vastuvoolu nimetusena üleslaadimiskiirus. ADSL puhul on mindud sellist teed, sest tavaliselt on kasutajate (näiteks kodud) suunas märgatavalt rohkem liiklust kui vastupidi. Näiteks veebilehe

külastamisel on HTTP päringud ja TCP kinnitused palju väiksema mahuga võrreldes serveri vastustega (HTML failid, pildid jms).

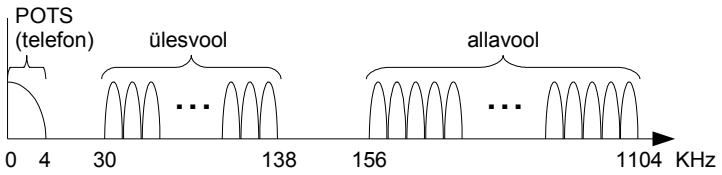
Tabel 17.1. Erinevate DSL perekonna tehnoloogiate andmed. Maksimaalsed väärtused sõltuvad eeskätt telefoniliini pikkusest, juhtme läbimõõdust ja häiretest. VDSL2-st on olemas kaks versiooni, millest alumine on kasutatav lühikestel vahemaadel. VDSL2 on tagasiühilduv ADSL2+'ga.

xDSL tüüp	vastu-voolu Mbit/s	päri-voolu Mbit/s	keerd-paare	maks. kaugus	modulat-siooni meetod	ITU tähis	kinni-tatud	telefoni toetus
ADSL	800 Kbit/s	8	1	5,5 km	DMT, CAP	G.992.1	1999	jah
ADSL2	1	12	1	7 km	DMT	G.992.3	2002	jah
ADSL2+	1	24	1	7 km	DMT	G.992.5	2003	jah
SHDSL	5,6	5,6			TC-PAM	G.991.2	2003	
HDSL	2,048	1,544	2	5,5 km	CAP, 2B1Q	G.991.1		ei
IDSL	144 Kbit/s	144 Kbit/s	1	11 km	2B1Q	-		ei
SDSL	2,3	2,3	1	6,9 km	2B1Q	-		ei
VDSL	15	55	1	0,9 km	QUAM	G.993.1	2004	ei
VDSL2	30	55			DMT	G.993.2	2005	
VDSL2	100	100			DMT	G.993.2	2005	

ADSL puhul on tänapäeval põhiliselt kasutusel modulatsioonisüsteem DMT (*Discrete Multi-Tone*), mis on ANSI standard "T1.413". Varem kasutati rohkem ka modulatsioonisüsteemi CAP (*carrierless amplitude/phase modulation*). ADSL on reversiivne, see tähendab, et sama traati kasutatakse mõlemas suunas edastuseks FDM (*frequency division multiplexing*) multipleksimise või kaja eemaldamise (*echo cancellation*) meetodil (kaetakse üle allavool ülesvoolu poolt ja nende eristamiseks kasutatakse kaja eemaldamise tehnikat). ADSL-i kasutades on telefoniliin jagatud kolmeks kanaliks (vaata ka joonist 17.1):

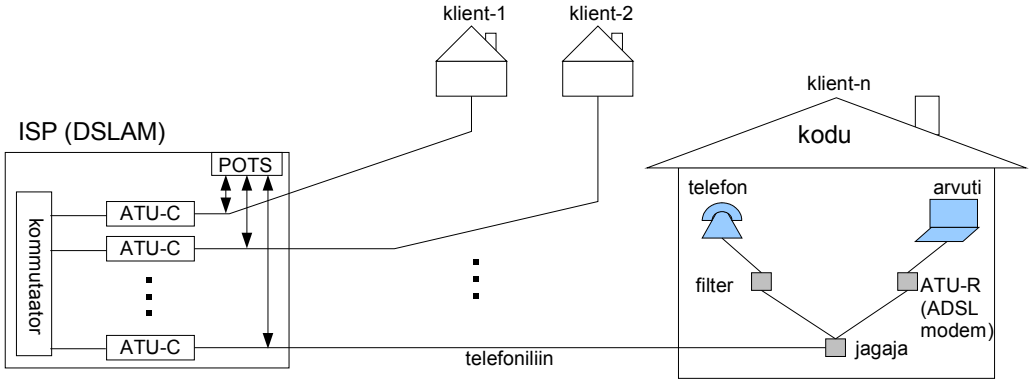
- POTS (*plain old telephone service*). Tavaline lauatelefoniühendus, mis töötab sagedusel 0-4 KHz.
- Ülesvoolukanal. Kasutab sagedusvahemikku 30-138 KHz. Sisaldab kokku 25 alamkanalit.
- Allavoolukanal. Kasutab sagedusvahemikku 156-1,1MHz. Sisaldab kokku 224 alamkanalit.

Kuna kõik kolm töötavad erinevatel sagedustel, siis nad on üksteisest sõltumatud. Seetõttu saab paralleelselt kasutada lauatelefoni ja Internetti. DMT puhul on sagedusvahemik jaotatud mitmetesse alamkanalitesse. Igal kanalil mõõdetakse kvaliteeti. Kõrgema sagedusega signaalid sumbuvad kiiremini, seetõttu ei pruugi kõrgemas sagedusalas olevad alamkanalid olla kasutatavad. Alamkanalite mittekasutatavus vähendab ühenduskiirust. Iga alamkanali sagedusvahemik on 4,3125 KHz.



Joonis 17.1: ADSL-i ribalaiuse kasutamine graafiliselt.

Ühenduse puhul on üks osapool kasutaja ehk klient ja teine pool võrk ehk ISP. ADSL kliendi poolset edastusseadet (tuntud ka kui CPE) nimetatakse ATU-R (*ADSL transceiver unit, remote terminal*) ja võrgu poolset seadet ATU-C (*ADSL transceiver unit, central office*). Kliendi pool jagatakse telefoniliin jagajaga kaheks, kus üks haru läheb ATU-R-i ehk ADSL modemisse ja teine haru läheb läbi filtri lauatelefoni. Filter filtreerib välja kõrgemad sagedused (ilma filtrita oleks lauatelefonis hääl mürane ja kõne katkeks tihti). ADSL modemis filtreeritakse omakorda välja telefoniliini kasutatav sagedusala. Kliendi poole minevale allavoolule pannakse ATU-C poolt väljuvale signaalile juurde lauatelefoni signaal. Kliendilt tuleva vastuvoolu puhul eraldatakse välja häälesignaal enne ATU-C'd. Mitu ATU-C-d on ühendatud DSLAM-ga (*DSL access multiplexer*). Vaata ka joonist 17.2. DSLAM on ATM või Etherneti kommutaator. Viimasel ajal on ATM põhised DSLAM-d asendatud Etherneti põhistega, mille põhjuseks võib pidada suuremaid ühenduskiiruseid, multiedastust, paketipõhist QoS, kõrget käideldavust, VPN ja muud funktsionaalsust, millest osa on raske implementeerida puhtas ATM-põhises keskkonnas.



Joonis 17.2: Joonis. ADSL ühenduste lihtsustatud ülesehitus. Internetiteenusepakkujaga (ISP) on telefoniliinide kaudu ühinenud mitu klienti.

ADSL2

ADSL2 on ADSL edasiarendus, mis on suurema jõudlusega ja enama funktsionaalsusega. Ta on standardiseeritud ITU-T poolt 2002. aastal standarditena G.992.3 ja G.992.4. ADSL2 puhul saavutati maksimaalseks allavoolukiiruseks 12 Mbit/s ja ülesvoolukiiruseks 1 Mbit/s.

ADSL2 täiendusi:

- Kiiruse ja liini pikkuse näitajate paranemine. Täiustati modulatsiooni efektiivsust, vähendati kadreerimise ballasti (ADSL puhul oli ballast fikseeritult 32 Kbit/s, ADSL2 puhul dünaamiliselt vahemikus 4-32 Kbit/s), muudeti efektiivsemaks kodeerimise tehnikad (Reed-Solomon kodeerimine), täiustati olekumasinate häälestust ja rakendati täiustatud signaalitöötluse algoritme, kärbiti elektrilist võimsust (vähendab ülekosteid ja väri-

naid) ja tehti muid madaltaseme laiendusi. ADSL2 on võimalik rakendada natuke pikemate ühenduste puhul kui ADSL-i.

- Lühem initsialiseerimisaeg. ADSL puhul umbes 10 sekundit, ADSL2-ga vähenes umbes kolmele sekundile.
- Kiiruse kohandumine. ADSL2 puhul kasutatakse automaatset kiiruse kohandumist. Vastuvõtja jälgib sideliini signaali ja müra suhet (*signal-to-noise ratio (SNR)*). Kui leitakse, et edastuskiirust on vaja muuta, siis saadetakse vastav signaal saatjale. Mürasid võivad tekitada teised telefoniliinid, mis võivad põhjustada ülekostet (tavaliselt koondatakse telefoniliinid kokku mitmekümne telefoniliiniga juhtmekimpudeks).
- Diagnostika võimekust on tublisti täiustatud (sideliini müra, sumbuvus, signaali ja müra suhe jms.), mille eesmärgiks on paremad veaotsimise ja jõudluse monitoorimise võimalused.
- Väiksem elektrikulu puhkerežiimis. ADSL transiiver töötas pidevalt täisvõimsusel, isegi kui sideliini ei kasutatud. ADSL2 puhul minnakse puhkerežiimi, kui ei ole pikka aega toimunud infovahetust ja väljutakse puhkerežiimist kiiresti, kui hakatakse ühendust kasutama. Selliselt võidakse kokku hoida palju elektrienergiat ja vähendada eeskätt ISP poolset seadmete jahutusprobleemi.
- Võimalus siduda kaks või enam traadipaari, võimaldamaks kiiremat ADSL2 ühendust (sisuliselt ühenduskiiruse kahekordistamine).
- Võimalus jagada ribalaius mitmeks kanaliks, mis võivad olla erinevate parameetritega. Näiteks saab võimaldada heliedastusprogrammidele eraldi kanalit, sest heliedastus võib olla kõrgema bitiveaga, kuid madala viivitusega. Andmete edastuse puhul on oluline madal vigade sagedus, kuid võib olla pikema viivitusega. ADSL2 võimaldab CVoDSL (*Channelized Voice over DSL*) kasutamist. Erinevalt VoIP-st toimib CVoDSL DSL füüsilisel tasemel. CVoDSL vajab 64 Kbit/s ribalaiust.
- Kui sideliin on seadistatud täielikult digitaalsena, siis on võimalik kasutada signaale alates 3 KHz-st, mis annab üleslaadimiskiirusele lisaks 256 Kbit/s lisaribalaiust (tavalist lauatelefoni ei saa sel juhul kasutada).
- Võimalus kasutada ADSL2 ühenduse peal paketipõhiseid teenuseid, näiteks Ethernet.

ADSL2-s on säilinud võimalus koos lauatelefoniga tegutsemiseks, kasutatakse samu elektrilisi nivoosid, sagedusvahemikke ja alamkanalite sagedusvahemikke.

ADSL2+

ADSL2+ standard anti ITU-T (G.992.5) poolt välja 2003. aastal. ADSL2+ muudatused võrreldes ADSL2-ga on järgmised:

- Sagedusala suurendati 2,2 MHz-ni.
- Suurendati alamkanalite arvu 512-ni. Suurendati ainult allavoolu alamkanalite arvu (vähem kasutusel oleva "Annex M" puhul on suurendatud ülesvoolu kiirust 3,5 Mbit/s).
- Suurendati allalaadimiskiirust 24 Mbit/s.
- Lisati mõningaid ülekostete vähendamise võimalusi.

ADSL2+ puhul on pikim distants maksimaalse kiiruse puhul 900 meetrit (ADSL puhul 1,8 km). Pikema distantsi puhul vähendatakse alamkanalite arvu (millega väheneb kiirus). ADSL2+ omab kiiruse kasvu võrreldes ADSL2-ga ainult lühemate distantside puhul.

17.2 PPP

PPP (*Point-to-Point Protocol*) (RFC 1661, 2153) on punktist-punkti sideliini kanalikihiprotokoll andmete edastamiseks, mida kasutatakse peamiselt WAN-is. PPP protokoll on paindlik ning seda on võimalik kasutada paljudel füüsilise kihi protokollidel ja tal on toetus mitmetele võrgukihi protokollidele ka samaaegselt. PPP ühendused saavad olla nii sünkroonsed kui ka asünkroonsed, kuid nad peavad olema täisdupleksühendused. PPP oli kasutusel näiteks sissehe-

listamisteenuste puhul. ADSL ühenduste puhul kasutatakse näiteks PPP vorme PPPoE ja PPPoA. PPP on ehitatud HDLC (*High-Level Data Link Control*) protokollile. PPP koosneb kahest alamprotokollist: LCP (*Link Control Protocol*) ja NCP (*Network Control Protocol*). LCP kasutatakse sideliinil ühenduse loomiseks, seadistamiseks ja testimiseks. NCP ülesanne on võrgukihi protokolli häälestamine.

LCP ülesanded:

- Paketi suuruse limiidiga tegelemine.
- Põhiliste vigaste seadistuste avastamine.
- Sideliini termineerimine.
- Sideliini funktsioneerimise jälgimine.

LCP poolt pakutavad valikud:

- Autentimine. Kasutatakse selleks, et määrata, kas ühenduse looja peab ennast autentima ja vajadusel autentimist teostama.
- Pakkimine. Saab määrata, kas kasutatakse andmete pakkimist.
- Vigade avastamine. Välditakse tsükleid, saates LCP teate koos kaasa pandud nn. maagilise numbriga. Kui saadakse tagasi sama maagilise numbriga LCP teade, siis järelikult on tegu tsükliga.
- Mitmik-sideliin. Võimaldab ühenduseks kasutada korraga mitut sideliini.
- Tagasihelistamine. Annab vastaspoolele märku, et helistataks tagasi ja alles seejärel luuakse ühendus. Seda kasutatakse turvalisuse tõstmiseks.

NCP protokoll tegeleb võrgukihi protokolliga, võimaldades sama sideliini kasutada mitmel võrgukihi protokollil. Selleks on igale protokollile oma NCP. Näiteks IP puhul IPCP ja IPX puhul IPXCP.

PPP kaadri väljad:

- Lipp. Osutab vastavalt kaadri algusele või lõpule. Välja binaarkuju on "01111110"
- Aadress. Välja binaarkuju "11111111" (leviedastus). PPP puhul ei määrata aadresse.
- Kontroll. Fikseeritud väärtus, binaarselt "00000011", mis viitab mittenummerdatud infole. 1 bait.
- Protokoll. Määrab kaadris kapseldatud edastatava protokolli. Välja esimesed bitid kuueteistkümnendkujul on järgmised: 0x0-0x3 kuuluvad võrgukihi protokollidele (nt. IPv4 on 0x0021); 0x4-0x7 kuuluvad vähese liiklusega protokollidele, mis ei ole seotud NCP-ga; 0x8-0xB kuuluvad NCP-dele (nt. 0x8021 kuulub IPCP-le); 0xC-0xF kuuluvad kanalikihi juhtprotokollidele (nt. 0xC021 on LCP, 0xC023 on PAP, 0xC223 on CHAP). 1 või 2 baiti.
- Andmed. Null või enam baiti.
- Kontrollsumma. 2 baiti.

PPP ametlikud koodid on kättesaadavad aadressil http://www.iana.org/assignments/ppp-numbers.

PPP ühenduse püstipanek toimub kolmes etapis:

1. Sideliinil ühenduse loomise faas. Osapoolte vahel luuakse suhtluskanal. Mõlemad pooled saadavad LCP kaadreid, millega seadistatakse ja testitakse sideliini. Seadistamisel määratakse ära MTU (*maximum transmission unit*), pakkimise kasutamine, kas kasutatakse autentimist (mittemääramisel kasutatakse vaikeväärtusi).
2. Autentimise faas. Kui eelmises faasis nõuti autentimist, siis alustatakse autentimisega, vastasel korral jäetakse autentimisfaas vahele. Selles faasis võib teostada ka sideliini kvaliteedi kontrolli, kas see on piisavalt hea võrgukihi protokolli jaoks.
3. Võrgukihi protokolli faas. Mõlemad osapooled seadistavad ühe või enam võrgukihi protokolli (näiteks IP). Kui see faas on läbi, siis on PPP ühendus mõlema osapoole vahel loodud ja saab hakata andmeid vahetama.

PPP ühendus suletakse, kui LCP või NCP kaadritega lõpetatakse ühendus, fikseeritud aja jooksul ei ole toimunud sideliinil liiklust või kui väliselt katkestatakse ühendus (näiteks kasutaja poolt).

PPP puhul on võimalik autentimisprotokollidena kasutada PAP (*Password Authentication Protocol*) või CHAP (*Challenge Handshake Authentication Protocol*) protokolle. PAP protokolli puhul saadab ühenduse alustaja autentimisinfo (kasutajatunnus ja parool) ja teine osapool kontrollib saadud infot ja saadab tagasiside autentimise õnnestumise või ebaõnnestumise kohta. Kui autentimine õnnestus, siis jätkatakse PPP ühendusega, vastasel korral suletakse ühendus. PAP puhul saadetakse kasutajatunnus ja parool avatekstina, seega on võimalik sideliini pealt kuulates autentimisinfo teada saada. PAP nimetatakse ka kaheosaliseks käepigistuseks.

Teine autentimisprotokoll on CHAP, mida nimetakse kolmeosaliseks käepigistuseks. CHAP puhul saadetakse ühenduse algatanud osapoolele juhuslikult genereeritud number, mille ta jätab omale ajutiselt meelde (autentimisfaasiks). Ühenduse algataja räsib saadud numbri oma parooliga ja saadab saadud räsi (räsimiseks kasutatakse näiteks MD5 räsialgoritmi). Seejärel teine osapool teostab kontrolli, mille käigus ta räsib samuti sama numbri parooliga ja kui saadud räsid langevad kokku, siis loetakse autentimine edukalt läbituks. Edasi saadetakse ühenduse algatajale tagasiside autentimise õnnestumise või ebaõnnestumise kohta ning minnakse edasi PPP ühendusega või katkestatakse ühendus. CHAP protokolli kasutamisel ei saa pealt kuulates parooli teada, seega on ta turvalisem PAP-st. Kuna saadetakse juhuslik arv, siis ei ole praktiliselt võimalik ka taasesitusrünne. Autentimist võidakse korrata suvalisel ühenduse ajal.

PPPoE

PPPoE (RFC 2516) teostab PPP ühendust üle Etherneti (Etherneti tüübivälja väärtus on 0x8863). PPPoE-d kasutatakse laialdaselt ADSL ühendustega. PPPoE puhul on vajalik alguses luua ühendus, mille järel saab luua PPPoE seansi. PPPoE ühendusel on kaks etappi: avastamise (*discovery*) ja seansi etapp (Etherneti kaadri tüübivälja väärtused on vastavalt 0x8863 ja 0x8864). Avastamise etapis saadakse teada osapoolte MAC-aadressid ja küsitakse serverilt seansi identifikaatorit. Avastamise sammud on järgmised:

1. Initsieerimine. PPPoE klient ei tea serveri MAC-aadressi. Selleks saadab ta välja PADI (*PPPoE Active Discovery Initiation*) paketi Etherneti leviedastusaadressil.
2. Pakkumine. PPPoE server, nimetatakse ka ligipääsukontsentraatoriks (*access concentrator*) (servereid võib olla ka mitu), vastab PADO (*PPPoE Active Discovery Offer*) paketiga kliendi Etherneti üksikedastusaadressil, mille sai PADI paketist.
3. Taotlus. Kui klient saab mitu PADO paketti (mitu serverit vastasid), siis valib välja klient sobiva, tavaliselt esimesena saadu. PPPoE klient vastab PADR (*PPPoE Active Discovery Request*) paketiga.
4. Seansikinnitus. PPPoE server genereerib unikaalse seansiidentifikaatori PPP seansiks ja vastab PADS (*PPPoE Active Discovery Session-confirmation*) paketiga kliendi üksikedastusaadressil.

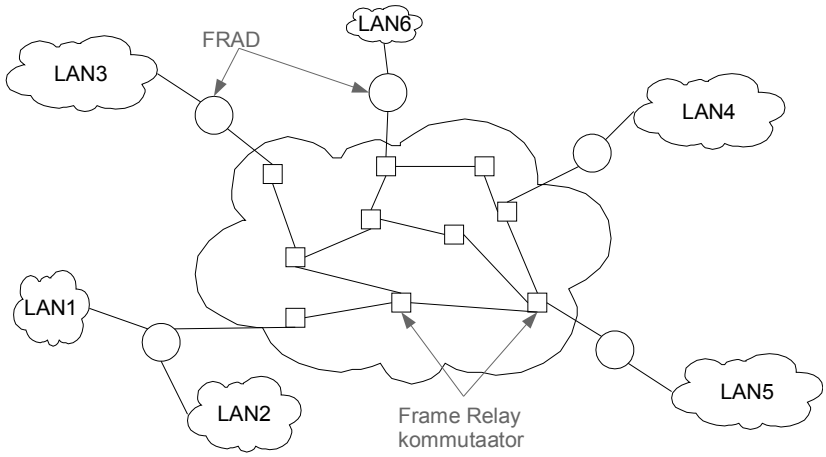
Peale edukat avastamisprotsessi kasutatakse saadud informatsiooni, et luua punktist-punkti ühendus üle Etherneti ehk jõutakse seansi etappi, kus saab vahetada andmeid.

Etherneti poolt ülekantava PPPoE kaadri väljad on järgmised:

- Versioon. PPPoE versioon. Peab olema 0x1. 4 bitti.
- Tüüp. Peab olema 0x1. 4 bitti.
- Kood. Vaata koodide registreeringuid aadressil: http://www.iana.org/assignments/pppoe-parameters. 8 bitti.
- Seansi identifikaator. 16 bitti.
- Ülekantavate andmete pikkus. 16 bitti.

17.3 Frame Relay

Kohtvõrkude ühendamiseks on võimalik kasutada punktist-punkti ühendusi, kuid mitmete ühenduste puhul on puhtakoelised punktist-punkti ühendused sideliini ressursikasutuse seisukohalt kallid ja ebaefektiivsed. Enamasti on olemas vajadus sideliine efektiivselt kasutada ja hoida kokku füüsilisi ühendusi, sest nende rajamine on kallis. Mõistlik on paigaldada üks suurema läbilaskevõimega sideliin ja jagada läbilaskevõimet väiksemateks osadeks, tehes seda virtuaalsete ühendustena. Järgnevalt vaatleme WAN-s kasutatavat protokolli Frame Relay, millega saab ühte või mitut sideliini kasutades luua mitu virtuaalset ühendust. Vaata ka joonist 17.3.



Joonis 17.3: Näide Frame Relay võrgust ja sellega ühendumisest. Igal füüsilisel sideliinil on defineeritud loogilised lülitused, mille põhjal suunatakse kaader edasi määratud suunda. FRAD on Frame Relay võrgu ääremasin, milleks võib olla näiteks marsruuter või tavaline personaalarvuti.

Frame Relay oli mõeldud protokolli X.25 väljavahetamiseks. Väljatöötamisel seati eesmärkideks põhiliselt võimalikult väike viivitus, protokolli lihtsus, väike maksumus ja efektiivsus (nt. vähe ballasti). X.25 disainimise ajal ei olnud sideliinid kuigi kvaliteetsed, mistõttu esines vigu suhteliselt sagedasti, kuid Frame Relay disainimise ajal oli sideliini kvaliteet tublisti kasvanud ja seetõttu eeldati, et sideliinidel on andmete saatmisel tekkivate vigade hulk tühine. Frame Relay on mõeldud kasutamiseks võrgu äärealadel. Frame Relay on OSI kanalikihi protokoll, mille jaoks ei ole oluline, millist ülemise kihi protokolli ta edastab. Frame Relay on pakett-kommutatsioon ja ühendus-orienteeritud protokoll, mis kasutab virtuaalseid lülitusi füüsilisel sideliinil. Frame Relay on nagu eraldi võrk, mida nimetatakse Frame Relay võrguks, milles asuvaid võrguseadmeid nimetatakse Frame Relay kommutaatoriteks (edaspidi Frame Relay osas nimetame neid lihtsalt kommutaatoriteks). Iga kommutaator omab kasutaja poolt defineeritud tabelit, mis sisaldab lähteliidest, lähteliidese virtuaalse lülituse numbrit, sihtliidest ja sihtliidese virtuaalse lülituse numbrit. Kui kommutaator saab kaadri, siis ta otsib tabelist kirjet, mis käib vastava võrguliidese kohta ja mille virtuaalse lülituse number vastab kaadris märgitud virtuaalsele lülitusele. Leidnud vastava kirje, saadab kommutaator kaadri edasi kirje juures märgitud võrguliidese kaudu ja vastava virtuaalse lülituse numbriga (selliselt on virtuaalsete lülituste numbrid sideliinidel autonoomsed).

Frame Relay kaadri formaat:

- Kaadri alguse eraldaja. Kaadri eraldajaks on kahendkujul "01111110". 1 bait.
- Aadress. 2 või 4 baiti. Koosneb alamväljadest:

- ◇ DLCI (*data link connection identifier*) – virtuaalse lülituse numbri esimene osa. 6 bitti.
- ◇ C/R (*command/response*) – rakendusespetsiifiline, Frame Relay ei kasuta. 1 bitt.
- ◇ EA (*extended address*) – laiendatud aadressi lipp. Kui lipp on püsti, siis aadress on lõppenud, kui ei ole, siis aadressiosa jätkub (uues baidis). Selliselt on võimalik kasutada tulevikus vajadusel pikemaid aadresse. 1 bitt.
- ◇ DLCI jätk.
- ◇ FECN (*forward explicit congestion notification*). Kasutatakse ummistustest teavitamiseks. Lipp pannakse püsti, kui kaader saadetakse võrguliidese kaudu, mis on ülekoormatud.
- ◇ BECN (*backward explicit congestion notification*). Lipp pannakse püsti ummistuse tekkimise vastassuunas (kaadritele, mis saadakse ummistunud võrguliidesest). BECN lipuga kaadriga proovitakse saatjat teavitada ummistusest, et vähendataks saatmise kiirust.
- ◇ DE (*discard eligibility*). See lipuke tähistab vähemtähtsat kaadrit. Kui peaks toimuma ummistus, siis esimeses järjekorras visatakse ära DE lipuga kaadrid.
- ◇ EA (*extended address*) – laiendatud aadressi lipp, mis võimaldab tulevikus veelgi laiendada aadressi pikkust. 1 bitt.

- ◆ Edastatavad andmed. Suurus on muutuv ja standard ei sea otseselt piirangut pikkusele, kuid tavaliselt on suurim kogu kaadri suurus 4096 baiti.
- ◆ Kontrollsumma. 2 baiti.

Frame Relay puhul eeldatakse, et kasutatava võrgu sideliinid on kvaliteetsed. Seetõttu ei saadeta probleemide esinemisel mitte mingit tagasisidet (näiteks IP võrgus saadeti osade probleemide korral ICMP pakett saatjale). Andmete taassaatmised peavad vajadusel olema realiseeritud ülemistes kihtides.

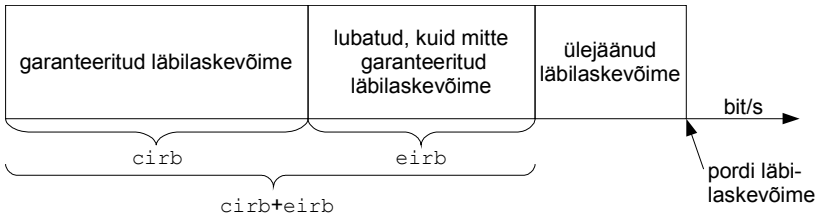
Frame Relay virtuaalsete ühenduste arv (DLCI väli) on maksimaalselt 2^{10}=1024. DLCI väärtuste vahemikud on määratletud:

- ◆ 0 – LMI (*Local Management Interface*) (ANSI, ITU). Kasutatakse kõne juhtimiseks.
- ◆ 1-15 – reserveeritud tulevikus kasutamiseks.
- ◆ 16-991 – DLCI numbrid, mida saab määrata virtuaalsete lülituste määratlemiseks. Virtuaalsete ühenduste numbrid on tavaliselt sõltumatud igal sideliinil. Seega sama ühenduse puhul erinevate kommutaatorite vahel võib DLCI number olla erinev ja erinevate ühenduste jaoks erinevatel sideliinidel võib kasutada sama numbrit.
- ◆ 992-1007 – CLLM (*Consolidated Link Layer Management*)
- ◆ 1008-1022 – reserveeritud tuleviku vajadusteks (ANSI, ITU).
- ◆ 1019-1020 – multiedastuseks (Cisco).
- ◆ 1023 – LMI (Cisco). Vahetatakse infot virtuaalsete ühenduste kohta.

Frame Relay võrguga kokku puutuvaid võrguseadmeid nimetatakse FRAD-ideks (*Frame Relay access device*, ka *Frame Relay Assembler/Disassembler*) (tavaliselt on marsruuterid), mis saavad näiteks IP paketi ja suunavad selle määratud virtuaalse ühenduse kaudu edasi mingile teisele FRAD-ile, mis saadab paketi edasi jällegi IP-võrku. FRAD-id on ühendatud Frame Relay kommutaatorite kaudu, mis suunavad kaadri edasi määratud loogiliste ühenduste kaudu soovitud sihtpunkti ehk FRAD seadmele.

Frame Relay igale virtuaalsele lülitusele on ette nähtud mingi määratud ühenduskiirus. Tavaliselt on ühenduskiiruste väärtused väiksemad kui sideliini ühenduskiirus (suurem ei saa olla), kuid summeerituna suurem kui sideliini ühenduskiirus (eeldatakse, et kõik virtuaalsed lülitused ei kasuta korraga kogu neile eraldatud ribalaiust). Samas peab virtuaalsetel lülitustel saatmine toimuma sideliini üldisel kiirusel. Arvepidamine ühenduskiiruste kohta toimub fikseeritud ajalõigus bitte loendades (tavaliselt sekund või vähem). Igas ajalõigus loendatakse iga virtuaalse lülituse edastatud bitte. Igale virtuaalsele lülitusele on garanteeritud mingi läbilaskevõime, mida nimetatakse CIR-iks (*committed information rate*) ja mõõdetakse bit/s (bitti sekundis). CIR ja

ajalõigu väärtusest tuletatakse garanteeritud bittide arv ajalõigus (tähistame siin `cirb` (CIR-i bitid)). Kui vastava ajalõigu sees peaks virtuaalset lülitust läbima rohkem bitte kui `cirb`, siis pannakse kaadris püsti lipp DE. Seega on võimalik kasutada kehtestatust suuremat ribalaiust, kui antud ajalõigus vaba ribalaiust on. Iga ajalõigu alguses nullitakse loendurid ja alustatakse loendamist uuesti. Frame Relay juures on olemas ka võimalus piirata CIR ületamise ulatust. Seda lisaribalaiust nimetatakse EIR-iks (*excess information rate*). Ajalõigu pikkusest ja EIR väärtusest tuletatakse lubatud bittide arv ajalõigus (tähistame siin `eirb` (EIR-i bitid)). Kui ajalõigus ületab virtuaalse lülituse loendur `cirb+eirb` väärtuse, siis visatakse vastava virtuaalse lülituse kaadrid antud ajalõigus koheselt minema. Vaata ka joonist 17.4. Iga ajalõik on sõltumatu. Kui ajalõigus reserveeritud osa ribalaius jääb kasutamata, siis ei anna see eelisõigust järgnevates ajalõikudes. Virtuaalse ühenduse CIR ja EIR väärtused võivad olla erinevates suundades samad või ka erinevad (vastavalt sümmeetriline või asümmeetriline).



Joonis 17.4: Frame Relay poolt eraldatavad ribalaiused.

Frame Relay proovib hoida järjekorrad võimalikult lühikesed, et andmete saatmisel oleksid viivitused väiksemad. Seetõttu rakendatakse erinevaid mooduseid, mismoodi hoida järjekorrad väikesed, kui need peaksid tekkima. Järjekordade tekkimisel visatakse esimestena järjekorrast välja kaadrid, millel on püsti DE lipp. Lisaks kasutatakse ummistustest teavitamiseks FECN ja BECN lippe. Frame Relay võrku läbides püsti pandud DE, BECN, FECN lippe maha ei võeta, mille tulemusena jõuavad nad FRAD-ni, mille vookontroll võib vastavat informatsiooni kasutada. Selle tulemusena võidakse näiteks pikendada aega, millal soovitakse kinnitust saadetule. Samuti saab TCP puhul vähendada akna suurust kuni ummistuse vähendamiseni. Viivituse minimeerimine on tihti väga oluline. Näiteks TCP loeb protokoll saadetud segmendi kaotsiläinuks, kui ei saada määratud aja jooksul vastuvõtjalt vastust kättesaamise kohta. Kordussaatmine aga suurendab niigi ummistunud sideliini koormust.

Frame Relay võrgus võib tekkida vajadus dünaamiliselt edastada informatsiooni võrgu oleku kohta, eeskätt FRAD-dele. Selleks on Frame Relayle laiendusena välja töötatud protokoll LMI (*Local Management Interface*). LMI jaoks on reserveeritud mõned DLCI numbrid. LMI-st on olemas mitu erinevat omavahel mitte ühilduvat protokolli: Cisco (esimene LMI laiendus), Ansi (ANSI standard T1.617 Annex D), q933a (ITU standard Q933 Annex A). LMI kasutatakse näiteks:

- Virtuaalse ühenduse funktsioneerimise kontrollimiseks. Selleks saadetakse iga 10 sekundi tagant elusoleku (*keep alive*) teateid.
- Multiedastuse teostamiseks. Näiteks kasutatakse marsruutimisuuenduste edastamiseks ja aadressilahenduste korral.
- Vookontrolliks.
- DLCI-dele globaalse tähenduse andmise võimalus.
- Virtuaalsete ühenduste staatuse mehhanism.

Kui FRAD-seade sisse lülitatakse, siis ta saadab staatuse päringu LMI-teate. Selliselt saadakse teada, millised virtuaalsed ühendused on olemas FRAD-i jaoks. Staatuste päringute tegemist jätkatakse ka määratud ajavahemiku tagant jooksvalt. Kui vajatakse virtuaalsele ühendu-

sele vastavat võrgukihi protokolli aadressi, siis kasutatakse "Inverse ARP" teadet, millele vastatakse võrgukihi aadressiga (mitme erineva võrgukihi protokolli puhul saadetakse vastus iga protokolli kohta).

Frame Relay võimaldab kasutada kahte tüüpi virtuaalseid ühendusi: PVC (*permanent virtual circuits*) ja SVC (*switched virtual circuits*). PVC on püsiv virtuaalne ühendus. SVC on virtuaalne lülitus, mis suletakse, kui fikseeritud aja jooksul ei ole seda virtuaalset lülitust kasutatud. Kui seda uuesti kasutama hakatakse, siis peab enne looma ühenduse, mis võtab aga natuke aega. Üldiselt on kasutusel PVC lülitused. SVC-d kasutati harva siis, kui oli vaja sulgeda ühendus, mille pealt maksti ajakasutustasu.

Frame Relay probleemiks on asjaolu, et kui peaks tekkima ummistus, siis hakatakse järjekordi vähendama. See aga võib tähendada, et Frame Relay võrku läbiva videoülekande osad kaadrid ei jõua kohale, kuid peale aegumist neid uuesti saata pole ka enam mõtet. See võib tähendada aga katkendlikku pilti ja heli. Selle jaoks on välja töötatud VOFR (*voice over frame relay*) FRAD-ide jaoks. VOFR täiustuste alla kuuluvad heli pakkimine, kõnepauside ja kaja elimineerimine ja viivituse juhtimise tehnikad (näiteks prioriteedid, kus heliedastusel on kõrgem prioriteet võrreldes andmete edastusega ja heliedastuse kaadrite tükeldamine). Frame Relay reputatsioonile on halvasti mõjunud mitmete Frame Relay teenust pakkuvate firmade klientidele eraldatud CIR-ide kogusumma liialt suur füüsilise sideliini läbilaskevõime ületamine, mille tulemusena kliendid tegelikult ei saagi kasutada kogu neile garanteeritud ribalaiust. Seda nimetatakse ka ülereserveerimiseks (*overbooking*). Paljud kliendid on suures osas seepärast üle läinud MPLS-il baseeruvatele protokollidele.

17.4 ATM

ATM (*Asynchronous Transfer Mode*) on ühendusorienteeritud protokoll, mille disainimisel peeti silmas:

- Toetust mitmele kasutajale paralleelselt.
- Toetust erinevatele reaalaja ja mitte-reaalaja kaugühenduste vajadustele (nt. telefon, videoedastus, andmeedastus, kohtvõrkude ühendamine jne.).
- Erinevate rakenduste vajadust erinevate ühenduskiiruste järele.

ATM mudelis loob saatja ühendused vastuvõtjatega ehk virtuaalsed lülitused. ATM kommutaatorid lisavad juurde vastava marsruudi. ATM lülitusi on seega võimalik teha punktist-punkti või multiedastuse põhimõttel. Kui virtuaalset lülitust enam ei vajata, siis vabastatakse virtuaalne ühendus ja eemaldatakse ATM kommutaatoritest marsruutimisinfo.

ATM kasutab fikseeritud pikkusega kaadreid, mida nimetatakse rakuks. ATM-i rakk on 53 baidi suurune, kus andmeid on 48 baiti ning 5 baiti on kaadri päis. ATM tükeldab andmed väikesteks tükkideks, millega saavutatakse väiksemad viivitused. Samas on ATM puhul ballast üldjuhul suurem, kui teistel protokollidel, näiteks võrrelduna Frame Relayga. ATM raku ballast on stabiilselt ~10%.

ATM on mõeldud rakendustele, mis nõuavad võimalikult väikest viivitust, nagu näiteks heli- ja videoülekanded, kus on vajalik stabiilse suurusega ja võimalikult väikese viivitusega sidekanal. ATM puhul on võimalik kasutada nii PVC kui ka SVC lülitusi, kuid põhiliselt kasutatakse PVC lülitusi (nagu ka Frame Relay puhul). ATM võimaldab samuti ühe füüsilise ühenduse peal luua mitu virtuaalset kanalit võrgu äärealadel. Kaadrite väljasaatmine toimub pidevalt, isegi kui midagi saata ei ole – siis saadetakse lihtsalt tühi kaader.

ATM kaadri ülesehitus on järgmine:

- Vookontroll. Pakub lokaalseid funktsioone, näiteks juhul, kui mitu masinat jagavad ühte ATM liidest. Ei kasutata tihti (märgitakse nulliks). 4 bitti.
- Virtuaalse tee identifikaator. 8 bitti.
- Virtuaalse kanali identifikaator. Saatmine otsustatakse marsruutimistabeli põhjal (koos virtuaalse tee identifikaatoriga). 16 bitti.

- Andmete tüüp. Eristatakse, kas tegemist on edastatavate andmetega või ATM juht-teadetega (bitt vastavalt 0 või 1). Teine bitt on ummistustest teavitamise lipp ja kolmas lipp annab teada, kas tegemist viimase rakuga kaadrist (kaader enamasti suurem, kui rakk, mistõttu saadetakse mitme rakuna). 3 bitti.
- Vähem tähtsa kaadri lipp. Kui lipp on püsti, siis tähistab, et tegemist on vähemtähtsa kaadriga, mis eelistatult visatakse ära ummistuse tekkimisel. 1 bitt.
- Päise kontrollsumma. Kontrollsumma arvutatakse ainult kaadri päise pealt, andmeid arvestamata. 8 bitti.

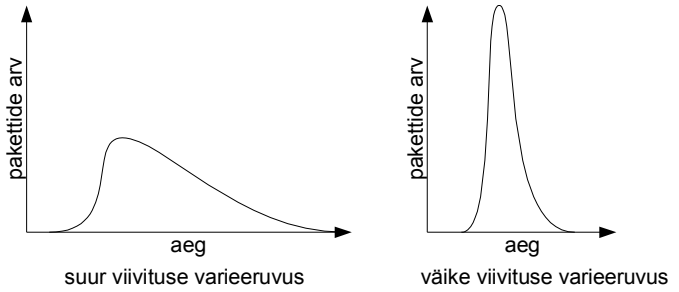
18. Teisi kasulikke tehnoloogiaid

18.1 QoS ehk teenusekvaliteet

Võrguliikluse puhul on tavapärane, et võrguseadmetes võivad tipptundide ajal tekkida ummistused, mis põhjustavad suuremaid viivitusi andmete edastamistel ja puhvrite täissaamisel pakettide äraviskamisi. Muidugi võiks panna kõikide otspunktide vahele piisavalt suure läbilaskevõime ja väikese viivitusega sideliinid ning võrguseadmed, et ka võrgu kasutuse haripunktis ei oleks ummistusi. Kuid see on väga kallis ja tihti praktiliselt mitteteostatav, sest otspunktid ei pruugi olla sama asutuse võrgus.

Erinevad arvutivõrgu kaudu pakutavad teenused vajavad erinevaid tingimusi. Näiteks telefonikõne vajab väikese viivitusega ja stabiilset ribalaiust, kus on lubatav mõnede pakettide kaotsiminek. Samas näiteks FTP-serverist faili transportimisel on oluline peamiselt ühenduse usaldusväärsus ja ka võimalikult suur läbilaskevõime, kuid pakettide saatmiste viivitused ja läbilaskevõime võib andmevahetuse käigus tugevasti varieeruda ilma suuremate probleemideta.

Teenusekvaliteediks ehk QoS-ks (*quality of service*) nimetatakse kogumikku kvaliteedinõudeid ühele või mitmele teenust pakkuvale objektile. Neli peamist teenusekvaliteedi parameetrit on läbilaskevõime, viivitus, usaldusväärsus ja viivituse varieeruvus (*jitter*). Viivituse varieeruvuse parameeter näitab pakettide saabumise aja ühtlust (vt. joonis 18.1). Mida ühtlasema ajaga paketid sihtpunkti jõuavad, seda parem. Viivituse varieeruvus on tähtis näiteks nõudevideo ehk VoD (*video on demand*) korral (võimaldab vaadata videosid kasutajale sobival ajal, mitte otseülekandena).



Joonis 18.1: Viivituse varieeruvus (*jitter*) graafiliselt. Väikese viivituse varieeruvuse korral jõuavad paketid sihtpunkti väiksema ajavahemiku sees, kui tavaliikluse pakettide korral.

QoS ei ole uus kontseptsioon. Juba telefoniühenduste puhul tekkisid vajadused prioritiseerida telefonikõnesid keskjaamas. QoS ei anna juurde täiendavat ribalaiust, kuid võimaldab pakkuda garanteeritud teenusekvaliteeti, kui võimalik. Teenusekvaliteedi tagamine ühele saatjale ummistunud ühendusteede puhul saab tulla teiste saatjate arvelt. Kui mitu osapoolt kasutavad samaaegselt teenusekvaliteedi teenust ja juba nende pärast tekivad ummistused, siis ei ole võimalik ka teenusekvaliteeti tagada. Sama kehtib ka rikete korral. Teenusekvaliteeti saab põhimõtteliselt rakendada ainult juhul, kui on tekkinud järjekord, mis üldjuhul saab tekkida saatmisel.

QoS-i on väga raske objektiivselt mõõta, nii tehniliselt kui ka teoreetiliselt. Ta ei peegelda otseselt võrgu jõudlust ja käitumist. Teenusekvaliteedi tagamiseks peavad kõik vahepealsed võrguseadmed teenusekvaliteedi funktsiooni toetama.

Teenusekvaliteedi tagamiseks ei ole ühest lahendust. Kasutatakse mitmeid lahendusi. Tihti realiseeritakse teenusekvaliteeti pakettide prioritiseerimist kasutades. Kõrgema prioriteediga

pakettidega tegeldakse esimesena, sellega vähenevad viivitused nende edastamisel. Ummistuste korral visatakse ära vähemprioriteetsemad paketid. IPv4 spetsifikatsioonis oli algselt 8-bitine teenusetüübi väli, kus esimesed kolm bitti tähistasid paketi prioriteeti, millede nimetused olid järgmised (tähtsuse järjekorras): 111 – võrgu juhtimine (*network control*); 110 – võrkude vaheline juhtimine (*internetwork control*); 101 – CRITIC/ECP (*critical and emergency call processing*); 100 – flash override; 011 – flash; 010 – kohene (*immediate*); 001 – prioriteetne (*priority*); 000 – tavaline (*routine*). Neljas bitt oli madala viivituse lipp, viies bitt kõrge läbilaske lipp, kuues bitt kõrge töökindluse lipp. Viimased kaks bitti olid reserveeritud tulevikus kasutamiseks. Kuna teenusetüübi välja oli kõigil võimalik muuta kõige prioriteetsemaks, siis tänapäeval on IPv4 teenusetüübi väli asendatud diferentseeritud teenuste väljaga. Kanalikihi tasandil on VLAN-de puhul loodud teenusekvaliteedi jaoks protokoll IEEE 802.1p, mida kasutavad ka standardid IEEE 802.1D (MAC-aadressipõhised sillad) ja IEEE 802.1Q (*trunking* protokoll). IEEE 802.1p võimaldab kaadrid määrata kaheksasse erinevasse teenuse klassi.

Viivituse varieeruvuse vastu aitab, kui vastuvõtja pool eelnevalt puhverdab vastuvõetut (nt. 5-15 sekundit), mis aitab näiteks nõudevideo vaatamisel, kuid näiteks telefoniühenduses ei ole see meetod kasutatav.

Mõnikord on võrgukasutus kindla mustriga (nt. teostatakse perioodilisi tagavarakoopiate tegemisi mitmest osakonnast või videokonverentse), siis on võimalik teenusekvaliteedi tagamises ISP-ga kokku leppida (kui ISP ei suuda soovitut täita, siis ei saa kokku leppida). Vastavat kokkulepet nimetatakse ka teenusetaseme lepinguks ehk SLA-ks (*service level agreement*).

Teenusekvaliteedi tagamiseks võidakse kasutada ka spetsiaalseid protokolle ressursside eelnevaks reserveerimiseks voole saatjast vastuvõtjani jäävates võrguseadmetes (võidakse keelduda, kui ei ole võimalik ressursse reserveerida). Põhiline protokoll on RSVP (RFC 2205). Tehtud reserveeringute põhjal loodud voog on simpleks-ühendus. Mõlemapoolse teenusekvaliteediga ühenduse jaoks on vaja luua eraldi kaks teenusekvaliteediga tagatud voogu. RSVP kasutab optimaalse voo teekonna jaoks olemasolevaid marsruutimisprotokolle. RSVP võimaldab nii otse- kui ka multiedastuse põhist edastust. Kui teekonna marsruuter ei oma RSVP toetust, siis ta saadab RSVP-teated lihtsalt edasi ega reserveeri vastavale voole ressursse.

Teenuse kasutatavuse huvides on osadel juhtudel võimalik vajadusel vähendada edastatava kvaliteeti, näiteks video- ja heliülekandel, kui see peaks vajalikuks osutuma. Parema teenusekvaliteedi tarvis võidakse kasutada ka eri liiki pakettide prioritiseerimist. Näiteks on mõistlik määrata TCP kinnituste paketid prioriteetsemaks (TCP segmendid, mis ei sisalda andmeid). Selliselt vähendame võimalust, et teine osapool saadab segmente uuesti, sest ta ei saanud enne kinnituse saamise aegumist andmete kättesaamist kinnitavat segmenti. Tavaliselt vajavad kiiret reaktsiooniaega interaktiivsed rakendused, mis enamasti ei edasta palju andmeid (nt. SSH sessioonid), mille samuti võiks prioriteetsemaks määrata.

18.2 VPN

Enamikul asutustest on olemas sisevõrk, kus on kättesaadavad paljud teenused ja ressursid, millele väljastpoolt sisevõrku ei ole ligipääsu. Paljud asutuse töötajad võivad siiski vajada väljaspool asutust ligipääsu asutuse sisevõrgu teenustele ja ressurssidele. Neid inimesi nimetatakse ka mobiilseteks töötajateks. Teisest küljest võib asutus kasvada piisavalt suureks, et peab paiknema mitmes geograafiliselt erinevas kohas, kuid on vajadus ühtse sisevõrgu järele, aga eraldi sideliini rajamine on enamasti liialt kallis ettevõtmine. Peamiselt nende probleemide jaoks on kasutatav VPN (*virtual private network*). VPN-iga luuakse turvaline kanal üle potentsiaalselt ebaturvaliste võrkude (enamasti üle Interneti), mida nimetatakse tunneliks (asutustel on üldjuhul olemas Internetiühendus). VPN abil saab krüpteeritud tunneleid kasutades ühendada mobiilseid töötajaid ja kontoreid üle Interneti.

Kuna VPN ülesandeks on luua üle potentsiaalselt ebaturvalise Interneti turvalisi ühendusi, siis peab ta täitma järgmisi funktsioone:

- Autentimine. Andmed on pärit usaldatud osapoolelt.
- Ligipääsukontroll. Mittevolitatud ligipääsu vältimine.
- Konfidentsiaalsus. Kui sideliini ka pealt kuulatakse, siis ei saa vahetatavaid andmeid lugeda.
- Andmete terviklikkus. Mitte keegi ei tohi andmeid vahepeal muuta, eemaldada või lisada märkamatult.

VPN tunnelite teostamiseks on kasutusel mitmeid protokolle, millest tuntumateks on IPsec (*IP security*), PPTP (*Point-to-Point Tunneling Protocol*), L2TP (*Layer 2 Tunneling Protocol*), OpenVPN ning L2F (*Layer-2 Forwarding*) protokoll (Cisco firmaomane).

PPTP, L2F ja L2TP on peamiselt mõeldud sissehelistamisteenuse kasutamiseks. IPsec on fokusseeritud kohtvõrkude vaheliseks kasutamiseks. PPTP on välja töötatud Microsofti poolt ning see oli esimene laiemalt kasutusele võetud VPN lahendus (tingitud sellest, et serveri toetus lisati operatsioonisüsteemile Windows NT 4.0 ja VPN-kliendi toetus Windows 95-le). PPTP töötab kanalikihis ja ta võib koos opereerida ka mitte-IP protokollidega (näiteks IPX, NetBEUI). L2F oli sarnaselt PPTP-le üks esimesi VPN lahendusi. Erinevalt PPTP-st on L2F võimeline otse töötama peale IP ka näiteks Frame Relayl ja ATM-il. L2TP on IETF poolt tunnustatud standard, mis on edasiarendus PPTP ja L2F protokollidele. OpenVPN on avatud lähtekoodil arendatav VPN lahendus, mis kasutab autentimiseks TLS/SSL protokolli ja tunneli turvamiseks IPseci ESP protokolli üle UDP.

IPsec

Algne IP protokoll ei sisaldanud turvalise andmeühenduse funktsioone, kuid turvaline andmeside on muutunud möödapääsmatuks vajaduseks, mistõttu on IP protokollile välja töötatud vastavaid turvateenuseid. IPsec (*IP security*) oli algselt sisse disainitud IPv6-le ja kohendatud ka IPv4 jaoks. IPsec on kogumik protokolle, mida võiks pidada pigem raamistikuks (RFC 2401-2411, 4301-4309). IPsec loetakse kolmandasse kihti kuuluvaks ja pakub järgmisi teenuseid:

- Ligipääsukontroll. Teenust saavad kasutada ainult lubatud osapooled.
- Terviklikkus. Tagatakse, et andmeid ei ole modifitseeritud.
- Autentimine. Tagatakse, et andmeid vahetatakse usaldatud osapoolte vahel.
- Salastatus. Andmed on krüpteeritud.
- Kordusesituse kaitse. Välditakse olukorda, et keegi ei saaks pakette taasesitada (ründaja ei pruugi täpselt teada, mida ta uuesti taasesitab, sest andmed on krüpteeritud, kuid võib tekitada lubamise korral kahju).

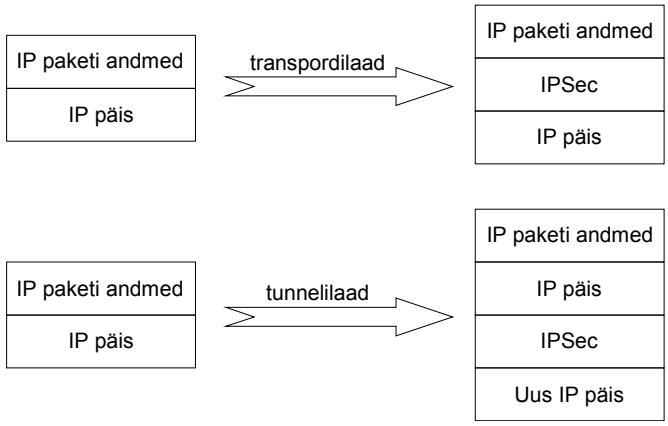
IPsec puhul luuakse alguses ühendus ja siis hakatakse andmeid vahetama. Ühenduse loomise käigus mõlemad osapooled

1. Autendivad üksteist.
2. Lepivad kokku krüptograafiaparameetrites.
3. Lepivad kokku jagatud salajastes võtmetes ja muudes parameetrites, kasutades IKE (*Internet Key Exchange*) protokolli.

Peale ühenduse loomist on võimalik hakata andmeid vahetama. IPsec puhul on arendatud kaks protokolli: AH (*Authentication Header*) ja ESP (*Encapsulating Security Payload*).

IPsec võib töötada kahes erinevas laadis: transpordi- ja tunnelilaadis. Pakett on tunnelilaadis, kui AH või ESP väli "järgmine väli" viitab IP protokollile. Vastasel korral on tegemist transpordilaadiga (viitab näiteks TCP, UDP, ICMP jne. protokollile). Transpordilaad pakub kaitset ülemiste kihtide protokollidele, ühendused on ainult kahe hosti vahel, krüpteeritakse ainult ülekantavaid andmeid ja ülekantavaid andmeid ja päist autenditakse (AH protokolli puhul). Tunnelilaadi puhul on kaitstud kogu IP-pakett (IP-pakett koos päisega pannakse uude paketti). Vaata ka joonist 18.2. VPN lahendustes kasutatakse tunnelilaadi.

Iga IP-aadresside paari ja turvaprotokolli (AH või ESP) jaoks on kasutusel oma turbeassotsiatsioon SA (*security association*), millele viidatakse pakettides SPI (*security parameters index*) väljaga.



Joonis 18.2: Paketi kapseldamine erinevate IPsec laadide korral.

AH protokolli (RFC 4302) kasutades on IP protokolli välja "protokoll" väärtuseks 51. AH kaitseb andmeid ja IP protokolli päise väljasid, mida võrku läbides ei ole vaja muuta, muutmise eest (muudetavad väljad: diferentseeritud teenuste väli, lipud, fragmendi nihe, aega elada ja päise kontrollsumma). AH paketi väljad on järgmised:

- Järgmine päis. Määrab ülekantava protokolli (nt. IP, TCP, UDP, ICMP). 1 bait.
- AH paketi suurus. 1 bait.
- Reserveeritud tuleviku jaoks. 2 baiti.
- SPI. 4 baiti.
- Järjekorranumber. Mõeldud taasesitusrünnete vältimiseks. Kui saavutab maksimaalse väärtuse (2^{32}-1), siis luuakse uus SA. Kui on vajalik andmete uuestisaatmine TCP poolt (pakett ei jõudnud sihtpuntki kohale), siis taassaadetud segmendil on uus järjekorranumber. 4 baiti.
- Autentimisinfo. Sisaldab räsi üle salajaste võtmete, ülekantavate andmete ja kaitstavate IP päise väljade. Muutuva suurusega.

ESP protokolli (RFC 2406) kasutades on IP protokolli väärtuseks 50. ESP pakub andmete autentimist ja krüpteerimist, kuid ei paku IP-paketi päise kaitset muutmise eest (AH pakub). Tunnelilaadis on kaitstud sisemine IP-pakett, kuid mitte välimine. ESP paketi väljad on järgmised:

- SPI. 4 baiti.
- Järjekorranumber. Samasugune funktsioon, nagu AH juures. 4 baiti.
- Ülekantavad andmed. Muutuva suurusega.
- Täidis. Vajatakse osade plokkšifrite puhul, mis vajavad kindla suurusega plokke. 0-255 baiti.
- Täidise suurus baitides. 1 bait.
- Järgmine päis. Samasugune funktsioon, nagu AH juures. 1 bait.
- Autentimisinfo. Sarnane funktsioon, nagu AH juures. Muutuva suurusega.

ESP krüpteerimisel on võimalik kasutada mitmeid erinevaid krüpteerimisalgoritme, näiteks 3DES (RFC 1851), DESE (RFC 1969), 3-DESE (RFC 2420). Terviklikkuse kontrolliks on AH ja ESP puhul võimalik kasutada mitmeid erinevaid räsifunktsioone. Näiteks MD2 (RFC 1319), MD4 (RFC 1320), MD5 (RFC 1321), SHA-1 (RFC 2841). MD algoritmid on nüüdseks muutunud ebaturvalisteks krüptoanalüüsi tõttu.

Peamiste erinevustena AH ja ESP vahel võib välja tuua, et AH omab kaitset ka osade IP protokolli päise väljade osas, mis annab natuke lisaturvalisust. Paraku seetõttu ei saa kasutada AH-d NAT-ga (IP-aadressi muutmise tõttu). AH tagab ainult terviklikkuse, kuid ESP protokollis

kasutatakse ka ülekantavate andmete krüpteerimist. AH-d ja ESP-d on võimalik ka koos kasutada, sellisel juhul kapseldatakse AH sisse ESP pakett (AH pakub tugevamat terviklikkuse kontrolli). Kuna NAT peab tõlkima ka pordinumbreid, kuigi see pole otseselt võimalik, siis peab NAT pakkuma eraldi tuge IPsec liiklusele või tuleb kasutada UDP kapseldust IPsec edastamiseks.

Osapoolte vahel toimub autentimine ja võtmevahetus, kasutades IKE (*Internet Key Exhange*) protokolli, mis põhineb Diffie-Hellmani võtmevahetusel.

18.3 UPnP

UPnP (*Universal Plug and Play*) on protokollide perekond seadmete automaatseks seadistamiseks, teenuste avastamiseks ja partnerilt-partnerile (*peer-to-peer*) andmeedastuse pakkumiseks üle IP-võrgu. UPnP ei ole tehniliselt seotud PnP (*Plug and Play*) tehnoloogiaga, kuid mõlemad pakuvad automaatset seadistust (PnP võimaldab arvutile lisaseadmeid lisada automaatse draiverite installeerimisega). UPnP on avaldatud 1999. aastal UPnP Forum (koduleht: http://www.upnp.org) poolt, mis ühendab üle 800 tootja. UPnP-ga kattuvaid funktsioone pakub Zeroconf (IP-aadressi hankimine, teenuste avastamine ja kohtvõrgus arvutite nimede lahendamine). Zeroconf on UPnP-st märgatavalt lihtsama ülesehitusega.

UPnP pakub järgmisi hüvesid (vaata ka UPnP arhitektuuri joonisel 18.3):

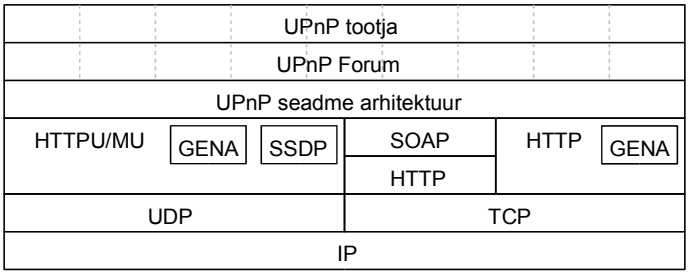
- Meediumist ja seadmetest sõltumatus. UPnP tehnoloogia saab töötada erinevate võrgutehnoloogiate peal. Näiteks: Wi-Fi, Bluetooth, koaksiaalkaabel, telefoniliin, elektrijuhtmed, Ethernet, IEEE 1394 (ehk FireWire) jt.
- Platvormist sõltumatus. Tootjad saavad kasutada suvalist operatsioonisüsteemi ja programmeerimiskeelt oma toodete tegemiseks.
- Internetipõhised tehnoloogiad. UPnP kasutab laialtlevinud standardeid ja tehnoloogiad, sealhulgas IP, TCP, UDP, HTTP ja XML.
- Kasutajaliidese loomise võimalus. UPnP arhitektuur võimaldab seadmetootjatel disainida seadme kasutajaliidest soovi korral ka veebibrauseripõhisena.
- Programne juhtimine. UPnP arhitektuur võimaldab traditsioonilist rakenduse programmilist kontrolli.
- Üldised baasseaded. Tootjad on baasprotokolli sätetes kokku leppinud seadmete baassätetes.
- Laiendatavus. Iga UPnP seade saab sisaldada lisateenuseid, mis on tootja poolt lisatud baasseadme arhitektuurile.

UPnP toetab nullseadistust ja automaatset avastamist, millega seade saab: dünaamiliselt ühenduda võrguga; hankida IP-aadressi; teatada oma nime; anda pärijale teada oma võimalustest; teada saada teiste seadmete olemasolu ja võimalusi; lahkuda vaikselt ja automaatselt võrgust, jätmata soovimatut olekuinformatsiooni. DHCP- ja DNS-serverid on ebavajalikud, kuid neid kasutatakse, kui nad on saadaval. Peale seadmete seadistamise pakub UPnP ka mitmeid rakendusi, sealhulgas andmete transporti üle võrgu (nt. Windows Media Connect kasutab UPnP-d heli- ja videovoo teostamiseks). Teenuste avastamiseks kasutatakse SSDP (*Simple Service Discovery Protocol*) protokolli, et avastada sobivaid servereid.

UPnP võrgukasutus jaguneb mitmeks etapiks:

1. Seadmele IP-aadressi hankimine. Kui seadmel pole IP-aadressi määratud, siis püütakse saada IP-aadressi DHCP-serverilt. Ebaõnnestumise korral valitakse IP-aadress lokaalse sideliini aadressihulgast 169.254.0.0/16 ja kontrollitakse unikaalsust kohtvõrgus (kasutades ARP-päringut). Perioodiliselt jätkatakse DHCP-serveri avastamisega (lootuses, et see saab millalgi kättesaadavaks).
2. Sobivate masinate avastamine. Seadmetel on olemas minimaalsed võimed. Seadmed kuulutavad enda olemasolust, kui nad on võrgus, ning värskendavad kuulutusi. Võrgust eemaldamisel lõpetavad kuulutamise. Kasutatakse protokolli SSDP.

3. Avastatud seadmete võimaluste teada saamine. Kirjeldusega antakse edasi tüüp, füüsiline andmestik (tootja, tootja URI, mudeli kirjeldus, mudeli nimi, mudeli number, mudeli URI, seerianumber, identifikaatorid), loogiline andmestik (teenuse tüüp, teenuse identifikaator, URI teenuse kirjelduse juurde, URI teenuste juhtimise juurde, URI teenuste muutumiste kuulamise juurde), muu andmestik (nt. ikoonid), teenused (sisaldab toimingute parameetreid ja muud sinna juurde kuuluvat), olekute muutujad (sisaldab muutuja kohta kirjeldavat infot). Informatsioon on esitatud XML formaadis.
4. Seadme kasutamine:
    1. Operatsioonide teostamine masinas. Saadetakse seadmele korraldusi, millele vastatakse. Kasutatakse SOAP (*Simple Object Access Protocol*) protokolli.
    2. Olekute muutumise kuulamine. Olekute muutumise kohta teavituste saamiseks peab esitama vastava tellimuse. Kasutatakse GENA (*General Event Notification Architecture*) protokolli, kasutades HTTP protokolli või administratiivselt piiratud multiedastust UDP-ga.
    3. Seadme kontrollimine ja informatsiooni vaatamine. Kasutatavaks vahendiks on tavaline veebibrauser.



Joonis 18.3: UPnP arhitektuuri protokollistik. UPnP protokollistiku kõik seadmed omavad teatud hulga ühist osa, millele on UPnP Forum poolt lisatud erinevat tüüpi seadmetele eraldi kiht, mis on samasugune kõikidel sama liiki saedmetel ja millel omakorda asub UPnP seadmete tootjate poolt lisatav täiendav funktsionaalsus. UPnP on riistvarast sõltumatu. HTTPU on HTTP variant, mis kasutab transpordikihina UDP protokolli, ja HTTPMU on multiedastuse variant.

UPnP ja NAT

UPnP protokoll aitab leevendada ka NAT-st tulenevaid probleeme, mis on praktikas UPnP üks peamine kasutusala. Nimelt on NAT-tud võrgus probleeme näiteks partnerilt-partnerile (*peer-to-peer*) ühendustega, reaalajakommunikatsiooniga, IPsec (ESP baasil) ühendustega, mitme osavõtjaga mängudega jne. UPnP pakub lahendust NAT probleemidele IGD (*Internet Gateway Device*) seadmekategooriasse kuuluva standardiga, mida tuntakse NAT läbikäigu (*NAT traversal*) all (ka lühendi NAT-T). NAT-T pakub järgmisi funktsioone:

- Saada teada NAT-marsruuteri avalikku IP-aadressi.
- Saada loend eksisteerivatest portide sidumistest.
- Lisada ja eemaldada portide sidumisi.
- Määrata sidumiste tähtaegu.

Eelnevalt loetletud funktsioonidega saab rakendusprogramm ise teha sobivad sidumised NAT-marsruuteris, millega saab sisevõrgu host kättesaadavaks välisvõrgust (NAT-st suunatakse edasi kõik paketid, mis on tulnud NAT-marsruuteri välisliidese registreeritud pordile). Selliselt saadakse üle paljudest NAT-ga seotud probleemidest, selleks peab programmidel olema NAT-T toetus sisse programmeeritud. NAT-T toetus on olemas paljudel kodukasutuseks mõeldud NAT-marsruuteritel (otsi spetsifikatsioonist märget UPnP toetuse kohta).

18.4 Võrguhaldus

Olemasolevat võrku on vaja hallata ja monitoorida, et võrk töötaks võimalikult heas seisukorras. Arvutivõrk on tänapäeval väga paljudele asutustele muutunud eluliselt tähtsaks elemendiks. Võrgu maasolek või probleemid jõudlusega tooksid kaasa väga suuri kahjusid ja häiriksid asutuse tegevust. Seetõttu on väga oluline probleeme ja kitsaskohti juba eos avastada ja nende vastu võidelda. Asutuse kasvades kasvab üldjuhul ka asutuse arvutivõrgu suurus ja keerukus.

Töökindluse ja kiiruse suurendamise üheks võimaluseks on dubleerimine. Nimelt on suures osas võimalik servereid, võimalikke ühendusteid, võrguseadmeid jne. dubleerida. Dubleerimine lisab kontekstist olenevalt keerukust. Keerukuse kasv võib omakorda tuua kaasa võimalikke ootamatuid olukordi, mil võrguadministraatori arvates peaks süsteem töötama, kuid seda siiski ei tee või toimib mõningate anomaaliatega. Arvestama peab ka mitte ainult arvutivõrgu kasvamisega, vaid ka uute tulevikus kasutatavate teenuste, protokollide ja muude nõudmistega.

Võrguadministraatori ülesannete hulka kuuluvad süsteemi lihtsa kasutatavuse tagamine, suhtlemine asutuse teiste IT-inimestega (näiteks serverite haldajatega, kasutajatoega, infosüsteemide arendajatega), taastamisvõimaluste tagamine (võimalike riistvara ja tarkvara rikete ning ka näiteks uue seadistuse probleemid), võrgu kättesaadavuse monitoorimine, automatiseerituse täiustamine (paljude tööde jaoks on võimalik kirjutada skripte ja programme, mis vähendavad ajakulu ja võimalikke inimlikke eksimusi), reaktsiooniaegade jälgimine (näiteks serveritelt vastuse saamine), turvaaspektidel silma peal hoidmine ning turvameetmete täiendamine; kasutajate lisamine ja eemaldamine võrgust ning olenevalt spetsiifikast ka muud tegevused.

Võrguhaldusprotokollid peaksid olema võimalikult vähe ise võrku mõjutavad (ribalaius, töötluse aeg, mälutarve jne.), lihtsad (nt. tarkvara kirjutamise jaoks), kergesti õpitavad ning töötama ka ebasoodsates tingimustes. Lisaks peaksid protokollid pakkuma tsentraalselt võrgu funktsioneerimise kontrollimist, kaughaldust, süsteemidest sõltumatult monitoorimist, mitme protokolli toetust ning keerukate lahenduste haldamist. Protokollid peaksid veel olema võimalikult läbipaistvalt töötavad, kiire reaktsiooniajaga, odavad ning peaksid rahuldama erinevate aspektide nõudeid (rakendusprogrammid, masinad, tehnoloogiad, turvalisus jne.). Võrguhaldusvahendite peamisteks eesmärkideks võib pidada võrgu või võrgu kaudu pakutavate teenuste maasoleku aja minimeerimist, ülevaate saamist võrgust ja teenuste toimimise näitudest (nt. reaktsiooniaeg).

Üks olulisi asju, mida võrguadministraator saab oma töö kergendamiseks ja parendamiseks teha, on kasutada automaatseid võrguhaldusvahendeid, mis toetavad ühte või mitut võrguhaldusprotokolli. Tuntuimateks protokollideks on:

- SNMP (*Simple Network Management Protocol*). Kõige sagedamini kasutatav võrguhaldusvahend, mida toetavad paljud seadmed ja tarkvaratooted.
- CMIP (*Common Management Information Protocol*). ISO OSI võrgu haldamiseks välja töötatud süsteem. SNMP konkurent, olles komplekssem, kuid seevastu ressursinõudlikum.
- WBEM (*Web-Based Enterprise Management*). WBEM koosneb mitmetest haldamise ja Interneti standardite tehnoloogiatest, mis on loodud hajutatud andmetöötlus-keskkondade ühendamiseks, soodustades andmete vahetust erinevate tehnoloogiate ja platvormide vahel. Windowsi all kasutatakse nimetust WMI (*Windows Management Instrumentation*).
- Netconf (*Network Configuration*) (RFC 4741). Seadmete seadistuse haldamiseks. Kasutab RPC mehhanismi (*Remote Procedure Call*).
- SGMP (*Simple Gateway Monitoring Protocol*) (RFC 1028). Vana võrguhaldusprotokoll, mis on asendatud SNMP protokolliga.

Võrguhalduse kohta on oma mudeli välja töötanud ka ISO, mida tuntakse MFA (*Management Functional Area*) nime all. MFA jagab võrguhalduse viide erinevasse kategooriasse (nimetatakse ka FCAPS mudeliks (akronüüm tuleneb nimetuste esitähtedest)):

- Rikkehaldus (*fault management*) – kannab hoolt võrgu ja tema komponentide normaalse toimimise eest. Vigade ilmnemisel on vajalik võimalikult kiire võrguhalduri teavitamine. Mõningatel juhtudel võib keskselt hallatav süsteem ise automaatselt vigu parandada. Vigadest teavitamise puhul on oluline vigade tõsidus ja prioriteet.
- Seadistuse haldus (*configuration management*) – ülesandeks on võrgus olevaid seadmeid identifitseerida, seadistada ja kontrollida. Seadistamiseks on vaja teada hallatava masina tüüpi, mille alusel on võimalik teda vastavalt kohelda. Siia alla kuuluvad: põhiline võrguseadistus (kasutusel olevad IP-aadressid, võrguliideste tüübid, serverid jne); ülevaade võrgu praegusest seisust (nii üksiku masina, kui ka kogu võrgu tasemel); seadete muutmine; masinate ja nende poolt pakutavate teenuste seiskamine, taaskäivitamine ja käivitamine; tarkvara levitamine hallatavatesse masinatesse (nt. tarkvarauuendused).
- Arvestuse haldus (*accounting management*) – tegeleb ressursside (nt. kettalimiit, sideliini kasutus) kasutamise statistikaga, teenuste arvetega ja kasutamise piirangutega.
- Jõudluse haldus (*performance management*) – jaguneb kahte kategooriasse: monitoorimine ja häälestamine. Monitoorida võidakse näiteks sideliinide hõivatust ning teenuste reageerimisaega. Monitoorimisel kontrollitakse, kas hallatav subjekt töötab etteantud piirides (piiri saab defineerida võrguadministraator). Häälestamisel kohandatakse subjekti ja eemaldamaks võimalikke pudelikaelu. Tulemusena saadakse näiteks teave võrgu kasutuse aktiivsusest eri aegadel, millega seonduvalt saab kindlaks teha kasutuse tippajad, kasutuse anomaaliad (mingi imelik käitumine, mida ei tohiks olla) jms.
- Turbe haldus (*security management*) – ülesandeks on määrata ligipääs võrgu ressurssidele vastavalt võrguadministraatori määrangutele, logida, autentida, krüpteerida, auditeerida jms.

18.5 SNMP

SNMP on suhteliselt lihtne, odav, paindlik ja lihtsalt toimiv võrguhaldusprotokoll, olles osa TCP/IP protokollistikust. SNMP on mõeldud eelkõige TCP/IP võrkude jaoks, kuid seda on võimalik kasutada ka teiste protokollistikega. Algselt oli SNMP peamiseks eesmärgiks marsruuterite haldamine, kuid nüüdseks on lisandunud ka teisi seadmeid (nt. personaalarvutid, serverid, kommutaatorid).

SNMP on levinuim võrguhaldamisprotokoll, millele on kõige laialdasem toetus. SNMP-st on olemas kolm põhiversiooni: SNMPv1 (RFC 1155-1157), SNMPv2 (RFC 1441, 2578-2580, 3416-3418) ja SNMPv3 (3410-3415, 3584). Isegi SNMPv3 ei sisalda kõiki ISO võrguhaldusstandardeid, sest mõningad nendest on tarbetult keerukad ja ebapraktilised. Peamiselt on kasutusel SNMPv1.

SNMP on rakenduskihi protokoll, mis on välja töötatud informatsiooni hankimiseks seadmetelt haldamise hõlbustamise eesmärgil. SNMP poolt hallatav võrk koosneb kolmest komponendist:

- Hallatavad seadmed, mida monitooritakse.
- Agendid. Hallatavas masinas asuv tarkvara, mis kogub informatsiooni SNMP protokollile sobival kujul.
- Võrguhaldussüsteem ehk NMS (*network-management systems*). NMS on tarkvara, mis kogub agentidelt informatsiooni ja pakub võrguhaldurile liidest võrgu haldamiseks. Asub ühes või mitmes arvutis.

Lihtsustatult öeldes omavad agendid hallatavas süsteemis mingeid muutujaid (nt. sideliinil esinenud vigaste TCP segmentide loendur (peamiselt mittesobiva kontrollsumma tõttu)), mille kohta NMS käib regulaarselt infot küsimas. Osasid muutujaid saab NMS ka muuta (nt. IPv4 pakettide "aega elada" vaikeväärtus) ning osade sündmuste kohta panna agent automaatselt NMS-i teavitama (nt. sideliini üles või maha minek).

Haldamise informatsiooni hoitakse objektidena hierarhilises andmebaasis, mida nimetatakse MIB-iks (*Management Information Base*) (RFC 1156, 1212, 1213). MIB sisaldab hallatavaid objekte (ehk MIB-objekte), kus hoitakse andmeid. Igal MIB-objektil on olemas unikaalne identifikaator, mis määrab ta asukoha MIB-i puus. Vaata MIB-i puu kohta ka joonist 18.4.

Hallatava informatsiooni kirjeldamise reeglid määrab SMI (*Structure of Management Information*) (RFC 1155, 2578), mis on ASN.1 kohandatud alamosa (ASN.1 on standard, mis annab paindliku esitusviisi andmestruktuuride kirjeldamiseks, kodeerimiseks ja edastamiseks). SMI määrab, kuidas nimetada, struktrueerida ja kodeerida hallatavaid objekte. Igal hallataval objektil peab andmetest olema nimi, süntaks (defineerib objekti andmetüübi) ja kodeering (kasutatakse BER (*Basic Encoding Rules*) formaati). Osa objekte on iseseisvad, osad on omavahel seotud ning grupeeritud MIB-i tabelis.

MIB-objektide väärtused saavad olla esitatud skalaartüübina või kahemõõtmelise massiivina (ehk tabelina), mille elemendid on skalaartüüpi. Skalaartüübid jaotatakse liht- ja *application-wide* tüüpideks. Lihttüüpideks on:

- Täisarv – vahemik -2 147 483 648 kuni 2 147 483 647 (32-bitine).
- Oktettsõne – baidijada, mis võib sisaldada suvalist binaarset või tekstilist informatsiooni ja mille pikkus on vahemikus 0 kuni 65 535 baiti (reaalsetel implementatsioonidel arvestatud tavaliselt kuni 255 baiti).
- Objekti identifikaator – vastavalt ASN.1 standardile.

Application-wide tüüpideks on:

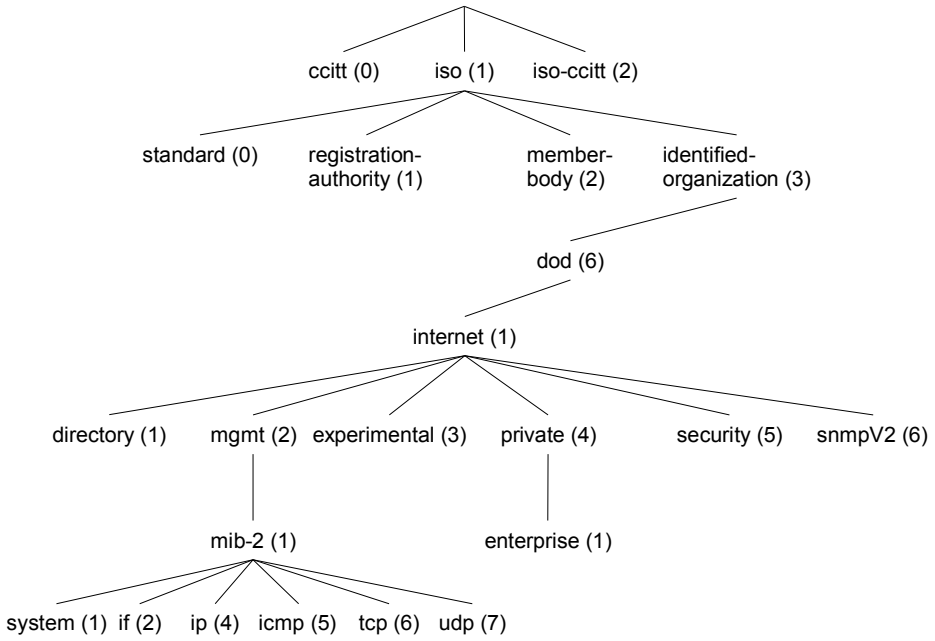
- *IpAddress* – IPv4-aadress.
- *Counter* – loendur, naturaalarv (vahemik 0 kuni 2^{32}-1), mida järjest suurendatakse (maksimaalse väärtuseni jõudes alustatakse nullist). SNMPv2-s nimetati ümber counter32-ks.
- *Gauge* – naturaalarv, mis võib suureneda ja väheneda (vahemik 0 kuni 2^{32}-1), maksimaalse väärtuseni jõudes ei lähe järgmise väärtusena nulliks. SNMPv2-s nimetati ümber gauge32-ks.
- *TimeTicks* – naturaalarv, mis loendab, mitu sajandiksekundit tagasi mingi sündmus aset leidis (maksimaalne aeg on ligikaudu 497 päeva).
- *Opaque* – suvaline kodeerimine, mida kasutatakse suvaliste informatsioonisõnede edastamiseks, mis ei vasta SMI poolt kasutatavale rangele andmete tüüpimisele.
- *Unsigned32* – lisatud SNMPv2 protokolliga. Naturaalarv vahemikus 0 kuni 2^{32}-1.
- *Counter64* – lisatud SNMPv2 protokolliga, mis on 64-bitine "Counter" tüüp.

SNMPv2 lisas juurde veel "bits" tüübi, mis on loetelu (*enumeration*) tüüpi, kus iga bitt tähistab mingit nimelist väärtust.

MIB-i struktuur

MIB on hierarhiline puu, kus puu juureks on tühi märgend. Igal järgneval märgendil on nimeline ja numbriline tähis (kirjutame antud peatükis nimeliste nimetuste järel sulgudes ka numbrilise nimetuse). MIB-i puu järgmisel tasemel asuvate objektide identifikaatorite nimeruumi haldavad standardiseerimisorganisatsioonid ISO ja ITU. Nemad delegeerivad edasi alamtasemeid teistele organisatsioonidele. Võrguadministraatorid puutuvad kõige enam kokku "iso" (1) haruga. "iso" haru jaguneb omakorda neljaks, kus kõige enam puudutab võrguadministraatorit haru "iso.org" (1.3) (märgend pikemalt "identified-organizations"). Haru "iso.org" kõige olulisem alamharu on "iso.org.dod" (1.3.6), mille kõige olulisem haru on omakorda alamharu "iso.org.dod.internet" (1.3.6.1). Siin harus on mitmeid kasutatavaid alamharusid. Alamharu "iso.org.dod.internet.private.enterprise" on eraldatud erinevatele organisatsioonidele. Näiteks "iso.org.dod.internet.private.enterprise.Estonian Educational and Research Network" (1.3.6.1.4.1.19974) on eraldatud EENetile (Eesti Hariduse ja Teaduse Andmesidevõrk). MIB-i alamharude registreerimised asutustele on saadaval aadressil http://www.iana.org/assignments/enterprise-numbers. MIB-i haru "iso.org.dod.internet.experimental" (1.3.6.1.3) on

katselistele protokollidele, mille registreeringud asuvad aadressil http://www.iana.org/assignments/smi-numbers. Harus "iso.org.dod.internet.mgmt" (märgend pikemalt "management") asuvad näiteks süsteemi, liideste, IP, ICMP, TCP, UDP, EGP ja teised alamharud. Vaata ka joonist 18.4 osalisest MIB-i puust.



Joonis 18.4: Osaline MIB puu alates juurest.

Algne MIB ehk MIB-I spetsifikatsioon sisaldas üle saja seadistatava hallatava objekti, mis jaotati kaheksasse gruppi:

- Süsteemiobjektigrupp ("1.3.6.1.2.1.1"). Sisaldab infot masina kohta, mida kasutavad eelkõige rikke- ja seadistushaldus.
- Liideseobjektigrupp ("1.3.6.1.2.1.2"). Sisaldab infot masina võrguliideste kohta. Kasutavad rikke-, seadistuse-, jõudluse- ja arvestuse haldus.
- Aadressi tõlkimise objektigrupp ("1.3.6.1.2.1.3"). Sisaldab füüsilise aadressi informatsiooni.
- IP objektigrupp ("1.3.6.1.2.1.4"). Sisaldab informatsiooni masina IP-aadresside, marsruutimistabeli kirjete, IP-paketi vigade, käideldud IP pakettide tüüpide jne. kohta. Kasutavad rikke- seadistuse-, jõudluse- ja arvestuse haldus.
- ICMP objektigrupp ("1.3.6.1.2.1.5"). Sisaldab infot masina ICMP kohta.
- TCP objektigrupp ("1.3.6.1.2.1.6"). Sisaldab infot masina TCP-ga seonduva kohta.
- UDP objektigrupp ("1.3.6.1.2.1.7"). Sisaldab infot masina UDP-ga seonduva kohta.
- EGP (*Exterior Gateway Protocol*) objektigrupp ("1.3.6.1.2.1.8"). Sisaldab infot teiste IP võrkude kättesaadavuse kohta, baseerudes EGP marsruutimisinfol. Kasutavad rikke-, seadistuse-, jõudluse- ja arvestuse haldus.

MIB-II, mille laiendused olid vajalikud SNMPv2 jaoks, lisas juurde kolm uut gruppi:

- CMOT (*Common Management Interface Protocol over TCP/IP*) objektigrupp ("1.3.6.1.2.1.9"). Eksisteerib ajaloolistel põhjustel.
- Edastuse objektigrupp ("1.3.6.1.2.1.10"). Sisaldab informatsiooni võrgumeediumi kasutamise kohta.

- SNMP objektigrupp ("1.3.6.1.2.1.11"). Sisaldab informatsiooni saadetavate ja vastuvõetud SNMP pakettide kohta. Kasutavad kõik halduse kategooriad.

MIB standardi kohaselt saab päringuid muutujatele teha ainult MIB puu lehtedele, kuid mitte sõlmedele. Informatsiooni küsimisel pannakse vastava skalaartüübi objekti identifikaatorile lõppu ".0".

Mõningaid näiteid olemitest:

- "iso.org.dod.internet.mgmt.mib.system.sysDescr" (1.3.6.1.2.1.1.1) – tekstiline väärtus, mis peaks sisaldama masina täisnime, operatsioonisüsteemi, riistvaratüüpi jms. Andmetüübiks on oktettsõne.
- "iso.org.dod.internet.mgmt.mib-2.tcp.tcpInErrs" (1.3.6.1.2.1.6.14) – näitab, mitu vigast segmenti on saadud (nt. vigase kontrollsummaga) (alates olemis "1.3.6.1.2.1.1.3" märgitud millisekundit tagasi). Andmetüübiks on Counter32.
- "iso.org.dod.internet.mgmt.mib-2.system.sysUpTime" (1.3.6.1.2.1.1.3) – näitab, kui kaua süsteem on järjest töötanud. Andmetüübiks on TimeTicks.
- "iso.org.dod.internet.mgmt.mib-2.ip.ipDefaultTTL" (1.3.6.1.2.1.4.2) – saab vaadata ja muuta IPv4 vaikimisi "aega elada" välja väärtust. Andmetüübiks on Integer32.

Suhtlus

SNMP kasutab suhtluseks UDP protokolli portidel 161 (agendi poolt kuulatav) ja 162 (NMS poolt kuulatav). Osadel implementatsioonidel võivad olla ka muud pordid. SNMP kasutab kahte erinevat porti, et samas masinas saaks korraga töötada nii agent kui ka NMS. Vahetatavad teadetetüübid on järgmised:

- Saa (*get*) – NMS päringu esitamiseks agendile.
- Sea (*set*) – agendi seadistamiseks NMS poolt.
- Püünis (*trap*) – Agent saab teavitada NMS-i mingist sündmusest, mille NMS on eelnevalt seadistanud. Näiteks võib seadistada, et iga kord, kui hallatav masin käivitub, saadetakse NMS-le vastav teade.

SNMP operatsioone saab sooritada, kasutades ainult käske:

- GetRequest – NMS saadab agendile, kui soovib saada mingit MIB-i muutujat.
- GetNextRequest – NMS soovib saada agendilt järgmist väärtust (jätkuks GetRequest käsule) massiivitüüpi andmete kohta.
- GetResponse – kasutab agent saatmaks NMS-le tagasi GetRequest ja GetNextRequest käsule vastust.
- SetRequest – NMS soovib muuta mingit MIB muutuja väärtust agendil.
- Trap – kasutab agent saatmaks teadet NMS-le, kui on toimunud mingi sündmus, mis NMS-i huvitab (NMS on eelnevalt seadistanud agenti).
- GetBulk – lisati SNMPv2-le, et NMS saaks efektiivsemalt saada massiivides sisalduvaid suuremaid andmeplokke.
- Inform – võimaldab saata infot ühelt NMS-lt teisele NMS-le ning vastata.

SNMP paketi väljad (esimesed kolm välja moodustavad päise):

- Versioon. SNMPv1 puhul 0, SNMPv2 puhul 1, SNMPv3 puhul 2.
- Kogukond (*community*). Kogukonna nimi määrab, milline NMS saab pärida ja seadistada agenti. Agent, mis saab SNMP teate, mille kogukonna nimi ei ole tema poolt aktsepteeritavate kogukonna nimetuste hulgas, viskab antud teate minema. Kogukond pakub triviaalset autentimist, mis siiski on ebaturvaline. Kogukonna alusel on võimalik määrata ühele kogukonnale ainult lugemise õigus (vaikimisi kogukonna nimi "public") ja mingile teisele kogukonnale MIB-objektide muutujate muutmise õigus (vaikimisi kogukonna nimi "private").
- Tüüp. Määrab teate tüübi (GetRequest=0, GetNextRequest=1, GetResponse=2, SetRequest=3, Trap=4) ja formaadi (get, request ja trap).
- Päringu identifikaator. Kasutatakse päringu ja saadava vastuse seostamiseks.

- Vea staatus. Antakse teada vea tüüp. GetRequest, GetNextRequest ja SetRequest teates ei kasutata (märgitakse nulliks). Kasutatakse vajadusel GetResponse teadetes.
- Vea indeks. Sisaldab täiendavat informatsiooni vea staatuse kohta (sõltub vea tüübist).
- Objekti identifikaator. Sisaldab küsitava MIB-objekti identifikaatorit.
- Väärtus.

Agendi poolt väljaga "vea staatus" raporteeritavaid vigu (sulgudes number tähistab veakoodi) (SNMPv2-s on vigade staatusi rohkem) (RFC 3584):

- "noError" (0) – operatsioon on edukas.
- "tooBig" (1) – vastus on saatmiseks liiga suur.
- "noSuchName" (2) – agent ei toeta küsitud objektiidentifikaatorit.
- "badValue" (3) – käsuga SetRequest edastati vigane väärtus.
- "readOnly" (4) – üritati muuta ainult loetavat väärtust.
- "genErr" (5) – tekkis viga MIB muutuja lugemisel või kirjutamisel.

SNMP püünisteate formaat (kolm esimest välja on samad):

- "Enterprise" – sisaldab agendi MIB objekti "sysObjectID" väärtust (OID väärtus on "1.3.6.1.2.1.1.2"), mis kirjeldab teate edastanud masinat.
- Agendi aadress – sisaldab agendi IP-aadressi, mis saadab püünisteadet.
- Üldine püünis (*generic trap*) – sisaldab teate koodi.
- Iseloomulik püünis (*specific trap*) – kui "üldine püünis" väärtus oli "enterpriseSpecific", siis sisaldab vastavat spetsiifilist koodi.
- Ajatempel (*time stamp*) – mitu tuhandiksekundit on möödunud agendi käivitamisest.
- Muutuja väärtus.

Võrgu haldamiseks NMS-i poolt peab NMS teadma agentide asukohtasid. Üks võimalus on käsitsi seadistada NMS. Automaatselt agentide ülesleidmiseks saab kasutada mitut moodust. Üks võimalus on kasutada ICMP protokolli (pingimine). NMS saab küsida masinate ARP-vahemälu, marsruutimistabeleid, kuulata marsruutimisprotokollide liiklust. Peale masinate avastamist peab NMS kindlaks tegema, kas masinas on olemas SNMP agent. Selleks saadetakse GetRequest teade, millele agendid vastavad GetResponse teatega. Eelnevalt peab võrguadministraator veenduma, et NMS ja agendid kasutavad sama SNMP versiooni. Vastasel korral on vajalik kasutada vahendajana SNMP-vahendajat (*proxy*).

SNMP protokolli disainimisel arvestati, et agendid peaksid võimalikult vähe mõjutama võrguliiklust, kasutama vähe hallatava masina mälu ja protsessoriaega. Seetõttu on püünisteateid suhteliselt vähe: kuumkäivitamine (muudetud ei ole seadistusi ega protokolli), külmkäivitamine (seadistus või protokoll on vahepeal muutunud), sideliini ülestulek, sideliini mahaminek, autentimise viga (sai teate, millel oli vigane kogukonna nimetus), EGP partneri naabrus-side on katkenud ning organisatsiooni spetsiifiline.

RMON ja vahendaja

SNMP hallatavas võrgus on võimalik kasutada RMON-e (*remote monitor*), mis koguvad lokaalselt informatsiooni ja saadavad NMS-le kokkuvõtte, kui viimane seda küsib. RMON-e nimetatakse ka sondideks (*probe*). RMON1 (RFC 2819) oli rohkem kontsentreeritud füüsilise- ja kanalikihi informatsiooni kogumisele. RMON2-ga (RFC 4502) lisati toetus kõrgemate kihtide monitoorimiseks. RMON on MIB-II standardi osa, mis ei vaja SNMP muutusi.

RMON1 MIB-i grupid on järgmised:

- Statistika grupp – haldab kasutuse ja vigade statistikat alamvõrgu või segmendi kohta. Näiteks: ribalaiuse kasutus, leviedastus, multiedastus, CRC kontrollid, fragmendid jne.
- Ajaloogrupp – hoiab grupi perioodilisi statistilisi tulemusi hilisemaks kasutamiseks. Näiteks: kasutus, vigade loendur, pakettide loendur jne.
- Alarmgrupp – võimaldab administraatoril saada määratud ajavahemike tagant monitoorimise tulemusi. Näiteks absoluutseid või relatiivseid väärtusi.

- Hosti grupp – mitmesugust tüüpi hosti liikluse mõõtmised. Näiteks: pakettide ja baitide saatmine ja vastuvõtmine, vead, levi-ja multiedastus pakettide loendur.
- Hostide top N grupp – annab informatsiooni mingi ajavahemiku kõige aktiivsemate ühenduste kohta.
- Liikluse maatriksi grupp – hoiab vigade ja kasutusstatistikat võrgus suhtleva kahe osapoole vahel. Näiteks: vead, saadetud baidid ja paketid.
- Filtrigrupp – teostab paketivoo kaadritest, mis langevad kokku kasutaja määratud filtriga.
- Paketihõive grupp – võimaldab hõivata (capture) liiklust defineeritud filtri põhjal ja salvestada või saata edasi soovitud kohta.
- Sündmuste grupp. Võimaldab logida sündmusi, kaasaarvatud püünissündmused (sisaldades toimumise aega). Selliselt saab koostada kohandatud raporteid võrguadministraatorile.
- Token Ring grupp – Token Ringi jaoks.

RMON2 lisas järgmised MIB-de grupid:

- Protokolli kataloog (*protocol directory*) – nimekiri protokollidest, mida ollakse võimeline uurima.
- Protokollide liigitus (*protocol distribution*) – sisaldab statistikat iga erineva protokolli kohta.
- Aadressikujutus – informatsioon võrgukihiaadressi (IP-aadressi) vastavusest kanalikihiaadressiga (MAC-aadress) (näiteks tõlgete lisandumised, kustutamised, tõlked).
- Võrgukihi host – võrgukihi liikluse statistika iga hostiga (nt. kui palju saadetud ja vastu võetud vastavalt hostilt).
- Võrgukihi maatriksigrupp – võrgukihipõhine statistika lähte- ja sihthostipaari kohta.
- Rakenduskihi host – protokollikasutusest võrgukihi või kõrgemate kihtide osas. Informatsioon liiklusmahu kohta protokollipõhiselt iga hosti osas.
- Rakenduskihi maatriks – protokollikasutusest võrgukihi või kõrgemate kihtide osas. Informatsioon liiklusmahu kohta protokollipõhiselt iga lähte- ja sihthostipaari osas.
- Kasutaja ajalugu – perioodilised seisundid kasutaja poolt eelmääratud muutujatest (võtab osaliselt üle NMS ülesandeid).
- Sondi konfigureerimine – sondi parameetrite kaugkonfigureerimine.

SNMP pakub ka vahendajate kasutamise võimalust suhtluses agentide ja NMS-de vahel. SNMP-vahendajad võivad olla vajalikud näiteks siis, kui agendid ja NMS ei kasuta sama protokolli, või võrgu administreerimise koorma vähendamiseks. SNMP-vahendajad on vahemäluks, mis hoiab sagedasti päritavat infot, aidates vähendada peamiselt hallatavate masinate koormust. Samuti võimaldavad SNMP-vahendajad jätta autentimine vahendaja ja NMS vaheliseks toiminguks, millega väheneb agentide ja administreerimise koormus. Vahendaja on vajalik, kui NMS ja mingi osa agente toetavad erinevaid võrguhaldusprotokolle või erinevaid SNMP versioone. Kui NMS ja osad agendid kasutavad erinevaid võrgukihi protokolle (nt. TCP/IP ja IPX/SPX), siis on võimalik kasutada lüüsivahendajat.

SNMP-vahendajat on võimalik kasutada ka juhul, kui jälgitav seade ei oma SNMP-agendi toetust ja seda ei saa sinna paigaldada. Sellisel juhul võib SNMP-vahendaja panna tegema päringuid vastava seadme kohta (nt. kas teenus töötab, milline on reageerimisaeg).

Arhitektuur

Võrgus võib olla ka mitu NMS-i, mis võivad omavahel olla võrdsed partnerid või olla hierarhias. SNMP võrguhalduse saab jagada kolmeks erinevaks arhitektuuriks:

- Tsentraliseeritud arhitektuur. Võrgus on ainult üks NMS ja mitu agenti. Info edastamine on tsentraliseeritud. Selline arhitektuur on sobilik väikese ja keskmise suurusega organisatsioonidele, kuid suuremate puhul võib tekkida ülekoormatus.

- Hajutatud arhitektuur. Võrgus on mitu NMS-i, mis haldavad erinevaid agente. Sellise süsteemi puhul on koormus hajutatum ja süsteem tõrkekindlam, mille tõttu on see kasutatav ka suurtes asutustes. Probleemiks on tsentraalse hallatavuse puudumine.
- Hierarhiline arhitektuur. NMS-d paigutuvad hierarhiliselt, kus juur-NMS kogub infot NMS-delt. Selline arhitektuur on tsentraalne, kuid samas kasutatav ka suurtes organisatsioonides. Ülem-NMS on paljud ülesanded delegeerinud alam-NMS-dele. Selle arhitektuuri probleemiks on keerukuse kasvades suurenevad nõudmised haldamisele ja administreerimisele.

SNMP protokolli teeb lihtsaks ja paindlikuks asjaolu, et agendi poole pöördumiseks on vajalik ainult agendi IP-aadressi ja kogukonna nime (*community name*). SNMP protokollist on olemas kolm erinevat põhiversiooni:

- SNMPv1 (RFC 1155, 1157, 1215). Senini laialdaselt kasutatav. Peamisteks probleemideks on turvalisus ja skaleeruvus. SNMPv1 abil on võimalik luua ainult tsentraliseeritud haldamise arhitektuuri, sest ei ole NMS-de vahelist päringute tegemist. SNMPv1 oli disainitud ainult TCP/IP jaoks ning sihiks oli lihtsus.
- SNMPv2 (RFC 1905-1907, 2578-2580) ei saanudki standardiks. Kõige enam on kasutusel SNMPv2c variant. SNMPv2 ja SNMPv3-ga on proovitud lahendada SNMPv1 puudujääke, kuid see on teinud nad keerulisemaks.
- SNMPv3 (RFC 2571-2575, 2741). Eesmärkideks on kasutada olemasolevat nii palju, kui võimalik (baseerudes mitteametlikel versioonidel SNMPv2u ja SNMPv2*) ning turvalisuse lisamine (paketid krüpteeritud, terviklikkuse kontroll ja osapoolte autentimine), mis seni oli suurim puudujääk. Üritatakse jätkuvalt hoida SNMP võimalikult lihtsana ja tulevikus pikalt kasutatava võrguhaldusprotokollina.

18.6 Syslog

Võrguadministraatorid ja teised infotehnoloogia valdkonnaga tegelejad vajavad moodust veasituatsioonidest teavitamiseks. Selleks otstarbeks on välja töötatud mitmeid mooduleid, teeke, vahendeid jne. Üks kasutatavamaid on syslog (RFC 3164, 3195). Syslog on standardvahend UNIX ja Linux operatsioonisüsteemides ning toetatav paljudes võrguseadmetes. Ka näiteks veebiprogrammeerimiseks kasutatavas keeles PHP (*PHP: Hypertext Preprocessor*) on sisseehitatud tugi syslogi kasutamiseks. Syslogi jaoks kasutatakse UDP protokolli (vaikimisi porti 514). Syslog eristab kriitilisusastmeid, milledeks on raskusastme järgi (sulgudes kood): hädaolukord (0) (süsteem on mittekasutatav); häire (1) (vajalik on kohene sekkumine); kriitiline (2) (kriitiline situatsioon); viga (3) (veasituatsioon); hoiatus (4) (hoiatuse situatsioon); märguanne (5) (normaalne, kuid tähelepanu vääriv situatsioon); informatiivne (6) (informatiivne teade); silumine (7) (silumistaseme teade). Teate formaat on soovituslik, kuid mitte kohustuslik. Kogunenud logi analüüsiks on saadaval mitmeid programme. IT-administraatori jaoks on oluline aegajalt logisid analüüsida. Vajadusel võtta kasutusele vajalikke meetmeid, kui avastatakse mingeid probleeme.

19. Arvuti seadistamine ja diagnostika-programmid

Oleme käsitlenud võrguliikluse toimimist ja selleks kasutatavaid mitmeid tehnoloogiaid ning protokolle. Edasi vaatleme praktilisemat poolt, eelkõige operatsioonisüsteemide võrgukonfiguratsiooni seadistamist. Vaatleme operatsioonisüsteemiga kaasas olevaid võrguga seonduvaid käsureaprogramme ning uurime mõnda eraldiseisvat programmi.

Operatsioonisüsteemidest vaatleme peamiselt Windowsit ja Linuxit. Kanalikihi protokollidena võtame aluseks peamiselt Etherneti, võrgukihiprotokollidena vaikimisi IPv4 ja ka IPv6 ning transpordikihis UDP ja TCP protokollid.

Enamus operatsioonisüsteemidega kaasas olevatest programmidest on käsureaprogrammid. Käskudes ja mujal kirjutame kasutajast sõltuvad parameetrid suurem-väiksem märkide sees (nt. "<mingi parameeter>"). Fakultatiivsed parameetrid kirjutame kantsulgudesse (nt. "[midagi]").

19.1 Operatsioonisüsteemide võrguvahendid ja seadistused

Elemendi hüpikmenüü avamiseks tuleb elemendi kohal vajutada hiire paremat nuppu. Paljudel juhtudel saab aktiivse elemendi hüpikmenüüd avada, kui vajutada klaviatuuril menüüklahvi, mis asub tavaliselt parempoolsest Ctrl klahvist vasakul (ei pruugi kõikidel klaviatuuridel olemas olla).

Tihti peab Windowsi keskkonnas enne mõnede operatsioonide läbiviimist valima soovitava võrguliidese. Vastavasse valikusse jõudmiseks on mitu teed:

- Vali Start menüüst "Run", sisesta "ncpa.cpl", vajuta sisestusklahvi (klahv Enter).
- Vali Start → Settings → "Control Panel" → "Network and Dial-up Connections".
- Töölaud → "My Network Places" → hüpikmenüü → Properties.

Mõnes kohas viitame Windowsi keskkonnas ka registrivõtmetele. Registrivõtmete puhul lühendame esimese astme nimetust järgnevalt: HKLM = HKEY_LOCAL_MACHINE.

Mõningatel juhtudel on viidatud teenustele. Windowsi keskkonnas saab teenuste käivitamiseks kasutada käsku "net start <teenuse nimi>". Teenuste peatamiseks on käsk "net stop <teenuse nimi>". Kui teenuse nimi sisaldab ka tühikut, siis peab teenuse kirjutama jutumärkidesse. Teenuste käivitustüüpe on kolm: automaatne (*automatic*) (teenus käivitatakse automaatselt arvuti käivitamisel); käsitsi (*manual*) (teenus tuleb kasutajal või mingil programmil käivitada); töövõimetu (*disabled*) (teenust ei ole võimalik käivitada). Teenuste käivitustüüpe saab muuta, kui võtta lahti Windowsi kataloogis alamkataloog "system32" ja käivitada fail "compmgmt.msc" või "services.msc". Olles sisseloginud tavakasutaja õigustes, peab Windows 2000 all vastavale failile tegema kiirkorralduse ja kiirkorraldusfaili atribuutides (valides kiirkorralduse hüpikmenüüst "Properties") märkima linnukese kasti "Run as different user", mille järel kiirkorraldust käivitades saab programmi käivitada administraatori õigustes. Windows XP puhul tuleb administraatorina käivitamiseks hüpikmenüüst valida "Run as...".

Linuxi all saab teenust käivitada käsuga "/etc/init.d/<teenuse nimi> start", tavakasutaja all käsuga "sudo /etc/init.d/<teenuse nimi> start" ("sudo" on käsk, mis võimaldab käivitada käsku mingi muu kasutaja õigustes). Teenuse peatamiseks kasutame parameetri "start" asemel parameetrit "stop" ja taaskäivitamiseks parameetrit "restart".

Linuxi keskkonnas on võimalik käskude kohta saada abiinformatsiooni käsurealt käsuga "man <käsk>". Näiteks "man arp" kuvab abiinformatsiooni käsu "arp" kohta. Manuaali lugedes on navigeerimiseks võimalik kasutada üles ja alla nooleklahve, tühikut (vaate võrra allapoole), "Home" ning "End" klahve. Väljumiseks tuleb vajutada klahvi "q". Lühiinformatsiooni on või-

malik saada käsu kohta, kui käsule anname ette lipu "--help" (nt. "arp --help"). Linuxiga arvutitel on tavaliselt Ethernet-tüüpi võrguliidese nimeks "eth0".

Linuxi keskkonnas on kasutatavad käsud ja nende parameetrid sõltuvuses distributsioonist. Linuxis saab muudatusi permanentseks muuta, modifitseerides konfiguratsioonifaile, mis asuvad olenevalt distributsioonist erinevates kaustades (nt. "/etc/network/interfaces", "/etc/sysconfig/network", "/etc/sysconfig/network-scripts"). Samuti võib vaadeldavad käsud lisada faili "/etc/rc.d/rc.local". Antud failis olevad käsud käivitatakse automaatselt arvuti käivitamisel.

Võrguseadistuse vaatamine

Windows 2000 ja XP keskkonnas pakub võrguseadistuse kohta informatsiooni käsk "ipconfig". Ilma parameetriteta kuvatakse põhiline informatsioon võrguliideste kohta. Lipuga "/all" (või ka "-all") kuvatakse liideste kohta põhjalikum informatsioon.

Windows 9x, ME all saab võrguseadistust näha programmiga "winipcfg.exe" (Start → Run → winipcfg).

Linuxi all saab võrguseadeid vaadata käsuga "ifconfig" (või "/sbin/ifconfig"), mis ilma parameetriteta väljastab info hetkel aktiivsete võrguliideste kohta. Andes parameetrina ette liidese, kuvatakse info ainult vastava liidese kohta. Kasutades lippu "-a", kuvatakse võrguinformatsioon kõikide liideste kohta. Lipuga "-v" kuvatakse täiendavat infot vigade korral. Alternatiivsed käsud on "ifstatus <võrguliides>" ja "ip addr".

MAC-aadressi vaatamine ja muutmine

MAC-aadressi teadasaamiseks Windowsis võib käsureal anda käsu "ipconfig /all". Alternatiivne variant on kasutada käsku "net config rdr". Windows XP keskkonnas on MAC-aadressi võimalik teada saada ka käsuga "getmac". Kasutades lippu "/V", kuvatakse ka liideste ühenduste nimed ja võrguadapterid.

MAC-aadressi muutmiseks valige soovitav võrguliides → hüpikmenüü → Properties → General → "Configure..." → Advanced → Properties → "Network Address" (või "Locally Administered Address") → Value. Siia lahtrisse saab kirjutada uue soovitava MAC-aadressi (ilma sidekriipsudeta). Alternatiivne variant on muuta registreid. Selleks avada registrivõti "HKLM\SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-11CE-BFC1-08002BE10318}". Siin on võtmed nimetustega "0001", "0002", ... Nendest tuleb üles otsida soovitav võti, mille jaoks võib vaadata väärtuse "DriverDesc" sisu. Seejärel tuleb lisada juurde võti "NetworkAddress" (või redigeerida olemasolevat), mille sisuks on uus MAC-aadress (tüübina valida REG_SZ). Peale muutmist tuleb arvutile teha taaskäivitus või muuta võrguliidese staatust "disabled" ja seejärel "enabled" (saab teha võrguliidese hüpikmenüüst).

Linuxi all saab MAC-aadressi vaadata väljundist käskudega "ifconfig", "ip link" ja "ip addr". MAC-aadressi saab muuta kahe järjestikulise käsuga "ifconfig <võrguliides> down hw ether <uus MAC-aadress>" ja "ifconfig <võrguliides> up" (parameeter "up" ja "down" vastavalt aktiveerib või deaktiveerib võrguliidese). Linuxi all on MAC-aadressi kahe kuueteistkümnendarvu eraldajaks koolon (Windowsi all sidekriips).

MAC-aadressi muutmist ei saa teostada tavakasutaja õigustes ja kõik võrgukaardid või seadmedraiverid ei pruugi toetada MAC-aadressi muutmist.

ARP-tabel

ARP-tabeliga ümberkäimiseks on Windowsi keskkonnas käsk "arp", mille lipud ja parameetrid on järgmised:

- "-a [<IP-aadress>]", "-g [<IP-aadress>]" – kuvatakse iga võrguliidese kohta ARP-tabeli sisu. Lisades lõppu IP-aadressi, kuvatakse ainult vastava IP-aadressiga seotud MAC-aadress.

- -s <IP-aadress> <MAC-aadress> – lisab ARP-tabelisse staatiliselt MAC-aadressi. Seda operatsiooni ei saa läbi viia tavakasutaja õigustes. Näidiskäsk: "arp -s 192.168.1.189 00-34-12-54-E5-5F".
- -d [<IP-aadress>] – kustutab ARP-tabeli sisu (ka staatilise). Andes parameetrina IP-aadressi, kustutatakse ainult vastav kirje ARP-tabelist.

ARP-tabeli väljundist näeme tavaliselt kirjeid, millel "Type" väärtuseks on "dynamic", mis tähendab, et vastav MAC-aadress on dünaamiliselt õpitud ARP-protokolli abil. Kui näiteks hetkel ei ole ARP-tabelis mõnda kohtvõrgu IP-aadressi, siis võime tema poole pöörduda näiteks pingides. Seejärel anda uuesti käsk "arp -a" ning nüüd peaks olema vastav kirje lisandunud. Dünaamilised kirjed aeguvad mingi aja jooksul. Kui "Type" veeru väärtuseks on "static", siis on tegemist käsitsi lisatud sidumisega. Käsitsi lisatud kirjed ei aegu.

Linuxi all on ARP-tabeliga ümberkäimiseks samuti kasutatav käsk "arp", mille lipud "-s" ja "-d" on analoogsed Windowsi omale. Mõningaid teisi kasutatavaid parameetreid:

- Parameetriteta, "-a" ja "-e" – kuvatakse ARP-tabeli sisu. Väljundis "Flags Mask" veerus "C" tähistab permanentset kirjet (nt. käsitsi lisatuid).
- -n – ei kuvata IP-aadressile vastavat domeeninime (hostid kuvatakse IP-aadressina).
- -i <võrguliides> – kuvab ainult etteantud võrguliidese ARP-tabeli sisu. ARP-tabelisse kirje lisamisel saab antud lipuga ette anda liidese, kuhu vastav kirje lisatakse. Ilma -i liputa lisamisel proovib tuum ise arvata, millise liidese ARP-tabelisse antud kirje peaks lisatama.
- -f [<faili nimi>] – sarnane lipu "-s" kasutamisele, kuid lisatavad kirjed võetakse etteantud failist. Failis kirjutatakse iga kirje eraldi reale kujul "<MAC-aadress> <IP-aadress>". Kui faili ei anta ette, siis võetakse kirjed failist "/etc/ethers".

Alternatiivne käsk on "ip neigh".

IP-aadressi seadistamine

Windows keskkonnas saab arvuti IP-aadressi seadistada, valides vastava võrguliidese → hüpikmenüü → Properties → "Internet Protocol (TCP/IP)" → Properties. Avanevas aknas saab seadistada IP-aadressi käsitsi või dünaamiliseks ehk automaatselt saadavaks (kasutatakse DHCP protokolli). Käsitsi seadistamisel peaks normaaljuhul seadistama IP-aadressi, võrgumaski, vaikevõrguvärava ja DNS-serveri.

DHCP-ga hangitud IP-aadressiga ümberkäimiseks saab kasutada käsurea käsku "ipconfig" parameetritega:

- /renew [võrguliides] – uuendab DHCP-serverist võrguseadistust kõikidel liidestel, mis on seadistatud DHCP-d kasutama võrguseadistuse saamiseks. Kui parameetrina on antud võrguliides, siis ainult vastaval võrguliidesel.
- /release [võrguliides] – vabastab IP-aadressi (saates teate DHCPRELEASE) ja tühistab ülejäänud võrguseadistuse, mis DHCP-ga saadi. Kui parameetrina on antud võrguliides, siis teostatakse operatsioon ainult vastaval võrguliidesel.

Kasutades võrguseadistuse jaoks DHCP-d, võtab arvuti käivitamine natuke rohkem aega, sest toimub IP-aadressi ja võrguseadistuse hankimine (DHCP-server võib pingides kontrollida IP-aadressi unikaalsust). Arvutil, mis on ainult ühes kohtvõrgus ja millel seadistust ei ole vaja kesksel muuta (näiteks kodudes), võiks seadistada võrguseadistuse staatiliseks. Selleks võib ka kasutada DHCP poolt seadistatud väärtusi (näeb "ipconfig -all" väljundist). Staatilise võrguseadistuse puhul on võimalik DHCP klientteenus "Dhcp" (ehk "DHCP Client") ära keelata või panna käsitsi käivituvaks, millega hoitakse kokku süsteemi ressursse ja arvuti alglaadimise aega. Samas, kui arvutiga ringi liigutakse (näiteks käiakse sülearvutiga tööl ja kodus), on mõistlik määrata arvuti võrguseadistust saama DHCP-serverist.

Linuxi all seadistatakse IP-aadress käsuga "ifconfig <liides> <uus IP-aadress> netmask <võrgumask>". Eraldi tuleb anda võrguvärava aadress, mida saab teha käsuga "route add default gw <võrguvärava IP-aadress> [<liides>]". Nimeserverite infot muudetakse failis "/etc/resolv.conf",

kuhu kirjutatakse iga nimeserveri kohta eraldi rida "nameserver <DNS-serveri IP-aadress>". Linuxi all saab seadistuse määrata DHCP-serverist saadavaks käsuga "ifup <võrguliides> dhcp" (käivitab skripti asukohas /sbin/ifup). Alternatiivne IP-aadressi lisamise käsk on "ip address add", näiteks "ip address add 192.168.0.123/24 dev eth0" (mask antakse ette kaldjoone formaadis). Eelnevalt lisatud IP-aadressi saab eemaldada, kasutades käsku "ip address del 192.168.0.123 dev eth0".

Hosti nime vaatamine ja muutmine

Windowsi keskkonnas saab arvuti nime vaadata käskude "ipconfig -all", "net config workstation" ja "hostname" (Windows 2000 all puudub) väljundist. Muuta saab hosti nime, kui minna töölaud → "My Computer" → hüpikmenüü → "Computer Name" → Properties → Change → "Computer name:" ja siia kirjutada uus soovitav hosti nimi (mõjumiseks vajatakse arvuti taaskäivitust).

Linuxi keskkonnas saab hosti nime vaadata käsuga "hostname". Hosti nime saab muuta käsuga "hostname <uus hostinimi>" (ei ole permanentne). Permanentselt saab hostinime muuta failis "/etc/hostname" (sisaldab ainult hosti nime).

Ühenduse kontrollimine

Pingimine on üks võimalus kontrollida osapoolte vahelist ühendust. Pingimiseks kasutatakse käsku "ping" (nii Windows kui ka Linuxi keskkondades). Pingides aadressi 127.0.0.1, saab teada kohaliku masina TCP/IP funktsioneerimise.

Windows keskkonna käsk "ping" pakub mitmeid lisaparameetreid ja lippusid:

- -t – pingitakse, kuni pingimine katkestatakse kasutaja poolt. Vahetulemusi on võimalik vaadata, vajutades klahvikombinatsiooni Ctrl+Break. Katkestamiseks on klahvikombinatsioon Ctrl+C.
- -n <täisarv> – antakse ette, mitu korda pingitakse. Ilma "-t" ja "-n" lippudeta pingitakse vaikimisi neli korda.
- -a – ei teostatata sihthosti IP-aadressile pöördteisendust (ehk ei leita vastavale IP-aadressile vastavat domeeninime). Tihti vajalik, sest DNS-serverilt vastuse saamine ja seetõttu ka väljundi kuvamine võtab aega ning DNS-server võib mitte olla kättesaadav.
- -l <täisarv> – määrab pingimisel saadetava "kaja päring" (*echo request)* päringu suuruse baitides. Vaikimisi on suuruseks 32 baiti.
- -f – saadetavas "kaja päring" paketis pannakse püsti fragmenteerimist mittelubav lipp DF (*don't fragment*).
- -i <naturaalarv> – määrab saadetava "kaja päring" paketi "aega elada" välja väärtuse.
- -r <naturaalarv> – kasutab "record route" valikut. Antud valiku puhul lisavad marsruuterid, mida pakett läbib, enda IP-aadressi paketi "record route" valiku vastavale väljale, millega salvestatakse paketi teekond, mille maksimaalne suurus on üheksa marsruuteri läbimist. Kui teekonnal on rohkem kui üheksa marsruuterit, siis alates kümnendast marsruuterist saadetakse pakett lihtsalt edasi. Parameetri väärtus saab olla vahemikus 1 kuni 9. Pikema vahemaa puhul saadakse teada ainult teekonna alguse marsruuterid.
- -j <IP-aadress1 IP-aadress2 ...> – saab kasutada "loose source route" valikut. Antud valiku puhul suunatakse pakett läbi määratud marsruuterite sihtpunktini. IP-aadresse saab olla maksimaalselt üheksa tükki. Käsus eraldatakse IP-aadressid omavahel tühikuga (näidiskäsk: "ping -j 192.168.3.2 192.168.8.1 192.168.101.1").
- -k <IP-aadress1 IP-aadress2> – saab kasutada "strict source route" valikut, mis on sarnane eelmisele, kuid nõuab rangelt ainult märgitud IP-aadressidega marsruuterite läbimist. Nimekirjas iga järgmine vastavat IP-aadressi omav marsruuter peab olema vahetult ühendatud eelmise marsruuteriga.

Linuxi all töötab käsk "ping" vaikimisi seni, kuni ta katkestatakse. Käsuga "ping" kasutatavad parameetrid:

- -c <naturaalarv> – määrab, mitu korda pingimist tehakse.
- -i <reaalarv> – määrab "kaja päring" pakettide saatmise tiheduse sekundites. Vaikimisi 1 sekund. Võimalik on kasutada ka täpsemaid kui sekundilisi vahesid. Näiteks käsuga "ping -i 0.02 192.168.1.4" saadetakse "kaja päring" pakette 0.02 sekundiliste vahedega.
- -a – annab "kaja vastus" paketi saamisel piiksuga märku.
- -b – kasutatakse leviedastusaadressi pingimisel.
- -l <naturaalarv> – saadetakse etteantud hulk "kaja päring" pakette, enne kui oodatakse vastust.
- -s <naturaalarv> – määrab "kaja päring" paketi andmete osa suuruse (lisatakse veel ICMP protokolli päis). Vaikeväärtus on 56.
- -p <kuueteistkümnendsüsteemis olev arv> – saab määrata "kaja päring" paketi poolt saadetavate andmete mustri ehk sisu baidid (näiteks kui pingime käsuga "ping math. ut.ee -p 5652", siis saadetav andmeosa on ASCII tekstina "VTVTVTVT<pikkus pingi paketi jagu>").
- -q – näidatakse ainult kokkuvõtet pingimisest.
- -t <naturaalarv> – määrab "kaja päring" paketi "aega elada" välja väärtuse.
- -M <"do", "dont" või "want"> – MTU-ga ümberkäimise valikud: "do" (pannakse püsti "ära fragmenteeri" lipp), "dont" (ei seata "ära fragmeteeri" lippu), "want" (fragmenteeritakse kohapeal, kui pakett on liiga suur)
- -R – kasutatakse "record route" laiendust (analoogne Windowsi ping käsu "-r" lipuga).
- -r – kasutatakse "loose source route" valikut (analoogne Windowsi ping käsu "-k" lipuga)
- -w <naturaalarv> – seatakse aeg sekundites, kui kaua pingimist teostatakse.
- -V – kuvatakse ping-programmi versiooniinfo.

Paljudes arvutites on pingimisele vastamine keelatud turvalisuse kaalutlustel. See tähendab, et ei vastata "kaja päring" teatele "kaja vastus" teatega.

Üks pingimise modifikatsioone on käsk "traceroute" (Linux, Mac OS X) või "tracert" (Windows). "Traceroute" võimaldab teada saada teekonnal läbitavaid marsruutereid. Samuti võib "traceroute" aidata lokaliseerida võrguprobleeme. "Traceroute" saadab alguses sihtpunktile paketi (Windowsis ICMP "kaja päring" pakett, Linuxis vaikimisi UDP-pakett), mille välja "aega elada" väärtus on üks. Kuna naabri juures jõuab paketi "aega elada" väärtus nulliks, siis saadab naaber-marsruuter tagasi ICMP veateate "eluaja ületamine" (*time exceeded*). Saanud kätte veateate, saadetakse järgmine pakett, mille "aega elada" välja väärtus on suurendatud ühe võrra (ehk kaks). Teele saadetud pakett jõuab nüüd teise marsruuterini, kust saadetakse tagasi ICMP teade "eluaja ületamine". Protsessi jätkatakse seni, kuni saadakse tagasi vastuse pakett, "aega elada" väärtus ületab 255 või saadakse muu ICMP-veateade. Iga marsruuter, mis saadab tagasi ICMP teate, paneb paketi lähteaadressiks oma IP-aadressi, mille põhjal on võimalik kokku panna täpne teekond. "Traceroute" tulemusi analüüsides peab arvestama, et alati ei pruugi paketid läbida sama teed. Iga hüppe kohta saadetakse kolm paketti, mille vastuse saamise ajad kuvatakse väljundis kolme tulbana.

Vaatleme käsuga "tracert" (Windowsi keskkonnas) kasutatavaid parameetreid:

- -d – ei teostatata sihthosti IP-aadressile pöördteisendust (ehk ei leita vastavale IP-aadressile vastavat domeeninime). DNS-serverilt vastuse saamine võtab aega.
- -h <hüpete arv> – antakse ette maksimaalne hüpete arv ehk paketi välja "aega elada" väärtus.
- -w <naturaalarv> – antakse ette aegumise aeg millisekundites, mil loetakse vastus kaotsiläinuks.

- -j <ip1 ip2> – kasutatakse "loose source route" valikut, millega antakse ette IP-aadressid, mille pakett peab läbima. Näidiskäsk: "tracert -j 10.100.0.1 10.100.3.1 10.79.2.1 kase.mahla.naide.ee".

Linuxi all kasutatava käsu "traceroute" parameetrid (ei pruugi kõikidel distributsioonidel olla samad võtmed ja kõiki võimalusi):

- "-4", "-6" – antakse ette kasutatav IP versioon (IPv4 või IPv6). Vaikimisi domeeninime lahendades valitakse kasutatav IP-aadressi protokoll automaatselt (kui saadakse ainult IPv6-aadress, siis kasutatakse "traceroute" tegemiseks IPv6-aadressi ja analoogselt ka IPv4 aadressi puhul). Kui saadakse mõlemad kirjed, siis vaikimisi eelistus oleneb realisatsioonist.
- "-I", "-T", "-U" – kasutatakse vastavalt ICMP "kaja päring", TCP (SYN lipuga) või UDP pakette. Vaikimisi kasutatakse lippu "-U" (ehk UDP pakette).
- -F – seatakse pakettidel püsti "ära fragmenteeri" lipp.
- -f <naturaalarv> – määrab esimesena saadetava paketi "aega elada" välja väärtuse. Vaikimisi on väärtuseks 1.
- -i <liides> – määrab liidese, mille kaudu paketid välja saadetakse (vaikimisi tehakse otsus marsruutimistabeli alusel).
- -m <naturaalarv> – määrab maksimaalse "aega elada" välja väärtuse, milleni jõudes "traceroute" lõpetatakse, kui pole muul põhjusel lõpetatud. Vaikeväärtus on 30.
- -N <naturaalarv> – määrab, mitu paketti korraga saadetakse. Mitme paketi saatmine kiirendab "traceroute" tulemuse saamist. Vaikeväärtuseks on 15.
- -n – ei kuvata IP-aadressi asemel domeeninimesid. DNS-päringud võtavad aega, mille tõttu on domeeninimedega väljundi kuvamine aeglasem.
- -p <pordi number> – määratakse päringute pordinumber UDP ja TCP puhul (hakatakse iga järgmise saatmise sammuga suurendama ühe võrra). Vaikeväärtus on 33 434.
- -w <sekundid> – määrab aja sekundites, kui kaua oodatakse vastust (vaikimisi 5 sekundit).
- -q <naturaalarv> – määrab, mitu paketti saadetakse hüppe kohta (vaikimisi 3).
- -s <IP-aadress> – määrab paketi lähteaadressi, mis peab olema üks arvuti IP-aadressidest. Vaikimisi võetakse paketi väljundliidese IP-aadress.
- -z <reaalarv> – minimaalne aeg pakettide saatmiste vahel millisekundites. Vajalik kasutada, kui mõned marsruuterid omavad ICMP teadete saatmise piiranguid.
- -V – kuvatakse programmi versiooniinfo.

Raja MTU leidmiseks kahe otspunkti vahel saab kasutada pingimist. Windowsi keskkonnas oleks käsk kujul: "ping <host> -l <suurus> -f". Näiteks "ping www.math.ut.ee -l 1480 -f". Selliselt seame IP-paketis püsti lipukese "ära fragmenteeri", mille tulemusena saadakse tagasi ICMP teade, et sel sammul ei toetata suuremat MTU väärtust. Vastavalt suurendades või vähendades paketi suurust saame leida kõige suurema väärtuse, millega pakett veel läbi läheb.

Linuxi all saab kasutada käsku "tracepath", mis on sarnane "traceroute" käsule, kuid näitab väljundis ka teekonna MTU-d.

Windows keskkonnas annab otspunktide vahelise teekonna kõikide sõlmede kohta viivituse informatsiooni käsk "pathping", mis kombineerib käske "traceroute" ja "ping". "Pathping" kasutab esimese etapina "traceroute"-mist, millega saadakse teada otspunkte ühendavad marsruuterid. Järgmisena pingitakse kõiki neid mitu korda (vaikimisi 100 korda), mille tulemusena esitatakse koondaruanne pingimise edukuse ja keskmise reageerimisaja kohta. "Pathping" parameetreid:

- -n – ei teostatata IP-aadresside kohta pöördteisendust (ehk ei leita IP-aadressile vastavat domeeninime). DNS-serverilt vastuse saamine võtab aega.
- -p <naturaalarv> – määrab "kaja päring" pakettide saatmise intervalli millisekundites. Vaikimisi 250 millisekundit.

- -q <naturaalarv> – määrab, mitu "kaja päring" paketti igale teele jäävale marsruuterile saadetakse. Vaikimisi 100 tükki.
- -R – tehakse iga teele jääva marsruuteri kohta kindlaks, kas toetatakse RSVP (*Resource Reservation Protocol*) protokolli (RSVP võimaldab reserveerida ribalaiust andmevoo jaoks).

Ühenduste info

Aktiivsete ühenduste kohta saab informatsiooni käsuga "netstat" (nii Windowsis kui ka Linuxis). Lisaks aktiivsetele ühendustele võimaldab "netstat" käsk vaadata ka marsruutimistabelit, Etherneti ja IP statistikat. Windowsi keskkonna käsu "netstat" parameetreid:

- -a – kuvatakse info nii kuulavate kui ühendatud soklite kohta. Ühenduste kohta tuuakse välja protokoll, pordid ning ühenduste seisundid. Väljundis "*" tähistab suvalist IP-aadressi või porti. Neid kuvatakse näiteks siis, kui kuulatakse mingit porti (protsess on hõivanud mingi pordi), oodates sissetulevaid ühendusi (näiteks veebiserveri külastust).
- -n – ei teostata IP-aadressile vastava domeeninime leidmist (väljundi kuvamine võtab märgatavalt rohkem aega). Samuti kuvatakse pordid numbriliselt, mitte nimeliselt.
- -p <protokoll> – võimaldab protokolli alusel ühendusi filtreerida. Protokollideks võib siin olla "tcp", "udp", "icmp", "ip", "tcpv6", "udpv6", "icmpv6" või "ipv6".
- -s – kuvab statistika protokollipõhiselt. Lipuga "-p" saab kuvada statistikat ka ainult ühe protokolli kohta.
- -r – väljastab marsruutimistabeli. Samaväärne käsuga "route print". Vaata jaotist "Marsruutimistabel", lk. 167.
- -o – väljastab lisaks ka ühendust omava protsessi identifikaatori ehk PID-i (*process identifier*). PID numbrile vastavat rakendust on võimalik ühe võimalusena teada saada, kui võtta lahti tegumihaldur ("Task Manager") (vajutades klahvikombinatsiooni Ctrl + Alt + Del ja valida "Task Manager" või hiirega kella peale minnes vajutades paremat hiireklahvi ja valides "Task Manager"). Tegumihaldurist valida View → "Select Columns ..." → panna linnuke kasti "PID (Process Identifier)" → OK. PID veerul vajutades sorteeritakse PID-i järgi protsessid. Antud lipp ei ole kasutatav Windows 2000 keskkonnas.

Linuxi keskkonnas on samuti olemas käsk "netstat", millega saab vaadata ühendusi, liideste statistikat ja multiedastusgruppidesse kuulumist. Informatsioon, mis netstat poolt väljastatakse, määratakse järgnevate parameetritega:

- ilma parameetriteta – kuvatakse kõik avatud ühendused. Väljundi tähendusi: proto – protokoll; Recv-Q – mitu baiti on vastuvõtupuhvris, mida programm ei ole omale veel ära kopeerinud; Send-Q – mitmele baidile andmetele ei ole teiselt osapoolelt veel kinnitust saadud; RefCnt – viidete loendur vastavale soklile.
- -g – kuvatakse multiedastusgruppidesse kuulumine.
- -i – kuvab võrguliideste kasutusaktiivsust (saadetud ja vastuvõetud kaadrid, vead, MTU suurus jms.). Väljundi lippude tähendusi: B – leviedastusaadress on olemas; L – tagasisidestusliides; M – liides töötab *promiscuous* laadis; N – lisasid välditakse; O – ARP-i ei kasutata vastaval liidesel; P – punktist-punkti ühendus; R – liides töötab; U – liides on püsti. Sama väljundi annab ka käsk "ifconfig -s".
- -s – kuvatakse erinevate protokollide loendurite seisundeid.

Käsu "netstat" parameetrid, millega muudetakse väljundit:

- -n – keelatakse numbriliste andmete tõlkimine nimedeks (pordid, IP-aadressid, kasutajate identifikaatorid).
- "--numeric-hosts", "--numeric-ports", "--numeric-users" – analoogne "-n" lipule, kuid vastavad lipud võimaldavad täpsemalt keelata soovitud numbriliste andmete tõlkimist nimedeks. Lipud käivad vastavalt IP-aadresside, pordinumbrite ja kasutaja identifikaatorite kohta.

- -A <protokoll1,protkoll2> – määrab väljastatavad protokollid. Kahe või enama protokolli väljastamiseks kirjutatakse nende vahele koma. Võimalikke variante: "inet" (IP), "inet6" (IPv6), "unix", "ipx", "ax25", "netrom" ja "ddp".
- -c – väljastab tulemuse uuesti iga sekundi tagant. Katkestamiseks klahvikombinatsioon Ctrl + C.
- "-e", "-ee" – väljastatakse täiendavat informatsiooni (nt. ühendussoklit omav kasutaja).
- -o – väljastatakse lisaks ka taimerite informatsioon.
- -p – väljastatakse lisaks ka protsessi identifikaator ehk PID (*process identifier*) ja protsessi nimi.
- -l – kuvab kuulamisseisundis olevad soklid (vaikimisi kuvatakse juba ühendatud soklid).
- -a – kuvab kõik soklid (nii kuulavad kui ühendatud). Kasutades koos lipuga "-i" kuvatakse kõikide liideste info.
- -v – väljastab info ka mitteseadistatud protokollide kohta.

DNS-kliendi päringud ja seadistamine

Windowsi keskkonnas saab kasutatavaid DNS-servereid vaadata käsu "ipconfig -all" väljundi realt "DNS Servers". Windowsi keskkonnas saab seadistada DNS-servereid, kui valida vastav võrguliides → hüpikmenüü → Properties → "Internet Protocol (TCP/IP)" → Properties. Antud aknas on kaks lahtrit DNS-serverite IP-aadresside jaoks. Vajadusel võib ka ainult esimese täita. Alternatiivset DNS-serverit kasutatakse juhul, kui eelistatud DNS-server ("Preferred DNS-server") ei ole kättesaadav või ei suuda nime lahendada. Kasutades DHCP-d võrguseadistuse jaoks, saadakse DNS-serveri IP-aadressid DHCP-serverist automaatselt koos muu võrguseadistusega.

DNS-infoga läbiviidavate operatsioonide jaoks saab kasutada käsku "ipconfig" parameetritega:

- /displaydns – kuvatakse arvutis puhverdatud DNS-i ressursikirjed.
- /flushdns – eemaldatakse arvutist puhverdatud DNS-i ressursikirjed.

Domeeninime lahendamiseks saab kasutada käsku "nslookup". Nslookup võib töötada nii interaktiivses (antakse lihtsalt käsk "nslookup" ilma parameetriteta) kui ka mitteinteraktiivses laadis (antakse käsuga ka kohe parameetrid). Mitteinteraktiivses laadis antakse käsurealt näiteks käsk "nslookup ave.ee", mis leiab domeeninimele "ave.ee" vastava IP-aadressi. Samamoodi antakse ette ka pöördteisendatav IP-aadress. Näiteks "nslookup 193.40.36.2".

Interaktiivses laadis kasutatavaid käskusid:

- exit – väljub programmist. Alternatiivne variant on klahvikombinatsioon Ctrl + C.
- help – kuvab abiinformatsiooni.
- lserver <DNS-server>, server <DNS-server> – määrab etteantud DNS-serveri vaikimisi kasutatavaks DNS-serveriks. Käsud "server" ja "lserver" erinevad omavahel sellepoolest, et esimene vaatab informatsiooni uue nimeserveri kohta jooksvalt kasutatava nimeserveri vahendusel, kuid teisel juhul kasutatakse esialgset nimeserverit.
- ls [valikud] domeeninimi ["> <failinimi>" või ">>failinimi"] – küsib jooksvast DNS-nimeserverist täielikku tsooniinfo ülekannet. Soovikorral salvestatakse tulemused faili (">" nullib olemasoleva faili sisu, ">>" kirjutab olemasoleva faili lõppu. Vajaduse korral luuakse fail automaatselt). Valiku "-t <päringutüüp>" korral saab küsida ainult soovitud kirjeid. Näiteid käskudest: "nslookup"; "server ns.ut.ee"; "ls ut.ee"; "ls math.ut.ee >> failikeutee"; "ls -t ns ut.ee".
- set võtmesõna[=väärtus] – seatakse seansiparameetreid. Väärtusteks võivad olla:
  - ◇ all – kuvatakse kõikide seadistuste väärtused.
  - ◇ debug, d2 – lülitavad sisse silumisinfo väljastamise. Silumisinfo väljastamist saab maha võtta vastavalt võtmesõnadega "nodebug" ja "nod2". Võtmesõna "d2" väljund on natuke detailsem.
  - ◇ ignore – ignoreeritakse pakettide kärpimise lippu (ei pöörduta TCP protokolliga

täiendava informatsiooni saamiseks, kui UDP datagrammi ei mahtunud kogu vastus). Ignoreerimise tühistamiseks on kasutatav võtmesõna "noignore".
    - ◇ port=<naturaalarv> – muudab DNS-serveri poole pöördumise porti. Vaikimisi 53.
    - ◇ querytype=<ressursikirje tüüp>, type=<ressursikirje tüüp> – määrab päringu ressursikirje tüübi. Vaikimisi on ressursikirjetüübiks "A". Võimalikke valikuid: A, ANY (kõik ressursikirjed), CNAME, GID (grupi nime identifikaator), HINFO, MB (postiserveri domeeninimi), MX, NS, PTR, SOA, TXT, UID (kasutaja identifikaator), UINFO (kasutaja informatsioon), WKS (*well-known service*) (pakutavad teenused).
    - ◇ recurse – määrab teostatavaks päringutüübiks rekursiivse päringu. Iteratiivsete päringute teostamiseks on märksõna "norecurse". Vaikimisi kasutatakse võtit "recurse".
    - ◇ retry=<naturaalarv> – määrab, mitu korda korratakse vastuseta päringut.
    - ◇ timeout=<naturaalarv> – määrab, mitu millisekundit oodatakse päringule vastust.
    - ◇ root <juurnimeserver> – muudab päringute juurnimeserveri.

DNS-serverile päringute tegemiseks on Linuxi all kasutatavad käsud "nslookup" ja "dig" ("dig" on uuem ja paindlikum). Käsu "dig" kasutamise parameetreid:

- Ilma parameetriteta – kuvatakse juurnimeserverid ehk tehakse päring domeenile ".".
- -b <aadress>[#<pordi number>] – määrab päringu IP-lähteaadressi, mis peab olema üks arvuti IP-aadressidest. Määrata saab samuti päringute lähteporti (port peab olema vaba, hõivatust saab kontrollida käsuga "netstat -n -A inet -a").
- -f <failinimi> – võtab päringud etteantud failist. Failis asuvad päringud peavad asetsema eraldi ridadel ja nad on samasugused nagu käsurea käsud.
- -p <pordi number> – määrab päringu DNS-serveri pordi. Vaikimisi on pordiks 53.
- "-4", "-6" – käsivad kasutada vastavalt IPv4 või IPv6 protokolli päringu edastamiseks.
- -t <päringutüüp> – määrab päringu tüübi. Vaikimisi on päringutüübiks "A".

Päringu valikuid:

- +[no]tcp – määrab, kas kasutatakse TCP protokolli päringu tegemiseks või mitte. Vaikimisi kasutatakse UDP protokolli ja vajadusel TCP protokolli, kui vastus oli liiga suur ühe UDP datagrammi jaoks.
- "+[no]aaonly", "+[no]aaflag" – määrab päringu AA (*authorative answer*) lipu.
- +[no]cl – määrab, kas kuvatakse väljundis CLASS veerg (tavaliselt alati väärtuseks IN). Kuvatakse vaikimisi.
- +[no]ttlid – määrab, kas kuvatakse välja "aega elada" ehk TTL väärtus. Kuvatakse vaikimisi.
- +[no]recurse – määrab, kas päring on rekursiivne või iteratiivne. Vaikimisi on päringud rekursiivsed v. a. valikute "+nssearch" ja "+trace" korral.
- +[no]trace – määrab, kas lahendatakse nimi ise alates juurserverist, kasutades iteratiivseid päringuid, mis kuvatakse ka väljundis. Vaikimisi ei kasutata.
- +[no]cmd – määrab, kas kuvatakse väljundi alguses dig'i versioon ja päringu valikud. Kuvatakse vaikimisi.
- +[no]comments – võimaldab väljundis kommentaare mitte kuvada. Kuvatakse vaikimisi.
- +[no]stats – määrab, kas väljundis kuvatakse päringu kohta infot (reageerimiskiirus, vastuse suurus, päringu sooritamise aeg, nimeserver). Kuvatakse vaikimisi.
- +[no]qr – määrab, kas väljundis kuvatakse päring. Vaikimisi ei kuvata.
- "+[no]question", "+[no]answer", "+[no]authority", "+[no]additional" – määrab, kas kuvatakse vastavalt küsimuse sektsioon (QUESTION SECTION), vastuse sektsioon (ANSWER SECTION), pädevuse sektsioon (AUTHORITY SECTION) või lisasektsioon (ADDITIONAL SECTION). Vaikimisi kuvatakse kõik sektsioonid.
- +[no]all – määrab, kas kuvatakse kõik sektsioonid või mitte midagi. Võimalik kombineerida teiste valikutega (järjekord on oluline, edasised väärtused katavad üle varasemad). Näiteks päring "dig www.math.ut.ee +noall +answer" kuvab ainult vastuse sektsiooni, aga näiteks "dig www.math.ut.ee +answer +noall" ei kuva mitte midagi).

- +[no]short – võimaldab kuvada väljundi lühiformaadis. Vaikimisi ei kasutata. Kasutades ka valikut"+[no]identify", kuvatakse ka informatsiooni allikas (vaikimisi ei kuvata).
- +time=<mitu sekundit> – määrab aja, kui kaua oodatakse vastust. Vaikimisi viis sekundit, minimaalselt üks sekund.
- "+tries=<naturaalarv>", "+retry=<naturaalarv>" – määrab, mitu korda proovitakse nimeserverile UDP päringuid saata. Vaikimisi kolm. Valiku "+retry" korral ei arvestata esimest korda sisse.
- +[no]ignore – määratakse, kas päring teostatakse TCP protokolli kasutades, kui UDP paketti ei mahtunud kogu vastus ära. Vaikimisi teostatakse päring TCP protokolli kasutades täiendava informatsiooni saamiseks.
- +[no]nssearch – otsitakse vastava domeeninime jaoks pädevat nimeserverit ja kuvatakse SOA kirjed, mis igal nimeserveril antud tsooni jaoks on.

Linuxi all sõltub domeeninimede lahendamise järjekord failis "/etc/host.conf" olevast seadistusest "order". Vaikimisi peaks seal sisalduma rida "order hosts, bind". "hosts" (aliased on "hosttable", "htable") puhul otsitakse failist "/etc/hosts" (nimelahendused on eraldi ridadel ja kujul "<IP-aadress> <nimi>[<alias1 alias2>]") sobivat nimelahendust ja "bind" (aliased on "dns", "domain") puhul kasutatakse DNS-nimelahendajateenust. Seega, kui kirjutada faili "/etc/hosts" rida "192.168.2.3 math.ut.ee", siis pöördutakse math.ut.ee poole pöördumiseks IP-aadressile 192.168.2.3, sest "hosts" oli järjekorras eespool kui "bind".

Marsruutimistabel

Windowsi keskkonnas on marsruutimistabeliga ümberkäimiseks käsk "route", mille süntaks on "route [-f] [-p] [<käsk> [<käsu võrk>] [mask <võrgumask>] [<võrguvärav>] [metric <meetrika väärtus vahemikus 1-9999>] [if <liides>]]"

Käsud on järgmised:

- print – kuvab arvutis sisalduva marsruutimistabeli. Sama tulemuse annab ka käsk "netstat -r". Marsruut 0.0.0.0/0 tähistab vaikemarsruuti.
- add – võimaldab lisada juurde uue marsruudi. Käsu üldine kuju on "route add <võrguaadress> mask <võrgumask> <järgmise hüppe aadress>".
- delete – kustutab marsruutimistabelist etteantud võrgu. Käsu üldine kuju: "route delete <võrguaadress>". Võimalik on kasutada ka metamärki "*". Näiteks kui soovitakse kustutada kõik võrgud, mis algavad oktettidega "172.20.", siis vastav käsk on "route delete 172.20.*".
- change – muudab olemasolevat marsruutimiskirjet.

Kasutada on võimalik lippe:

- -f – puhastab marsruutimistabeli, jättes alles ainult mõned marsruudid, nende seas tagasisidestusvõrgu (127.0.0.0/8), masina enda IP-aadressi maskiga /32 ja kohaliku võrgu leviedastusaadressid (maskiga /32). Peale marsruutimistabeli kustutamist kustutatakse ka vaikemarsruut, mille tagasisaamiseks tuleb vaikemarsruut uuesti lisada. Vastavaks käsuks on "route ADD 0.0.0.0 MASK 0.0.0.0 <marsruuteri IP-aadress>".
- -p – lisab persistentse marsruudi marsruutimistabelisse, mis säilib ka peale arvuti taaskäivitamist (vaikimisi ei ole persistentne) (marsruut lisatakse registrisse asukohaga HKLM\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\PersistentRoutes). Parameetri "-f" kasutamisel kustutatakse ka persistentsed marsruudid.

Linuxis saab marsruutimistabelit kuvada käsuga "route", mille käsud on kokkuvõtlikult järgmised (ei ole permanentne mõju):

- route [-CFvnee] – marsruutimistabeli sisu vaatamine.
- route [-v] [-A <protokoll>] add [<"-net" või "-host">] <sihtaadress> [netmask <võrgumask>] [gw <võrguvärava IP-aadress>] [metric <naturaalarv>] [mss <naturaalarv>] [reject] [[dev] <võrguliides>] – lisab marsruutimistabelisse uue kirje (võtmesõna on "add"). Näide: "route add -net 192.168.208.0 netmask 255.255.254.0 dev eth0"

- route [-v] [-A <protokoll>] del [<"-net" või "-host">] <sihtaadress> [gw <võrguvärava IP-aadress>] [netmask <võrgumask>] [metric <naturaalarv>] [[dev] <võrguliides>] – eemaldab marsruutimistabelist etteantud kirje (võtmesõna on "del").
- route [-V] [--version] [-h] [--help] – muud käsud.

Käsu "route" parameetreid:

- -n – IP-adresse ei teisendata domeeninimedeks.
- "-v", "-n" – kuvatakse ka lisainformatsioon (nt. meetrika).
- -ee – kuvatakse kõik marsruutimistabeli kirjete parameetrid.
- -A <protokolli tähis> – kuvatakse soovitud protokolli marsruutimistabel, vaikimisi väärtuseks "inet" (IPv4). Mõningaid teisi valikuid: inet6 (IPv6), ipx (IPX), x25 (X.25).
- -C – kuvab Linuxi tuuma marsruutimise vahemälu.
- "-net" või "-host" – ütleb, mismoodi sihtaadressi käsitletakse, kas vastavalt võrguna või hostina.
- netmask <võrgumask> – määrab võrgu võrgumaski (nt. "255.255.128.0").
- gw <võrguvärava IP-aadress> – määrab võrguvärava (BSD süsteemidega ühilduvuse jaoks).
- metric <naturaalarv> – määratakse marsruudi meetrika.
- mss <naturaalarv> – määrab marsruudi maksimaalse segmendi suuruse ehk MSS-i baitides (vaikimisi MTU miinus päised või raja MTU).
- reject – vastav marsruut määratakse ebaõnnestumis-marsruudiks, kuhu saadetud paketid minema visatakse.
- [dev] <võrguliides> – määratakse, millisest võrguliidesest saadetakse välja marsruuditav pakett.

Marsruutimistabeli kuvab ka käsk "netstat -r". Marsruutimistabeli vaatamisel on väljundis mitmeid tähiseid ja lühendeid: A – lisatud võrguaadressi automaatseadistuse poolt; D – marsruut on dünaamiliselt loodud; G – võrguvärav; H – sihtaadress on host; U – marsruut on püsti, kasutatav; ! – ebaõnnestumise marsruut. Marsruutimistabeli vaatamiseks ja sinna muudatuste tegemiseks on alternatiivseks käsuks veel "ip route".

Käsu "netsh" kasutamine

Võrguseadistuse vaatamiseks ja muutmiseks on võimalik Windowsi keskkonnas kasutada käsku "netsh". Netsh on käsurea skriptimise töövahend, mis võimaldab kohalikus või ka kaugmasinas kuvada ja muuta võrguseadistust arvutites. Samuti võimaldab netsh skriptimise abil teha hulgitöötlust, salvestada konfiguratsiooni skriptina (tagavaraks või teiste arvutite kiireks seadistamiseks). Käivitades "netsh", avaneb interaktiivne käsurida. Netsh kasutamise käsurida on hierarhiline, seda nimetatakse kontekstiks. Kontekste pakuvad erinevad pluginad, mida nimetatakse "helper"-teks. Kontekstis olemisest saab informatsiooni käsureaviida järgi, mis on sarnane käsuga, millega konteksti mindi (algselt "netsh>"). Vaatleme mõningaid käskusid.

- "?", "help" – kuvatakse vastava taseme käsuloend (kasutatav ka teistes tasemetes).
- ".." – minnakse ülemtasemesse.
- "bye", "quit", "exit" – sulgeb netsh.
- alias [<aliase nimi>] [<aliasele vastav käsk>] – võimaldab asendada mingi käsu kasutajale sobiva aliasega. Näiteks peale käsu "alias q bye" andmist võib anda käsu "q", mis käivitab käsu "bye". Defineeritud aliaseid ja nende sisu saab näha, kui anda käsk "alias" (ilma parameetriteta). Aliast saab tühistada käsuga "unalias <aliase nimi>".
- dump – genereerib skripti, mis sisaldab jooksvat konfiguratsiooni.
- exec <skriptifailinimi> – käivitab etteantud failis olevad käsud.
- "online", "offline", "commit", "abort" – netsh saab töötada kahes laadis: "online" ja "offline". Laadis "online" viiakse kõik seadistused kohe täide. Laadis "offline" ei viida seadistusi täide enne, kui antakse käsk "commit", millega viiakse täide kõik tehtud sea-

distused. Tehtud seadistuse tühistamiseks on "abort" (käsud "commit" ja "abort" ei oma mõju "online" laadis).

- show <valikud: alias, helper, mode> – kuvab informatsiooni vastavalt parameetrile. "show alias" on analoogne käsule "alias"; "show helper" kuvab kõik helperid ehk netsh kontekstide pluginad, kuvades GUID-i (*globally unique identifier*), plugina failinime ja konteksti saamise käsu; "mode" näitab töötamise laadi ("online" või "offline").
- "add helper <faili nimi>", "delete helper <faili nimi>" – võimaldab vastavalt lisada või eemaldada helper-pluginat "netsh" programmis.
- interface – siirdutakse alamkonteksti, kus käsureaviidaks saab "interface>". Antud kontekst käib võrguliideste kohta, milledest olulisim on "ip" alamkontekst, kus saab seadistada TCP/IP protokollistikku antud arvutis (käsureaviidaks saab "interface ip>"):
  - ◇ set address [name=]<võrguliidese nimetus> [source=]dhcp – arvuti sätitakse IP-seadistust saama DHCP-serverist.
  - ◇ set address [name=]<võrguliidese nimetus> [source=]static [addr=]<IP-aadress> [mask=]<võrgumask> [gateway=]<võrguvärava IP-aadress või puudumisel "none "> [gwmetric=]<meetrika naturaalarvuna> – seadistab staatiliselt IP-aadressi. Kui võrguväravat millegipärast ei seadistata, siis meetrika parameetrit ei ole vaja.
  - ◇ set dns [name=]<võrguliides nimetus> [source=]static [addr=]<DNS-serveri IP-aadress> [ddns=<"enable" või "disable">] [suffix=<"interface" või "primary">] – seadistab staatilise DNS-serveri aadressi. Kui käsus on "ddns=enable", siis uuendatakse oma IP-aadressi ja hostinime DNS-serveris (kasutatakse DDNS-i). Kui käsus on "suffix=interface", siis registreeritakse DNS-serveris hostinimi ja ühendusepõhine nimi, "suffix=primary" korral ainult hostinimi. Parameetrid "ddns" ja "suffix" ei ole kasutatavad Windows 2000 keskkonnas.
  - ◇ set dns [name=]<võrguliidese nimetus> [source=]dhcp – seadistab arvuti saama DNS-serveri aadressid DHCP-serverist (ei saa kasutada staatiliselt seatud IP-aadressiga).
  - ◇ add address [name=]<võrguliidese nimetus> [addr=]<IP-aadress> [mask=]<võrgumask> [[gateway=]<võrguvärava IP-aadress> [gwmetric=]<võrguvärava meetrika>] – lisab võrguliidesele uue IP-aadressi (võrguliidesel on pärast seda mitu IP-aadressi).
  - ◇ add dns [name=]<võrguliidese nimetus> [addr=]<DNS-serveri aadress> [index=<naturaalarv>] – lisab DNS-serveri arvuti Windowsi kasutatavate DNS-serverite nimistusse. Kasutamise järjekorra saab täpsemalt seadistada parameetriga "index".
  - ◇ delete address [name=]<võrguliidese nimetus> [addr=]<IP-aadress> [[gateway=]<võrguvärava IP-aadress või "all" (kõikide kustutamiseks)>] – kustutab soovitud IP-aadressi (ainukest allesjäänut ei kustutata).
  - ◇ delete dns [name=]<võrguliidese nimetus> [addr=]<DNS-serveri IP-aadress või kõikide kustutamiseks "all"> – kustutab määratud või kõik seadistatud DNS-serverite IP-aadressid.
  - ◇ show address [[name=]<võrguliidese nimetus>] – kuvab kõikide võrguliideste IP-aadressid. Kui võrguliides on ette antud, siis kuvatakse ainult vastava võrguliidese aadressid.
  - ◇ show config [[name=]<võrguliidese nimetus>] – kuvab võrguliideste või määratud liidese kohta seadistusinformatsiooni (põhjalikum kui "show address").
  - ◇ delete arpcache [[name=]<võrguliidese nimetus>] – kustutab kõigi või määratud liideste ARP-tabeli kirjed (ka staatilised). Analoogne käsule "arp -d *".
  - ◇ dump – kuvab praeguse seadistuse käskudena (võimalik skriptifaili panna). Skriptifail või käsu väljund on võimalik suunata faili, kui kirjutada käsu järele ">> ", ning failinimi määrata näiteks käsuga "netsh interface ip dump > c:\temp\ip_seadistus.txt". Antud faili on võimalik käivitada käsuga "exec", millega saab seadistust kiiresti muuta (kasulik, kui vaja seadistada käsitsi mitut masinat).

- ras – sisenetakse RAS (*remote access servers*) konteksti, mis on kaugühenduste jaoks (näiteks PPP seadistamine). Näiteks PPP tegevuste logimist saab sisse lülitada käsuga "set tracing PPP enable" (salvestatake faili "%SystemRoot%\tracing\ppp.log").
- routing – sisenetakse marsruutimise alamkonteksti. Põhiline alamkontekst on "ip", kus seadistatakse TCP/IP protokollistiku marsruutimisseadeid, sealhulgas marsruutimisprotokolle (nt. RIP, OSPF), NAT-i, DHCP-serverit, DHCP-releesid, DNS-vahendajat, IGMP-marsruuterit/vahendajat ja marsruuterite avastamise protokolli.
- popd, pushd – võimaldavad käsurea kontekste muuta magasini ehk LIFO (*last in, first out*) põhimõttel. Kasutatavad eelkõige skriptimisel. Näitekäsud (koos käsureaviidaga): "netsh>pushd"; "netsh>interface"; "ip interface ip>popd"; "netsh>".

Käsku "netsh" saab kasutada ka mitteinteraktiivselt. Näiteks IP-aadressi muutmiseks võib kasutada käsku "netsh interface ip set address "Local Area Connection" static 192.168.1.10 255.255.255.0 192.168.1.1 1". Netsh käsureal võib käskude võtmesõnad jätta poolikuks, mille netsh ise automaatselt lõpetab (sama algusega käskude korral teeb ise valiku, mis aga ei pruugi olla kasutaja poolt soovitud). Võrguliideste nimetused saab näiteks käsuga "netsh interface ip dump".

Mõningate käskude juures (nt. IP-aadressi muutmisel) on oluline, et oleks käivitatud teenus "Remote Registry Service" (saab käivitada käsuga "net start RemoteRegistry"). Samuti on mõningate käskude (nt. käsu "interface ip show tcpconn" jaoks) juures vajalik teenuse "Routing and Remote Access" töötamine (saab käivitada käsuga "net start remoteaccess"). Teenused peavad olema käivitatud enne "netsh" käivitamist.

Kus asub ja kellele kuulub?

Võib tekkida huvi teada saada, kus mingi IP-aadress füüsiliselt asub, näiteks info saamiseks serveri turvalogis oleva lähte-IP-aadressi kohta. Selleks saab kasutada ühe moodusena IP-aadresside jaotamise infot. Selleks tuleb leida, millisele RIR-le vastav IP-aadress on edasi delegeeritud. Seda informatsiooni näeb aadressil http://www.iana.org/assignments/ipv4-address-space. Kui RIR on leitud, siis tuleb suunduda vastava RIR-i "whois" teenuse poole läbi veebiliidese:

- AfriNIC – http://www.afrinic.net/cgi-bin/whois
- APNIC – http://wq.apnic.net/apnic-bin/whois.pl
- ARIN – http://www.arin.net/whois/
- LACNIC – http://lacnic.net/cgi-bin/lacnic/whois
- RIPE – http://www.ripe.net/perl/whois/

Päringu vastuse lehelt saab informatsiooni, mis organisatsioonile vastav IP-aadressivõrk on registreeritud ja ka selle orienteeruva paiknemise.

Alternatiivselt saab Linuxi keskkonnas kasutada käsku "whois", mida võib kasutada kujul: "whois [-h <whois-server>] <IP-aadress>", näiteks käsu "whois -h whois.ripe.net 193.40.36.2" abil saame teada, et võrk 193.40.36.0/23 on EENet'i kaudu antud TÜ Matemaatika-informaatikateaduskonnale.

Teine mugav moodus on kasutada spetsiaalsete andmebaaside abi. Üheks selliseks andmebaasiteenuse pakkujaks on MaxMind, mille otsivorm asub aadressil: http://www.maxmind.com/app/locate_ip.

Olemas on ka mitmeid programme, mis näitavad sihtpunkti ja vahepealsete marsruuterite geograafilise paiknemise, kuid üldjuhul on nad tasulised (nt. VisualRoute, NeoTrace).

Domeeni ".ee" alla olevate teise taseme domeenide registreerimiste kohta saab informatsiooni EENet'i whois-serverist whois.eenet.ee, mille veebiversioon asub aadressil http://www.eenet.ee/EENet/ee-whois. Näiteks käsuga "whois -h whois.eenet.ee ut.ee" (veebist päringuga "ut.ee") saab teada, mis asutusele on vastav domeen registreeritud ja domeeni kontaktisiku kontaktandmed.

Domeenide ".com", ".net" ja ".edu" all olevate teise taseme domeenide registreerimiste kohta saab informatsiooni VeriSign whois-serveri abil, mille veebiliides on kättesaadav aadressil http://registrar.verisign-grs.com/whois.

IPv6 käsud

Windowsi keskkonnas on IPv6 käsud eraldi käskudena. IPv6 käske:

- ipv6 – käsk, millega saab vaadata ja muuta IPv6 protokolli seadeid. Valikuid:
  - ◇ if [<liidese indeks>] – kuvatakse liideste info. Kui liides on ette antud, siis kuvatakse ainult vastava liidese seadistus.
  - ◇ nc – kuvatakse vahemälu naabritest (liides, IPv6 aadress, kanalikihi aadress, kättesaadavuse olek).
  - ◇ rc – marsruutimise vahemälu.
- ping6 – IPv6 analoog IPv4 käsule "ping".
- tracert6 – IPv6 analoog IPv4 käsule "tracert".
- ipsec6 – võimaldab vaadata ja muuta IPseci seadistust.

19.2 Wireshark

Võrguliikluse pealtkuulamine on võrguspetsialistile väga oluline abivahend võrguliiklusest parema ülevaate saamiseks, võrguliikluse kontrollimiseks ja probleemide lahendamiseks.

Vabavaralistest programmidest üks kasutatavamaid võrguliikluse pealtkuulamise ja analüüsimise tarkvarasid on Wireshark, endise nimega Ethereal (nime vahetati kaubamärgi probleemide tõttu). Wiresharki koduleht asub aadressil http://www.wireshark.org. Wireshark töötab paljudel operatsioonisüsteemidel, kaasaarvatud Windows (uuemad Wireshark versioonid alates Windows 2000-st), Linux, BSD, Mac OS X.

Wireshark võimaldab järgmist:

- Paljude protokollide (sadades) analüüsimise toetus.
- Jooksev hõivamise ja hilisem analüüsimise võimalus.
- Graafilise kasutajaliidese ehk GUI (*graphical user interface*) poolt pakutavad mugavused: kolm eraldi vaadet paanidena, sorteerimised, paketiloendis värvide kasutamine jpm.
- Filtrite kasutamise võimalus.
- VoIP analüüsi vahendid.
- Mitmete hõivefailiformaatide toetus (avamine ja salvestamine).
- Dekrüpteerimise toetus mitmetele protokollidele, näiteks IPsec, ISAKMP, Kerberos, SNMPv3, SSL/TLS, WEP ja WPA/WPA2.
- Väljundi eksportimise toetus XML, PostScript, CSV ja lihtteksti.

Wireshark installeerimisel Windowsi keskkonda paigaldatakse ka WinPcap teek (Windowsi versioon teegist libpcap). Kui on soovi võimaldada Wiresharkil hõivata ehk pealt kuulata võrguliiklust ka tavakasutaja õigustes, siis installeerimise käigus tuleb panna linnuke kasti "Start WinPcap service "NPF" at startup". Sellega registreeritakse teenus nimega "NPF" (ehk WinPcap), mis käivitatakse automaatselt arvuti algkäivitamisega (teenus töötab administraatori õigustes) ja tavakasutajaõigustes Wireshark saab tänu "NPF" teenusele hõivata võrguliidese liiklust (tavakasutajal ei ole selleks vastavaid õigusi).

Wiresharkiga võrguliikluse pealtkuulamiseks tuleb valida menüüst Capture → "Interfaces...". Ilmuvas aknas saab valida liidest, mida soovitakse pealt kuulata. Vajadusel saab liidese kohta vaadata detaile, vajutades nupule "Details". Kohe hõivamisega alustamiseks tuleb vajutada nuppu "Start". Hõive valikute muutmiseks võib vajutada nupule "Options". Avanevas aknas tiitelribaga "Wireshark: Capture Options" tuleb panna linnuke kasti "Capture packets in promiscuous mode", mis käivitab valitud liidese *promiscuous* laadis. *Promiscuous* laadis olev liides edastab võrgukihti ka kaadreid, mis ei ole mõeldud antud arvutile (kanalikihi aadressi alusel). Näiteks hubide kasutamisel saadetakse kaadrid edasi kõikidele masinatele vastavas kolli-

sioonidomeenis, samuti ka Wi-Fi puhul saab kõikide teiste kaadreid hõivata *promiscuous* laadis. Arvuti ei tööta vaikimisi *promiscuous* laadis, mille puhul kanalikihiaadressi põhjal temale mitte mõeldud kaadrid visatakse minema, koormates vähem masina keskset protsessorit. Tekstikasti "Capture Filter:" saab kirjutada filtri, kui ei soovita kogu läbivat liiklust hõivata (hõivefiltri süntaks ei ole samasugune analüüsimisfiltri süntaksiga). Kasti "File:" failinime kirjutades, salvestatakse hõivatu vastavasse faili, mida saab hiljem Wiresharkiga avada ja analüüsida. Soovi korral saab ka määratud aja või failimahu täissaamisel hakata salvestama hõivatud liiklust uude faili (kasulik, kui tegeletakse suurte andmemahtudega).

Peaaknas on kolm paani, mis asetsevad algselt üksteise peal, seda on võimalik seadetes muuta (Edit → Preferences... → "User Interface" → Layout). Edaspidi lähtume algseadistusest. Ülemine paan on hõivatud liikluse kaadrite kirjed, keskmine paan sisaldab ülemises paanis valitud kirje protokollide analüüsi ning viimane paan on HEX-vaataja valitud kirje kohta (keskmises paanis mingile analüüsitud väljale minnes tõstetakse esile vastavad baidid HEX-vaatajas).

Filtrite rakendamine

Wireshark võimaldab mugavalt ja paindlikult rakendada hõivatud liiklusele filtreid, mida saab kasutada huvipakkuvate kirjete kuvamiseks. Filtrid kirjutatakse üleval paremal olevasse lahtrisse "Filter:".

Filtrites kasutatakse järgmisi operatsioone (mõningal juhul ka tähelised alternatiiv-variandid, mis on toodud sulgudes):

- x – filtri läbivad kirjed, mis sisaldavad protokolli "x".
- x.y – filtri läbivad kirjed, mis sisaldavad protokolli "x" välja "y".
- !x (not x) – filtri läbivad kirjed, mis ei sisalda protokolli "x".
- x&&y (x and y) – tingimuste konjunktsioon. Filtri läbimiseks peavad mõlemad tingimused olema täidetud.
- x||y (x or y) – tingimuste disjunktsioon. Filtri läbimiseks peab vähemalt üks tingimus olema täidetud (võivad olla täidetud ka mõlemad tingimused).
- x==y (x eq y) – peab kehtima võrdus filtri läbimiseks.
- x!=y (x ne y) – peab kehtima mittevõrdus filtri läbimiseks.
- x>y (x gt y), x<y (x lt y), x>=y (x ge y), x<=y (x le y) – peab kehtima võrratus filtri läbimiseks.
- x contains "midagi" – protokoll "x" peab sisaldama stringiliselt sõne "midagi".
- x matches "regulaaravaldis" – saab kasutada Perliga ühilduvaid regulaaravaldisi ehk PCRE (*Perl-compatible regular expression*) protokolli või mingi välja osas.
- x.y[m:n] – võimaldab välja võtta x.y alamsõne kohast m pikkusega n. Näiteks x.y[0:5] võtab x.y välja viis esimest märki. Siin on võimalikud ka alternatiivid:
  - ◇ x.y[m] – samaväärne kujuga "x.y[m,1]" (võetakse positsioonil m asuv üks märk).
  - ◇ x.y[:n] – samaväärne kujuga "x.y[0,n]" (võetakse n esimest märki).
  - ◇ x.y[m:] – saadakse sõne alates positsioonist m kuni välja x.y lõpuni.
  - ◇ x.y[m-n] – saadakse sõne alates positsioonist m kuni positsioonini n.
  - ◇ x.y[-m:n] – saadakse sõne, mille algpositsioon on tagantpoolt -m märki ja pikkusega n märki.
  - ◇ x.y[-m:] – saadakse sõne, mille algpositsioon on tagantpoolt -m märki kuni sõne lõpuni.
- x & y – kasutatakse loogilist (bitiviisilist) korrutamist. Näiteks "tcp.flags & 0x08" annab kõik kirjed, kus on püsti "PSH" lipp (samaväärne filtriga "tcp.flags.push == 1").
- upper(x.y) – funktsioon, mis teisendab sõne suurtähtedeks. Kasutatav, kui soovitakse teostada võrdlust ilma tõstutundlikkuseta. Näiteks näidisfilter: upper (ncp.nds_stream_name) contains "MACRO".
- lower(x.y) – funktsioon, mis teisendab sõne väiketähtedeks.

Erinevatel loogilistel tehetel on erinevad prioriteedid, mis tähtsuse järjekorras on "!", "&&" ja "||". Tehetejärjekorra muutmiseks on kasutatavad sulud. Näiteks avaldis "x || y && z || t" on ekvivalentne avaldisega "x || (y && z) || t". Soovides filtreerida välja kirjed, kus on täidetud tingimus x või y ning ka tingimus z või t, siis tuleb kirjutada "(x || y) && (z || t)". Sõnede kirjutamiseks kasutatakse jutumärke. Kui sõnes on vaja kasutada jutumärke või kaldkriipsu "\", siis peavad nad olema varjestatud, kirjutades nende ette kaldkriipsu ("\"). Näiteks sõne 'valem. "uit"\"ohutus" ' varjestatud kuju on 'valem. \"uit\"\\\"ohutus\" '. Kui kirjutatakse mingi avaldis, näiteks x.y="midagi", siis kasutatakse automaatselt filtrit, mis filtreerib välja kõik kirjed, milles ei eksisteerinud protokolli "x" ja tema välja "y".

Wiresharkis on võimalik ette anda numbreid nii kaheksand-, kümnend- kui kuueteistkümnendsüsteemis. Kümnendsüsteemis oleva arvu kirjutame nagu tavaliselt, kaheksandsüsteemiarvule lisame ette numbri "0" ja kuueteistkümnendsüsteemiarvule "0x". Näiteks kümnendsüsteemiarvu 12 puhul oleksid esitused järgmised: 014, 12, 0xC. Kontrollides protokolli mingi lipu püsti- või maasolekut, tähistatakse püsti olekut numbritega "1" ja maasolekut "0" (näiteks "x.y == 1" korral filtreeritakse välja need kirjed, kus protokolli x lipp y on püsti).

Filtrite hulgas on ka metaprotokolle ja -väljasid. Näiteks "frame" tähistab suvalist kaadrit. Filtris "tcp.analysis.duplicate_ack" on "analysis" metatähendusega, mis tähistab, et tegemist on TCP protokolli duplikaat-kinnitussegmendiga.

Filtrite kirjutamisel peab arvestama tähtede tõstutundlikkust. Mittetäheliste võrdlus- ja loogiliste tehete vahele ei pea jätma tühikut. Samuti võib kasutada sulgusid ka seal, kus nad ei ole otseselt vajalikud (näiteks "!(arp)" on ekvivalentne filtriga "!arp"). Filtrite nimetuste abiinformatsioon on kättesaadav aadressidel http://www.wireshark.org/docs/dfref ja http://www.wireshark.org/docs/man-pages/wireshark-filter.html.

Filtrite rakendamisel ei eemaldata kirjeid. Kui kasutatakse uut filtrit, siis rakendatakse teda ka eelmise filtri poolt mittekuvatud kirjetele ja filtri kustutamisel kuvatakse jälle kõik hõivatud kirjed.

Filtrite näiteid

Toome järgnevalt välja mõningaid potentsiaalselt kasutatavaid filtreid, mida saaks koheselt rakendada ja mida kirjeldame lühidalt.

Väljundis on kindlasti mitmeid ARP-protokolli pakette, mille välja filtreerimiseks saame kasutada filtrit "!arp". Soovides näha ainult ARP-protokolli pakette, kasutame filtrit "arp". Soovides samaaegselt ka mitte kuvada DHCP protokolli pakette, võime kasutada filtrit "!arp && ! bootp".

Soovides näha suhtlust ainult valitud IP-aadressiga, võime kasutada filtrit: "ip.dst == 193.40.5.130 || ip.src == 193.40.5.130" (me tahame nii väljuvaid kui ka sisenevaid pakettide kirjeid välja filtreerida). Alternatiivselt võime kasutada filtrit "ip.addr == 193.40.5.130", kus väli "addr" on metaväli, mis tähistab lähte- või sihtaadressi.

Kirjed, mille siht-MAC-aadress on "ff:ff:ff:ff:ff:ff" (ehk leviedastusaadressiga Etherneti kaadrid): "eth.dst == ff:ff:ff:ff:ff:ff". Selleks, et filtreerida OUI prefiksi alusel välja näiteks Inteli võrgukaartide poolt edastatud kaadreid, saab kasutada filtrit "eth.src[0:3] == 00:19:D2".

Kirjed, mille HTTP päis sisaldab sõnet ".ut.ee": 'http contains ".ut.ee"'.

Kirjed, mille andmeosa baidid on kuuteistkümnendkujul "b8:73:35:d1:76:d2:f2:8c:35:ab:44:db" (mingid binaarandmed): "data[0:] == b8:73:35:d1:76:d2:f2:8c:35:ab:44:db". Soovides kirjeid, mille FTP käsuks on "NOOP" (NOOP (*no operation*) käsku kasutatakse tavaliselt FTP-ühenduse avatuna hoidmiseks, kui pikka aega mingeid muid korraldusi ei kasutata), saab kasutada filtrit: 'ftp.response.arg[0:4] == "NOOP"'.

Kirjed, mille kaadri pikkus on vähemalt tuhat baiti: "frame.len >= 1000".

Kirjed, mille IP-paketi sihtaadress jääb vahemikku 80.0.0.0 kuni 100.0.0.0: "ip.dst >= 80.0.0.0 && ip.dst <= 100.0.0.0". Võrku kuulumist on võimalik filtreerida ka maski järgi. Näi-

teks soovides filtreerida kõik kirjed, mis on saadetud sisevõrgu arvutitele: "ip.addr == 192.168.0.0/16". IP-aadressi asemel võib kasutada ka domeeninime, mis tõlgitakse automaatselt IP-aadresssiks.

Soovides filtreerida ainult neid kirjeid, mis olid TCP ühenduse algatamise kõige esimesed segmendid, võime kasutada filtrit "tcp.flags.syn == 1 && tcp.flags.ack == 0" (teatavasti ühenduse loomise esimene segment on ainukene, millel on püsti SYN ja maas ACK lipp). Soovides näha ainult neid ühenduse loomise segmente, mis on loodud mingisse konkreetsesse võrku (näiteks võrku 193.40.4.0 maskiga /22 ehk 255.255.252.0), võime kasutada filtrit "tcp.flags.syn == 1 && tcp.flags.ack == 0 && ip.dst == 193.40.5.130/22" (me ei pea kasutama võrguaadressi, piisab suvalisest IP-aadressist).

Kirjed, mille HTTP protokolli tulemuskood on 404 (tähendab, et vastavat lehekülge ei leitud): "http.response.code == 404". Tavaliselt on veebilehtede tulemuskoodiks "200", mis tähistab, et kõik oli korras, ja kood "304", mis tähendab et lehekülg ei ole vahepeal muutunud. Soovides kuvada kõik teised vastused (õigemini vastuste esimesed segmendid), võime kasutada filtrit "http.response.code != 200 && http.response.code != 304". Kuvamaks HTTP POST tüüpi päringud, saab kasutada filtrit 'http.request.method == "POST"' (samaväärne on ka filter "http.request.method == 50:4f:53:54", kus "POST" on esitatud baitidena kuueteistkümnendarvudes).

Soovides eemaldada kirjed, milleks on suhtlus veebiserveriga (nii päringud kui ka vastused), saame kasutada filtrit "tcp.srcport!=80 && tcp.dstport!=80". Kasutada on võimalik ka metafiltrit "tcp.port", millega annab filtrit lühemalt kirjutada, kuid mille kasutamine võib tekitada segadust. Nimelt kasutades filtrit "tcp.port!=80", ei ole tulemus ekvivalentne eelmisega, sest antud juhul kontrollitakse, kas "tcp.srcport!=80" või "tcp.dstport!=80". Selle filtriga eemaldatakse ainult need kirjed, kus nii TCP lähte- ja sihtport on 80. Õige oleks kirjutada "!tcp.port==80".

Wireshark pakub ka võimalust graafilise kasutajaliidese abil filtreid koostada. Selleks võib kasutada lahtri "Filter:" järel olevat nuppu "Expressions...". Avanevas aknas tuleb esimesena valida valikupuust "Field name" protokoll ja vajadusel vastava protokolli väli. Edasi tuleb valida teostatav tehe valikute "Relation" hulgast. Kui valiti mingi võrdlustehe, siis tuleb järgmisena kirjutada lahtrisse "Value (protocol)" väärtus, millega tahetakse teostada võrdlemist. Mõningatel juhtudel on võimalik kasutada eeldefineeritud väärtusi, mis tuuakse ära kastis "Predefined values:". Näiteks "wlan.fc.type" (vaata "IEEE 802.11" alt) korral pakutakse valikuteks "Management frame", "Control frame" ja "Data frame" (käsitlesime peatükis "IEEE 802.11", lk. 119), millele vajutades väärtustatakse tekstikast "Value (protocol)" vastava tüübi numbriga. Stringide ja protokollide puhul on võimalik võrrelda ka valitud alamsõne või määratud baite väärtusega kastis "Range (offset:length)" (kujul "<algus>:<pikkus>").

Valides menüüst Analyze →"Display Filters", avatakse aken, kus on juba mõningad filtrid olemas ning neid on võimalik ka ise juurde defineerida (vajutades nupule "New"). Defineeritud filtrid salvestatakse ja neid on võimalik kunagi hiljem kasutada või olemasolevat täiendada, mis on kasulik pikemate ja keerulisemate filtrite korral.

Vaata informatsiooni filtrite kohta ka leheküljel http://www.wireshark.org/docs/man-pages/wireshark-filter.html.

Filtrite rakendamine hõivamisel

Wireshark võimaldab rakendada filtreid juba hõivamisel, et välja selekteerida huvipakkuv liiklus ning mitte koormata arvutit ebahuvipakkuvaga. Hõivefiltrid tuleb anda libpcap filtrikeeles. Filtrite põhikuju on "[not] <tingimus> [<"and" või "or"> [not] <tingimus> [<jne.>]]".

Soovides eemaldada ainult IP protokolli mittesisaldavaid kirjeid, saame kasutada filtrit "ip". Soovides hõivata liiklust ainult määratud IP-hostiga, saame kasutada filtrit "host 193.40.5.130". Hõivamaks liiklust ainult hostiga math.ut.ee, kuid mitte soovides suhtlust HTTP- ja SSH-serveriga, on kasutatav filter "host math.ut.ee and not (port 80 or port 22)". Kirjutades märksõna "port" ette "src" või "dst", saab spetsifitseerida vastavalt lähte või sihtporti. Portide vahemiku jaoks võime kasutada märksõna filtrit "portrange <algus>-<lõpp>".

Vaata lisainformatsiooni hõivefiltrite koostamiseks aadressidelt http://www.tcpdump.org/tcpdump_man.html ja http://wiki.wireshark.org/CaptureFilters.

Muid kasulikke seadistusi ja funktsioone

Wireshark kuvab filtripõhiselt erinevat värvi kirjeid ülemises paanis. Värvide kasutamine annab visuaalselt parema ülevaate, aidates leida soovitavat (näiteks saab värvidega eristada päringut ja vastust). Värve on võimalik muuta, avades menüüst View → "Coloring Rules...". Avanevas aknas on filtrid ja neile vastavad kirjete värvused. Filtrid on nimekirjas prioriteetsuse järjekorras. Esimesena kontrollitakse kirje sobivust esimese filtriga nimekirjas. Kui see sobis, siis värvitakse vastav kirje antud värvifiltri kirjes märgitud värvidega. Kui filter ei sobinud, siis võetakse järgmine värvifiltri kirje ja võrreldakse sellega jne. Oluline on arvestada, et spetsiifilisemad reeglid oleksid eespool üldisematest. Kirjutades näiteks nimekirja esimeseks värvifiltrikirjeks filtri sisuga "frame", siis kõik hõivatud kirjed värvitakse selle värvifiltrikirje alusel.

Kiiruse huvides võib eemaldada nimekirjast värvimise reegleid, mida ei rakendata kindlasti antud võrgus või mis ei paku huvi eraldi värvituna. Vaikimisi Wiresharkiga kaasas olev värvimisreegli filter, mille sisuks on "hsrp.state != 8 && hsrp.state != 16", ei leia tavaliselt rakendamist (HSRP (*Hot Standby Router Protocol*) (RFC 2281) on protokoll, mis lubab hostidel kasutada mitut marsruuterit ühe virtuaalse marsruuterina, tõstmaks töökindlust võrguväravaks oleva marsruuteri maha kukkumisel). Enamasti ei ole paljude asutuste võrkudes kasutusel marsruutimisprotokolli OSPF. Samuti võib mõningaid reegleid optimeerida vastavalt kontekstile. Näiteks värvimisreegli filterist, mille sisuks on "cdp.checksum_bad == 1 || edp.checksum_bad == 1 || ip.checksum_bad == 1 || tcp.checksum_bad == 1 || udp.checksum_bad == 1", võib välja visata disjunktsioonielemendid "cdp.checksum_bad == 1" ja "edp.checksum_bad == 1" (CDP (*Cisco Discovery Protocol*) on peamiselt Cisco võrguseadmetes kasutusel olev protokoll naaberseadmete kohta informatsiooni saamiseks, et lihtsustada administreerimist) (EDP (*Extreme Discovery Protocol*) on sarnane protokoll CDP funktsioonidele). Mitte ühegi värvireegli rakendumisel kuvatakse vastav kirje vaikimisi värvidena (must tekst valgel taustal).

Mugav moodus kogu TCP-voo jooksul vahetatud andmete vaatamiseks on kasutada funktsiooni, mille saab kätte ülemises paanis TCP kirje hüpikmenüüst valides "Follow TCP Stream". Avanevas aknas näidatakse kogu vastava TCP-ühenduse jooksul vahetatud andmed, kus ühe otspunkti andmed on punast ja teise andmed sinist värvi. Näiteks on hea antud funktsionaalsust kasutada veebiprogrammeerijatel silumiseks (nt. AJAX-i (*Asynchronous JavaScript and XML*), mida on tavavahenditega raskem teha).

Wireshark pakub erinevat statistikat hõivatu kohta menüüs "Statistics". Üldstatistikat pakutakse järgmistel kujudel:

- Kokkuvõtlik (*summary*) statistika, mis näitab aega, pakette, baite, keskmisi tulemusi jms. Siin näeb ka hõive- ja analüsaatorifiltrite rakendatust.
- Protokollihierarhia (*protocol hierarchy*) statistika näitab kokkuvõtlikku statistikat iga protokolli kohta hierarhiliselt. Statistika käib peaaknas filtreeritud kirjete kohta.
- Sideseansside (*conversations*) statistika näitab sideseansse erinevate kihtide protokollide tasemel (nt. Ethernet, IP, TCP, UDP). Antud aknas mingil kirjel hüpikmenüüd avades pakutakse erinevaid filtreid filtreerimiseks, kirjete otsimiseks ja sideseansside värvimiseks.
- Lõpp-punktide (*endpoints*) statistika on sarnane sideseansside statistikale, kuid erinevus ilmneb väga suurte kirjete töötlemise kiiruses.
- Sisendi-väljundi-diagramm (*IO graph*) näitab liiklust diagrammina, kus on võimalik rakendada erinevaid värve eraldi filtrite jaoks, mis annavad visuaalse ettekujutuse filtreeritavate kasutusaktiivsusest ajaliselt ja mahuliselt.

Peale üldstatistika on menüüs "Statistics" saadaval ka mitmeid protokollispetsiifilisi statistikafunktsioone, millest on paljudele võimalik rakendada ka eelnevat filtreeringut. Näiteks "HTTP" alamenüü alt leiab HTTP protokolli kasutuse kohta: "Load Distributions...", näitab IP-

aadresside kaupa need kohad, kuhu ja kui palju päringuid on teostatud (siin on näha, kui mõnel veebilehel käiakse rohkem või sisaldab see endas rohkesti pildikesi jm. faile); "Packet Counter..." näitab HTTP-päringute vastuste koodide loendureid; "Requests..." näitab tehtud päringute loendureid (näiteks näeb, milliseid faile on kõige enam küsitud). Huvi võiks pakkuda ka menüüelement "Flow Graph...", mis näitab voodiagrammina liiklust erinevate hostide vahel koos lühiinfoga.

Mõnikord võib vaja olla Wiresharki jaoks käsitsi määrata kasutatav dekodeerija. Näiteks vahel on veebiserver töötamas mingil muul pordil kui 80, mistõttu Wireshark ei oska valida sobivat dekodeerijat (rakenduskihi protokollid tehakse kindlaks pordinumbri alusel). Et näidata kirjeid ikkagi dekodeerituna, tuleb vajutada vastava kirje peal ja avada hüpikmenüüst "Decode As...", valides sobiva saki (HTTP puhul "Transport") ja seejärel sobivad parameetrid ning lõpetuseks vajutada "OK" või "Apply" nuppu. Kasutaja defineeritud dekodeerijate kasutamisi saab vaadata ja tühistada, kui valida Analyze → "User Specified Decodes...".

Wiresharki allosas oleval olekuribal on paremal pool tähed ja nende järel numbrid. "P:" tähistab hõivatud pakettide arvu ja "D:" ülemises paanis kuvatavate kirjete arvu (saab vaadata, kui paljud kirjed läbisid rakendatud filtri). "M:" tähistab märgitud kirjete arvu (märgistada saab, kui ülemises paanis kirje peal hüpikmenüüst valida "Mark Packet (toggle)" (sama moodi saab ka märgistust maha võtta)).

Wireshark sisaldab veel mitmeid lisavõimalusi, mis annavad lisafunktsionaalsust või kergendavad tööd: makrod filtrite mugavamaks sisestuseks (View → "Display Filter Macros..."); ekspertinfo, mis toob välja võimalikud tähelepanu väärivad kirjed, mis on selekteeritavad tõsiduse järgi (nt. dubleerivad TCP kinnitussegmendid, taassaatmised, kontrollsumma vigane, pakett ebakorrektne jne.) (Analyze → "Expert Info"; Analyze → "Expert Info Composite"); dekodeerimise reeglite määramine, mis võib vajalik olla, kui näiteks veebiserver töötab mingil muul pordil kui 80.

Nõuandeid ja probleemilahendusi

Mõnede võrgukaartide, eriti Wi-Fi korral, võib tekkida probleem, et sättides Wiresharki võrguliidest pealt kuulama, ei pruugi mitte ühtegi paketti näha. Sellisel juhul tasuks peatada võrguliidese liikluse pealtkuulamine ja valida Capture → "Interfaces..." → soovitava võrguliidese reast "Options". Avanevas aknas tuleb võtta ära linnuke kastist "Capture packets in promiscuous mode".

Uuemate võrgukaartidega arvutitel võib Wiresharki väljundis esineda tihti viga "TCP CHECKSUM INCORRECT", mis tähendab, et TCP poolt arvutatud kontrollsumma on vigane. Põhjustatud on see sellest, et uuemad võrgukaardid võimaldavad kontrollsumma arvutamise jätta võrgukaardi teostada, mitte lasta teha seda tarkvaraliselt. Võrgukaardi draiveri poolt jäetakse kontrollsumma arvutamata. Jättes kontrollsumma arvutamise võrgukaardi teostada, vähendatakse keskse protsessori koormust. Et ignoreerida TCP kontrollsummade viga, saab Wiresharkis keelata TCP kontrollsummade kontrolli. Selleks peab Wiresharkis liikuma Edit → Preferences... → Protocols → TCP. Siit tuleb võtta ära linnuke kastist "Validate the TCP checksum if possible". Samuti võib keelata kontrollsumma arvutamise võrgukaarti delegeerimise (*checksum offloading*) (Windows keskkonnas saab seda muuta, kui avada vastav võrguliides → General →> "Configure..." → Advanced → "Offload Checksum" → Value → Disable), kui see on võimalik (mõistlik teha ajutiselt Wiresharkiga hõivamise ajaks).

Jooksvalt pakette analüüsida on mugavam, kui Wireshark ei kasuta automaatset kerimist. Selleks valida View → "Auto Scroll in Live Capture" (kehtib antud programmi kasutusseansi käigus) või Edit → "Preferences" → "User Interface" → "Capture" → "Automatic scrolling in live capture" (kehtib ka järgmine kord Wiresharki käivitades).

Wireshark võtab TCP puhul automaatselt järjekorranumbri välja algväärtuse relatiivselt nulliks, mis võib alguses tekitada segadust. Minnes "Sequence number" väljale (keskmises paanis), on näha järjekorranumbri tegelik väärtus HEX-vaate aknas.

Mõnikord võib probleemiks olla, et valitakse pealtkuulamiseks vale liides. Hõivamise ajal saab kontrollida valitud liidest, kui vaadata olekuriba vasakule poole, kus on toodud hõivatav võrguliides.

19.3 TCPView

Wiresharkiga liiklust hõivates ei ole võimalik näha, millisele protsessile vastav liiklus on suunatud. Käsuga "netstat" saame teada, missugune avatud sokkel missuguse identifikaatoriga protsessile kuulub. Protsessi identifikaatori abil saame näiteks tegumihalduris vaadata, millise protsessiga on tegemist. Töö lihtsustamiseks on Windowsi keskkonnas ühe programmina kasutatav TCPView, mille saab tõmmata aadressilt http://www.microsoft.com/technet/sysinternals/Networking/TcpView.mspx. Tegemist on vabavaralise programmiga. Käivitades TCPView, kuvatakse nimekiri avatud TCP ja UDP ühenduste lõpp-punktide kirjed, mis sisaldab veergusid: käivitatud programm, protokoll, kohalik aadress, sihtaadress ja ühenduse seisund. Vaikimisi uuendatakse nimekirja iga sekundi tagant, seda on võimalik muuta ja käsitsi värskendada menüüst "View". Uued lisandunud ühendused on rohelise taustaga ja kustutatud ühendused punase taustavärviga. Soovi korral on võimalik loodud ühendusi ja protsesse sulgeda, valides hüpikmenüüst vastavalt "Close connection" või "End Process". Hüpikmenüüst on samuti võimalik eemalasuva otspunkti kohta kasutada "whois" teenust. Vaikimisi lahendatakse IP-aadressid domeenideks ja pordinumbrid asendatakse teenuste nimetustega. Neid valikuid on võimalik peale ja maha keerata menüüst "Options". Hetkeseisu saab salvestada tekstifaili "File" menüüst.

19.4 Võrguprobleemide lahendamine.

Proovime järgnevalt välja tuua tegevused ilmnenud võrguprobleemide lahendamiseks:

1. Probleemi ilmnemisel proovi võimalikult palju konkretiseerida probleemi ja tema olemust, püüdes paralleelselt analüüsida saadud infot.
2. Proovi lokaliseerida probleemi esinemine.
3. Probleemi oletatava allika leidmisel mõtle välja esialgsed lahendussammud.
4. Vajadusel ja võimalusel tee tagavarakoopia olemasolevast süsteemist, juhuks kui probleemi lahendamisel läheb probleem suuremaks või tekivad muud tõsisemad komplikatsioonid, et saaks ennistada teostatud muudatusi.
5. Kõrvalda oletatava probleemi allikas.
6. Testi süsteemi, et veenduda soovitud tulemustes. Kontrolli ka muid süsteemi osasid, et probleemi lahendus ei oleks neid mõjutanud.
7. Dokumenteeri probleem ja tema lahendusskeem, juhuks kui probleem peaks tulevikus uuesti või mujal ilmnema.

Probleemi lokaliseerimisel tasub lähtuda OSI kihtidest, kontrollides esimesena, kas OSI esimeses (ehk füüsilises) kihis on probleeme. Kui füüsiline kiht on korras, siis peaks liikuma järgmise kihi (ehk kanalikihi) juurde. Mõningaid probleemide allikaid kihtide kaupa:

OSI esimene ehk füüsiline kiht:

- Kas kõik vajalikud seadmed on ikka sisse lülitatud ja töötavad tavarežiimis?
- Kaablite ühendatuse korrektsuse kontrollimine. Kaablid ei pruugi olla korrektselt ühendatud (kas otsikud on lükatud lõpuni, korralikult kinni keeratud), nad võivad olla ühendatud valesse porti, võibolla on kasutatud vale kaablit (nt. ristkaabli asemel on kasutatud otsekaablit või vastupidi; kas DCE ja DTE otsad on õigel pool) jne.
- Kaablite töökorras oleku kontrollimine, mida võib võimalusel testida testriga või muude

seadmete vahele ühendades. Samuti võib testida kaabliga, mis teatakse olevat töökorras. Kontrollida tuleb kaabli otsikuid, vajadusel võrrelda samalaadsetega (nt. midagi on kõveraks läinud, korrodeerunud jne.).

- Võrguliidese töökorras oleku kontrollimine. Näiteks on võrguliides ülepinge tõttu rikki läinud (tihti äikese tõttu) või on see lihtsalt praak. Proovida tuleks võrguliidese asendamist samalaadsega või töötava võrguliidese asendamist potentsiaalselt rikkis olevaga.
- Ühendus on katkendlik. Proovida pingida erineva suurusega pakettidega sideliini teist poolt. Kui pakettide suuruse kasvades resultaadid muutuvaks kehvemaks, siis on võimalik kaabli või võrguliidese probleem (signaale ei suudeta tihti korralikult välja lugeda).
- Kas võrguseade või võrguliidese kaart on korrektselt seadistatud? Tuleks uuendada või taasinstalleerida draivereid.
- Kas "auto negotiation" on automaatseks seadistatud (soovitatav) mõlemas liideses?

OSI teine kiht ehk kanalikiht:

- Vigaselt konfigureeritud liides.
- Võrguliidesed on erinevalt konfigureeritud. Näiteks erinev kapseldus või mittekokkusobivad muude parameetrite seadistused.

OSI kolmas kiht ehk võrgukiht:

- IP-aadressi seadistuse ebakorrektsus. Näiteks on tehtud trükiviga IP-aadressi määramisel, IP-võrguskeem on vigaselt disainitud, võrgumask on ebakorrektne, võrguvärav on vale või puudub.
- Kas hosti ja marsruuterite marsruutimistabelid on ikkagi korras? Põhjuseks võivad olla vigased staatilised marsruudid või marsruutimisprotokollide vigane seadistus.

Kõrgemad kihid:

- Kas DNS-lahendaja arvuti funktsioneerib korralikult. Kontrollimiseks võib kasutada vastavalt keskkonnale "traceroute" või "tracert" käsku, mille sihtaadressina kasutada IP-aadressi (võib olla suvaline üksikedastuseks kasutatav IP-aadress).
- Tulemüürist tingitud probleemid.

Testides ühenduse olemasolu "ping" või "traceroute" käsku kasutades, peab arvestama, et ühendus peab toimima mõlemas suunas. Ühes suunas ühenduse toimivust saab kontrollida näiteks pingides masinast A masinat B, kusjuures masinas B on töötamas Wireshark või mingi muu taoline programm. Kindluse mõttes võib pingimisel kasutada ka ise etteantud baidimustrit (Linuxi "ping" käsuga).

Mõnikord juhtub Windowsi keskkonnas, et domeeninimede lahenduseks kasutatav teenus ei funktsioneeri korralikult. Antud juhul on abi, kui teha teenusele "DNS Client" taaskäivitus (käsk on "net restart Dnscache").

Soovitusi arvutivõrgu paigaldamiseks

Arvutivõrku planeerimisel on mitmeid soovitusi, mida peaks jälgima, et lõpptulemus oleks võimalikult efektiivne ja väikeste kuludega. Järgnevalt mõningaid soovitusi, mida oleks kasulik järgida:

- Kaabeldust planeerides paigalda pigem kõrgema kategooria kaabel, sest tavaliselt paigaldatakse kaabel aastateks. Tänapäeval tasuks juba mõelda Cat 6 või Cat 6a kaabli paigaldamisele, eeskätt suuremat läbilaskevõimet vajavate sideliinide osas.
- Väldi võrgukaablite lähestikku (eelkõige paralleelselt) sattumist elektrijuhtmetega (mida tugevam pinge, seda suurem mõju). Elektrijuhtmed kiirgavad elektromagnetlaineid, mis tekitavad võrgukaablis müra. Kaabli paigaldamisel nurkadesse peab arvestama, et kaabel ei tohiks olla järsult murtud, vaid kaarjalt keeratud. Kaabli paigaldamisel ei tohiks kasutada klambripüstolit. Kaabeldust on soovitatav testida ja testitulemused dokumenteerida (et hiljem vajadusel võrrelda paigaldamisjärgsete andmetega).
- Kui kaablit on vaja vedada õuest, siis kasuta spetsiaalse mantliga kaablit, mis on küll kallim, kuid ilmastikukindlam. Tavalise UTP kaabli mantel muutub päikese ultra-

violettkiirguse mõjul rabedaks ja läheb katki.

- Vaskkaablit õue paigutades (naabermajade vahel) arvesta äikeseohuga.
- Võrgu magistraalühenduste jaoks kasuta kiiremaid ühendusi, võimalusel ka dubleerimist.
- Kaabelduse ja kommutaatorite paigaldamisel ei ole alati efektiivne osta üks rohkete portidega kommutaator ja ühendada kõik arvutid jm. seadmed sellega – vajatav kaabli hulk võib minna suureks. Võibolla on mõtekas kasutada mitut kommutaatorit, mille tulemusena kulub vähem kaablit ja haldamine muutub tihti lihtsamaks. Vajalik on arvestada sideliinide ribalaiustega, et kohtvõrgus ei tekitataks pudelikaelu.
- Püüa planeerimisel, haldamisel ja muudatuste tegemisel kinni pidada süstematiseeritusest. Näiteks IP-aadresside jaotus, kaablite ühendamised portidesse, tekstilised tähistused, värvide kasutamine. Süstematiseeritusega paraneb võrguhaldus ja tekib vähem eksimusi ning olemasolevate probleemide lahendamine on efektiivsem.
- Võrku disainides dokumenteeri planeeritu ja hiljem juhindu koostatud dokumendist. Kui realiseerimisel tekib vajadus teisiti toimida, siis täienda ka dokumenti. Koostatud dokument peab olema kasutatav (ka teistele).

20. Võrguturve

Peale selle, et võrk oleks funktsioneeriv, skaleeruv, kiire ja muude heade omadustega, peab arvestama ka turvalisuse aspekti. Turve on võrgutehnoloogias väga oluline teema, sest Interneti kaudu on põhimõtteliselt sekunditega kättesaadav suvaline host maakeral. Järjest enam teenuseid on kättesaadavad Interneti vahendusel (räägitakse ka e-riigist) ning see tõstab järjest enam potentsiaalse kahju suurust ja ohtu. Seetõttu on turvalisuse valdkond muutunud järjest olulisemaks ja eeldatavasti kasvab olulisus seda enam, mida enam sõltutakse arvutitest. Me ei hakka siinkohal vaatlema füüsilist turvalisust ((loodus)õnnetused, vargused jms.), mis on samuti väga oluline osa, vaid käsitleme põgusalt pigem mittefüüsilist turvet.

Võrgutehnoloogia areng on loonud väga palju mugavaid võimalusi, kuid samal ajal ka tekitanud juurde uusi potentsiaalseid nõrkusi. Võrgu turvalisusprobleemide korral võivad sissemurdmised olla raskesti avastatavad või saada teatavaks pikka aega peale intsidendi toimumist. Jälgi ei pruugi palju jääda ning needki võivad olla hägused ja kaudsed. Ründaja võib asuda suvalises maailma nurgas. Kuni 2000. aastate alguseni esinesid peamiselt oma lõbuks tehtavad ründed. Peale seda on kasvanud märgatavalt kasusaamise ja spionaaži eesmärgil tehtavad ründed ja eeldatavasti kasvab nende osakaal veelgi. Tavaelust võiks tuua näite turvalisuse ja mugavuse osalisest vastandlikkusest koduukse luku vajalikkuse näitel. Äärmiselt mugav oleks, kui puuduks vajadus ukselukkude ja signalisatsiooni järele. Kuid turvalisuse huvides oleme sunnitud neid kasutama ja mõnikord oleme ka ise selle ohvrid (nt. võtme kaotus või unustamine).

Täiesti vale on laialt levinud suhtumine, et kes peaks tahtma minu arvutisse sisse tungida. Minu arvutis ei ole mitte midagi olulist varjata, mida keegi võiks omale tahta. Aga võib kohe välja tuua, et viimasel ajal on olnud uudiseid pangakontodelt raha varastamise juhtumitest, mis on saanud teoks ohvri arvutisse spetsiaalse programmi sokutamise tõttu, millega varastatakse märkamatult paroole jm. infot. Kuid pangakontode paroolivarguste eesmärgil tehtavad ründed ei ole ainukesed "täiesti tähtsusetu arvuti" ründamisest materiaalse kasu saamise võimalused.

Mõningaid võrguturbe põhimõisteid

Põhilisteks turvanõueteks loetakse:

- Konfidentsiaalsus. Andmed ei tohi olla nähtavad volitamata osapooltele.
- Käideldavus. Teenus peab olema kasutatav.
- Terviklikkus. Andmed ei tohi olla muudetavad (ka osaliselt välja lõigatud või mingit suvalist osa dubleeritud) volitamata osapoolte poolt.

Ründajad võib üldiselt jagada järgmiselt:

- Häkkerid. Ründavad peamiselt oma lõbuks või enda proovilepanekuks. Üldiselt midagi kurja ei tee. Häkkereid, kes teavitavad turvaaukudest administraatorit, võib nimetada ka eetilisteks häkkeriteks.
- Skriptijuntsud. Mitteprofessionaalsed arvutikasutajad, kes ei tea täpselt, mida nad teevad, vaid kasutavad kellegi kolmanda osapoole kirjutatud programmikesi ja õpetusi.
- Kräkkerid. Pahatahtlikud häkkerid, kes arvutisüsteemide ründamisega teevad kurja (kustutavad andmeid, paigaldavad pahavara), mille eesmärk võib olla ka otsese kasu saamine (tellimustööd, info hankimine jms.). Tihti nimetatakse kräkkereid häkkeriteks, mis rikub häkkerluse mainet.

Ründajad võib jagada välisteks ja sisemisteks. Välised ründajad üritavad väljastpoolt võrku (asutuse võrk) ründeid teostada. Sisemised ründajad on sisevõrgust ründajad. Sisevõrgust ründajad on väliste ründajatega võrreldes palju ohtlikumad, sest neil on mitmeid eeliseid. Enamus ründeid pannakse toime sisemiste ründajate poolt.

Arvutivõrkudes võib põhiliste ohtudena välja tuua:

- Info pealtkuulamine. Ründaja kuulab meie ühendust passiivselt pealt, kuid ei muuda midagi või pole selleks võimeline. Pealtkuulamisega on võimalik varastada ebaturvaliste protokollidega kasutatavaid paroole (näiteks HTTP, FTP, IMAP jne.).
- Vahendusrünne, ka vahemehe-rünnak (*man-in-the-middle attack*). Lisaks osapoolte vahelise ühenduse pealtkuulamisele saab ründaja muuta, lisada ja eemaldada edastatavaid andmeid kahe või enama osapoole vahel. Ründaja võib osapoolte A ja B vahel olles esitleda ennast A-le B-na ja B-le A-na, kontrollides sellega kogu A ja B vahelist liiklust.
- Taasesitusrünne. Ründaja saab teha sama päringut uuesti (kuigi ta ei pruugi saada takistada andmete saatmist ega muudetud andmeid lisaks saata krüpteerimise tõttu).
- Paroolide äraarvamine või erinevate kombinatsioonide kontroll (tihti nn. sõnaraamatu baasil).
- Seansi kaaperdamine. Serveriga suheldes luuakse peale autentimist kasutajale seansivõti või mingi seansiomane asi (nt. identifikaator, järjekorranumber). Kui ründaja selle seansivõtme ära arvab või teada saab, siis on tal võimalik teisest arvutist seanss üle võtta ning teostada soovitud operatsioone.
- Tarkvara vead. Tihti on põhjusteks puhvrite ületäitumised, mis võimaldavad ründajal jooksutada ründaja poolt saadetud programmikoodi, mis võib paigaldada nt. tagaukse.
- Matkimine. Ründaja väidab midagi, mida ta ei ole või milleks tal õigusi ei ole. Näiteks võltsitud lähteaadressiga e-mailid, veebipäringud, võltsitud lähteaadressiga paketid ja kaadrid (näiteks MAC, IP), e-arved (inimeselt petetakse raha välja) jne.
- Teenusetõkestusrünne. Ründaja halvab rünnatava ummistamisega, tehes suure hulga päringuid. Selliselt muutub süsteem mittekäideldavaks ja samuti võivad ülekoormuse tõttu üles öelda serverid, võrguseadmed jms. Alamliigiks on hajus teenusetõkestusrünne, kus paljud erinevad masinad teostavad samaaegselt teenusetõkestusrünnet, millega nende mõju kumuleerub.
- Ussid, viirused jms. Levivad enamasti Interneti ja võrgu teel, kasutades ära turvaauke.
- Tagauksed (*back doors*). Arvutisse on mingil moel paigaldatud programmid, mis avavad ründajale ligipääsu subjekti (nt. arvutisse) ning mille kaudu saab ründaja arvutit omatahtsi kasutada. Tagaukse loomine võib olla teostatud ka mitte võrgu teel või kaudselt (keegi installeeris või kasutaja ise sattus ahvatluse lõksu (näiteks mingit "kasulikku" programmi käivitades, e-maili manuses asuvat programmi avades).
- Loogikapommid. Loogikapomm on programmi kirjutatud kood, mis mingi sündmuse korral teostab eelprogrammeeritud pahatahtliku operatsiooni (näiteks peale vallandamist kustutab andmed vmt.). Loogikapomm võib aktiveeruda näiteks siis, kui ründaja ei saa lükata edasi programmi aktiveerumist (nt. vallandamise tõttu). Seetõttu on kasulik vallandatud kasutajate arvutitele teha puhtale kettale installeerimine (nt. ketta tõmmise abil).
- *Bot*-iks muutmine. Arvutisse murtakse sisse ja paigaldatakse programmike (trooja hobuse eriliik), mis teostab ründaja poolt ettenähtud ülesandeid. Masinat kasutatakse ressursina (võrguribalaius, töötlemisvõimsus, andmehoidla). Arvuti võib saada ka vahejaamaks ehk hüppelauaks rünnakute allika raskemini jälgitavamaks tegemiseks. Kogumikku *bot*-e, mida on võimalik kellegi poolt kontrollida nimetatakse *botnet*-iks. *Botnet*-e kasutades teostatakse tavaliselt suuremaid hajusaid teenusetõkestusründeid, levitatakse rämpskirju, proovitakse rünnata uusi arvuteid (*bot*-iks tegemise eesmärgil).
- Sisevõrgu mõne masina nõrgaks lüliks olemine, mille kaudu on võimalik anda ründajale paremad eeldused teiste masinate ründamiseks.
- Kasutajate teadmatus ja laiskus ning selle ära kasutamine.

Usse, viirusi, tagauksi, loogikapomme jms. programme kutsutakse üldnimetusega pahavara, ka kurivara.

Võrguturbe seisukohalt võib põhiliste probleemidena välja tuua järgmist:

- Üleliigsete teenuste pakkumine. Tihti jookseb serveris või töökohaarvutis teenuseid ja programme, mille poole on võimalik võrgust pöörduda, kuid mida reaalselt ei kasutata. Mida enam teenuseid jookseb, seda suurem on tõenäosus, et mõnes nendest võib olla turvaauk, mida ründaja leiab ja oskab ära kasutada. Üleliigsed teenused kulutavad ka asjatult masina ressursse (mälu, protsessor, võrgu ribalaius, algkäivitamise pikem aeg).
- Teenuste liialt laialt pakkumine. Tihti on mõeldud, et teenust saaks kasutada ainult sisevõrgust või ainult valitud hostid, kuid tegelikult pääsevad ligi ka kõrvalised hostid või koguni kogu Internet.
- Tulemüüri mittekasutamine. Töökohaarvutis on tungivalt soovitatav kasutada tulemüüri (viirusvastane programm on tulemüüri kõrval vähem prioriteetsem).
- Nõrkade paroolide kasutamine. Paroolid peavad olema piisavalt pikad ja keerulised. Ründaja võib kasutada ka sõnaraamatut ja proovida erinevaid sõnu ja nende kombinatsioone. Lisaks võidakse kasutada krüptoanalüüsi toimuva liikluse pealt. Oluline ka parooli räside turvalisus (näiteks MD5 räsid on tänapäeval juba ebaturvalised).
- Tarkvara uuenduste mittepaigaldamine.
- Vabalt ligipääsetavad või nõrga turbega (nt. WEP-i kasutades) traadita kohtvõrgud.
- Muud asjatundmatud seadistamised. Laialt on levinud suhtumine, et kui asi töötab, siis on kõik korras, kuid turvalisusele ei teata või ei viitsita tähelepanu pöörata (kes ikka tahaks või viitsiks teda rünnata).

20.1 Rünnakutest üldiselt

Võrguturbe juures peab arvestama, et ründaja ja ka rünnakuks kasutatavad moodused pole teada. Ei ole võimalik kaitsta kõikide rünnakumustrite vastu. Eeldatakse, et ründajal on rünnaku ettevalmistamiseks tohutult aega.

Tavaliselt kogutakse enne rünnaku alustamist informatsiooni ja kombitakse süsteemi. Kompimise all mõistetakse seda, kui ründaja saadab erinevaid päringuid ja analüüsib vastuseid, saamaks teada, milliseid teenuseid pakutakse, mis tarkvara versiooni kasutatakse, milline on konfiguratsioon jne. Informatsiooni kogumine on ründajale vajalik eduka rünnaku teostamiseks. Samas on kaitsjal võimalik juba eos avastada potentsiaalne rünnakukatse. Selleks, et vähem kahtlust äratada võib informatsiooni kogumine mõnikord olla väga hajutatud (muidugi peab rünnatav kasutama monitooringut ja liiklust analüüsima, teavitades kahtlastest asjadest võrguadministraatorit). Üks informatsiooni kogumise mooduseid, millele tihti ei osata tähelepanu pöörata, on rakendussotsioloogia (*social engineering*). Rakendussotsioloogiat rakendades võidakse saada inimestelt juhuslikult ja näiliselt tühiseid informatsioonikildusid, milledeks võivad olla näiteks andmed käitumisharjumuste, serverite ja võrgutopoloogia kohta. Võimalusel võidakse tuhnida paberites, uurida prügikasti sisu jne. Rakendussotsioloogia võib lihtsustada ründajal mõista potentsiaalset rünnatavat ja anda olulist taustinformatsiooni või koguni midagi konkreetsemat, mida saab ära kasutada.

Uusi ohvermasinaid otsivad ründajad ei teosta seda tavaliselt käsitsi, vaid kasutavad nende arsenalis olevaid automaatseid programme, mis ise otsivad potentsiaalseid ohvreid, kombivad neid ja rakendavad erinevaid ründeid (tuntakse *auto-rooter* nime all). Enamasti on eesmärgiks istutada tagaukseprogramm (*back door*), tihti *rootkit* omadustega. *Rootkit*-programmid peidavad ennast süsteemi võimalikult hästi nii, et neid ei ole näha protsesside loendis. Nende faile ei ole samuti näha ning kasutajal on väga keeruline süsteemis olevast *rootkit*'st teada saada.

Ründaja sihtmärkide järgi võib ründamise jagada kaheks. Esimesel juhul võib eesmärgiks olla rünnata mingit konkreetset asutust, serverit, teenust. Antud rünnete eesmärgiks võivad olla näiteks väljapressimised (katuse pakkumine), spioneerimine või mingite muude eesmärkide saavutamine. Teisel juhul ründaja otsib hoste, mis on lihtsamini rünnatavad, sisaldades näiteks mingit kindlat lappimata turvaauku. Ründaja eesmärgiks on tavaliselt võimalikult paljude masi-

nate saamine oma kontrolli alla, et kasutada nende ressursse (suurendada ründaja *botnet*'i) või varastada juhuslike inimeste tähtsaid andmeid (nt. internetipangast raha varastamiseks).

Ründaja, kelle eesmärgiks on konkreetse subjekti ründamine, esimeseks tegevuseks on üldjuhul hosti või hostide kompimine, et leida nõrkusi. Näiteks võidakse hakata kasutama pordiskaneerijat, leidmaks avatud porte. TCP-l baseeruvate teenuste korral proovitakse luua TCP-seanss, läbides kolmeosalise käepigistuse faasi, mille järel katkestab ründaja ühenduse. Kui antud pordi peal ei ole ühtegi teenust, siis server ei alusta TCP-seansi loomist. UDP portide sondeerimisel oodatakse ebaõnnestumisel ICMP teadet "port kättesaamatu" (*port unreachable*). Kuna vastavat ICMP-teadet ei pruugita saata või lasta läbi tulemüüri poolt, siis võidakse pöörduda pordi poole konkreetse protokolli teatega (nt. marsruutimisprotokolli RIP-i teatega). Ründaja töö lihtsustub, kui teenused asuvad vaikeportidel.

Kohtvõrgus olles on võimalik teiste masinate olemasolu kindlaks teha, kasutades ARP-tabelit. Pöördudes vastava sisevõrgu IP-aadressi poole (näiteks pingides), ei pruugi meie päring õnnestuda. Tavaliselt ARP-päringutele siiski vastatakse ja ARP-kirje ikkagi reedab vastava hosti elusoleku.

Rünnates suvalisi hoste, on vaja ründajal alguses need hostid üles leida. Selleks on tal mitmeid mooduseid: pöörduda suvalise IP-aadressi poole, kasutada DNS- ja whois-serverist saadavat infot.

20.2 Võrgu pealtkuulamine

Võrgu pealtkuulamine on passiivne rünne, mida kasutatakse informatsiooni kogumiseks. Näiteks on võimalik näha teiste inimeste omavahelist suhtlust, paroole, uurida, millega nad tegelevad, mis hostidega suhtlevad ja millise intensiivsusega. Seda informatsiooni saab hiljem ära kasutada. Pealtkuulamist võidakse kasutada otseste rünnakute teostamiseks. Näiteks on pealtkuulamise vastu kaitsetud protokollidest FTP, Telnet, HTTP, SNMPv1. Võrguliikluse kuulamiseks ja analüüsiks on olemas mitmesugust erinevat tarkvara, millest vaatlesime Wireshark nimelist programmi.

Võrgu pealtkuulamine on kõige lihtsam jagatud meediumi puhul, sest kõik näevad toimuvat liiklust. Masinate võrguliidesed viskavad tavarežiimis ära kaadrid, mis ei olnud adresseeritud neile. Valdaval osal võrgukaartidest on võimalik kasutada vahettegematut režiimi (*promiscuous mode*) (vt. lk. 171). Sellist tegevust nimetatakse *sniffing* ja vastavaid arvutiprogramme *sniffer*'teks.

Kaitseks pealtkuulamiste vastu on kasutada protokolle, mis tarvitavad krüpteerimist. Näiteks FTP asemel SFTP; Telnet asemel SSH; IMAP, POP3 ja SMTP juures krüpteeritud versiooni, HTTP asemel HTTPS.

Üle võrgu tuleva liikluse suhtes ei tohi midagi eeldada, ka mitte madalama taseme protokollide (näiteks Ethernet, IP, TCP, UDP jne.) osas. Kõiki nende protokollide väljasid on võimalik võltsida, sõltudes siiski ründaja või hõivatud masinate asukohast, näiteks kui ründaja ei kontrolli mitte ühtegi masinat kohtvõrgus, siis ta ei saa kanalikihi protokolli välju võltsida. Tihti kasutatakse lähte-IP-aadressi võltsimist (*IP spoofing*). IP võltsimise korral ründaja ei ootagi vastust. Näiteks on ründajal võimalik selliselt teha liikluse taasesitust, millega mittearvestamine võib olla ära kasutatav (võimaldab operatsioone pimesi korrata).

Kommutaatori vastane rünne

Üks tuntumaid ründeid kommutaatori vastu on MAC-tabeli üleujutamine, mille tulemusena hakkab kommutaator käituma sisuliselt hubina, saates kaadrid kõikidesse portidesse, sest ta ei leia MAC-tabelist sihtmasina MAC-aadressi. MAC-tabeli üleujutamiseks saadab ründaja suures koguses kaadreid, millel on erinevad lähte-MAC-aadressid. Kommutaator õpib vahendatud võltskaadri lähte-aadressi järgi, et vastavas pordis on mingi uus masin, mille MAC-aadressi ta lisab MAC-tabelisse. Kui MAC-tabel saab täis, siis ei saa enam uusi kirjeid lisada, kuni olemas-

olevad aeguvad. Ründaja jätkab võltsitud lähteaadressiga kaadrite saatmist, et hoida MAC-tabel üleujutatud seisundis. Üleujutatud seisundis saab ründaja kuulata pealt kommutaatorit läbivat liiklust, kasutades näiteks Wiresharki.

Kaitseks võidakse piirata pordis lubatavate MAC-aadresside arvu, mille ületamisel võidakse näiteks vastav port lukustada (kommutaator peab seda võimaldama).

Kommutaatorit läbiva liikluse osaliseks pealtkuulamiseks või halvamiseks on võimalik kasutada ka MAC-aadressi dubleerimise rünnet. Kui ründaja kasutab väljasaadetavate kaadrite lähte-MAC-aadressina kohtvõrgu mingi muu masina MAC-aadressi, siis kommutaator "saab teada", et rünnatav masin on hoopis uue pordi kaudu kättesaadav ning saadab edaspidi kõik rünnatavale mõeldud kaadrid ründajale (rünnatav neid pakette kätte ei saa). Valesse porti saatmine toimub seni, kuni rünnatav ise saadab välja kaadri ja kommutaator jällegi saab teada, et rünnatav masin asub uues pordis, ja teeb MAC-tabelisse vastavasisulise muudatuse. Kaitse oleneb kommutaatori võimalustest. Näiteks võidakse rakendada probleemsete portide lukustamist, mille saab taasavada võrguadministraator. Ennetavalt võidakse käsitsi või automaatselt teha MAC-aadressid sõltuvaks pordist (esimese kaadri põhjal lisatakse MAC-aadress automaatselt MAC-tabelisse jäädavalt või pikaks ajaks). Kui kommutaator saab varasemaga võrreldes kaadri mingi muu pordi kaudu, siis võidakse näiteks vastav kaader ära visata, port blokeerida või võrguadministraatorile teade saata. Lisaks võidakse piirata ka pordi kohta lubatud MAC-aadresside hulka, mida MAC-tabelisse lisatakse.

20.3 Teenusetõkestusrünne

Teenusetõkestusrünne ehk **DoS** (*denial of service*) on üks enamkasutatavaid ja laia spektriga rünnakuviise, kus proovitakse teenus, server või võrk normaalse töö takistamiseks päringutega üle koormata. Subjekt on seda haavatavam, mida enam ta peab massiliste päringute tingimustes tegema tööd või eraldama ressursse, võrreldes ründajaga. DoS rünnakute vastu on üldiselt raske võidelda, sest teenus (näiteks veebileht) peab olema käideldav ehk kättesaadav. Siiski on võimalik paljude teenusetõkestusrünnete mõju tunduvalt vähendada.

Tihti võivad programmid ekstreemse koormuse tingimustes käituda ettearvamatult, mille põhjusteks võivad olla näiteks:

- Programmi disainimisel ei ole arvestatud ekstreemsete tingimustega. Paljudel serveritel on lisatud juurde võimalus määrata samaaegselt teenindavate ühenduste arv. Suurtel koormustel võivad tekkida näiteks lukustamistega tupikud, mistõttu jääb protsess või lõim igavesti ootama lukkude vabastamist.
- Programmeerimisel on sattunud sisse vigu, mis tavaolukorras ei avaldu ning mida on ka keeruline ja töömahukas testida. Tulemusena võivad programmid kokku joosta või teha mingit lubamatut tegevust, mis võib andmeid kahjustada.
- Serverite, võrguseadmete ja masinate ülekuumenemine. Ülekuumenemise tõttu võivad seadmed rivist välja minna ja vajada asendamist.
- Mälu täissaamine, mille tõttu ei ole võimalik enam mälu hõivata.
- Protsessitabel saab täis, mille tulemusena ei ole võimalik luua uusi protsesse.
- Ülekoormus võib kahjusid põhjustada ka kaudselt, näiteks võib rünnakulogi tõttu kõvaketas täis saada, mille tõttu ei pruugi saada ka muid operatsioone läbi viia (logimine peaks toimuma võimalusel eraldi füüsilisele või loogilisele kettale).

Teenusetõkestusrünnakuid, mida tehakse paljudest erinevatest arvutitest koordineeritult samal ajal sama subjekti vastu, nimetatakse **hajusaks teenusetõkestusründeks** ehk **DDoS**-iks (*distributed DoS*). DDoS-ründega on võimalik teostada massiivsemaid ründeid, sest korraga ründab mitu masinat. Tavaliselt on rünnataval rohkem ressursse kui ühel ründajal, kuid koostöös võib nende jõud rünnatavast üle käia. DDoS-rünnete puhul kasutatakse tavaliselt *botnet*'e.

Tihti tehakse DoS rünnakuid võltsitud lähte-IP-aadressidega. Lähteaadressi võltsimisel on

rünnataval raskem rakendada filtreid ja teada saada ründajaid. Selle vastu aitab, kui rünnaku lähtevõrgu marsruuter kontrolliks lähte-IP-aadressi kuulumist vastavasse võrku. Mittekuulumise korral visatakse pakett minema. Selliselt ei ole võimalik antud võrgupiirkonnast teostada rünnakut võltsitud lähte-IP-aadressiga.

Kõige sagedamini korraldatakse teenusetõkestusründeid veebiserverite vastu. Veebiserverite vastaseid ründeid on osaliselt aidanud pehmendada vahendajaserverid. Teenusetõkestusrünnakute õnnestumisele "aitab suuresti kaasa" halvasti kirjutatud tarkvara. Tihti peetakse oluliseks võimalikult paljude funktsioonide olemasolu ja arendamise kiirust. Tarkvara arendamisel on probleemideks näiteks: korralikult analüüsimata arhitektuur, ebaefektiivsed ja mitteskaleeruvad algoritmid, programmeerimisvead, liiga palju kihte ja raamistike ebaefektiivne kasututamine.

Vaatleme järgnevalt lähemalt mitmeid erinevaid teenusetõkestusrünnete liike.

Ping-uputus

Ping-uputusrünne on väga lihtsakoeline ründeviis. Ründavad hostid saadavad rünnatavale hostile massiliselt "kaja päring" pakette, ummistades ühendusteed, mille tõttu tavaliiklus on tugevalt häiritud. Eduka ping-uputusründe jaoks peab ründavatel masinatel olema rohkem ribalaiust kui rünnataval. Kaitseks võib tulemüürides piirata "kaja päring" pakettide hulka sekundis või keelata üldsegi mingiks ajaks "kaja päring" pakettide edastamine. Hostides võidakse keelata üldsegi "kaja päring" pakettidele vastamine.

SYN-uputus

TCP-ühendus luuakse kahe osapoole vahele kolmeosalise käepigistuse abil. SYN-uputuse (*SYN flood*) teenusetõkestusründe korral saadab ründaja hästi palju kolmeosalist käepigistust alustavaid SYN-pakette, millede lähteaadress on tavaliselt võltsitud. Rünnatav masin salvestab saadud järjekorranumbri ja saadab võltsitud aadressil omakorda segmendi, millele jääb tagasisidet ootama. Sellist ühendust nimetatakse poolavatud ühenduseks. Rünnaku põhimõte seisneb selles, et iga selline poolavatud ühenduse loomine rünnatavas masinas võtab ressursse (siiski mingi aja jooksul kuni ühenduse loomise aeg aegub). Mingi aja jooksul võib TCP poolavatud ühenduste limiit täis saada, millega on antud server kättesaamatu ka teiste kasutajate jaoks, näiteks firma kodulehe külastaja jaoks.

Kaitsena saab vähendada ühenduste aegumise aega, suurendada lubatud ühenduste arvu, kasutada mõnda tuvastusprogrammi. SYN-uputuse rünnakust on võimalik aru saada, kui andes käsurealt käsu "netstat -n" on näha väga palju ühendusi SYN-RECEIVED olekuga. Windowsiga masinas saab vastavaid parameetreid muuta registrikataloogis HKLM\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters olevate võtmete "SynAttackProtect", "TcpMaxHalfOpen", "TcpMaxHalfOpenRetried" ja "TcpMaxPortsExhausted" abil.

SYN-uputuse rünnaku vastu on võimalik kasutada SYN-küpsiste tehnikat, et tagada rünnaku ajal potentsiaalselt reaalsete klientide teenindamine. SYN-küpsiste tehnika võetakse kasutusele, kui TCP poolavatud ühenduste hulk jõuab mingi piirini. Olles SYN-küpsise laadi lülitunud, ei loo rünnatav enam poolavatud ühendusi, kuid siiski saadab vastussegmendi, milles on püsti SYN ja ACK lipud ning järjekorranumbri välja bittide väärtuseks paneb:

- 1. kuni 5. bitile: loendur mooduli 32 järgi. Loendurit suurendatakse iga 64 sekundi tagant ühe võrra.
- 6. kuni 8. bitile: kodeeritud maksimaalse segmendi suuruse valik serveri poolt.
- 9. kuni 32. bitile: räsifunktsiooni väärtus, mis arvutatakse serveri IP-aadressi, serveri pordi, kliendi IP-aadressi, kliendi pordi ning loenduri väärtuste põhjal.

Järjekorranumbri juures on oluline, et ründaja ei saaks pimesi arvata (võltsitud lähte-IP-aadressiga) serveri saadetud järjekorranumbrit.

Kui lülitutakse ümber SYN-küpsiste laadile, siis TCP ühenduse käigus kokkulepitavad valikud ei ole kasutatavad, sest esimene ühenduse initsialiseerimise segment visatakse minema. See

ei ole siiski suur probleem, sest see ei takista ühenduse loomist. SYN-küpsiste laadis töötav server on võimeline SYN-uputuse rünnaku tingimustes töötamist jätkama. SYN-küpsiste toetus on lisatud mitmele operatsioonisüsteemile, kuid pole vaikimisi kasutusel.

UDP-uputus

UDP-uputuse (*UDP flood*) puhul proovib ründaja tekitada võimalikult palju liiklust ja tööd rünnatava jaoks. Eelnevalt võidakse proovida leida sobivaid teenuseid rünnatavas masinas, mis tekitaksid võimalikult palju liiklust või peaksid kulutama rünnatavate ressursse (mälu, kõvaketaruum, protsessori aeg, erinevad olekute seisundid jne.). Ründajale on meeltmööda ka asjaolu, kui ohver vastab ICMP teatega (näiteks ei olnud vastavat teenust, masinat), mis tõstab rünnatava masina koormust. Tavaliselt võltsib ründaja paketi lähte-IP-aadressi.

Üks ekstreemsem näide on, kui esimeses rünnatavas masinas jookseb "chargen" teenus, mis saadab tagasi mingi juhusliku jada sümboleid (vaikimisi port 19). Teises masinas jookseb "echo" teenus, mis saadab tagasi iga vastu võetud märgi (vaikimisi port 7). Selliselt saab nende kahe masina vahele tekitada jäävat liiklust. Kui "aidata" uusi ühendusi serverite vahele luua, siis võivad masinad üksteist jäädagi koormama. Ründaja saab anda veelgi hoogu juurde, kuni selleni, et hostide vaheline ribalaius on täielikult hõivatud ja tekitatud masinatele enestele ka asjatut koormust.

Unix-tüüpi masinates tasuks maha keerata "chargen" ja "echo" teenused, kommenteerides välja failis "/etc/inetd.conf" read, mis algavad "chargen" või "echo". Tulemüürides võib rakendada sarnast kaitset, mida ping-uputuse korral.

Smurf-rünne

Asutuse marsruuterid, mis lubavad suunatud leviedastust, on altid smurf-rünnakule (*smurf attack*). Ründaja saadab võltsitud lähte-IP-aadressiga "kaja päring" ICMP teate (ehk proovib pingida) rünnatava võrgu leviedastusaadressile. Seepeale vastavad võrkudes olevad masinad "kaja vastus" ICMP teatega võltsitud IP-aadressile. Mida rohkem on võrgus pingimisele vastavaid masinaid, seda suuremat võrgukoormust põhjustatakse rünnatavas võrgus ning võrgule, mille IP-aadress oli märgitud "kaja päring" paketi lähte-IP-aadressiks. Smurf-rünnaku edasiarendus on *fraggle* rünnak, mis kasutab UDP protokolli, olles UDP-uputuse ründe võimendatud variant, kusjuures võimalik on kasutada ka "chargen" ja "echo" teenuseid koos smurf-rünnakuga.

Smurf-rünnaku vastu kaitseks piisab, kui marsruuterites keelata suunatud leviedastus. Tänapäeval ei ole suunatud leviedastus üldiselt vaikimisi lubatud ning ka hostides on see tihti ignoreeritud.

Jõulupuupaketi rünne

Jõulupuupaketiks nimetatakse suvalise protokolli paketti, kus on võimalikult palju erinevaid lippe püsti ja valikuid kasutusel. Nimetatakse ka kamikaze-paketiks. Taoliste pakettide töötlemiseks kulub marsruuteritel ja lõpp-hostil tavalise paketiga võrreldes rohkem aega. Erinevad süsteemid võivad jõulupuupaketiga käituda erinevalt, võimaldades ründajal saada informatsiooni rünnatava kohta. Kuna jõulupuupaketid ei ole enamuse protokollide puhul normaalne nähtus, siis mõned tulemüürid filtreerivad vastavaid pakette.

Tšornobõlipaketi rünne

Pakett saadetakse läbi võrguvärava mingisse teise võrku, kus lähte-IP-aadressiks on ühe võrgu leviedastusaadress ja siht-IP-aadressiks teise võrgu leviedastusaadress. Eesmärgiks on teenusetõkestusründe võimendamine.

Marslasepaketi rünne

Paketi lähteaadressiks pannakse mittemarsruuditav aadress. Näiteks on sisevõrgus kasutusel 192.168.0.0 võrku kuuluvad aadressid, kuid paketi lähteaadressiks on 172.16.0.1. Antud juhul on 172.16.0.1 nn. Marsilt pärit pakett, mida ei oska marsruuter kuskile edasi saata.

E-maili pomm

Ründaja võib rünnatava mingile konkreetsele e-maili aadressile saata tohutult e-maile. Rünnaku tulemusena võidakse üle koormata ka e-maili servereid. Suurema kahju tekitamiseks võidakse e-maile saata listidesse ning kasutada mitut sihtaadressi. E-maili lähteaadress on samuti üldjuhul võltsitud, mistõttu võivad vastused (vigade, kinnituste kirjad jms.) veelgi ummistust suurendada.

Kaitseks on võetud kasutusele rämpskirjade filterprogrammid (*spam filter*). Samas on võimalik filterprogrammiserverit omakorda üle koormata rämpskirjadega. Üks moodus on saata e-maile, mis sisaldavad manuses pakitud faile. Pakitud failid on äärmiselt homogeense sisuga, et kokkupakitus oleks võimalikult suur. Näiteks pakkides ZIP arhiveerimisega faili, mis sisaldab ainult "x" tähte 6,8 miljonit korda, siis pakitud faili suurus on ainult 6,6 KB. RAR-ga pakitud faili maht on ainult ligi 600 baiti. Rämpsposti filterserveris võib see tekitada probleeme, kui kontrollitakse manuste sisu. Tihti pakitakse fail lahti kuskile kõvakettale ja siis hakatakse faili kontrollima viiruste vastu. Samas võib pakitud failis sisalduv fail olla märksa suurem, näiteks sajad megabaidid või enamgi ja neid kirju võib tulla minutis sadades või enamgi.

Üldisemalt nimetatakse pakitud andmetega ründeid lahtipakkimispommiks, mida on võimalik rakendada ka teisiti kui e-mailidega. Näiteks ka pildiformaatidega PNG ja GIF, mida saab üles panna veebilehele.

Veebilehtedega seotud ründed

Kaudselt on võimalik serverit ja ühendusi asjatult koormata, kui kuskile paljukülastatavatele veebilehtedele saab varjatult panna tõmbama mingeid suuri faile (näiteks suuri pilte). See võib olla võimalik näiteks XSS (*Cross-site scripting*) rünnakut teostades (veebilehele kirjutatakse ründaja poolt näiteks foorumi postitusena JavaScripti kood, mis järgnevate lehe vaatajate jaoks täidetakse JavaScriptina). Samuti võivad olla kasutatavad näiteks erinevad Java-appletid või ActiveX komponendid. Samas pakuvad mõned veebikeskkonnad ka legaalseid mooduseid piltide kuvamiseks mujalt serverist, näiteks foorumipostitustega.

Teenusetõkestusründeid on võimalik teha ka veebiserveritele, kui lihtsalt väga paljud korraga hakkavad "külastama" veebilehte. Paljud veebiportaalide veebilehed on suure mahuga, ulatudes mitmesaja kilobaidini ja mille sisu taasgenereeritakse vähemalt osaliselt iga kord, mis tähendab aga veebiserverile rohket tööd. Ründe efektiivsemaks muutmiseks võivad ründajad teha päringuid, mis nõuavad veebiserverilt rohkem tööd, näiteks: veebikeskkonnasisesed otsingud; postitamispäringud, mis vajavad serveril andmete salvestamist (nt. gallupitele vastamine, foorumipostitamised, kommenteerimine); veasituatsioonide ärakasutamine (enamasti logitakse hulga informatsiooni veasituatsiooni kohta informatsiooni saamiseks). 2007. aasta aprillisündmuste ajal Eestis olid paljud uudisteportaalid väga aeglased või kättesaamatud.

Veebiserverite kaitseks kasutatakse piiratud ühenduste arvu, et veebiserveri koormus ei tõuseks üle kriitilise piiri. Paljud veebiserverid kasutavad genereeritud lehtede osalist või täielikku puhverdamist, mis võimaldab neil rohkem kliente teenindada. Samuti võidakse piirata ligipääsu serverile. Näiteks 2007. aasta aprillisündmuste ajal piirati korduvalt liiklust välismaalt mitmetele veebiserveritele. 2001. aasta 1. septembri terroriakti puhul Maailma Kaubanduskeskusele ja Pentagonile oli CNN-i veebileht ebanormaalselt suure koormuse all (mitte küll tingitud peamiselt teenusetõkestusründest), mille tõttu eemaldati ajutiselt kõik üleliigsed pildid, funktsioonid ning muu kõrvaline sisu.

Mõningaid teisi teenusetõkestusründeid

Mõistmaks rünnakute mooduseid, toome illustreerimiseks lühidalt välja ka mõned tuntumad DoS-ründed, mille "hiilgeajad" on jäänud minevikku lapitud tarkvara ja erinevate meetmete tõttu. Need rünnakud on näidiseks leitud vigadest tarkvara implementatsioonidest.

Surma ping (*ping of death*) rünnakus saadab ründaja fragmenteeritud IP-pakette, kus on modifitseeritud IP-päist, kus kõikide fragmentide andmeid on kokku rohkem, kui lubatud paketi suurus (ehk 2^{16}-1). See võib põhjustada rünnatava hosti kokkujooksmise. Enamus süsteeme teeb nüüdseks lisakontrolli, et fragmenteeritud paketi kogusuurus ei oleks suurem kui lubatud.

Teardrop rünnaku IP-paketi fragmentide andmed katavad üksteist üle ja võivad olla üle lubatud suuruse. Fragmentide kokkupanemisel jooksevad operatsioonisüsteemid kokku.

Land.c rünnak on SYN-uputuse rünnaku erijuht, kus IP-paketi lähteaadress võltsitakse siht-aadressiga ekvivalentseks, samuti TCP pordid. Selle tulemusena saadetakse ohvri arvutis ühenduse alustamise pakett iseendale, mille tulemusena mõnede operatsioonisüsteemide (nt. Windows XP SP2 ilma tulemüürita hangus mõneks sekundiks) TCP/IP protokollistik jooksis kokku.

Sunkill rünnak on näidis tarkvara veast. Nimelt, kui Sun Solaris operatsioonisüsteemi proovida telneti kaudu sisse logida ja kasutajanime küsimise peale ujutada ühendus üle ^D (ASCII 4. märk) märkidega, siis jookseb rünnatav arvuti kokku.

Teenusetõkestusrünnete vastane kaitse

Esimese abinõuna on vaja hoida tarkvara uuendatuna, eriti operatsioonisüsteem. IPv6 puhul aitab teenusetõkestusründeid hajutada suvaedastuse kasutamine, millega hajutatakse päringud erinevate serverite vahel. Suvaedastuse puhul ei saa kliendi rollis olev ründaja koondada rünnakut mingi konkreetse serveri vastu. Kasulik on hoolitseda selle eest, et liikluse ligipääsukontrolli nimekirjade moodustamine oleks rünnaku korral kiirelt rakendatav. Jooksvalt peab arvestama, et mida rohkem informatsiooni ründaja võrgu kohta saab, seda haavatavam võrk potentsiaalselt on. Mitteennetavate meetmetena võib DDoS-ründe korral näiteks kasutada järgmisi võimalusi: DNS-serveril lasta vastata osadele päringutele erinevalt (nt. kui samalt IP-aadressilt küsitakse nimelahendust väga tihti, siis võidakse talle vastata nimelahenduse IP-aadressina nt. 127.0.0.1 või privaatkasutuseks mõeldud IP-aadress); muuta IP-aadressi (nt. muudeti Valge Maja veebilehe IP-aadressi ussi *Code Red* rünnakust pääsemiseks); avastada ründaja kontrollarvuti, kust ta kogub informatsiooni rünnatava töötamise kohta; avastada *botnet*'i juhtmasin.

20.4 Muid rünnakuviise

DNS

Tavaliselt pöördub inimene või programm serveri poole domeeninime alusel, mis viiakse vastavusse IP-aadressiga. Ründajatel on võimalik domeeninime viia vastavusse endale sobiva IP-aadressiga, millega kasutaja suunatakse vale IP-aadressi poole, milleks võib olla näiteks võltsitud internetipanga lehekülg. Võltsitud internetipank võib ka olla vahendusründaja osas.

DNS-serverite vastu on ründeid olnud, kuid need on vaatamata sellele, et DNS ei ole tõestatava autentimisega protokoll, siiski piisavalt raskesti rünnatavad ja enamasti kasutatakse teisi kergemini rünnatavaid võimalusi. Kergem on rünnata kohaliku masina hostifaili, lisades sinna uusi ridasid, mis lahendaks mõningad domeeninimed ründaja jaoks sobivateks IP-aadressideks (vaata peatükki "DNS-kliendi päringud ja seadistamine", lk. 165). Selleks peab ründajal muidugi olema kirjutamisõigused vastavatele failidele, mis kuuluvad üldjuhul administraatorile või juurkasutajale.

Möödahiilimised.

Kui tulemüüri poolt on keelatud lasta sisse liiklust mingile pordile, siis on sellest võimalik mööda hiilida, kasutades ära paketi fragmenteerimist. Nimelt fragmenteerib ründaja paketid piisavalt väikesteks, et TCP või UDP pordi number jääks teise fragmenti. Lõpp-host paneb fragmendid kokku, millega hiilitakse mööda tulemüürist. Kaitsena võivad tulemüürid ise paketi fragmentidest kokku panna.

Veebilehe näotustamine

Veebilehe kaudu saadakse asutuse kohta vajalikku informatsiooni. Veebilehe näotustamise ründe korral muudetakse asutuse veebilehte, kirjutades mingeid teateid, mis üldjuhul häbistavad rünnatavat. Veebilehe näotustamisega tekitatakse mainekahju. Samuti võidakse kustutada või üle kirjutada vajalikke andmeid, mis võib omakorda veel lisakahju tuua.

Liba-DHCP-server

Tihti on arvutid seadistatud saama võrguseadistuse DHCP-serverist. Seda võib ära kasutada kohtvõrgus mõne arvuti üle kontrolli omav ründaja, kes paneb püsti liba-DHCP-serveri, hakates alternatiivseks DHCP-serveriks. Selliselt on võimalik sisevõrgu teistele arvutitele anda ette ründajale sobiv seadistus (nt. võrguvärav, DNS-serveri aadress), mille baasil on võimalik teostada näiteks vahendusründeid. Kaitsena võib võrguadministraator kontrollida kõiki päringutele vastavaid DHCP-serverid, et seal ei oleks üleliigseid.

Teisi ründeid

Ründeid on erinevaid, näiteks:

- ARP-tabeli võltsimine, mille puhul vastab ARP-päringule vale masin (*ARP spoofing*, *ARP poisoning*). Selliselt tehakse teiste masinate poolt vale MAC-aadressi ja IP-aadressi vaheline sidumine, mille tulemusena paketid saadetakse valele hostile (nt. ründajale).
- Marsruutimisinfo võltsimine, milleks võidakse kasutada marsruutimisprotokolle või marsruuteritesse sisse murdmist ja staatiliste marsruutide lisamist.
- Õngitsemine (*phishing*) on üks rakendussotsioloogia tehnika variante. Ründaja saadab rünnatavale e-maili, mis esitleb ennast mingi tuntud keskkonna poolt saadetud (nt. pangad, veebikeskkonnad) teatena, millele proovitakse kasutaja reageerima saada. Näiteks turvaintsident või andmete kaotsi minek (väidetavalt riknesid serveri andmed), mistõttu on vaja koheselt etteantud lingi kaudu sisse logida, et antaks vajalikud andmed. Teine variant on meelitada mingit uut võimalust või teenust kasutama, milleks peab sisse logima ründaja poolt ette antud kohast (nt. otse e-mailist). Tavaliselt saadetakse õngitsemise e-maili suurtes kogustes, lootes mõne lihtsameelse õngeminekut.
- Brauseripõhised ründed. Brauserid on väga laialt kasutatavad, mistõttu on nad soodne pinnas, et ära kasutada brauserites olevaid turvaauke või mängida kasutaja rumalusele (proovitakse kasutajat meelitada installeerima mingit vinget programmi või ActiveX komponenti, mis sisaldab varjatult aga pahavara). Veebilehtede vahendusel teostavatest rünnetest üks tuntumaid on XSS-ründed, mis võivad ära kasutada brauseri turvaauke, proovida õngitseda, varastada andmeid jms.

20.5 Kaitse

Vaatleme ka mõningaid vahendeid, mida potentsiaalne rünnatav saaks kaitseks kasutusele võtta. Loomulikult ei ole alltoodu kaugeltki lõplik hulk meetmeid.

Tulemüürid

Tulemüüriks nimetatakse tarkvaralist või riistvaralist seadet, mis on seadistatud lubama ja keelama liiklust usaldatud ning välise võrgu vahel. Tulemüüride ülesanne on mingil määral kontrollida liiklust, et ta vastaks reeglitele.

Tulemüüride seadistuses on võimalik määrata, milliseid erinevaid reegleid rakendatakse välis- ja sisevõrgu vahelisele võrguliiklusele. Näiteks võidakse määrata, et mingile sisevõrguhostile saab ligi pääseda ainult määratud IP-aadressidelt või võrkudest, samuti määrata lubatavaid porte. Ka võimaldavad tulemüürid rakendada keerulisemaid reegleid. Näiteks tihti on vaja avaliku IP-aadressiga võrku eraldada Internetist, et võrgus olev arvuti saaks teha ühendusi väljapoole, kuid väljastpoolt ei saaks ühendusi teha (kasutades TCP protokolli). Selleks on võimalik kasutada asjaolu, et TCP segmendi korral on erandlik juhtum, kui SYN lipp on püsti ja ACK lipp on maas. Nimelt sellises kombinatsioonis on lipud siis, kui ühenduse looja saadab esimese segmendi sihtpunkti. Seda asjaolu saab ära kasutada asutuse tulemüür, keelates ära sisse tulevad segmendid, millel on SYN lipp püsti, kuid ACK lipp maas.

Tulemüüre võib jagada olekuta ja olekuga tulemüürideks. Olekuta tulemüürid lihtsalt filtreerivad informatsiooni paketipõhiselt. Kõik paketid, millele rakendatakse eraldi ettemääratud reegleid, on neile sõltumatud. Olekuga tulemüürid peavad arvet ühenduste kohta, saades ise aru ühenduste loomisest ning lõpetamistest. Selliselt saab vältida seansside kaaperdamist ja mõningaid teenusetõkestusrünnete liike.

Tavaliselt töötavad tulemüürid kuni OSI neljanda kihi tasemeni. Mõningad tulemüürid võimaldavad reegleid rakendada ka rakenduskihiprotokollidele, mistõttu neid nimetatakse rakenduskihi tulemüürideks. Rakenduskihi tulemüürid peavad toetama kontrollitavaid protokolle. Kontrollida võidakse näiteks viiruste sisaldumist veebiliikluses ehk HTTP protokollis, FTP-ga tõmmatavates failides, e-kirjade sisus. Lisaks sellele võidakse mõningaid protokolle modifitseerida erinevatel põhjustel, näiteks võidakse eemaldada või asendada HTTP päises "User-Agent" andmed, et suurendada kasutajate privaatsust. Rakenduskihi tulemüürid võivad olla kombineeritud ka vahendaja funktsionaalsusega. Samuti võib tulemüür sisaldada IDS (*intrusion detection system*) või IPS (*intrusion prevention system*) laiendusi. IDS laiendus proovib avastada võimalikke ründeid, mida logitakse ja teatatakse toimunust võrguadministraatorile. IPS on IDS-i edasiarendus, mis püüab takistada rünnakut, blokeerides vastava liikluse.

Üks liik tulemüüre on personaalarvutis olevad personaalsed tulemüürid, millega saab turvareegleid rakendada lisaks tavatulemüüride poolt pakutavale ka rakendusprogrammide kaupa, mis lisab suuresti turvalisust. Turvalisuse varjuküljeks on asjaolu, et programmid, mille võrgukasutust piiratakse, asuvad samas masinas tulemüüri tarkvara endaga. See loob võimalused kompromiteerida tulemüüritarkvara nii teiste programmide kui ka kasutajate poolt. Jällegi on oluline mitte kasutada masinat administraatori või juurkasutajana. Tulemüüri võib pidada olulisemaks vahendiks kui viirusetõrjeprogrammi. Üldiselt ei tohiks personaalarvuti olla Internetti otse ühendatud, kui tal ei ole paigaldatud ja töötav tulemüür, eriti juhul, kui operatsioonisüsteem ja muu võrku kasutav tarkvara ei ole kõige viimasemate turvauuendustega. Kui operatsioonisüsteem ja muud programmid (nt. brauser) on viimaste turvauuendusteta, siis ei pruugi ka tulemüürist kasu olla.

Meepotid

Kaitsjad võivad kasutada spetsiaalseid hoste, mis emuleerivad haavatavat süsteemi ning püüavad saada ründaja kohta võimalikult palju informatsiooni, milleks võib olla ründaja rünnakumoodused ja taktika ning tarkvara, mida sinna "istutatakse". Ründaja, kes satub meepotiga võitlema, võib reeta oma võtteid, mida võivad analüüsida turvatarkvara tegijad.

Kursis olek

Võrguadministraator ja üldse infotehnoloogiaga tegelevad spetsialistid peavad ennast hoidma kursis globaalselt toimuvaga, ennast harima ja hallatavat võrku täiustama. Turvalisus on muutunud järjest olulisemaks teemaks. Internetis on mitu turvaaukudele pühendunud veebisaiti, kus on informatsiooni uutest leitud turvaaukudest ja aktuaalsetest ussidest.

Üks veebisaitidest, mis pakub arvutiturvalisuse koolitust, sertifitseerimist ja uuringuid on SANS (*SysAdmin, Audit, Network, Security*), mille koduleht asub aadressil: http://www.sans.org. Avastatud turvaaukude kohta saab informatsiooni saidilt CVE (*Common Vulnerabilities and Exposures*), mille aadress on http://cve.mitre.org. Samuti on suuremaid andmebaase nõrkuste kohta NVD (*National Vulnerability Database*), mille aadress on http://nvd.nist.gov.

Müüdid

Järgnevalt toome välja mõningaid eksiarvamusi, mis on seotud võrguturbega.

- Antiviirusprogrammid kaitsevad minu süsteemi. Antiviirusprogrammid töötavad põhiliselt viiruste signatuuride põhjal, mistõttu nad ei saa kaitsta otseselt uute viiruste eest. Samas on olemas ka pahavara, mille kood muutub, mille tõttu ei ole võimalik neid identifitseerida signatuuri järgi. Osad viirused on nn. polümorfsed, mis krüpteerivad ennast (tavaliselt väga lihtne krüpteering) või metapolümorfsed viirused, mis muudavad oma koodi järjekorda ja vastavalt hüppamisi ning lisaks sisaldavad juhuslikku surnud koodi, mida mitte kunagi ei täideta. Antiviirusprogrammid "õpivad" pahavara eemaldama tagantjärele ja vähelevinud ning eriotstarbelist pahavara ei pruugita kunagi osata ära tunda. Pahavara modifikatsioone lisandub päevas juurde tuhandeid. Antiviirusprogrammide efektiivsuse trend on vähenemise suunas, ulatudes hinnangute kohaselt kolmandikuni.
- Turvaline on avada ainult tuttavatelt tulnud e-maile (lähteaadressi on lihtne võltsida).
- Viirused ei tee mu süsteemile suurt kahju. Ühe viiruse või ussi hävitustöö põhjustatud kahjud võivad ulatuda maailma kontekstis sadadesse miljonitesse dollaritesse.
- Kes peaks tahtma minu süsteemi sisse murda, sest siit ei ole midagi saada. Otseselt tahavad *botnet*'i omanikud, kes vajavad antud arvuti poolt kasutatavat ressurssi. Kasutaja enda jaoks tundub Internet kuidagi aeglane ja masin võib olla ebastabiilne.
- Kui ma kunagi midagi alla ei tõmba, siis ma pole ohus. Turvaaugud, mida ussid ära kasutavad, on ikkagi ohuks.
- Minu arvuti on ainult mõned tunnid päevas sees, seega risk on äärmiselt väike. Lappimata Windowsi masin saab nakatatud umbes mõne minuti jooksul.
- Kui kõik kasutaksid krüpteerimist, siis me oleksime kaitstud rünnete eest. Krüpteerimine ei kaota ründeid, kuid muudab nad raskemaks, sest paljusid meetodeid ei saaks praktiliselt kasutada.
- Kui ühendun veebiserveriga üle HTTPS-i, siis informatsioon on kaitstud.

Soovitusi

Võrgu turvalisust on alati võimalik tõsta. Järgnevalt mõningaid soovitusi, kuidas saaks turvalisust tõsta (tegemist ei ole kaugeltki ammendava loeteluga):

- Kasutada turvalisi paroole. Paroolid peaksid olema vähemalt kaheksa või enam märki pikad. Nad ei tohi olla seotud isikuga (parooli ära arvamine), sõnaraamatus olevad sõnad

(sõnastikurünne), peavad sisaldama erinevaid tähti, numbreid, suuri ja väikeseid tähti. Ei tohiks kasutada eelmainitud sõnu, kus teatavad tähed on asendatud numbritega (nt. "e"⇒"3", "a"⇒"4", "l"⇒"1", "o"⇒"0", "õ"⇒"6", "ä"⇒"2", "ö"⇒"5", "ü"⇒"y"). Parool peaks olema võimalikult juhuslik, kuigi keeruliste paroolide meeldejätmine võib olla problemaatiline. Üks võimalik variant on võtta mingid suvalised sõnad ja kombineerida ja modifitseerida neid. Näiteks tulevad pähe sõnad "katus" ja "massaaž". Nendest tuletatud parool võiks olla "gAdus5masaz" (fantaasia vili). Tasuks jälgida, et parooli oleks võimalik kiiresti sisestada (juhuks, kui keegi näeb parooli sisestamist).

- Vahetada paroolid niipea kui võimalik, kui parool võib olla lekkinud (nt. pealtkuulamiskahtlus, trooja hobuse avastamine, keegi nägi parooli sisestamist pealt).
- Kasuta erinevates kohtades erinevaid paroole. Kui paroole on liialt palju ja nad hakkavad ununema või sassi minema, siis võimalik kasutada spetsiaalseid programme paroolide turvaliseks hoidmiseks. Üks selliseid programme Windows keskkonnale on "Password agent", mille saab aadressilt http://www.moonsoftware.com/pwagent.asp (eestlastele on tasuta). Antud programmi paroolifaili parool peaks olema eriti turvaline (eeldatavasi on programmi krüpteerimisalgoritmi AES kasutamine turvaliselt realiseeritud) ja antud programmi tuleks kasutada ainult siis, kui teised ei ole monitori nägemisulatuses. Samuti ei tohiks vähemalt oluliste paroolide hoidmise faile avada suvalistes arvutites. Ohuks on klahvivajutuste nuhkprogrammid (*keylogger*).
- Paroolide valesti proovimistele tuleks seada limiit. Kui kasutaja proovib näiteks rohkem kui viis korda valede paroolidega sisse logida, siis tuleks kasutaja blokeerida mingiks ajaks. Samas peaks arvestama ründega, millega proovitakse kasutaja blokeerida meelega, millega põhjustatakse ebamugavusi, sest kasutaja ei saa ajutiselt oma kontot kasutada (näiteks pangakontod). Antud juhul on kaitseks võimalik kasutada IP-aadressi või terve aadressivahemiku järgi mingiks ajaks blokeerimist.
- Kasutajatele ei tohiks anda süsteemis õigusi, mida tal ei ole oma tööks vaja.
- Hoida arvutivõrk ka füüsiliselt turvatud.
- Asutustes peaks olema konsultatsioonipunkt või inimene, kelle poole saab pöörduda, kui kasutaja on sattunud kahtlasevõitu olukorda. Suures osas on arvutivõrgu nõrgim lüli teadmatu inimene.
- Võimaluse korral automatiseerida tegevusi, et vältida juhtumeid, millele aitab kaasa inimlik laiskus, mugavus, eksimus ja unustamine. Täpne rakendamine on täiesti rakendusspetsiifiline, kuid üldiselt võiks välja tuua: paroolide vahetamine (nt. iga aasta tagant), uuenduste automaatne paigaldamine (hea, kui administraatori eelneva nõusolekuga või hulgitöötlusena), logifailide automaatsed analüsaatorid (teavitavad ainult tõsisematest juhtumitest ja annavad perioodiliselt üldist statistikat).
- Ära kasuta arvutit administraatori või juurkasutaja õigustes, sest ka käivitatud programmid omavad siis administraatori õigusi. Ole sisse logitud tavakasutajana ja vajadusel käivita administraatori õigusi vajav programm administraatori turvakontekstis. Linuxis kirjuta ühe võimalusena käsule ette "sudo ". Windows keskkonnas saab Exploreris käivitatava faili hüpikmenüüst valida "Run as...".
- Windowsi paigaldamisel eelistada NTFS (*New Technology File System*) failisüsteemi, mitte FAT-i. Kindlasti peaks olema NTFS-failisüsteemis ketas, kus paikneb Windowsi kataloog ja ka installeeritud programmid. Kui viirus on nakatanud kasutaja õigustes programmi ja administraator käivitab vastava programmi, siis võivad nakatuda ka teised programmid, millele ainult administraatoril on kirjutamisõigused.
- Ära ole hoolimatu. Täida turvalisuse tagamiseks seatud ettekirjutusi. Reeglitest on vähe kasu, kui neid ei täideta.
- Pöörata tähelepanu pealtkuulamisohule ühissõidukis, kohvikus või mujal oma materjale lugedes ja nendega töötades või suheldes kaaslasega. Juhuslikult võivad asjast mingil moel huvitatud inimesed materjale näha või pealt kuulata.

- Võõras arvutis on alati oht klahvilugejate, vahemälu jms. osas.
- Lugege kõiki veateateid hoolega ja proovige neist aru saada.
- Kui märkate, et käivitatavaid või teegifaile on muudetud (võimaldavad paljud tulemüüri ja viirustõrjetarkvarad) ja ei ole teostatud tarkvarauuendust, siis on see alarmiks.
- Personaaltulemüüri abil laske suhelda ainult neil programmidel, mida teate, miks nad vajavad ligipääsu võrgule (nt. veebibrauser). Kui kahtlete, siis ärge lubage programmil võrku kasutada. Otsige vajadusel infot ka Internetist.
- Veendu, et kuskilt (nt. avalikust FTP-serverist) allalaaditud tarkvara on modifitseerimata (ründaja võis paigutada programmi lisaks trooja hobuse). Paljudes tarkvara allalaadimiskohtades on välja toodud installeerimisfaili räsisid, mida saab kasutada, veendumaks mitte modifitseerituses (nt. MD5, SHA-1).
- Ära käivita tundmatu päritoluga programme, eriti kahtlastest veebisaitidest või FTP-serveritest. Tihti võivad sellised programmid tekstide järgi lubada paljut, kuid mis on tegelikult mõeldud kergeusklike meelitamiseks programmi käivitama (programm võib salaja trooja hobuse arvutisse paigutada).
- E-maili programmis keelata e-kirjades olevate väliste failide laadimine. Vastasel juhul võimaldab rämpspostitajal saada teada, et antud e-mail on aktiivselt kasutusel.
- Keerata maha arvutites ja võrguseadmetes kõik teenused, mida ei kasutata ja mis on ebaturvalised (näiteks SNMP). Kõik üleliigsed teenused võivad olla asjatuteks turvaaukudeks.
- Keerukus on turvalisuse vaenlane ehk järgi KISS (*Keep It Simple, Stupid*) printsiipi. Mida keerukamat süsteemi ehitatakse, seda vähem kontrollitakse detailselt olukorda ja rohkem võimalikke eksimusi võib tekkida.
- Kvaliteet ja turvalisus on tihedalt seotud. Kvaliteedi osas ei tohiks järeleandmisi teha, sest võivad tekkida ootamatud probleemid.
- Kui server on sattunud rünnaku ohvriks, siis tasub server installeerida puhtalt lehelt, sest ründaja võis maha jätta *rootkit*'i, mille abil ta järgmine kord uuesti ja kergemini masinasse pääseb.
- Pea meeles, et kui asi tundub liiga hea, et tõsi olla, siis reeglina see nii ka on, s.t. ilmselt on tegemist pettusega (nt. "Nigeeria pärandus", lotovõidud).

Lõpetuseks

Arvutivõrgu turvalisus ei ole midagi sellist, mis saab kunagi valmis ja temaga ei pea enam tegelema. Kui arvame, et kõik peaks olema korras, siis võib ründaja selle ikkagi ümber lükata, sest ründaja kohta pole midagi teada. Efektiivsed rünnakud on need, mis kasutavad midagi ootamatut. Näiteks uss "SQL Slammer" kasutas nakatamiseks UDP paketti, mille suuruseks oli ainult 376 baiti, mida enamus tulemüüre ei osanud kahtlustada pahavarana tema väiksuse tõttu ning mistõttu oli uss väga edukas.

Tihti öeldakse, et arvuti on turvatud, kui ta on tundmatusse kohta maha maetud, kuigi ka siis veel mitte sajaprotsendiliselt. Me peame arvestama, et turvalisuse üks nõudeid on käideldavus. Nimelt ei ole tundmatusse kohta maha maetu enam käideldav.

Eeldatavasti on turvaprobleeme rohkem kui avalikult räägitakse, sest paljudel juhtumitel ei oldagi kursis nende vastu toimunud õnnestunud ründest. Firmad aga võivad peljata maine kahjustumisega kaasnevaid probleeme.

21. Sõnastik

21.1 Eesti - Inglise

abonendisiin – *local loop*
administratiivne kaal – *administrative distance*
aega elada – *time to live*
aeglaselt alustamise algoritm – *slow-start*
aegmultipleksimine – *time-division multiplexing*
ainumood – *single-mode*
akna suurus – *window size*
ala – *area*
alamvõrk – *subnet*
allavool – *downstream*
edastus – *transmission*
automaatkätlus – *auto-negotiation*
baasjaam – *access point*
demarkatsioonipunkt – *demarcation point*
eluaja ületamine – *time exceeded*
esimese taseme domeen – *top level domain*
esituskiht – *presentation layer*
farssbitt – *bit stuffing*
fragmenditu – *fragment-free*
füüsiline kiht – *physical layer*
globaalvõrk – *global area network*
hajus teenusetõkestusrünne – *denial of service*
hõivamine – *capture*
häkker – *hacker*
hübriidkommuteerimine – *adaptive switching*
hüpe – *hop*
infrastruktuuri laad – *infrastructure mode*
insertsiooni kadu – *insertion loss*
internetiteenusepakkuja – *Internet service provider*
iteratiivne päring – *iterative request*
jadaedastus – *serial transmission*
jaotur – *hub*
juhuvõrk – *ad-hoc*
juurkommutaator – *root bridge*
juurnimeserver – *root name server*
juurport – *root port*
jõulupuupaketi rünne – *christmas tree attack*
jäik kaabel – *solid cable*
järgur – *repeater*
kaader – *frame*
kaaperdamine – *hijacking*
kaja päring – *echo request*
kaja vastus – *echo reply*
kanalikiht – *data link layer*
kanalkommutatsioon – *circuit switched*
keerdpaarkaabel – *twisted pair cable*
keskjaam – *central office*
kiht – *layer*
kiire taastumine – *fast recovery*
kiireloomulise teate viit – *urgent pointer*
klassidega adresseerimine – *classful addressing*
klient-server – *client–server*
koduvõrk – *home-area network*
kogukond – *community*
kohtumispunkt – *rendezvous point*
kohtvõrk – *local area network*
kollisioonidomeen – *collision domain*
kolmeosaline käepigistus – *three-way handshake*
kommutaator – *switch*
komposiitmeetrika – *composite metric*
kräkker – *cracker*
käideldavus – *availability*
laiendatud tähttopoloogia – *extended star topology*
laivõrk – *wide area network*
leviedastus – *broadcast*
leviedastusaadress – *broadcast address*
leviedastustorm – *broadcast storm*
levimise kiirus – *propagation delay*
lokaalselt administreeritud aadressid – *locally administered addresses*
loogikapomm – *logic bomb*
läbilaskevõime – *thtoughput*
lülitus-kommuteeritud – *circuit-switched*
lüüs – *gateway*
maashoidmise taimer – *holddown timers*
magistraalala – *backbone area*
magistraalvõrk – *backbone network*
marsruut – *route*
marsruuter – *router*
marsruuteri kuulutus – *router advertisement*
marsruuteri leidmise palve – *router solicitation*
marsruutimine – *routing*
marsruutimisprotokoll – *routing protocol*
marsruutimistabel – *routing table*
meepott – *honeypot*
meetrika – *metrics*

mitmekiuline kaabel – *stranded cable*
mittepädev – *non-authoritative*
multiedastus – *multicast*
multikodu – *multihoming*
multimood – *multi-mode*
multiplekser – *multiplexer*
murdumisnäitaja – *refractive index*
märgend – *label*
märgutulekaader – *beacon frame*
määratud marsruuter – *designated router*
määratud port – *designated port*
naabri kuulutus – *neighbor advertisement*
naabri leidmise palve – *neighbor solicitation*
nimelahendaja – *name resolver*
nimeruum – *namespace*
nimeserver – *name server*
nullala – *area 0*
oktett – *octet*
orinimeserver – *slave name server*
otsekaabel – *straight-through cable*
pahavara – *malware*
pakett – *packet*
pakettkommutatsioon – *packet-switched*
partner – *peer*
partnerilt-partnerile – *peer-to-peer*
peegelport – *port mirroring*
pehme kaabel – *stranded cable*
personaalne tulemüür – *personal firewall*
personaalvõrk – *personal area network*
pilu – *slot*
polüpropüleen – *polyethylene*
polüvinüülkloriid – *polyvinyl chloride, PVC*
pooldupleks-edastus – *half-duplex transmission*
pordi peegeldus – *port mirroring*
port – *port*
primaarne nimeserver – *primary name server*
protokoll – *protocol*
protokolli andmeüksus – *protocol data unit*
protokollistik – *protocol stack*
puhvermälu nimeserver – *caching-only name server*
punktist-punkti sideliin – *point-to-point link*
puutopoloogia – *tree topology*
pädev nimeserver – *authoritative name server*
päis – *header*
pöördteisendus – reverse lookup
radavektorprotokoll – *path vector protocol*
rakenduskiht – *application layer*
regionaalsed Interneti registrid – *regional Internet registries*
regionaalvõrk – *metropolitan area network*
rekursiivne päring – *recursive query*
repiiter – *repeater*
ressursikirje – *resource record*
ribalaius – *bandwidth*
ringtopoloogia – *ring topology*
ristkaabel – *crossover cable*
rumala akna sündroom – *silly window syndrome*
rändlus – *roaming*
rööpedastus – *parallel transmission*
ründaja – *attacker*
saba – *trailer*
sagedusmultipleksimine – *frequency division multiplexing*
salvestusvõrk – *storage-area network*
seansikiht – *session layer*
segment – *segment*
segmenteerimine – *segmentation*
sekundaarne nimeserver – *secondary name server*
sideliin – *link*
sideliini-oleku-marsruutimisprotokoll – *link-state routing protocol*
sidumispunkt – *demarcation point*
siintopoloogia – *bus topology*
sild – *bridge*
simpleks-edastus – *simplex transmission*
skriptijuntsu – *script kiddie*
staatiline marsruut – *static route*
staatiline marsruutimine – *static route*
sumbuvus – *attenuation*
supervõrk – *supernet*
surma ping – *ping of death*
suunatud leviedastus – *directed broadcast*
suvaedastus – *anycast*
sünkroonne edastus – *synchronous transmission*
taasesitusrünne – *replay attack*
tagasipõrke kadu – *return loss*
tagasisidestusaadress – *loopback address*
tagasivõtu algoritm – *backoff*
tagauks – *backdoor*
teenus – *service*
teenusekvaliteet – *quality of service, QoS*
teenusetaseme leping – *service level agreement*
teenusetõkestusrünne – *denial-of-service attack; DoS attack*
toesepuu algoritm – *spanning-tree algorithm*
topoloogia – *topology*
traadita ühendus – *wireless connection*
transpordikiht – *transport layer*

transpordilaad – *transport mode*
tulemüür – *firewall*
tunnelilaad – *tunnel mode*
tupikmarsruuter – *stub router*
tähttopoloogia – *star topology*
täisdupleks-edastus – *full-duplex transmission*
täisvõrktopoloogia – *full mesh topology*
ummistuste vältimine – *congestion avoidance*
ummistussignaal – *jam signal*
uss – *worm*
vaatevälja jagamine – *split horizon*
vahehoidega edastus – *store-and-forward*
vahendusrünne – *man-in-the-middle attack*
vahetult ühendatud võrk – *directly connected network*
vaikemarsruut – *default route*
vaikevõrguvärav – *default gateway*
valge müra – *white noise*
varjestamata keerdpaar – *unshielded twisted pair*
varjestatud keerdpaar – *shielded twisted pair*
veebilehe näotustamine – *defacement*
viirus – *virus*
virtuaalne kohtvõrk – *virtual LAN (local area network)*
vookontroll – *flow control*
vooledastus – *cut-through*
võrguaadress – *network address*
võrgukaart – *NIC (network interface card)*
võrgukiht – *network layer*
võrgumask – *network mask*
võrguvärav – *gateway*
võrktopoloogia – *mesh topology*
väli – *field*
värin – *jitter*
õngitsemine – *phishing*
ühekiuline kaabel – *solid cable*
üksikedastus – *unicast*
ülekoste – *crosstalk*
ülemnimeserver – *master name server*
ülereserveerimine – *overbooking*
ülesvool – *upstream*
ümbersuunamine – *redirect*

21.2 Inglise - Eesti

access point – baasjaam
ad-hoc – juhuvõrk
adaptive switching – hübriidkommuteerimine
administrative distance – administratiivne kaal
anycast – suvaedastus
application layer – rakenduskiht
area 0 – nullala
area – ala
attacker – ründaja
attenuation – sumbuvus
authoritative name server – pädev nimeserver
auto-negotiation – automaatkätlus
availability – käideldavus
backbone area – magistraalala
backbone network – magistraalvõrk
backdoor – tagauks
backoff – tagasivõtu algoritm
bandwidth – ribalaius
beacon frame – märgutulekaader
bit stuffing – farssbitt
bridge – sild
broadcast address – leviedastusaadress
broadcast storm – leviedastustorm
broadcast – leviedastus
bus topology – siintopoloogia
caching-only name server – puhvermälu nimeserver
capture – hõivamine
central office – keskjaam
christmas tree attack – jõulupuupaketi rünne
circuit switched – kanalkommutatsioon
circuit-switched – lülitus-kommuteeritud
classful addressing – klassidega adresseerimine
client–server – klient-server
collision domain – kollisioonidomeen
community – kogukond
composite metric – komposiitmeetrika
congestion avoidance – ummistuste vältimine
cracker – kräkker
crossover cable – ristkaabel
crosstalk – ülekoste
cut-through – vooledastus
data link layer – kanalikiht
defacement – veebilehe näotustamine
default gateway – vaikevõrguvärav
default route – vaikemarsruut
demarcation point – demarkatsioonipunkt
demarcation point – sidumispunkt
denial of service – hajus teenusetõkestusrünne
denial-of-service attack; DoS attack – teenuse-

tõkestusrünne
designated port – määratud port
designated router – määratud marsruuter
directed broadcast – suunatud leviedastus
directly connected network – vahetult ühendatud võrk
downstream – allavool
echo reply – kaja vastus
echo request – kaja päring
extended star topology – laiendatud tähttopoloogia
fast recovery – kiire taastumine
field – väli
firewall – tulemüür
flow control – vookontroll
fragment-free – fragmenditu
frame – kaader
frequency division multiplexing – sagedusmultipleksimine
full mesh topology – täisvõrktopoloogia
full-duplex transmission – täisdupleks-edastus
gateway – lüüs, võrguvärav
global area network – globaalvõrk
hacker – häkker
half-duplex transmission – pooldupleks-edastus
header – päis
hijacking – kaaperdamine
holddown timers – maashoidmise taimer
home-area network – koduvõrk
honeypot – meepott
hop – hüpe
hub – jaotur
infrastructure mode – infrastruktuuri laad
insertion loss – insertsiooni kadu
Internet service provider – internetiteenusepakkuja
iterative request – iteratiivne päring
jam signal – ummistussignaal
jitter – värin
label – märgend
layer – kiht
link – sideliin
link-state routing protocol – sideliini-oleku-marsruutimisprotokoll
local area network – kohtvõrk
local loop – abonendisiin
locally administered addresses – lokaalselt administreeritud aadressid
logic bomb – loogikapomm
loopback address – tagasisidestusaadress
malware – pahavara
man-in-the-middle attack – vahendusrünne
master name server – ülemnimeserver
mesh topology – võrktopoloogia
metrics – meetrika
metropolitan area network – regionaalvõrk
multi-mode – multimood
multicast – multiedastus
multihoming – multikodu
multiplexer – multiplekser
name resolver – nimelahendaja
name server – nimeserver
namespace – nimeruum
neighbor advertisement – naabri kuulutus
neighbor solicitation – naabri leidmise palve
network address – võrguaadress
network layer – võrgukiht
network mask – võrgumask
NIC (network interface card) – võrgukaart
non-authoritative – mittepädev
octet – oktett
overbooking – ülereserveerimine
packet – pakett
packet-switched – pakettkommutatsioon
parallel transmission – rööpedastus
path vector protocol – radavektorprotokoll
peer – partner
peer-to-peer – partnerilt-partnerile
personal area network – personaalvõrk
personal firewall – personaalne tulemüür
phishing – õngitsemine
physical layer – füüsiline kiht
ping of death – surma ping
point-to-point link – punktist-punkti sideliin
polyethylene – polüpropüleen
polyvinyl chloride – polüvinüülkloriid
port mirroring – peegelport
port mirroring – pordi peegeldus
port – port
presentation layer – esituskiht
primary name server – primaarne nimeserver
propagation delay – levimise kiirus
protocol data unit – protokolli andmeüksus
protocol stack – protokollistik
protocol – protokoll
PVC – vt. *polyvinyl chloride*
QoS – vt. *quality of service*
quality of service – teenusekvaliteet
recursive query – rekursiivne päring
redirect – ümbersuunamine
refractive index – murdumisnäitaja
regional Internet registries – regionaalsed

Interneti registrid
rendezvous point – kohtumispunkt
repeater – järgur
repeater – repiiter
replay attack – taasesitusrünne
resource record – ressursikirje
return loss – tagasipõrke kadu
reverse lookup – pöördteisendus
ring topology – ringtopoloogia
roaming – rändlus
root bridge – juurkommutaator
root name server – juurnimeserver
root port – juurport
route – marsruut
router advertisement – marsruuteri kuulutus
router solicitation – marsruuteri leidmise palve
router – marsruuter
routing protocol – marsruutimisprotokoll
routing table – marsruutimistabel
routing – marsruutimine
script kiddie – skriptijuntsu
secondary name server – sekundaarne nimeserver
segment – segment
segmentation – segmenteerimine
serial transmission – jadaedastus
service level agreement – teenusetaseme leping
service – teenus
session layer – seansikiht
shielded twisted pair – varjestatud keerdpaar
silly window syndrome – rumala akna sündroom
simplex transmission – simpleks-edastus
single-mode – ainumood
slave name server – orinimeserver
slot – pilu
slow-start – aeglaselt alustamise algoritm
solid cable – jäik kaabel
solid cable – ühekiuline kaabel
spanning-tree algorithm – toesepuu algoritm
split horizon – vaatevälja jagamine
star topology – tähttopoloogia
static route – staatiline marsruut
static route – staatiline marsruutimine
storage-area network – salvestusvõrk
store-and-forward – vahehoidega edastus
straight-through cable – otsekaabel
stranded cable – mitmekiuline kaabel
stranded cable – pehme kaabel
stub router – tupikmarsruuter
subnet – alamvõrk
supernet – supervõrk
switch – kommutaator
synchronous transmission – sünkroonne edastus
three-way handshake – kolmeosaline käepigistus
thtoughput – läbilaskevõime
time exceeded – eluaja ületamine
time to live – aega elada
time-division multiplexing – aegmultipleksimine
top level domain – esimese taseme domeen
topology – topoloogia
trailer – saba
transmission – edastus
transport layer – transpordikiht
transport mode – transpordilaad
tree topology – puutopoloogia
tunnel mode – tunnelilaad
twisted pair cable – keerdpaarkaabel
unicast – üksikedastus
unshielded twisted pair – varjestamata keerdpaar
upstream – ülesvool
urgent pointer – kiireloomulise teate viit
virtual LAN (local area network) – virtuaalne kohtvõrk
virus – viirus
white noise – valge müra
wide area network – laivõrk
window size – akna suurus
wireless connection – traadita ühendus
worm – uss

22. Aineregister