ИНТЕРНЕТ ПРОТОКОЛИ И ТЕХНОЛОГИЈЕ

Верица Васиљевић

Висока школа електротехнике и рачунарства
струковних студија
Београд, 2013.

Верица Васиљевић

Интернет протоколи и технологије

Висока школа електротехнике и рачунарства
струковних студија
Београд, 2013.

др Верица Васиљевић
ИНТЕРНЕТ ПРОТОКОЛИ И ТЕХНОЛОГИЈЕ
прво издање

Рецензенти: др Жарко Марков, др Борислав Одаџић

Издавач: Висока школа електротехнике и рачунарства струковних студија у Београду

За издавача: др Драгољуб Мартиновић

Лектор: Анђелка Ковачевић

Техничка обрада: др Верица Васиљевић, Ана Милетић, Веселин Илић, Марко Караџић, Никола Кајгана, Зоран Трандафиловић, Мирко Ступар

Дизајн: Мирко Ступар

Штампа: Школски сервис Гајић, Београд

Тираж: 50

Наставно веће Високе школе електротехнике и рачунарства струковних студија у Београду, на седници одржаној 3.10.2013. године, одобрило је издавање и примену овог уџбеника у настави.

CIP - Каталогизација у публикацији
Народна библиотека Србије, Београд

004 . 72 . 057 . 4 (075.8)

ВАСИЉЕВИЋ, Верица, 1952-
Интернет протоколи и технологије / Верица Васиљевић. - Београд : Висока школа електротехнике и рачунарства струковних студија, 2013 (Београд : Школски сервис Гајић). - V, 582 стр. : илустр. ; 22 cm + 1 електронски оптички диск (CD-ROM)

Тираж 50. - Библиографија: стр. 581-582.

ISBN 978-86-7982-188-1

а) Мрежни протоколи
COBISS.SR-ID 202409740

Предговор

Уџбеник Интернет протоколи и технологије резултат је дугогодишњих припрема предавања за предмете Рачунарске мреже и Протоколи у рачунарским мрежама на Високој школи електротехнике и рачунарства струковних студија у Београду. Уџбеник садржи теоретске и искуствене основе о протоколима жичних и бежичних умрежених система. Посебна пажња посвећена је протоколима мрежног, транспортног и апликативног слоја који су методички обрађени у 13 целина.

Уџбеници Рачунарске мреже и Интернет протоколи и технологије, оба од истог аутора и издавача, припремљени као мултимедијалне публикације дају систематски увид у област умрежавања.

Београд, Аутор
октобра 2013. године

Садржај

7. АПЛИКАЦИОНИ СЛОЈ И СИСТЕМ ИМЕНА ДОМЕНА

8. ВЕБ СИСТЕМИ

9. ЕЛЕКТРОНСКА ПОШТА

1. Протоколи и архитектуре

Концепти дистрибуираног процесирања и рачунарских мрежа условљавају да целине у различитим системима имају потребу да међусобно комуницирају. Користимо термин целина[1] и систем у општем смислу. Пример целине су: кориснички апликациони програм, пренос датотека, системи за управљање базама података, уређаји за електронску пошту и терминали. Пример система су: рачунар, терминал и удаљени сензор. Уочимо да су у неком случају целина и систем коегзистентни. У општем случају целина је било шта способно да шаље и прима информације, а систем је физички одвојен објекат који садржи једну или више целина.

Две целине које треба да комуницирају „морају да говоре истим језиком". Шта комуницира, како се комуницира и када се комуницира мора да се повинује неком опште прихватљивом скупу конвенција целина, које су укључене у комуникацију. Скуп конвенција назива се протокол, и он се може дефинисати као скуп правила која управљају разменом података између две целине. Кључни елементи протокола су:

- синтакса - укључује формат података, кодирање и нивое сигнала,
- семантика - укључује управљачке информације за координацију и вођење рачуна о грешкама,
- временска синхронизација[2] - укључује подешавање брзина и секвенционирање.

Један од примера је често коришћени протокол HDLC[3]. Подаци које треба разменити морају се слати у рамовима специфичног формата (синтакса). Управљачко поље обезбеђује различите регулаторске функције као што су постављање врсте рада и успостављање везе (семантика).

---

[1] *Entity* - ентитет

[2] *Timing*

[3] *High Level Data Link Control*

Резултати стандарда су следећи:

- произвођачи се подстичу да примене стандарде због очекивања да ће се, због широке употребе стандарда, њихови производи више продавати,
- корисници захтевају да стандарде примењују сви произвођачи јер могу опрему набављати од било кога од њих.

Задатак да се обезбеди комуникација између апликација на различитим рачунарима превише је комплексан да би се посматрао као једна целина. Проблем се мора раздвојити у целине које се могу лакше реализовати. Тако, пре него што неко развије стандарде, мора да постоји структура или архитектура која дефинише комуникационе циљеве.

Интернационална организација за стандардизацију (ISO[1]) је 1977. године формирала комисију са задатком да развије такву архитектуру. Резултат је отворени систем за међусобно повезивање (OSI[2]), референтни модел који је оквир за дефинисање стандарда за повезивање хетерогених рачунара. OSI модел обезбеђује основу за повезивање отворених система дистрибуираних апликативних процеса. Термин „отворен" одређује способност да било која два система која подржавају референтни модел и одговарајуће стандарде могу међусобно да се повежу.

1.1 Мрежна архитектура

Због комплексности у пројектовању многе мреже су организоване као низ слојева или нивоа. Број слојева, као и њихова имена разликују се од мреже до мреже. У свим мрежама намена сваког слоја је да пружи одређену услугу вишим слојевима, штитећи те нивое од детаља (нпр. како се услуга реализује).

Слој N на једном рачунару води конверзацију са слојем N на другом рачунару. Правила и конвенције које се користе у овој конверзацији заједничке су и познате као протокол N-тог слоја.

Практично, подаци се не шаљу директно из слоја N једног рачунара ка слоју N другог рачунара (осим најнижег слоја). Уместо тога, сваки слој шаље податке и управљачке информације ка слоју који је одмах

[1] *International Organization for Standardization*

[2] *Open System Interconnection*

испод, докле год не стигне до најнижег слоја. На најнижем слоју постоји физичка комуникација са другим рачунаром, супротно од виртуелне комуникације коју користе виши слојеви. Између сваког пара суседних слојева постоји интерфејс[1]. Интерфејс[2] слојева дефинише које основне операције и услуге нижи слој нуди вишем слоју N. Када пројектант мреже одлучи колико слојева ће укључити у мрежу и шта ће који од њих да ради, најважније је јасно дефинисати интерфејсе између слојева. Јасно дефинисање слојева, заузврат, захтева да сваки слој извршава специфични скуп јасно дефинисаних функција. Последица тога је да се један слој може заменити потпуно другим слојем (нпр. телефонска линија се може заменити сателитском везом), пошто је све што се тражи од новог слоја да пружи исти скуп услуга за слој изнад себе као што је претходно био обезбеђен.

Скуп слојева и протокола назива се слојевита архитектура.

Спецификација архитектуре мора да садржи довољно информација да омогући ономе ко њу имплементира да изгради програм за сваки слој, тако да програм исправно реализује одговарајући протокол. Ни детаљи примене, ни спецификација интерфејса нису делови архитектуре. У ствари, чак ни сви интерфејси на свим рачунарима нису исти, а ипак је обезбеђено да сваки рачунар може правилно да користи протокол.

Пример вишеслојне архитектуре је следећи[3]: замислите да два истраживача (парњаци[4] слоја 3) од којих је један у Београду а други у Минхену, желе да комуницирају (слика 1.1). Пошто немају заједнички

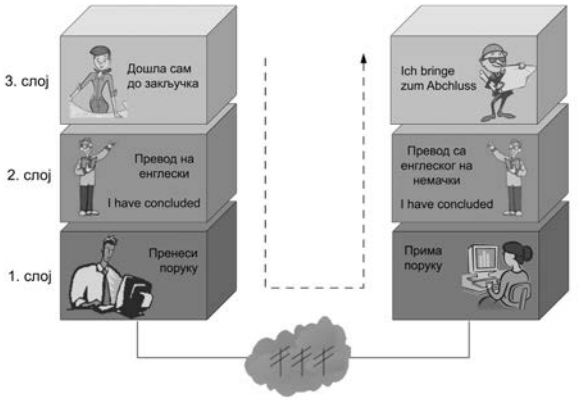
[1] Неке од општих дефиниција интерфејса:

- Тачка или средство интеракције са системом, без обзира да ли је корисник човек или други систем.
- У комуникационим системима интерфејс се везује за границу између два дела опреме преко које су сви сигнали који пролазе пажљиво дефинисани. Дефиниција обухвата нивое сигнала, импедансу, временски редослед, редослед рада и значење сигнала.
- Софтверски интерфејс може да се односи на низ различитих типова интерфејса на различитим нивоима: оперативни систем може имати интерфејс (везу) са хардвером; апликације или програми који раде на оперативном систему можда ће морати да комуницирају преко токова (*streams*); у објектно оријентисаним програмима објекти у оквиру апликације ће можда морати да комуницирају преко метода.

[2] Користи се и термин сучеље.

[3] Пример преузет из [10]..

[4] *Peer*



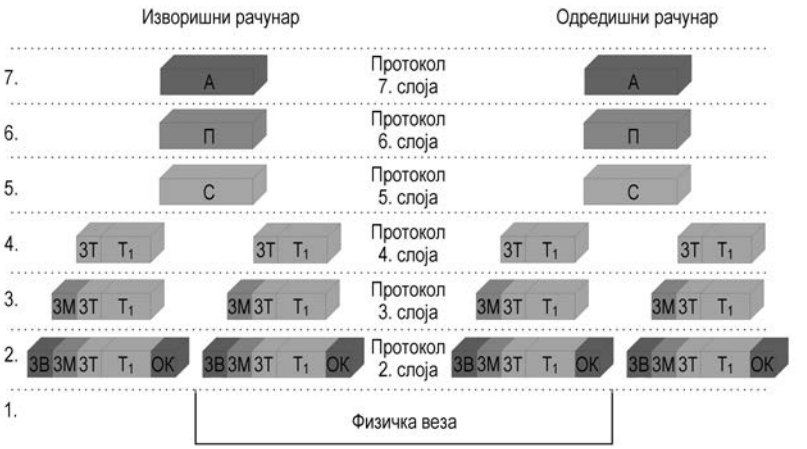
Слика 1.1 Пример комуникације истраживача – парњака

језик они укључују преводиоце (парњак - процес 2. слоја), а сваки од њих контактира секретара (парњак- процес 1. слоја).

Истраживач 1 жели да пренесе своје закључке свом парњаку. Он шаље своје закључке на српском језику као поруке преко интерфејса 3. и 2. слоја, ка свом преводиоцу који реченицу „Закључио сам...” преводи у „I have concluded....”, или на француски језик, зависно од протокола 2. слоја. Преводилац затим даје поруку свом секретару који поруку преноси телеграмом, телефоном, рачунарском мрежом или неким другим средством, зависноод тога како су се унапред договорили (протокол 1. слоја). Када порука стигне она се преводи на немачки и преноси преко интерфејса 2. и 3. слоја истраживачу 2 у Минхену. Уочимо да је сваки протокол комплетно независан од других докле год се интерфејс не мења. Преводилац се може мењати од енглеског на француски подразумевајући да се оба слажу и да ниједан не мења интерфејс ка слоју 1 или слоју 3.

Посматрајмо технички пример: како обезбедити виртуелну везу ка највишем слоју седмослојне мреже. На слици 1.2 поруку А генерисао је процес који се одвија на 7. слоју. Порука се заједно са заглављем прослеђује са 7. до 6. слоја преко интерфејса 6/7 слоја.

У овом примеру 6. слој трансформише поруку, на пример врши компресију текста, а затим прослеђује нову поруку П ка 5. слоју преко интерфејса 5/6 слоја.



Слика 1.2 Пример виртуелне везе

Слој 5, у овом примеру, не модификује поруку, већ једноставно регулише смер преноса (нпр. спречава долазеће поруке да дођу у 6. слој) док је он заузет и води рачуна о серији одлазећих порука ка 5. слоју.

У нашем примеру, као и у другим стварним мрежама, не постоји ограничење у величини поруке коју може 4. слој да прими, али постоји ограничење које поставља 3. слој. Посматрајући даље, 4. слој мора да разбије долазећу поруку Т у мање јединице, додајући заглавље (ЗТ) свакој јединици. Ово заглавље укључује управљачке информације, као што је нпр. редни број[1], да би омогућио 4. слоју на одредишном рачунару да повеже делове у правилном редоследу.

На 3. слоју свакој поруци се додаје заглавље (ЗМ) и прослеђује 2. слоју. Други слој поред заглавља (ЗВ) додаје и ознаку о крају поруке (ОК). Тако формиран рам прослеђује се физичком везом.

Важна ствар за разумевање слике 1.2 је веза између виртуелне и стварне комуникације и разлика између протокола и интерфејса. Процес парњак[2] на 7. слоју, на пример, размишља о својој комуникацији као да је „хоризонтална", користећи протокол 7. слоја. Сваки од њих има процедуру која се назива „пошаљи на другу страну" и процедуру „узми са друге

[1] Користе се и термини редоследни и секвенцијски број (*sequence number*).

[2] *Peer process*

стране", иако ове процедуре стварно комуницирају са нижим слојевима преко интерфејса 7/6 слоја, а не са другом страном.

Апстракција са процесима парњака је суштинска за пројектовање мрежа. Без ове технике не би било могуће раздвојити пројектовање комплетне мреже (нерешиви проблем) у неколико мањих целина, које се могу надгледати, пројектовати и које се називају пројектовање појединих слојева.

1.1.1 Циљеви у пројектовању слојева

Сваки слој мора да има механизам за успостављање везе. Пошто се мрежа нормално састоји од много рачунара, а неки од њих садрже више процеса, потребан је начин да се процесу на једном рачунару укаже са ким он то жели да комуницира. Адресирање је потребно у било ком слоју где постоји више одредишта.

Као што постоји механизам за успостављање везе кроз мрежу, тако постоји и механизам за раскидање везе, када она више није потребна.

Други скуп одлука у пројектовању су правила за пренос података. Подаци могу да се преносе:

- само у једном смеру, што се назива једносмерна (симплекс) комуникација,
- у оба смера (али не истовремено), што се назива наизменична двосмерна (полудуплекс[1]) комуникација, или
- у оба смера истовремено, што се назива истовремена двосмерна (потпуна дуплекс[2]) комуникација.

Протокол мора, такође, да одреди колико логичких канала вези припада и какви су приоритети. Многе мреже обезбеђују најмање два логичка канала по једној вези, на пример: један за нормалне податке а један за брзе податке, један за корисничке а један за управљачке податке, један за један а други за други смер.

Контрола грешака важна је компонента када комуникациони путеви нису савршени. Познати су многи кодови за детекцију и корекцију грешака, али оба краја везе морају се договорити који се користи.

[1] *Semiduplex* (*half duplex*)

[2] *Full duplex*

Такође, прималац мора да има начин да укаже пошиљаоцу које поруке су коректно примљене, а које нису.

Није тачно да баш сви комуникациони канали задржавају редослед порука које се шаљу. Да би се могао разрешити могући губитак редоследа, протокол мора да направи експлицитну припрему пријемнику да би омогућио да се делови могу исправно повезати. Видљиво решење је пребројавање делова, али ово решење још увек оставља отвореним питање шта треба урадити са деловима који стижу ван редоследа.

Питање које се појављује на сваком слоју је како спречити брзог пошиљаоца да преплави подацима спорог примаоца. Различита решења постоје. Сва укључују неку врсту повратне спреге од пријемника ка предајнику, директно или индиректно, зависи од тога каква је текућа ситуација пријемника.

Други проблем који се мора разрешити на различитим нивоима јесте способност процеса да прихвати произвољно дугачке поруке. Ова карактеристика доводи нас до механизма разлагања (сегментација), слања и поновног обнављања (асемблирање) поруке. Проблем који се јавља је и шта радити када процес инсистира на слању јединица података које су тако мале да је слање сваке понаособ неефикасно. Овде је решење скупити више малих порука које се шаљу ка заједничком одредишту у једну велику поруку, и раздвојити велику поруку на другом крају.

Када није згодно, или је скупо успостављање одвојених путева за сваки пар комуникационог процеса, слој испод може да одлучи да користи исти пут за више различитих веза. Ово мултиплексирање и демултиплексирање, уколико се ради транспарентно, може да користи било који слој. Мултиплексирање је потребно 1. слоју, на пример, где се сав саобраћај за све везе мора послати преко једног, или у најбољем случају неколико физичких канала.

Када постоји неколико могућих путева између изворишта и одредишта на истој тачки хијерархије, одлука о рутирању мора се донети. На пример, при слању података из Лондона у Москву одлучивање на вишем нивоу упутиће их преко Француске или Немачке, у зависности њихових закона, а одлучивање на нижем нивоу помоћи ће да се одабере расположиви канал у зависности од текућег саобраћаја.

1.1.2 Услуга и интерфејс

Функција сваког слоја је да обезбеди услугу слоју изнад себе. У овом делу анализираћемо појам услуге.

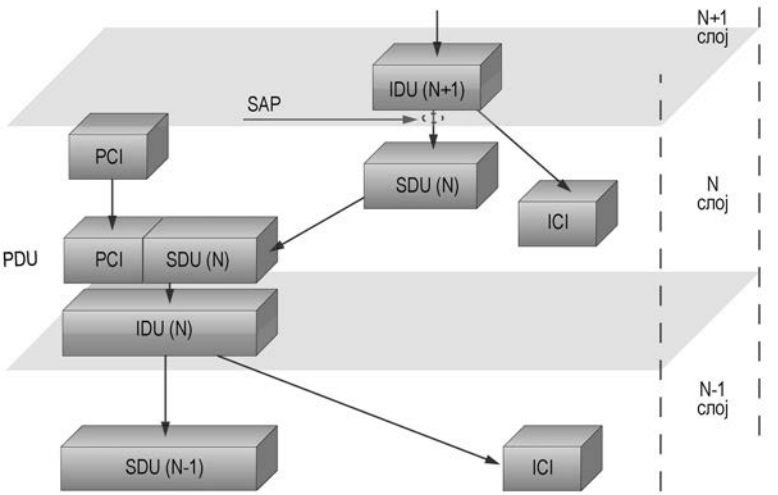
Активни елеменат у сваком слоју обично се назива целина[1]. Целина може бити софтвер (нпр. процес) или хардвер (као што је интелигентни улазно/излазни чип). Целине на истом слоју, на различитим машинама називају се парњак целине[2]. Целина у слоју N имплементира услугу коју користи слој N+1. У овом случају слој N назива се давалац[3] услуге, а слој N+1 назива се корисник услуге. Слој N може да користи слој N-1 да би обезбедио себи услугу. Могуће је обезбедити више класа услуга нпр. брза, скупа комуникација, или спора, јефтина.

Услуга се добија на тачкама приступа услузи које се означавају са SAP[4]. Тачка приступа услузи N-тог слоја је место где слој N+1 може да приступи понуђеној услузи. Свака тачка приступа има адресу која је једнозначно идентификује. Да бисмо ово разјаснили тачке приступа у телефонском систему су утичнице у које се телефони могу прикључити, а адреса тачке приступа је телефонски број утичнице. Да бисте некога позвали морате знати његову адресу тачке приступа. У поштанском систему, адреса тачке приступа је улична адреса или поштански фах. Када треба послати писмо мора се знати адресу онога коме је писмо упућено.

Да би два слоја размењивала информације мора да постоји усаглашени договор око скупа правила везаних за интерфејс. Интерфејс целине слоја N+1 преноси јединице података интерфејса (IDU[5]) ка N-том слоју преко тачке приступа (SAP), као што је приказано на слици 1.3. Јединица података интерфејса (IDU) састоји се од јединице података услуге (SDU[6]) и управљачких информација (ICI[7]). Јединица података услуге је информација која се прослеђује преко мреже ка парњак целини и затим ка N+1 слоју.

---

[1] *Entity*
[2] *Peer entities*
[3] *Provider*
[4] *Service Access Points*
[5] *Interface Data Unit*
[6] *Service Data Unit*
[7] *Interface Control Information*



Слика 1.3 Пренос јединица података између слојева

Управљачке информације потребне су нижим слојевима да би обавиле функцију као што је нумерисање бајтова у јединици података услуге али нису део самих података. Да би пренела јединицу података услуге, целина слоја може да је дели у неколико делова. Сваки од тих делова може да има своје заглавље (PCI[1]) и да буде послат као посебна јединица података протокола (PDU[2]). Заглавље јединица података протокола (PCI) користе парњак целине да би реализовале свој парњак протокол. Оно садржи информације о томе која јединица протокола садржи податке, која садржи управљачке информације, која садржи редне бројеве, итд.

1.1.3 Услуге са и без успоставе везе

Слојеви нуде два различита типа услуга ка слоју изнад њих: са успоставом везе[3] и без успоставе везе[4].

Услуга са успоставом везе моделирана је по узору на телефонски систем у коме да бисте са неким разговарали подижете слушалицу,

[1] *Protocol Control Interface – Header*
[2] *Protocol Data Unit*
[3] *Connection Oriented*
[4] *Connectionless*

окрећете број, разговарате и на крају разговора спуштате слушалицу. Слична је употреба услуге са успоставом везе: корисник услуге прво успоставља везу, користи везу и раскида је. Суштински веза функционише као цев: пошиљалац шаље објекте (битове) на једном крају, а пријемник их преузима у истом редоследу на другом крају.

Услуга без успоставе везе моделирана је по узору на поштанску услугу. Свака порука (писмо) садржи комплетну адресу и свака је упућена (рутирана) независно од осталих. Нормално, када се две поруке шаљу на исто одредиште, прва која се пошаље прва ће и стићи. Може се десити да прва порука која се пошаље закасни тако да друга стигне пре ње. Са системом са успоставом везе ово не може да се деси.

Свака услуга може се окарактерисати преко квалитета услуге (QoS[1]). Неке услуге су поуздане у смислу да је вероватноћа губитка података занемарљива. Обично се, за поуздану услугу користи потврда[2], тако да је пошиљалац сигуран да је порука стигла на одредиште. Процес потврде уводи додатне податке (потврду, поновно слање[3]) и додатно кашњење, које је некада вредно тога а некада је непожељно.

Пренос датотека је типична ситуација где је поуздана услуга са успоставом везе неопходна. Власник датотеке жели да буде сигуран да су сви битови тачно примљени и у истом редоследу у коме су и послати. Мали је број корисника који би пристао да дође до мешања или губитка само пар битова, без обзира што би процес преноса био бржи.

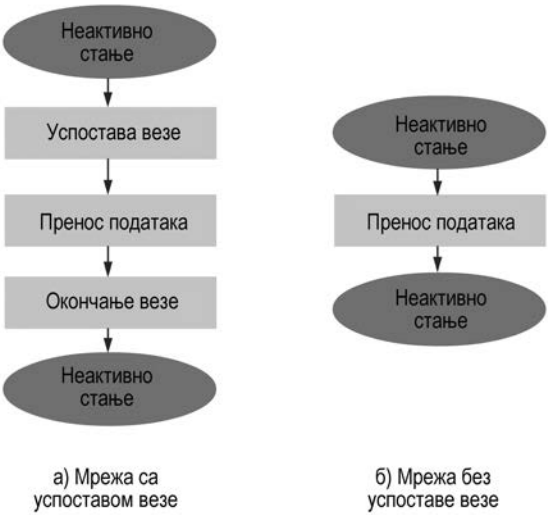
У неким применама кашњења која се уносе при слању потврде су неприхватљива. Једна таква апликација је пакетизовани говор. За корисника телефона боље је да има шум са линије него уношење кашњења које се добија коришћењем потврде. Исто је и са преносом пакетизоване живе слике.

Постоје апликације којима није потребна успостава веза. На пример електронска пошта. Потребно је послати поруку и велика је вероватноћа да ће она бити примљена, али се не гарантује. Пакетска порука којом

[1] *Quality of Service*

[2] Позитивна и негативна

[3] *Retransmission*



Слика 1.4 Карактеристике услуга са успоставом и без успоставе везе

се шаље целокупна информација са адресама пошиљаоца и примаоца назива се датаграм.

1.1.4 Примитиве услуге

Услуга се формално специфицира преко скупа примитива (операција) које су доступне кориснику или другој некој целини и преко којих он приступа услузи. Ове примитиве налажу целини која пружа услугу да изврши неку активност, или да извести о некој активности коју су парњак целине обавиле. Један од начина да се класификују примитиве услуге је да их групишемо у четири класе, као што је приказано у табели 1.1.

Примитиве	**Значење**
Request - Захтев	Корисник тражи од целине која нуди услугу да одради неки посао
Indication - Индикација	Корисник је информисан о догађају
Response - Одговор	Одговор Корисник жели да одговори на догађај
Confirmation - Потврда	Одговор на претходно послати захтев је стигао

Табела 1.1 Четири класе примитива услуга

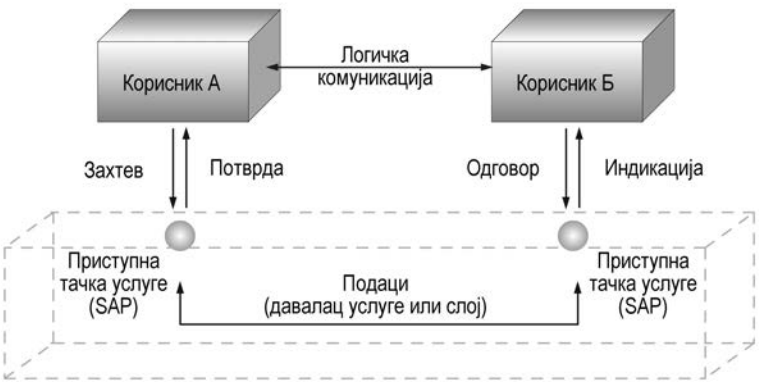
Да бисмо илустровали употребу примитива, размотримо како се веза успоставља и раскида. Целина која је иницијатор извршава примитиву *Request*, која резултује слањем јединице података протокола. Пријемник добија примитиву *Indication* (обавештење) да нека целина негде жели да успостави са њим везу. Користи *Response* примитиву која указује да ли ће да прихвати или одбаци позив. Целина која је иницирала *Request* примитиву проналази шта се дешава посредством *Confirm* примитиве (слике. 1.5 и 1.6).

Примитиве могу да имају параметре и многе од њих и имају. Параметри за *Request* примитиву могу да специфицирају рачунар са којим треба успоставити везу, тип услуге који се жели и максималну величину поруке која се у вези користи.

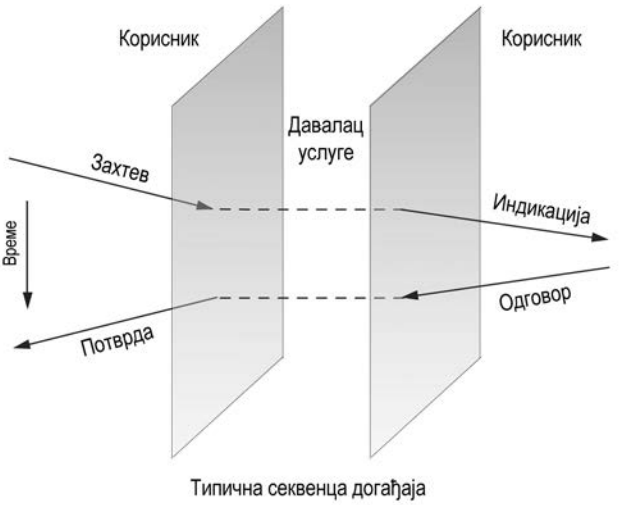
Параметар за *Indication* примитиву може да садржи идентитет позивајуће целине, тип услуге који се жели и препоручену максималну величину поруке. Уколико се позвана целина не слаже са препорученом максималном величином поруке, може да направи другачији предлог у *Response* примитиви, која ће бити доступна пошиљаоцу преко *Confirm* примитиве. Детаљи договора су део протокола.

1.1.5 Релација између услуге и протокола

Услуге и протоколи имају различите концепте, који се често мешају. Услуга је скуп примитива (операција) које слој обезбеђује слоју изнад себе.



Слика 1.5 Успостављање логичке везе



Слика 1.6 Размена примитива

Услуга дефинише које операције је слој спреман да изврши на захтев корисника, али не указује ничим како су ове операције примењене. Услуга је везана за интерфејс између два слоја, са нижим слојем који је давалац услуге и вишим слојем који је корисник услуге.

Протокол, на другој страни, је скуп правила која одређују формат и значење рамова, пакета или порука које се размењују преко парњак целина у оквиру слоја. Целине користе протокол у циљу имплементације дефинисане услуге. Могу слободно да мењају протокол али не могу да мењају услугу која је видљива њиховом кориснику. На овај начин услуга и протоколи потпуно су раздвојени.

Направићемо аналогију са језиком за програмирање. Услуга је као апстрактни тип података или објекат у објектно - оријентисаним језицима. Он дефинише операције које треба да се изврше над објектима, али не специфицира како су ове операције имплементиране. Протокол се односи на имплементацију услуге и као таквог га корисник услуге не види.

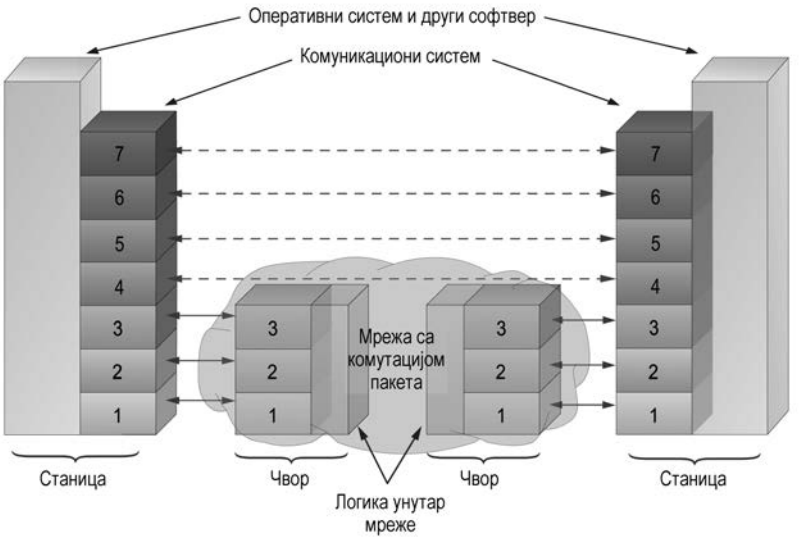
Пошто смо размотрили слојевитост на апстрактан начин, у овом поглављу, анализираћемо три важне мрежне архитектуре, OSI, X.25 и TCP/IP референтне моделе.

1.2 OSI референтни модел

OSI модел (слика 1.7) базиран је на препорукама које је развила интернационална организација ISO. Модел се назива ISO OSI референтни модел за међусобну комуникацију отворених система[1] зато што се бави повезивањем отворених система – система који отворено комуницирају са другим системима.

1.2.1 Процес укалупљивања

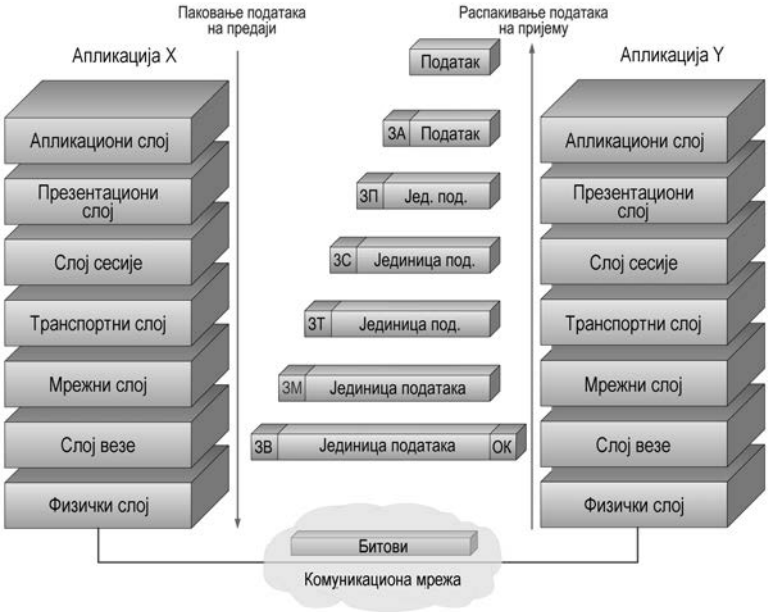
Слика 1.8 приказује OSI референтни модел и употребу јединице података у оквиру OSI референтног модела. Када апликација X има податке за апликацију Y она их шаље целини (процесу) на апликационом слоју. Подацима се додаје заглавље ЗА (тзв. процес укалупљивања[2]) које садржи управљачке информације потребне парњаку, 7. слоју на одредишту. Оригинални подаци заједно са заглављем сада се прослеђују 6. слоју. Презентациони слој посматра јединицу података као јединствену целину и додаје своје заглавље ЗП (друго укалупљивање).



Слика 1.7 OSI референтни модел

[1] *Open System Interconnection reference model*

[2] Користи се и термин учауривања (*encapsulation*).



Слика 1.8 Ток података (укалупљивање)

Овај процес се наставља све до слоја везе (линка) који додаје и ознаку о почетку (ЗВ) и крају[1] (ОК) јединици података. Јединица података 2. слоја, која се назива рам (оквир) преко физичког слоја прослеђује се на трансмисиони медијум. Када рам стигне до одредишног система на њему се одвија процес обрнутог редоследа. Сваки слој примивши јединицу података уклања управљачке информације (заглавље) њему упућене а остатак прослеђује вишем слоју.

1.2.2 Физички слој

Физички слој дефинише физички интерфејс између уређаја и правила по којима се сигнали преносе од једног до другог. Физички слој има четири важне карактеристике:

- механичку,
- електричну,
- функционалну,
- процедуралну.

[1] *Trailer*

Пример стандарда за физички слој је EIA[1] 232-F.

1.2.3 Слој везе

Док физички слој обезбеђује само услугу која се састоји у формирању сигнала (нпр. као серије битова), задатак слоја везе[2] је да обезбеди да веза буде и поуздана. Слој везе обезбеђује и начин да се успостави, одржава и раскине веза. Основна услуга, коју обезбеђује слој везе вишим слојевима, је детекција грешке и управљање везом. Тако, са потпуно функционалним протоколом слоја везе, следећи виши слој може претпоставити виртуелни пренос без грешака. У ствари, уколико је комуникација између два система који нису директно повезани, веза између тих система састоји се од већег броја физичких веза (линкова), при чему свака физичка веза функционише независно. На тај начин виши слојеви нису ослобођени одговорности за контролу грешке.

1.2.4 Мрежни слој

Основна услуга мрежног слоја је да обезбеди механизме за транспарентан пренос података између целина транспортног слоја (транспортних целина). Мрежни слој ослобађа транспортни слој потребе да „зна“ било шта о преносу података или техникама комутације које се користе да би се повезали системи. Услуга мрежног слоја (мрежна услуга) је одговорна за успостављање, одржавање и раскидање везе преко комуникационих путева који су у ту везу укључени.

То је тачка у којој је концепт протокола помало нејасан. Ово је најбоље илустровано на слици 1.8 која показује две станице које међусобно комуницирају не преко директног линка већ преко мреже са комутацијом пакета. Станице имају директне линкове ка мрежним чворовима. Протоколи 1. и 2. слоја су протоколи станице-чвора (локални). Јасно је да су протоколи од 4. до 7. слоја протоколи између 4. до 7. целине двеју станица. Протокол 3. слоја је помало од оба.

Принципијелни дијалог одвија се између две станице и чвора; станица шаље адресиране пакете ка чвору за прослеђивање ка мрежи. Чвор захтева комуникацију преко виртуелног кола, користи везу да би пренео

[1] Раније означен као RS-232.

[2] *Data Link*

податке, и завршава везу. Ово је све урађено протоколом станица-чвор. У ствари, пошто се пакети размењују и виртуелни канал се успоставља између две станице, постоји такође и аспект протокола станица-станица.

Постоји широки спектар комуникационих карактеристика које се јављају, а које може да подржава мрежни слој. На једном крају најједноставнија је директна веза између станица. У овом случају, може да постоји мало, или нимало потребе за мрежним слојем, пошто слој везе може да извршава неопходне функције за одржавање везе између крајњих тачака везе.

Уобичајена услуга 3. слоја је да води рачуна о детаљима коришћења комуникационе мреже. У овом случају мрежна целина у станици мора да обезбеди мрежу потребним информацијама о комутирању и рутирању података ка другој станици.

1.2.5 Транспортни слој

Слој 4 и слојеви изнад у OSI моделу генерално се називају виши слојеви. Протоколи на овим слојевима су с краја - на крај[1] и не воде рачуна о мрежи. Намена 4. слоја је да обезбеди поуздан механизам размене података између процеса на различитим системима.Транспортни слој гарантује да се јединице података прослеђују без грешака, по правилном редоследу, без губитка или удвостручавања. Транспортни слој може такође да води рачуна о оптимизацији коришћења услуге мрежног слоја и обезбеђивању захтеваног квалитета услуге за целине слоја сесије. На пример, целина сесије може да специфицира прихватљиву вероватноћу грешке, максимално кашњење, приоритет и поузданост. Практично, транспортни слој служи као корисничка веза са комуникационим уређајима.

Величина и комплексност транспортног протокола зависе од типа услуге коју може да добије од 3. слоја. За поуздан рад 3. слоја, са виртуелним каналом, потребне су минималне функције 4. слоја. Уколико је 3. слој непоуздан, или само подржава рад са датаграмима[2], протокол

---

[1] *End to end, E2E*

[2] *Datagram* - јединица података протокола која садржи адресу изворишта и одредишта, податке и која се потпуно независно може преносити кроз рачунарску мрежу. Представља основну јединицу преноса без успоставе везе (*connectionless*).

4. слоја укључиће механизме за детекцију и опоравак од грешке. ISO је дефинисао четири класе транспортног протокола. Свака обезбеђује различите услуге.

1.2.6 Слој сесије

Слој сесије обезбеђује механизам за дијалог између апликација. Минимално, слој сесије обезбеђује начин на који ће два апликациона процеса да успоставе и користе везу која се назива сесија. Поред тога он може да обезбеди следеће услуге:

- тип дијалога који може да буде двострано симултан, двострано наизменичан или једносмеран;
- опоравак од грешке је последица тога што слој сесије може да обезбеди механизам за проверу. Уколико се неки тип грешке појави између контролних тачака слој сесије може поново да пошаље све податке од последње контролне тачке.

1.2.7 Презентациони слој

Слој презентације (презентациони слој) бави се синтаксом података који се размењују између апликационих целина. Његова намена је да разреши разлике у формату и репрезентацији података.

Презентациони слој дефинише синтаксу која се користи између апликационих целина и обезбеђује одабир даљих модификација репрезентација које ће се користити.

Пример презентационог протокола је телетекст, видеотекст, шифрирање и протокол виртуелног терминала.

1.2.8 Апликациони слој

Слој апликације (апликациони слој) обезбеђује апликационом процесу приступ OSI окружењу. Овај слој садржи функције управљања и генерално корисне механизме за подршку дистрибуираним апликацијама. Пример протокола на овом слоју је протокол за пренос, приступ и управљање датотекама (FTAM[1]).

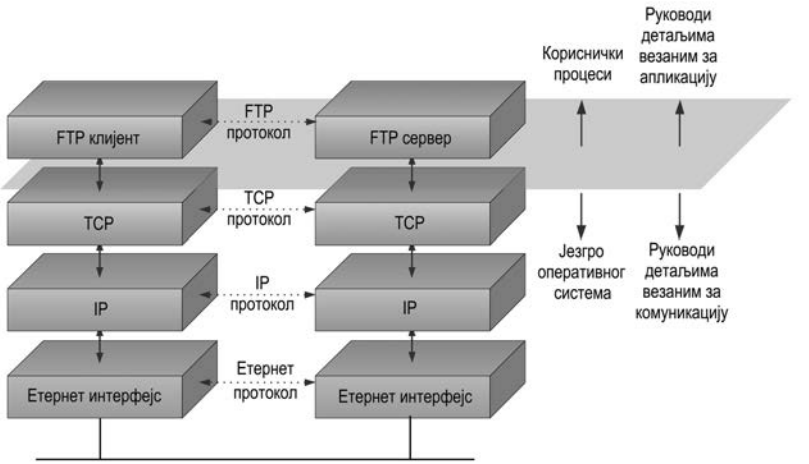
[1] *File Transfer Access and Management*

1.3 TCP/IP референтни модел

TCP/IP[1] је четворослојни референтни модел[2] чији је један пример приказан на слици 1.9. Развила га је 1960. године агенција DARPA[3] за мрежу са комутацијом пакета. Постао је начин за повезивање рачунара који се највише користи, тј. *de facto* стандард. TCP/IP протоколи су основа Интернета.

Скуп TCP/IP протокола је комбинација различитих протокола са различитих слојева. Сваки од слојева има различита задужења:

- слој везе (среће се у литератури и под називом приступ мрежи) води рачуна о свим детаљима физичког медијума који се користи;
- мрежни слој (среће се и под називом интернет слој) води рачуна о кретању пакета кроз мрежу. Преусмеравање (рутирање) пакета извршава се на овом слоју. Протоколи IP, ICMP и IGMP чине мрежни слој у TCP/IP скупу протокола;



Слика 1.9 TCP/IP референтни модел

[1] *Transmission Control Protocol/Internet Protocol*

[2] По стандарду RFC1122 и у литератури [9] и [10]. У литератури [5] број слојева је 5.

[3] *Defense Advanced Research Projects Agency*. Агенција Министарства одбране САД одговорна за развој нових технологија за војску САД. DARPA је учествовала у развоју технологија које су умногоме промениле свет, укључујући рачунарске мреже (ARPANET - *Advanced Research Projects Agency Network*), основе глобалног позиционог система (GPS - *Global Positioning System*), развој вештачке интелигенције (AI - *Artificial intelligence*) и многе друге.

- транспортни слој апликационом слоју изнад себе обезбеђује проток података између две станице (рачунара). У TCP/IP скупу протокола најважнија два протокола су:
  - TCP протокол који обезбеђује поуздан проток података између рачунара, најчешће преко виртуелних веза. Од података које добија од апликације: формира целине (делове) које прослеђује мрежном слоју, шаље потврде о успешном пријему пакета од удаљених рачунара, поставља времена на часовницима, итд.,
  - UDP[1] протокол обезбеђује много једноставнију услугу апликационом слоју. Он једноставно шаље од једног до другог рачунара пакете података (датаграме). Све захтеве за поузданошћу мора да преузме апликациони слој;
- апликациони слој води рачуна о карактеристичним детаљима сваке апликације. У скоро свим имплементацијама обезбеђени су протоколи за следеће апликације:
  - Telnet - за удаљени приступ,
  - FTP[2] - за пренос датотека,
  - SMTP[3] - за електронску пошту,
  - SNMP[4] - за надзор и управљање рачунарском мрежом.

У поглављима које следе биће детаљно обрађени протоколи TCP/IP референтног модела.

1.4 ATM[5] референтни модел

ATM је систем са комутацијом и преносом пакета који је тачно одређене величине од 53 бајта. Назива се и систем са комутацијом ћелија. На слици 1.10 приказан је ATM референтни модел.

Стандардје донела интернационална организација ITU-T[6] а референтни модел се разликује и од OSI и од TCP/IP референтних модела. На слици 1.12 представљен је ATM референтни модел. Физички слој води рачуна о

[1] *User Datagram Protocol*
[2] *File Transfer Protocol*
[3] *Simple Mail Transfer Protocol*
[4] *Simple Network Management Protocol*
[5] *Asynchronous Transfer Mode*
[6] *International Telecommunication Union - Telecommunication Standardization Sector*

Слика 1.10 ATM референтни модел

специфичностима трансмисионог медијума и техникама кодирања.

Два слоја референтног модела односе се на ATM функције: ATM слој заједнички за све услуге које омогућавају пренос и слој за прилагођавање на ATM слој (AAL[1]). ATM слој дефинише пренос података помоћу ћелија и дефинише начин коришћења виртуелних (логичких) веза.

Намена адаптационог слоја је да обезбеди да и други (не само ATM) протоколи користе ATM системе. AAL слој повезује информације виших слојева у ATM ћелије (на изворишној страни), усмерава пренос ћелије преко ATM мреже и на одредишној страни преузима информације из ATM ћелија и испоручује их вишим слојевима.

Референтни модел је тродимензионалан и сачињавају га следеће одвојене равни:

- корисничка раван - обезбеђује кориснику све информације потребне за пренос података заједно са управљачким информацијама као што су контрола тока, контрола грешке итд...
- контролна раван - извршава све операције везане за управљање позивом и везом,
- управљачка раван и управљање слојевима - воде рачуна о функцијама везаним за надзор ресурса и координацију између слојева.

[1] *ATM Adaptation Layer*

1.5 X.25 референтни модел

Да би се спречило да свака од земаља прави међусобно некомпатибилне интерфејсе, интернационална организација ITU-T[1] донела је стандарде за протоколе за приступ мрежи са комутацијом пакета. Заједно су ови стандарди познати као X.25 и представљају најстарију пакетску технику. Стандард X.25 дефинише (слика 1.12):

- интерфејс између опреме крајње станице[2] која се означава као опрема крајње станице DTE[3] и опреме носиоца (оператера) која се означава DCE[4]. Опрема чвора се означава са DSE[5];
- формат и значење информација које се размењују преко DTE - DCE интерфејса и то за протоколе слојева 1, 2 и 3 (слика 1.12). Пошто овај интерфејс раздваја опрему носиоца (DCE) и опрему корисника (DTE) важно је да је интерфејс пажљиво дефинисан;
- три слоја комуникације: физички слој, слој везе и слој пакета.

Физички слој се бави физичким интерфејсом између крајњих станица (рачунара, терминала) које се прикључују на мрежу и физичке везе (линка) која повезује станице са пакетском мрежом. У реализацији интерфејса користе се спецификације за физички слој X.21 или EIA -232.

Слика 1.11 Упоређивање X.25 и OSI референтних модела

[1] *International Telecommunication Standardization Sector* преименована 1993. године од организације *International Telegraph and Telephone Consultative Committee* (*CCITT* на француском: *Comité Consultatif International Téléphonique et Télégraphique*) која је основана 1956. године.

[2] Користи се и термин хост (*host*).

[3] Опрема терминала за податке (*Data Terminal Equipment*)

[4] *Data Circuit terminating Equipment*

[5] *Data Switching Exchange*

Слој везе обезбеђује поуздан пренос података преко физичког линка. На слоју везе користи се протокол за балансирани приступ (LAPB[1]) који је сличан HDLC[2] протоколу[3].

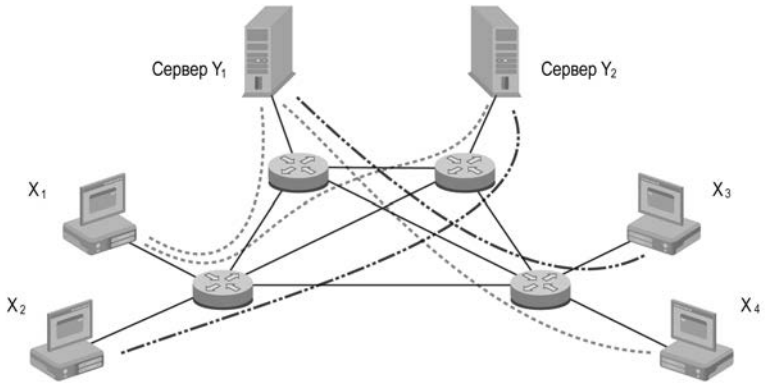
Пакетски слој обезбеђује услугу успостављања и одржавање логичке везе - виртуелног канала између два корисника.

У примеру са слике1.12 станице (персонални рачунари) успоставиле су са серверима логичке везе и комуницирају преко одговарајућих виртуелних канала: станице X_1, X_3 и X_4 комуницирају са сервером Y_1, станице X_1 и X_2 комуницирају са сервером Y_2.

1.6 Штафетни пренос рамова

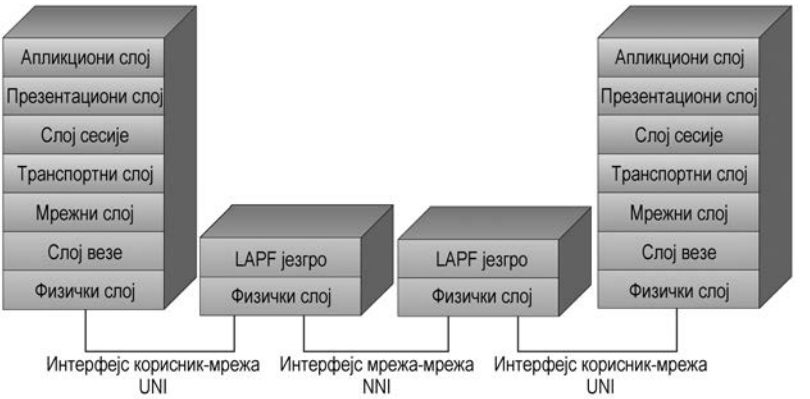
Данашњи преносни системи на великим растојањима[4] највише су ослоњени на оптичке каблове и значајно мање подложни сметњама него мреже пре двадесетак година. Као резултат тога примењен је нови концепт комутације - штафетни пренос рамова (FR[5]) или комутација рамова.

Основна идеја комутације рамова је следећа: корекције грешака и контрола протока се из чворова мреже (у којима су имплементирани само



Слика 1.12 Начин употребе виртуелног канала

[1] *Link Access Protocol Balanced*
[2] *High Level Data Link Control*
[3] Детаљније објашњење дато је у литератури [1], 11. поглавље.
[4] *Long distance digital transmission system*
[5] *Frame Relay*



Слика 1.13 Упоређивање FR и OSI референтних модела на интерфејсима корисник- мрежа[1] и мрежа-мрежа[2]

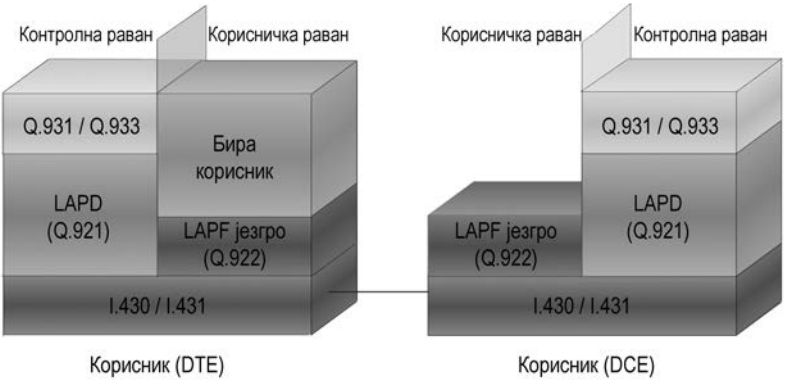
протоколи прва два слоја: физичког и слоја везе LAPF[3]), пребацују у крајње станице мреже (слика 1.13). Крајње станице као што су персонални рачунари, сервери, велики рачунари[4] користећи протоколе виших слојева (од 3. до 7.) могу да обезбеде сопствену контролу грешака. Овај концепт одлично функционише док су грешке у преносним системима мале. Уколико се овај, основни предуслов, промени онда долази до поновног слања података кроз целу мрежу уместо од једног до другог чвора (комутатора).

Циљ мрежа са штафетним преносом рамова је да омогуће ефикаснији пренос података од X.25 мрежа са комутацијом пакета. За X.25 мреже карактеристично је да:

- управљачки пакети који се користе за успоставу и раскидање виртуелних канала преносе се истим виртуелним каналима као и пакети података[5],
- мултиплексирање виртуелних канала обавља се на 3. слоју,
- и 2. и 3. слој користе механизме за контролу тока и грешака.

Препорука I.233 (ITU-T) специфицира употребу штафетног преноса рамова за брзине приступа од 2Mb/s. Користе се и веће брзине приступа.

[1] UNI (*User to Network Interface*)
[2] NNI (*Network to Network Interface*)
[3] *Link Access Procedure for Frame Relay*
[4] *Mainframe* – мејнфрејм
[5] *In band signaling*



Слика 1.14 Стандарди у мрежи са штафетним преносом рамова[1]

На слици 1.14 приказана је слојевита архитектура протокола која се користи у мрежама са штафетним преносом рамова. Уочавају се две одвојене равни:

- контролна раван која се користи код успостављања и окончавања логичке везе,
- корисничка раван која је одговорна за пренос података између претплатника.

На тај начин протоколи контролне равни су између претплатника и мреже, а протоколи корисничке равни су између крајњих корисника[2].

У табели 1.2 дата је намена стандарда означених на слици 1.14.

Намена стандарда	**ITU-I стандард**	**ANSI стандард**
Архитектура и опис услуга	I.233	T1.606
Слој за податке	Q.992	T1.618
Управљање перманентним виртуелним каналом (PVC[2])	Q.933	T1.617
Регулисање загушења	I.370	T1.606a
Сигнализација комутираним виртуелним каналом (SVC[3])	Q.933	T1.617

Табела 1.2 Ознаке и намене стандарда код мрежа са комутацијом рамова

[1] LAPD (*Link Access Procedures, D channel*)
[2] *End to end*
[3] *Permanent Virtual Circuit*
[4] *Switched Virtual Circuits*

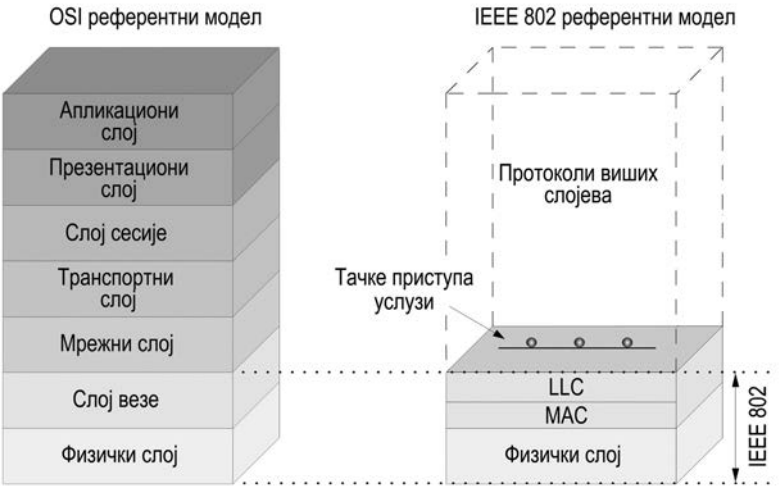
1.7 Питања и задаци

1. Који су кључни елементи сваког протокола?
2. Шта се све постиже стандардизацијом протокола?
3. Како се дефинише и реализује протокол N-тог слоја?
4. Какаву намену има интерфејс у слојевитој архитектури протокола?
5. Описати циљеве у пројектовању слојева.
6. Како се дефинише појам услуге у модулу слојевите архитектуре?
7. Описати механизам укалупљивања у OSI референтном моделу.
8. Како се најједноставније може описати OSI референтни модел?
9. Колико слојева је дефинисано у OSI референтном моделу? Набројати их и описати функцију сваког од њих.
10. Описати разлику између протокола и услуге.
11. Дефинисати појам примитиве. Колико врста примитива је дефинисано? Набројати их.
12. ICI је скраћеница од:
    а) *Interface Conrol Information*
    б) *Interface Circuit Information*
    в) *Integration Circuit Information*
13. PDU је скраћеница за:
    а) *Paging Domen Unit*
    б) *Protocol Digital Uplink*
    в) *Protocol Data Unit*
    г) *Partial Domen Unit*
14. Колико слојева је дефинисано TCP/IP референтним моделом? Набројати их и описати намену сваког од њих.
15. Који слојеви TCP/IP референтног модела су део оперативног система?
16. Описати разлику измеђуTCP и UDP протокола.
17. Навести неке протоколе апликационог слоја.
18. Укратко описати на шта се односи X.25 стандард.
19. Шта се све може преносити ATM мрежом?
20. Где се обављају корекције грешака и контрола протока у мрежама са штафетним преносом рамова?

2. Слој везе података

Други слој OSI референтног модела је слој везе података[1] или само слој везе. На том слоју реализоване су основне функције многих жичних и бежичних мрежа. На пример: Етернет (IEEE802.3), IEEE802.11, IEEE802.15 и IEEE802.16 означавају се као технологије слоја везе. Скуп уређаја повезаних на слоју везе уобичајено се сматра једноставно мрежом.

Слој везе је концепцијски подељен у два подслоја: управљање логичком везом (LLC[2]) и управљање приступом медијуму (MAC[3]). Раздвајање на два подслоја засновано је на архитектури која се користи у IEEE[4] пројектима (слика 2.1). Раздвајањем функција LLC и MAC подслојева умногоме је олакшан међусобни рад различитих мрежних технологија.



Слика 2.1 Однос архитектуре IEEE802 и OSI

[1] *Data link layer*
[2] *Logical Link Control*
[3] *Media Access Control*
[4] *Institute of Electrical and Electronics Engineers*

LLC подслој обезбеђује функције које су потребне за успоставу и управљање логичком везом између локалних уређаја у мрежи. Он обезбеђује услуге слоју мреже (изнад себе) и штити више слојеве од детаља везаних за различите технологије преноса које слој везе реализује. Већина технологија локалних рачунарских мрежа користи IEEE802 LLC протокол.

Функције које реализује слој везе су следеће:

- MAC подслој. Обезбеђује процедуре које користе уређаји да би управљали приступом трансмисионом медијуму. Пошто многе мреже користе дељене трансмисионе медијуме неопходно је да постоје и да се примењују правила која спречавају да дође до истовременог преноса података са више станица а самим тим и сукобљавања на медијуму;
- Прављење рамова[1]. Слој везе је задужен за завршно укалупљивање[2] јединица података виших слојева који се шаљу у мрежу преко физичког слоја;
- Адресирање. Слој везе је најнижи слој OSI референтног модела који води рачуна о адресама: ознаке које указују на одређену локацију изворишта и одредишта. Сваки уређај на мрежи има јединствен број, који се уобичајено назива хардверска или MAC адреса. Користи је протокол слоја везе да обезбеди да подаци намењени одређеном уређају до њега и стигну;
- Откривање и вођење рачуна о грешкама. Слој везе води рачуна о грешкама које се јављају на најнижем слоју скупа протокола. На пример циклична редундантна провера (CRC[3]).

Физички слој и слој везе међусобно су уско повезани. Најчешће су спецификације за физички слој део спецификације слоја везе одређене технологије. Најбољи пример су Етернет и IEEE802.3 стандарди који садрже и спецификацију слоја везе и спецификацију различитих физичких слојева.

Пошто су слој везе и физички слој тако уско међусобно повезани велики број хардверских уређаја се на њих односи. На мрежним

[1] *Data framing*

[2] *Encapsulation*

[3] *Cyclic Redundancy Check*

интерфејсним картицама (NIC[1]) је имплементирана одређена технологија слоја везе па се често називају етернет картице, WiFi картице. Оне такође представљају уређаје за међусобно повезивање који раде на другом слоју пошто одлучују о томе шта треба урадити са подацима које добијају анализирајући адресу из рама података. Неке од највише коришћених технологија и протокола су: Етернет, IEEE802.3, IEEE802.16 (WiMAX), IEEE802.11 (WiFi), IEEE802.15 (Bluetooth, ZigBee) и PPP[2].

2.1 Методи приступа трансмисионом медијуму

Уколико исти медијум треба да користи (дели) више радних станица неопходно је да постоји начин који регулише приступ више станица трансмисионом медијуму. Дефинишу се два приступа: статички (FDM[3] и TDM[4]) и динамички (Aloha[5], CSMA[6], мреже са жетоном...).

Највише коришћени представник динамичких метода приступа трансмисионом медијуму је вишеструки приступ са ослушкивањем носиоца.

2.1.1 Вишеструки приступ са ослушкивањем носиоца

CSMA (вишеструки приступ са ослушкивањем носиоца) је метод приступа каналу који се користи да би се умањила могућност сукоба. Идеја је да када станице желе да шаљу податак испитују (ослушкују) да ли је неко заузео трансмисиони медијум (кабл). Ако је трансмисиони медијум заузет, станица чека да се он ослободи. Ако је слободан одмах започиње слање. Уколико две или више станица истовремено почну да шаљу преко слободног трансмисионог медијума, долази до сукоба. Могућност конфликта (судара) постоји у кратком временском интервалу који се назива време рањавања[7] (време рањивости) и једнак је максималном времену пропагације између најудаљенијих корисника.

---

[1] *Network Interface Card*

[2] *Point to Point Protocol*

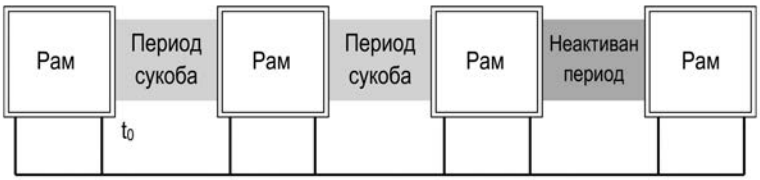
[3] FDM (*Frequency Division Multiplex*) - фреквенцијски мултиплекс

[4] TDM (*Time Division Multiplex*) - временски мултиплекс

[5] *Aloha* (здраво на Хавајском) - станица шаље када има пакет за слање.

[6] CSMA (*Carrier Sense Multiple Access*) - станица прво ослушкује преносни медијум, ако је слободан шаље.

[7] *Vulnerable time*



Слика 2.2 Временски периоди у слању код CSMA/CD метода

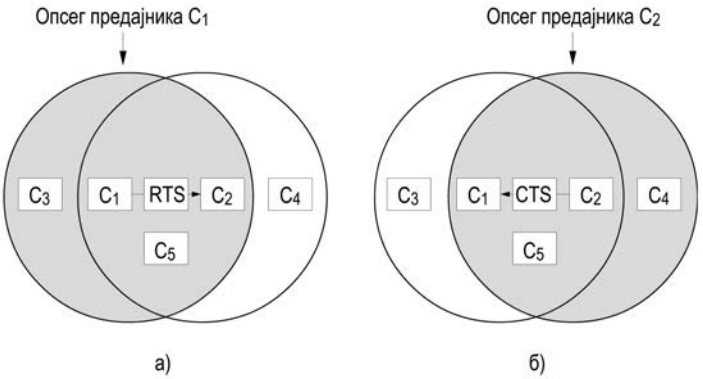
Метод CSMA/CD[1] (ослушкивање носиоца, вишеструки приступ и откривање судара) широко је распрострањен у локалним рачунарским мрежама. Припада подслоју за приступ трансмисиони медијуму (MAC[2] подслоју). Као и многи други протоколи у локалним рачунарским мрежама користи модел који се може објаснити помоћу слике 2.2. У тренутку означеном са t_0 станица завршава слање свог рама. Било која друга станица која има рам за слање може сада да покуша да то уради. Уколико две или више станица покушају то даураде, доћи ће до колизије тј. судара. По детектовању колизије станица прекида слање, чека случајни период времена и поново покушава са слањем. То значи да модел CSMA/CD чине периоди слања, сукобљавања и неактивни периоди.

2.1.2 Протокол за бежичне локалне рачунарске мреже

Вишеструки приступ са избегавањем колизије (CSMA/CA[3]) први је протокол пројектован за бежичне локалне рачунарске мреже. Основна идеја је да пошиљалац шаље примаоцу кратак рам, тако да га станице које су у близини могу да детектују и „избегну" слање за време слања долазећег, великог рама података. Протокол CSMA/CA је илустрован на слици 2.3.

Посматрајмо како станица C_1 шаље рам ка станици C_2. Започиње слањем рама „захтев за слањем" (RTS[4]) ка станици C_2 (приказано је на слици 2.3а). Овај кратак рам (30 бајтова) садржи дужину рама података који треба да буде послат. Станица C_2 одговара рамом „спреман за пријем" (CTS[5]). У раму CTS налази се дужина рама података (ископираног из рама

[1] *Carrier Sense Multiple Access with Collission Detection*
[2] Управљање приступа медијуму - *Medium Access Control*
[3] *Carrier Sense Multiple Access with Collision Avoidance*
[4] *Request To Send*
[5] *Clear To Send*



Слика 2.3 CSMA/CA протокол: а) C_1 шаље RTS ка C_2, б) C_2 одговара са CTS ка C_1

RTS). По пријему рама CTS, станица C_1 започиње слање података. Како станице које „чују" рамове RTS и CTS реагују? Свака станица која детектује („чује") рам RTS довољно је близу станице C_1 и мора да остане „мирна" толико дуго да може рам RTS да стигне до станице C_2 без колизије. Свака станица која „чује" рам CTS је близу станице C_2 и мора да остане „мирна" док се не заврши слање података дужине која се може видети из рама CTS.

На слици 2.3 станица C_3 је у опсегу станице C_1, али није у опсегу станице C_2. Због тога „чује" рам RTS од станице C_1, али не и рам CTS од станице C_2. С друге стране станица C_4 је у опсегу станице C_2 али не и станице C_1. Она не чује рам RTS али „чује" рам CTS. Њој пријем рама CTS указује да је близу станице која треба да прими рам, тако да одлаже слање било чега за време трајања рама података. Станица C_5 „чује" обе управљачке поруке, и као и станица C_4 остаје „мирна" док се се не заврши слање рама података. И поред свих мера предострожности до колизије може да дође. На пример станице C_2 и C_3 могу да пошаљу истовремено рам RTS ка станици C_1. Доћи ће до судара рамова (колизије). У случају колизије пошиљалац рама RTS (онај који не добије одговор - рам CTS после одређеног времена) чека одређени временски интервал одабран по случајној расподели и покушава касније. Алгоритам који се користи је бинарни експоненцијални[1].

[1] *Binary exponential backoff* – алгоритам који се користи да распореди поновљене ретрансмисије истог блока података као део процеса избегавања загушења у мрежи. Користи се и код CSMA/CD протокола.

2.2 Формати рамова Етернет II и IEEE802.3

Без обзира што етернет и IEEE802.3 стандард нису идентични, термин Етернет се користи за мреже које подржавају стандард IEEE802.3 тј. локалне рачунарске мреже са протоколом CSMA/CD.

Стандард познат у литератури као Етернет II[1] донет је 1982. године а бројни додаци стандарду IEEE802.3 дати су у табели 2.1. Стандард IEEE802.3 разликује се од етернет спецификације (слика 2.4а) у томе што се једно од поља заглавља међу њима разликује (IEEE802.3 дужина поља је коришћена за тип пакета у Етернету II). У раму за Етернет II поља имају следеће значење:

- Уводни део (претходни, преамбула[2]) од 8 бајтова садржи комбинацију 10101010_2 при чему последњи бајт има вредност 10101011_2. Има за циљ да обавести и омогући синхронизацију мрежне картице са долазећим подацима;
- Одредишна и изворишна адреса (од по 6 бајтова) називају се адресе MAC[3] подслоја или физичка адреса;
- Поље тип указује на то који је протокол имплементиран на вишим слојевима. На пример уколико је у раму упакован IP пакет, поље тип ће садржати вредност 0800_{16}. Ове вредности додељује организација IEEE. На овај начин омогућено је да различити протоколи виших слојева могу да користе исту мрежну картицу;
- Поље за податке су укалупљени (енкапсулирани) пакети протокола виших слојева, од мрежног до апликационог. Дужина може да им буде најмање 46_{10}, а највише 1500_{10} бајтова;
- FCS[4] је секвенца која омогућава детекцију грешака. Користи се тридесетдвобитни циклични кôд (CRC[5]), који се генерише над пољима адресе, типа и података. Пријемна мрежна картица може да генерише идентичан циклични кôд над пољима адресе, типа и података и упоређује их са предајним циклични кодом. Уколико

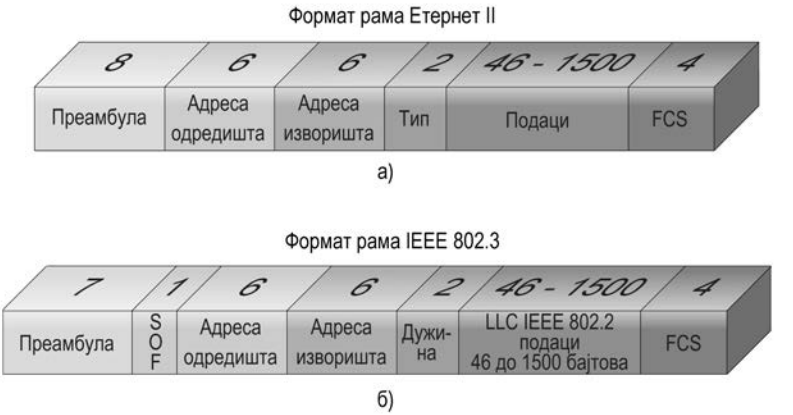
[1] Претече су биле XEROX *Ethemet* као Етернет I (специфицирале су га компаније XEROX, DEC, INTEL за 10Mb/s локалну рачунарску мрежу).

[2] *Preamble*

[3] *Media Access Control*

[4] *Frame Check Sequence*

[5] *Cyclic Redundancy Check*



Слика 2.4 Формати рамова а) Етернет II б) IEEE 802.3

су идентични пренос је обављен без грешке. Са цикличним кодом од 32 бита може да се детектује импулсна сметња од 31 бита са 100% тачношћу.

Структура рама по стандарду IEEE802.3 приказана је на слици 2.4б. Сваки рам почиње уводним делом (преамбулом) од седам бајтова а сваки садржи комбинацију 10101010_2. Следеће поље је ознака о почетку рама (SOF[1]), бајт који садржи 10101011_2 чиме је означен почетак самог рама. IEEE802.3 преамбула и ознака о почетку рама (SOF) имају исту вредност као и преамбула Етернет II рама.

Рам садржи две MAC адресе, једну одредишта[2] а другу извориште[3]. Стандард допушта двобајтне и шестобајтне адресе, али параметри дефинисани за 10Mb/s стандард користе шестобајтне адресе.

Поље дужина[4] показује колико бајтова је присутно у пољу података, од минималних 0 до максималних 1500_{10}. Ако је дужина података мања од 46_{10} бајтова користе се додатна[5] поља да допуне рам до минималне величине. Уколико је ова вредност већа од 1500_{10} сматра се да поље указује на тип а не на дужину рама података.

[1] *Start Of Frame*
[2] *Source*
[3] *Destination*
[4] *Length*
[5] Пад (*pad*)

2.3 Адресе MAC подслоја

MAC адреса се формира према правилима једног од три система који је прописала организација IEEE: MAC-48, EUI[1]-48, и EUI-64. Сва три система користе исти формат и разликују се само по дужини адресе (тј. по идентификатору).

Организација IEEE сматра термин MAC-48 застарелим. Овај термин се користио за специфичан тип EUI-48 идентификатора за адресирање хардверских интерфејса као што су мрежне интерфејсне картице (NIC[2]). Препоручује се да се у будућности користи ознака EUI-48.

Прва три октета (рачунавши редослед слања) одређује произвођач (организација) који додељује адресу и називају се јединствени идентификатор организације (OUI[3]). Следећа три октета (код MAC-48 и EUI-48 адреса) или пет (код EUI-64 адреса) додељујуе произвођач уз основно ограничење да су једнистевни у адресном простору произвођача (слика 2.5). Очекује се да ће се 48-битни адресни простор „потрошити“ до 2100. године. За адресни простор EUI-64 не очекује се да ће се потрошити у догледној будућности.

Бит најмање тежине у бајту највеће тежине одредишне адресе је 0 за обичне адресе[4], а 1 за групне адресе. Редослед слања је као на слици 2.5: шаље се прво бит најмање тежине. Кад је рам послат на групну адресу све станице у групи га примају. Слање групи станица назива се вишеструко емитовање[5]. Адреса која садржи све јединице резервисана је за слање свима (бродкаст[6]). Рам који садржи све јединице у пољу одредишне адресе биће послат свим станицама у мрежи и примиће све станице у мрежи. Још једна интересантна особина адресирања је коришћење бита (суседног бита до бита најмање тежине у бајту највеће тежине) за разликовање локалне[7] од глобалне[8] адресе (слика 2.5). Локалне адресе

[1] *Extended Unique Identifier*
[2] *Network Interface Controllers*
[3] *Organizationally Unique Identifier*
[4] *Unicast*
[5] *Multicast* - једна станица шаље групи станица.
[6] *Broadcast*- једна станица шаље свим станицама.
[7] *Locally administered addresses*
[8] *Universally administered addresses*

Слика 2.5 Формат адресе MAC-48 и EUI-48 адреса

додељују администратори локалне мреже и немају никакв значај изван локалне мреже. Глобалне адресе додељује IEEE тако да две станице нигде на свету немају исту глобалну адресу. Са 46 битова[1] доступно је $7x10^{13}$ глобалних адреса. Идеја је да било која станица јединствено адресира било коју другу станицу дајући тачан 46-битни број. На мрежном слоју је да лоцира одредиште.

2.4 PPP протокол и TCP/IP референтни модел

TCP/IP скуп протокола генерално је пројектован да подржи протоколе од трећег слоја навише. Такође TCP/IP референтни модел дефинише слој приступа мрежи који одговара слоју везе података, другом слоју OSI референтног модела. У многим класичним мрежама са TCP/IP протоколима претпоставља се да функције другог слоја обезбеђују технологије локалних мрежа или мрежа ширег подручја као што су: Етернет, IEEE802.3, IEEE802.11, IEEE802.16,...Ове технологије одговорне су за класичне функције другог слоја: адресирање, управљање приступом трансмисионом медијуму и најважније формирање рамова од датаграма примљених од трећег слоја.

Без обзира што TCP/IP референтни модел базично дефинише слој за заједнички приступ мрежи у оквиру овог слоја се разликује физички слој (који одговара физичком слоју OSI референтног модела) и слој везе. Најнижи слој OSI и TCP/IP референтних модела је физички слој који је одговоран за пренос података са једног места на друго у оквиру мреже.

[1] 48-2 = 46

Слој везе TCP/IP скупа протокола преузима IP пакет на страни извориштa (пошиљаоца) и смешта га у структуру рама који се користи између извориштa и одредиштa (нпр. етернет рам). После тога слој везе прослеђује тај рам података физичком слоју који рам шаље као серију битова преко линка (везе). На одредишту (пријемној страни) физички слој и слој везе из серије добијених битова реконструишу рам података, издвајају IP пакет и прослеђују га IP слоју одредиштa.

Интерфејси за IP пакете дефинисани су за следеће врсте локалних мрежа и мрежа ширег географског подручја:

- Етернет - направљен од стране компанија Digital Equipment Corporation, Intel, и Xerox (некада се означава као DIX Ethernet);
- IEEE802.3 - локалне рачунарске мреже, укључујући све варијанте до 10Gb/s;
- SONET/SDH[1] - оптичке мреже великих брзина за пренос података у мрежама ширег подручја;
- IEEE802.11 (WiFi) и IEEE802.16 (WiMax) бежичне мреже-технологије;
- PPP протокол - развијен од стране IETF[2] заједнице и није ограничен само на пренос IP пакета;
- X.25 - интернационални стандард за јавну мрежу са комутацијом пакета и успоставом везе[3];
- Штафетни пренос рамова[4] - интернационални стандард за јавну мрежу са комутацијом пакета и успоставом везе заснованом на X.25 стандарду;
- ATM[5] - интернационални стандард за јавну мрежу са комутацијом ћелија и успоставом везе;
- Мрежа интегрисаних услуга (ISDN[6]) – јавна мрежа за пренос података. Данас се користи највише као приступна мрежа;

[1] *Synchronous Optical Network/Synchronous Digital Hierarchy*
[2] *Internet Engineering Task Force*
[3] *Connection-oriented*
[4] *Frame Relay*
[5] *Asynchronous Transfer Mode*
[6] *Integrated Services Digital Network*

- Дигитална претплатничка петља (DSL[1]) – користи се за приступ Интернету;
- Дигитална претплатничка петља велике брзине (VDSL[2]) – врста DSL приступа који користи оптичка влакна да би се приближио на удаљеност мању од једног километра до корисника. Често се назива оптичка веза до „скоро до куће“ (FTTN[3]);
- Интернет протокол преко серијске линије (SLIP[4]) и компримовани SLIP (CSLI[5]) - старији начин слања IP пакета преко мрежа са комутацијом линија[6] (развијен у оквиру IETF заједнице);
- Кабловски модеми (CMODEM[7]) – метод за слање IP пакета преко ТВ кабловске инфраструктуре;
- Пасивни оптички гигабитни Етернет (GE-PONS[8]) – део су глобалне оптичке мреже (FTTH[9]). Заснована на IEEE802.3ah стандарду ова технологија може да подржи гигабитне брзине у оба правца и услуге као што је пренос говора (VoIP[10]) и телевизијског сигнала (IPTV[11]);
- На неким местима брзи приступ Интернету обезбеђује се електричном мрежом која је део технологије широкопојасних енергетских линија (BPL[12]). Преноси се преко истих утичница[13] као и електрична енергија.

Последица тога што се мрежни слој не везује за одређену врсту везе на нижим слојевима јесте флексибилност интернет протокола: IP пакети

---

[1] *Digital Subscriber Line*

[2] *Very-high-speed Digital Subscriber Line*

[3] *Fiber To the Neighborhood*

[4] *Serial Line Internet Protocol*

[5] *Compressed SLIP*

[6] *Dial-up*

[7] *Cable Modem*

[8] *Gigabit Ethernet Passive Optical Network*

[9] *Fiber To The Home* – оптичка веза до куће

[10] *Voice over IP*

[11] *Internet Protocol TeleVision*

[12] *Broadband Power Line*

[13] Среће се и: комуникација енергетским водовима (PLC - *Power Line Communication*), дигитална претплатничка петља преко енергетских водова (PLDSL - *Power Line Digital Subscriber Line*), телекомуникације преко енергетских водова (PLT - *Power Line Telecom*), умрежавање преко енергетских водова (PLN - *Power Line Networking*). Стандард IEEE1901 за пренос података енергетским водовима усвојен је септембра 2010. год.

могу да се преносе било којом мрежом. Нове врсте мрежних интерфејса могу се додавати без икаквих проблема. Такође међусобно повезивање мрежа значајно је олакшано. Све TCP/IP имплементације треба да подрже у крајњем случају бар један од горенаведених мрежних интерфејса. Етернет приступ трансмисионом медијуму (и интерфејс) најчешће је коришћени приступ слоја везе којим се преносе IP пакети, нарочито до крајњих тачака мреже.

Технологије које се највише користе за приступ Интернету су SONET/SDH, DSL, Етернет и бежичне технологије. У литератури се више пажње посвећује Етернету и бежичним технологијама него SONET и DSL технологијама. Адресирање на слоју везе је код SONET/SDH и DSL приступа много једноставније од адресирања у локалним рачунарским мрежама (жичним и бежичним). Постоје само две крајње тачке код тачка-тачка везе и увек корисник зна на ком је он крају. Било шта што се шаље намењено је другом крају везе и било шта што се прима долази такође са другог краја везе.

Два протокола су до сада дефинисана и имплементирана на слоју везе TCP/IP референтног модела: SLIP и PPP[1] протоколи (слика 2.6).

2.4.1 Протокол PPP и дигитална претплатничка петља

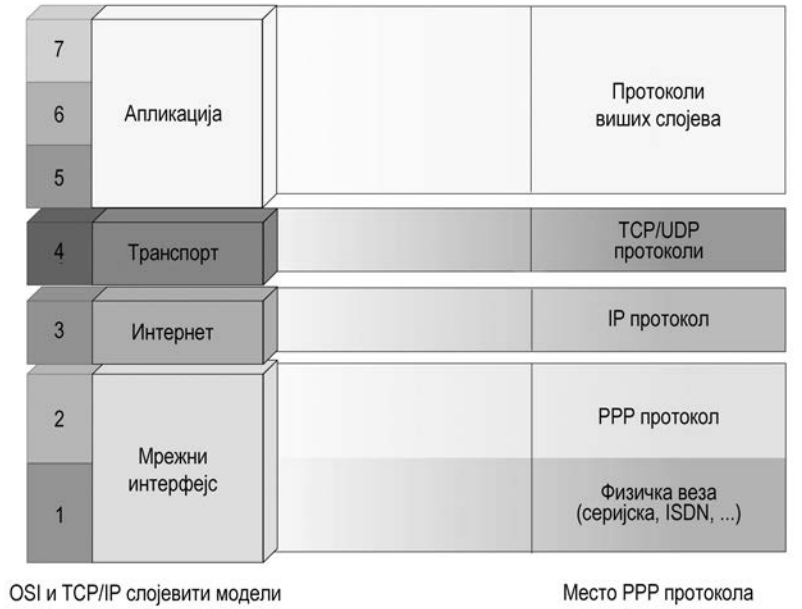
Поставља се питање зашто се PPP протокол користи у дигиталним претплатничким петљама (DSL) и оптичким преносним системима (SOET/SDH)? Главни разлог је што је интернет посредницима била потребна нека врста тунел протокола[2]. Као што је познато податак који треба пренети укалупљује се на сваком слоју: додају се заглавља апликације, транспортног слоја, слоја мреже и слоја везе на коме се формира рам. Када се тако формиран рам преноси другом врстом рама реч је о преносу кроз тунел. Многе методе прављења тунела су стандардизоване на различитим слојевима TCP/IP референтног модела.

Код преноса дигиталном претплатничком петљом (DSL) тунел протокол треба да обезбеди тачка-тачка везу од централне мрежне локације до локације корисника[3] и преко дељеног медијума локалних рачунарских мрежа

[1] *Point to Point Protocol* - протокол тачка - тачка

[2] *Tunneling protocol*

[3] *Customer's premises*



Слика 2.6 Протоколи слоја везе у TCP/IP слојевитом моделу протокола

до корисничког уређаја (крајње станице, хоста). Постоје бројни начини да се овај задатак одради као што су: IP-y-IP тунел, виртуелне приватне мреже (VPN[1]) или прављење тунела на нижим слојевима. Интернет посредници су одабрали PPP протокол као решење за дигиталне претплатничке петље. Годинама посредници Интернет услуга су користили PPP протокол за приступ удаљених корисника преко јавне комутиране мреже[2]. PPP протокол је омогућавао једноставно доделу и управљање IP адресним простором, наплату... Као технологија локалног умрежавања Етернет не поседује ниједну од тих карактеристика. PPP протокол такође дозвољава аутентификацију корисника помоћу RADIUS[3] протокола који не постоји у највећем броју технологија локалног умрежавања (ако сте у локалној жичној мрежи значи да њој припадате). Наравно, примењујући PPP протокол значи

[1] *Virtual Private Network*
[2] *Dial-in users*
[3] *Remoute Authentication Dial in User Service*

да се PPP рамови смештају унутар етернет рамова а као резултат добија се PPP протокол преко Етернета (PPPoE[1]) описан у RFC2516 документу.

2.4.2 Начин функционисања PPP протокола

PPP протокол није једини протокол другог слоја који је IETF организација специфицирала. Пре него што је PPP протокол постао широко примењен користили су се SLIP и CSLIP протоколи. Они су омогућавали повезивање индивидуалних персоналних рачунара преко комутиране телефонске мреже, асинхроном аналогном телефонском линијом са модемима. Протоколи SLIP и CSLIP (специфицирани у документима RFC1055/STD47) такође су се користили за повезивање рутера на удаљеним издвојеним TCP/IP мрежама преко асинхроних аналогних изнајмљених линија користећи модеме.

PPP протокол може се користити на асинхроним и синхроним везама и омогућава више од формирања рама податак са IP пакетом. Стандард за PPP протокол дефинише различите функције за тестирање, надзор и управљање квалитетом линије, договор око параметара везе итд. Протокол PPP дефинисан документом RFC1661 не зависи од осталих протокола и његова примена није ограничена на пренос IP пакета.

PPP протокол представља у ствари скуп протокола који покривају различите аспекте рада PPP протокола. Протоколи који се сматрају најважнијим су:

Протокол за управљање линком (LCP[2] протокол) је најважнији протокол у PPP скупу протокола. Он је одговоран за конфигурисање, одржавање и окончавање целокупне PPP везе. Два уређаја која користе PPP протокол размењују скуп LCP рамова који руководе радом LCP протокола.

Мрежни управљачки протокол (NCP[3] протокол) користи се за конфигурисање PPP протокола за различите протоколе слоја мреже. Пошто је PPP веза успостављена коришћењем LCP протокола, сваки протокол мрежног слоја захтева успостављање одговарајућег NCP линка (везе). Најважнији од NCP протокола је интернет управљачки протокол

[1] *Point-to-Point Protocol over Ethernet PPP*
[2] *Link Control Protocol*
[3] *Network Control Protocols*

(IPCP[1] протокол) који омогућава да се пакети (датаграми) преносе PPP везама.

Поред IPCP протокола, који се за сада највише користи, дефинисани су и протоколи: IPXCP[2] (за IPX[3] протокол), NBFCP[4] (за NetBEU[5] протокол) и IPv6CP[6] (за IPv6 протокол).

Протоколи који се користе за аутентификацију за време успоставе PPP везе јесу протокол за аутентификацију са слањем корисничке лозинке (PAP[7]) и протокол за међусобну размену аутентификационих порука (CHAP[8] протокол[9]).

Размена порука LCP и NCP протокола на PPP линку у TCP/IP мрежи обавља се на следећи начин:

- Изворишни PPP систем (корисник) шаље серију порука за конфигурисање и тестирање везе (линка);
- Оба краја везе размењују LCP поруке да би се договориле и подесиле карактеристике линка које ће се користити;
- Изворишни PPP систем шаље серију NCP порука за успоставу протокола слоја везе (на пример IP, IPX[10]);
- NCP и LCP поруке се користе за затварање везе.

LCP протокол

Постоје три битне фазе у животном циклусу везе и LCP протокол игра кључну улогу у њима. То су:

- конфигурисање везе[11]: процес постављања и међусобног договора око параметара везе;
- одржавање везе[12]: управљање и надзор отворене везе;

---

[1] *Internet Control Protocol*
[2] *Internetworking Packet eXchange Control Protocol*
[3] Мрежни протокол *Novell Netware* оперативног система
[4] *PPP NetBIOS Frames Control Protocol*
[5] *MS Windows* оперативни систем
[6] *PPP IP Version 6 Control Protocol*
[7] *Password Authentication Protocol*
[8] *Challenge Handshake Authentication Protocol*
[9] Тачније би било: аутентификациони протокол са изазовом и троструком разменом порука.
[10] *Internetworking Packet eXchange* - мрежни протокол *Novell Netware* оперативног система.
[11] *Link Configuration*
[12] *Link Maintenance*

- окончавање везе[1]: процес затварања постојеће везе када она више није потребна.

Дијаграм на слици (2.7) представља преглед LCP порука које се

Слика 2.7 Размена LCP порука

[1] *Link Termination*

размењују за време различитих фаза PPP везе. Конфигурисање везе је представљено као једноставна размена *Configure-Request* и *Configure-Ack* порука. После размене PPP порука за аутентификацију и конфигурацију једног или више NCP линкова, веза прелази у фазу *Link Open*. У овом примеру поруке *Echo-Request* и *Echo-Replay* користе се за испитивање везе после чега следи пријем и слање података крајњих уређаја. На слици 2.6 приказан је и пример одбацивања једне *Data* поруке која је имала погрешно *Code* поље. Веза се окончава разменом *Terminate-Request* и *Terminate-Ack* порука.

IPCP протокол

Начин рада мрежних управљачких протокола као што је IPCP протокол веома је сличан начину рада LCP протокола. У примеру на слици 2.8 представљена је размена поруке IPCP протокола.

По завршетку размене LCP порука конфигурација везе (укључујући и аутентификацију) је окончана. Поруке IPCP *Configure-Request* и IPCP *Configure-Ack* користе се за успоставу IPCP везе па се подаци (IP пакет) могу слати преко те везе. Уколико више не постоји потреба за IPCP везом она се може оконачати после чега LCP веза остаје отворена за пренос друге врсте података. Неопходно је да се експлицитно оконча IPCP веза пре затварања LCP везе.

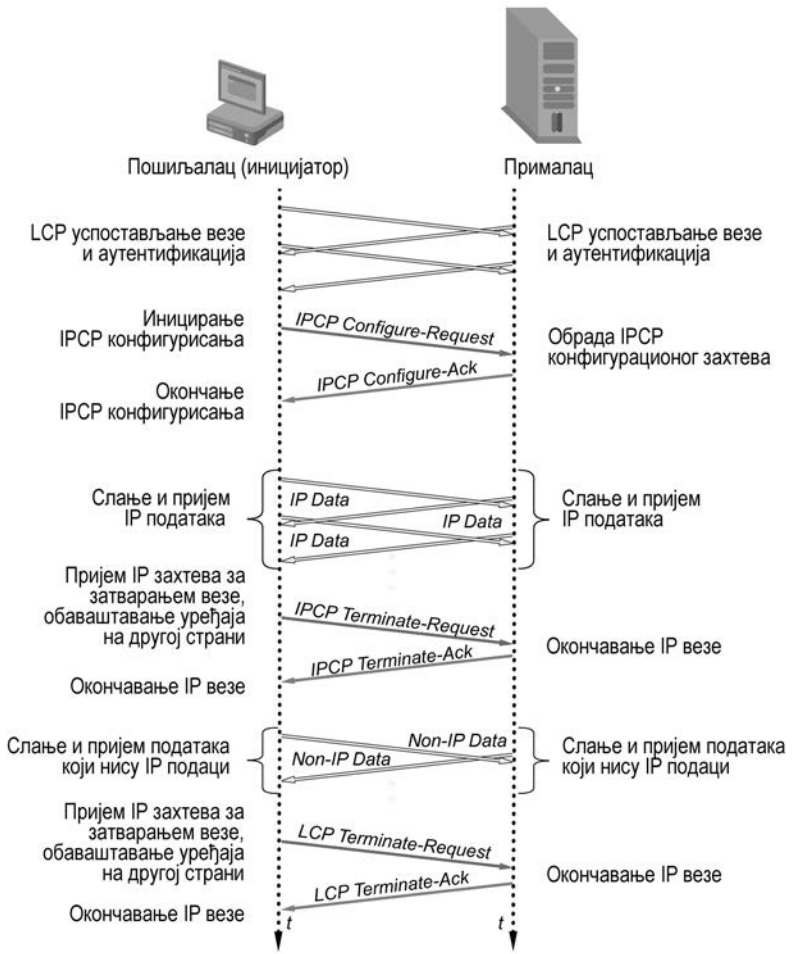
2.4.3 Протоколи за аутентификацију PPP везе

Скуп PPP протокола је тако пројектован да могу да се користе опциони протоколи за аутентификацију на везама где је битна аутентификација. За време фазе успоставе везе са LCP протоколом крајњи уређаји могу да се договарају око тога који ће протокол за аутентификацију да користе. Уколико постигну договор серија аутентификационих порука се шаље да би се потврдио идентитет уређаја који је иницирао успоставу везе. Само ако је процес аутентификације успешан може се процес конфигурисања везе наставити.

У оквиру скупа PPP протокола дефинисана су два различита протокола за аутентификацију: PAP[1] и CHAP[2] протоколи. Поред њих могуће је користити

[1] *Password Authentication Protocol*

[2] *Challenge-Handshake Authentication Protocol*



Слика 2.8 Размена IPCP порука

аутентификационе шеме произвођача[1]. То захтева да се одговарајуће конфигурационе опционе вредности програмирају у LCP протоколу и замене постојеће у *Authenification-Protocol* конфигурационим опцијама.

PAP протокол (слика 2.9) ради на тај начин што размењује поруке (захтеве) који садрже име и лозинку и одговор који указује да ли јесте или није аутентификација успешна.

[1] *Proprietary*

Слика 2.9 PAP аутентификација

Основна разлика између PAP и CHAP протокола је што CHAP протокол не шаље лозинку преко линка (везе). Када се користи PAP извориште поручује одредишту „шаљем лозинку коју знам, упореди да ли одговара твојој“. Када се користи CHAP протокол сваки крајњи уређај користи лозинку да би одрадио криптографско израчунавање и упоређује се да ли су оба уређаја (и извориште и одредиште) добили исти резултат. Уколико јесу они имају исту лозинку.

Процес се одвија у три корака и приказан је на слици 2.10:

1. Изазов (*Challenge*): аутентификатор генерише рам који се назива *Challenge* и шаље га извориштy (иницијатору везе). Рам садржи једноставан текст. Порука нема никакво специјално значење тако да није битно и ако је неко пресретне. Битно је да после пријема *Challenge* рама оба уређаја имају исту поруку - изазов[1].

[1] *Challenge message*

2. Одговор (*Response*): извориште (иницијатор) користи лозинку (или неку дељену тајну[1] коју аутентификатор такође зна). Лозинка и изазов су улаз у криптографску функцију извода (хеш[2]). Он шаље натраг аутентификатору тако креирани извод као *Response* рам.
3. Успех (*Success*) или неуспех (*Failure*): аутентификатор извршава исти поступак креирања извода на поруци - изазову као што је то урадио иницијатор. Уколико аутентификатор добије исти резултат као и резултат који је извориште (иницијатор) послало у *Response* раму онда аутентификатор зна да је иницијатор имао исправну лозинку када је креирао извод. Тада аутентификатор шаље назад *Success* рам. У супротном шаље *Failure* рам (поруку).

Слика 2.10 Размена порука код CHAP протокола

1 *Shared secret*
2 *Hash*

Слика 2.11 Формат рама PPP протокола

2.4.4 Формат рама PPP протокола

На слици 2.11 приказан је формат рама PPP протокола (PPP рам) који је направљен слично HDLC протоколу (документ RFC1662).

Рам PPP протокола садржи следећа поља:

- Ознаке (индикатор). Сваки PPP рам почиње и завршава се једнобајтном ознаком са вредношћу 01111110_2;
- Адреса (1 бајт). Једина могућа вредност за ово поље је 11111111_2 што се увек користи да укаже на „све станице“ или адресу упућену свима[1] (бродкаст). Треба уочити да ни једно друго поље PPP заглавља не садржи адресу изворишта. PPP протокол води рачуна само о одредишту чија је адреса увек 11111111_2 што суштински значи „било који уређај на другом крају везе који препознаје овај рам“;
- Управљање (1 бајт). Вредност на коју је постављено ово поље је 00000011_2 што одговара формату ненумерисане информације (UI[2]). Указује да се користи IP протокол без успоставе везе. С обзиром на то да и адресно и управљачко (контролно) поље тренутно могу имати само фиксне вредности, поставља се питање зашто су та поља уопште и дефинисана. У спецификацији за PPP протокол (RFC1662) каже се да друге вредности „могу да се дефинишу касније”, мада их до данас нико није дефинисао. Имајући у виду да та поља имају фиксне вредности, PPP протокол дозвољава предајнику да не шаље те бајтове, штедећи тако два бајта додатног оптерећења у PPP раму;

[1] *Broadcast address*

[2] *Unnumbered Information*. Код HDLC протокола (детаљније погледати 11. поглавље у уџбенику Рачунарске мреже истог аутора) контролним пољем су дефинисане различите врсте рамова: информациони, управљачки и ненумерисани.

- Протокол (2 бајта). Поље за протокол указује PPP пријемнику на садржај PPP рама. То је у ствари ознака за протокол вишег слоја коме припадају укалупљени подаци. По пријему PPP рама, PPP пријемник ће проверити његову исправност и затим проследити укалупљене податке одговарајућем протоколу. Документи RFC1700 и RFC3232 дефинишу 16-битне кодове за протоколе које PPP протокол користи. Вредности су следеће:
  - UDP за LCP протокол вредност је $C021_{16}$,
  - за NCP16 протокол вредност је 8021_{16},
  - за IP протокол (односно, подаци укалупљени у PPP рам у виду IP датаграма) вредност је 0021_{16},
  - за *AppleTalk* протокол вредност је 0029_{16}.
- Информације (поље променљиве дужине). Ово поље садржи укалупљени пакет (за PPP то су подаци) који је послао протокол горњег слоја (на пример, IP). Максимална подразумевана дужина поља за информације је 1500 бајтова, мада то може да се промени када се веза први пут конфигурише;
- Контролни збир (2 бајта). Поље за контролни збир користи се за откривање грешака у пренесеном раму.

2.4.5 Уметање бајтова

Шта се дешава ако се секвенца битова идентична са ознаком и сама појављује на неком другом месту у раму? На пример, шта се догађа ако се вредност поља индикатора 01111110 појављује у пољу за информације? Да ли ће пријемник неисправно открити крај PPP рама?

Решење које је преузето у PPP протоколу и многим другим протоколима (нпр. HDLC протоколу) је да се користи техника позната као уметање бајтова. PPP протокол користи специјалан контролни бајт 01111101. Ако се секвенца 01111110 појављује било где у раму, изузев у пољу ознаке (флега), PPP протокол ставља испред те секвенце контролни бајт. То значи да „умеће" (додаје) контролни бајт у низ података који се преноси, пре бајта 01111110, да би указао да бајт 01111110 који следи није вредност ознаке него, у ствари, стварни податак. Пријемник који види 01111110 коме претходи 01111101 ће, наравно, уклонити попуњен контролни бајт да би реконструисао оригиналне податке. На тај начин, када пријемник угледа

један контролни бајт у низу података, он зна да је бајт био уметнут у том низу. Пар контролних бајтова један до другог значи да се један од њих појављује у оригиналним подацима који се шаљу.

2.4.6 Пренос IP пакета DSL везама

Поставља се питање како се IP пакети преносе DSL везама? DSL спецификација дефинише формирање и пренос основног DSL рама на физичком слоју, али IP пакети се не стављају директно у ове рамове. IP пакети се прво стављају у PPP рамове а затим се PPP рамови укалупљују (енкапсулирају) у етернет рамове (PPPoE[1]). Затим се етернет рамови смештају у DSL рамове и шаљу ка DSL приступном модулу (DSLAM[2]) који се налази у телефонској централи.

Када DSL рам стигне у телефонску централу изгледа најједноставније да се издвоји етернет рам и проследи рутерима. Показује се да су скоро сви DSLAM уређаји повезани међусобно са ATM[3] технологијом или штафетним преносом рамова[4]. На тај начин уместо да се за повезивање DSLAM система и интернет посредника користе три слоја: мрежни, слој везе и физички слој уобичајено се користи пет слојева:

- IP пакет који садржи корисничке податке налази унутар PPP рама који је
- унутар етернет рама који се преноси преко DSL рутера (PPPoE) који је
- унутар серије ATM ћелија (или рама мреже за штафетни пренос рамова), које се шаљу преко физичког медијума као серија битова.

2.5 Уређаји за међусобно повезивање станица

Слојевите архитектуре протокола (TCP/IP, OSI) омогућавају међусобно повезивање станица у мрежама[5]. Ови протоколи могу се употребити за повезивање станица у рачунарским мрежама у једној згради, блоку

[1] *PPP over Ethernet*

[2] *DSL Access Module*

[3] *Asyncronous Transfer Mode* – користи ћелије уместо рамова за пренос информација.

[4] *Frame relay* Customer's premises

[5] *Internetworking, Interworking*

зграда, граду, држави или у целом свету. Нису сви уређаји за међусобно повезивање исти. У општем случају када се пројектују рачунарске мреже које обухватају и станице ван локалних рачунарских мрежа користе се четири врсте уређаја за међусобно повезивање станица: обнављивачи[1], мостови[2], комутатори[3] и рутери[4].

Не тако давно мрежна конфигурација и доступни уређаји одређивали су која ће врста уређаја за међусобно повезивање бити употребљена. Данашње мрежне конфигурације постају све сложеније и уређаји који се могу наћи на тржишту често у себи садрже карактеристике више уређаја. Ако се жели једноставно извршити категоризација уређаја за међусобно повезивање онда је то на основу различитих слојева TCP/IP референтног модела на коме раде као што је то приказано на слици 2.12.

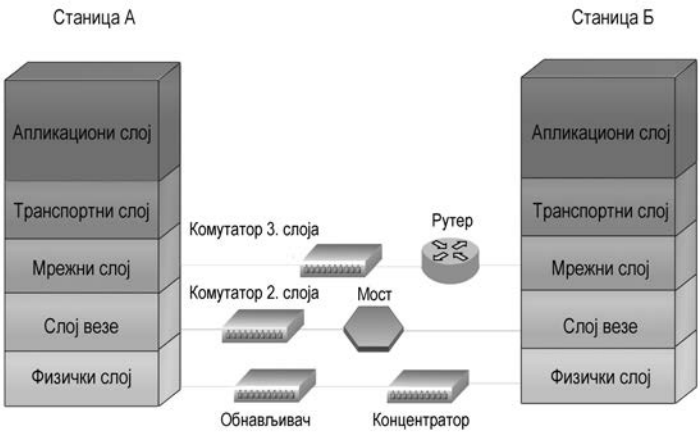
Грубо речено обнављивачи, вишепортни обнављивачи[5] (концентратори, хабови[6]) преусмеравају битове података са једног сегмента локалне рачунарске мреже на други, мостови и комутатори (слоја 2) преусмеравају рамове а рутери и комутатори (слоја 3) пакете. То значи да:

Обнављивачи и вишепортни обнављивачи (концентратори, хабови) не анализирају адресе;

Мостови и комутатори слоја 2 анализирају адресу другог слоја (MAC адресу) и на основу ње врше преусмеравање и филтрирање рамова. Подсетимо се да мост и комутатор 2. слоја праве табеле MAC адреса на основу којих доносе одлуку о преусмеравању;

Рутери и комутатори 3. слоја преусмеравање врше на основу адресе 3. слоја односно IP адресе. Комутатор 3. слоја осим прикупљања[7] информација која MAC адреса је придружена[8] ком порту прикупља и информације о томе која IP адреса је придружена ком интерфејсу. На тај

[1] *Repeaters*
[2] *Bridges*
[3] *Switches*
[4] *Routers*
[5] *Multiport repeaters*
[6] *Hub*
[7] *Learn*
[8] *Associated*



Слика 2.12 Позиција уређаја за међусобно повезивање у TCP/IP референтом моделу

начин комутатор 3. слоја усмерава саобраћај на основу информација о IP адреси.

Обнављивач посматра бит по бит рама, док мостови и комутатори другог слоја посматрају рамове као целине. Рутери раде на слоју мреже што представља основни начин рада на Интернету.

2.5.1 Сегментирање локалних рачунарских мрежа

Мрежни администратори и пројектанти често се суочавају са потребом да увећају количину пропусног опсега која је доступна корисницима, повећају број станица у мрежи или прошире домет локалне рачунарске мреже.

Некада се циљ може једноставно постићи: уколико долази до загушења саобраћаја у мрежи са уређајима брзине (протока) 100Mb/s[1] повећање пропусног опсега постиже се када се они замене са уређајима брзине 1Gb/s[2]. Међутим, то значи да је потребно заменити сву активну мрежну опрему: интерфејсне картице, комутаторе, мостове и рутере што може да буде веома скупо.

Други начин да се сваком кориснику омогући већи пропусни опсег је сегментирање локалне рачунарске мреже. Сегментирање не

[1] *Fast Ethernet*
[2] *Gigabit Ethernet*

захтева замену комплетне опреме корисника. Као што и само име каже сегментирањем се локална рачунарске мреже дели у мање целине које се међусобно повезују одговарјућим уређајима.

Друга последица тога што уређаји за међусобно повезивање станица раде на различитим слојевима је различит број колизионих и бродкаст домена који се формирају. Метод CSMA/CD за приступ трансмисионом медијуму може довести до судара (колизије) када станице покушају да користе медијум у исто време. Судари „троше“ пропусни опсег пошто долази до оштећења рамова и станице које су се сударили морају да сачекају одређени период времена и да поново започињу процедуру слање. Чак и када у етернет (или IEEE802.3) мрежама не долази до колизије, рамове упућене свим станицама (бродкаст) морају да приме и анализирају све станице у једном домену. Пропусни опсег се без потребе троши када се рамови упућени свим станицам (бродкаст рамови) шаљу и станицама које нису заинтересоване за њихов садржај. Код TCP/IP скупа протокола ARP[1] рамови су најчешћа врста бродкаст рамова које станице и шаљу и примају. Важно је нагласити да без обзира што је CSMA/CD протокол део и гигабитног Етернета он суштински није присутан код 10Gb/s Етернета.

Проширивањем локалне рачунарске мреже преусмеравањем битова коришћењем обнављивача или концентратора (хаба), остварује се један колизиони и један бродкаст домен. Број колизионих и бродкаст домена за различите уређаје за међусобно повезивање дат је у табели

Уређај за међусобно повезивање	**Колизиони домен**	**Бродкаст домен**
Обнављивачи (концентратори, хабови)	Један	Један
Мост	Већи број	Један
Рутер, комутатор 3. слоја	Већи број	Већи број
Комутатор 2. слоја	Већи број	Зависи од VLAN конфигурације

Табела 2.2 Колизиони и бродкаст домени различитих уређаја за међусобно повезивање

[1] *Adress Resolution Protocol* – протокол за разрешавање адресе. Користи се када је позната IP адреса одредишта а непозната његова MAC адреса.

2.2. Коришћење једног уређаја у једној мрежи не искључује коришћење других уређаја у тој истој мрежи. Другим речима рутер се може користити за сегментирање локалне рачунарске мреже у два (или више сегмената) а сваки резултујући сегмент може бити даље подељен помоћу моста. У крајњем случају сваки корисник или систем има доступан комплетан пропусни опсег што на пример омогућава комутатор 2. слоја.

Обнављивачима се не може вршити сегментирање мреже. Обнављивачи преусмеравају битове из једног сегмента у други и немају могућност да анализирају формат података. Уколико рам садржи грешке, крши правило о минималној или максималној величини рама прописаној за ту мрежу, или било шта што је погрешно обнављивачи и концентратори ће их, без обзира на све, преусмерити.

2.5.2 Мостови

Етернет спецификација ограничава број станица (уређаја) који се може прикључити на један сегмент локалне рачунарске мреже и свеукупну раздаљину коју локална рачунарска мрежа досеже. Ако је потребно додати уређај у локалну рачунарску мрежу која је достигла једно од поменутих ограничења тада се може употребити мост за повезивање сегмената те локалне рачунарске мреже.

Локална рачунарска мрежа која користи мостове филтрира рамове и не прусмерава све рамове на све сегменте повезане на мост. То је разлог што мостови обезбеђују више од једног колизионог домена. Међутим, локалне рачунарске мреже повезане мостовима и даље сачињавају један бродкаст домен. Иако се име „мост“ често користи у литератури мостови као и вишепортни обнављивачи (концентратори, хабови) данас се веома мало користе.

Процес филтрирања који је примењен у мостовима разикује се у зависности од специфичне технологије локалних рачунарских мрежа које мост повезује. Етернет користи транспарентно премошћавање[1] за повезивање сегмената локалне рачунарске мреже. Транспарентни мост анализира одредишну MAC адресу и на основу ње одлучује да ли ће рам бити:

[1] *Transparent bridging*

- преусмерен[1] - рам се шаље само на сегменте локалне рачунарске мреже где се налази одредиште;
- филтриран - мост одбацује[2] рам. Никаква порука се не шаље извориште;
- преплављен[3] - рам се шаље ка сваком сегменту мреже који је повезан на мост. Реализује се за саобраћај усмерен ка свим станицама (бродкаст) и за саобраћај усмерен ка групи станица (мултикаст[4]).

Мостови представљају напредније уређаје у односу на обнављиваче, али још увек имају бројне недостатке. Поруке које генерише протокол за разрешавање адресе (ARP протокол), који се користи за повезивање IP адреса слоја 3 са MAC адресама локалних мрежа слоја 2, преносе се кроз све мостове. Саобраћај упућен свима, који се на тај начин генерише не представља недостатак мостова. Саобраћај упућен групи станица (мултикаст) такође се преноси плављењем[5] и мултимедијалне апликације као што су видеоконференције могу лако да преоптерете рачунарску мрежу међусобно повезану мостовима.

Етернет мостови треба да обезбеде функцију разгранатог стабла. Ови мостови могу да открију затворене петље у међусобно повезаним локалним рачунарским мрежама. Петље представљају проблем у мрежама са мостовима пошто се може десити да се исти рамови прослеђују (преплављују) на све сегменте. Преплављивањем се умножава укупан број рамова који се преносе мрежом. Петље умножавају рамове све док се не достигне тачка засићења[6].

2.5.3 Рутери

Мостови су додали нове функције међусобно повезаним мрежама пошто раде на вишем слоју од обнављивача. Мостови раде на другом слоју и обезбеђују више колизионих домена, оно што обнављивачи (и

[1] *Forwarded*
[2] *Drop*
[3] *Flooded*
[4] *Multicast*
[5] *Flooded*
[6] *Saturation*

вишепортни обнављивачи, хабови) не могу. Рутери обезбеђују и већи број колизионих домена а такође и већи број бродкаст домена.

У локалним мрежама са обнављивачима и мостовима све станице припадају истој подмрежи. Адресе 3. слоја (на пример IP адресе) састоје се од мрежног дела адресе и дела адресе станице (хоста)[1]. Локалне рачунарске мреже повезане рутерима имају више бродкаст домена и сваки сегмент локалне рачунарске мреже припада различитој подмрежи.

Када постоји више подмрежа међусобно повезаних рутерима TCP/IP уређаји раде другачије. Мостови који повезују TCP/IP крајње станице транспарентни су за цео систем али рутери који повезују крајње станице нису. Крајња станица мора да зна адресу најмање једног рутера (подразумеваног рутера[2]) како би послала пакете ван локалне подмреже.

Мостови се некада називају уређајима који су „независни од протокола“ што значи да се мостови могу користити за повезивање сегмената локалне рачунарске мреже независно од тога да ли се користи TCP/IP скуп протокола или неки други скуп протокола. Међутим рутери морају да користе софтвер 3. слоја у зависности од протокола 3. слоја који се у тој мрежи користи. Највећи број рутера, нарочито оних који су повезани на Интернет користе и раде само са TCP/IP протоколом. Неки рутери могу да раде са више различитих протокола 3. слоја.

2.5.4 Комутатори локалних рачунарских мрежа

Комутатор[3] је уређај који има бројне заједничке карактеристике са мостом и рутером. Комутатор локалне рачунарске мреже је сложени мост са већим бројем интерфејса а комутација у локалним рачунарским мрежама је коначан наставак вишепортног премошћавања[4]. У локалним рачунарским мрежама са комутаторима сваки уређај има свој сегмент чиме је омогућено да је целокупан трансмисиони медијум на располагању том уређају. Више уређаја може да шаље истовремено осим у случају колизије на порту до које долази када више станица у исто време покушава да шаље рамове преко истог порта комутатора.

---

[1] Детаљно ће бити обрађено у 4. поглављу овог уџбеника.

[2] *Default router*

[3] *Switch*

[4] *Multiport bridging*

Сви портови комутатора формирају своје сопствене бродкаст домене. Када се појаве рамови које генерише ARP протокол, или мултикаст саобраћај комутатор прослеђује те рамове (преплављује) на све портове осим оног са кога је рам пристигао. То значи да комутатори нису ништа бољи од мостова и обнављивача у погледу бродкаст и мултикаст саобраћаја. Постоји унапређење у односу на обнављиваче (концентраторе, хабове) пошто не може да дође до колизије која би довела до поновног слања рамова (ретрансмисије).

Да би превазишли овај проблем комутатори локалне рачунарске мреже могу да обезбеде да само одређени број портова припада једном бродкаст домену. Тако конфигурисан бродкаст домен омогућава да је бродкаст и мултикаст саобраћај ограничен на одређени домен. То значи да није могуће ниједном раму да пређе границу бродкаст домена коме припада.

Рутери су уређаји од којих се очекује да међусобно повезују домене. Када комутатори[1] дефинишу више бродкаст домена они на тај начин формирају виртуелне локалне рачунарске мреже (VLAN[2]).

2.6 Виртуелна локална рачунарска мрежа

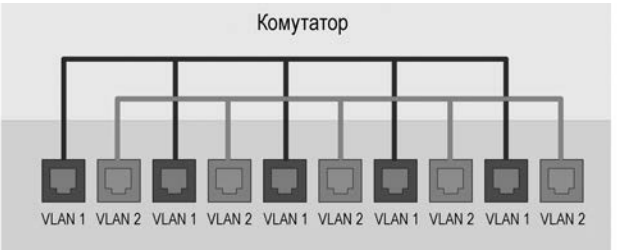
Виртуелна локална рачунарска мрежа по спецификацији IEEE дефинише бродкаст домене на слоју 2. Без обзира што виртуелне локалне рачунарске мреже немају никакве везе са TCP/IP скупом протокола оне су унеле значајну разлику у рад комутатора и рутера који раде у TCP/IP мрежама.

Рутери не преусмеравају бродкаст саобраћај као што то раде мостови тако да рутери аутоматски дефинишу посебне бродкаст домене на сваком свом интерфејсу. Комутатори слоја 2 логички праве бродкаст домене у зависности од тога како су конфигурисани. Конфигурација указује комутатору како да усмери рамове упућене свим станицама:

Када се комутатори користе за повезивање сегмената локалне рачунарске мреже бродкаст домен се не може одредити једноставним

[1] У мрежама ширих подручја термин „комутатор“ односи се на мрежне чворове који се разликују од рутера. То су на пример комутатори у мрежама са штафетним преносом рамова (*Frame Relay*) или у ATM (*Asynchronous Transfer Mode*) мрежама.

[2] *Virtual LAN*



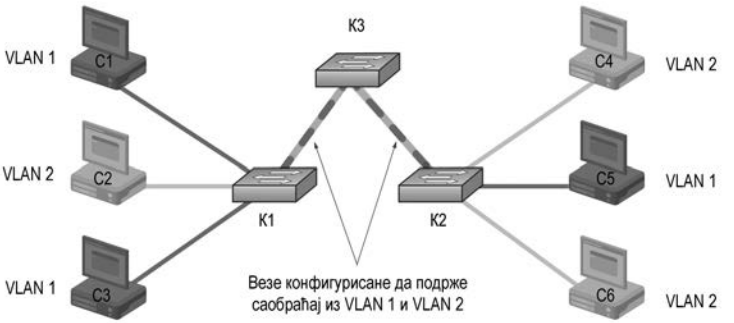
Слика 2.13 Виртуелне локалне рачунарске мреже у оквиру комутатора

прегледом мрежне топологије. Станице могу да припадају различитим, истим или неколицини бродкаст домена. Конфигурациона датотека у комутатору одређује границе ових домена као и чланове домена. Бродкаст домени су сада логичке целине повезане виртуелним мостовима[1] као што је приказано на слици 2.13.

Сваки виртуелни мост конфигурисан у комутатору успоставља посебан бродкаст домен или виртуелну локалну рачунарску мрежу (VLAN). Рамови из једне виртуелне мреже не могу се проследити директно другој виртуелној мрежи у оквиру комутатора слоја 2. Рамови адресирани на све станице (бродкаст поруке) из виртуелне локалне рачунарске мреже VLAN1 шаљу се само у VLAN1 бродкаст домен. Такође се бродкаст поруке из виртуелне локалне рачунарске мреже VLAN2 шаљу само у VLAN2 бродкаст домен.

Анализираћемо мрежу која се састоји од две виртуелне локалне мреже међусобно повезане са три комутатора 2. слоја приказану на слици 2.13. На комутатор К1 повезане су станице С1, С2 и С3 а на комутатор К2 станице С4, С5 и С6. Станице С1, С3 и С5 представљају виртуелну мрежу VLAN1 а станице С2, С4 и С6 представљају виртуелну мрежу VLAN2. Комутатори К1 и К2 неозначене рамове које примају од станица означавају у зависности од тога којој виртуелној мрежи станице припадају. Комутатор К3 повезује комутаторе К1 и К2 и станице које припадају виртуелним мрежама VLAN1 и VLAN2. Комутатори К1 и К2 повезани су са комутатором К3 сабирним везама[2] које су тако конфигурисане да се њима

[1] *Virtual bridge*
[2] *Trunk*



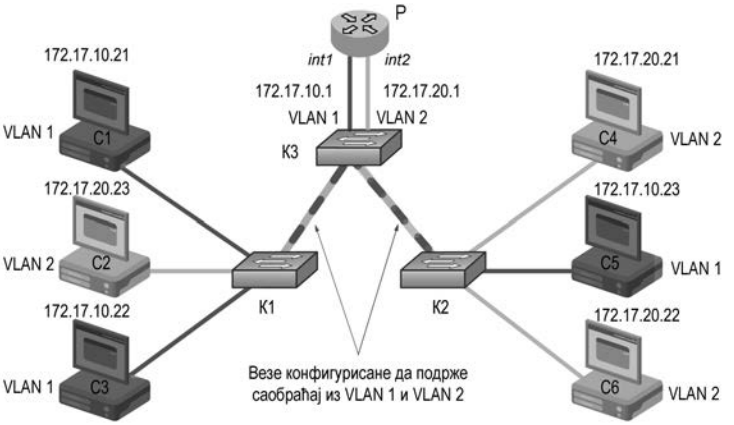
Слика 2.14 Две виртуелне мреже повезане комутаторима 2. слоја

преносе означени рамови обе виртуелне мреже. Уколико станица C1 шаље бродкаст рам (нпр. ARP рам) онда ће тај рам комутатор K1 означити ознаком VLAN1 и проследити ка станици C3 и комутатору K3. Комутатор K3 ће означени рам проследити комутатору K2. Комутатор K2 одстрањује VLAN1 ознаку у раму и прослеђује га ка станици C5. Треба нагласити да комуникација између виртуелних мрежа VLAN1 и VLAN2 није омогућена уколико је уређај K3 комутатор 2. слоја.

Уређај за међусобно повезивање мрежа који ради на трећем слоју, као што су рутер или комутатор 3. слоја мора се употребити за повезивање виртуелних мрежа. На тај начин се обезбеђује међусобно повезивање виртуелних мрежа и истовремено задржавају одвојени бродкаст домени. Сви уређаји који могу да комуницирају директно без рутера (или других уређаја који раде на 3. или вишим слојевима) деле исти бродкаст домен.

2.6.1 Контролисање бродкаст домена са комутаторима и рутерима

Раздвајање великих бродкаст домена у више мањих умањује обим саобраћаја упућеног свим станицама (бродкаст) и на тај начин се побољшавају перформансе мреже. Раздвајање домена у виртуелне локалне рачунарске мреже побољшава сигурност при размени информација унутар једне организације. Бродкаст домени могу се раздвојити или прављењем више виртуелних рачунарских мрежа (помоћу комутатора) или помоћу рутера. Рутер је потребан увек када уређаји на различитим мрежама на 3. слоју треба да комуницирају, без обзира да ли се користе или не користе виртуелне мреже.



Слика 2.15 Контролисање бродкаст домена са комутаторима и рутерима

2.6.2 Комуникација унутар виртуелне локалне мреже

На слици 2.15 станица C1 жели да комуницира са станицом C5. Обе станице C1 и C5 су на виртуелној мрежи VLAN1 и на истој мрежи чија је адреса 172.17.10.0/24. Комуникација се реализује на следећи начин:

1. Станица C1 у виртуелној мрежи VLAN1 шаље свој рам *ARP request* (упућен свима) ка комутатору K1. Комутатор K1, K2 и K3 шаљу *ARP request* на све портове виртуелне мреже VLAN1;
2. Комутатори у мрежи преусмеравају *ARP reply* рам (уникаст) ка станици C1. Станица C1 прима одговор који садржи MAC адресу станице C5;
3. Станица C1 сада има одредишну MAC адресу станице C5 и користи је да направи рам упућен станици C5. Комутатори K1, K2 И K3 испоручују рам станици C5.

2.6.3 Комуникација између виртуелних локалних мрежа

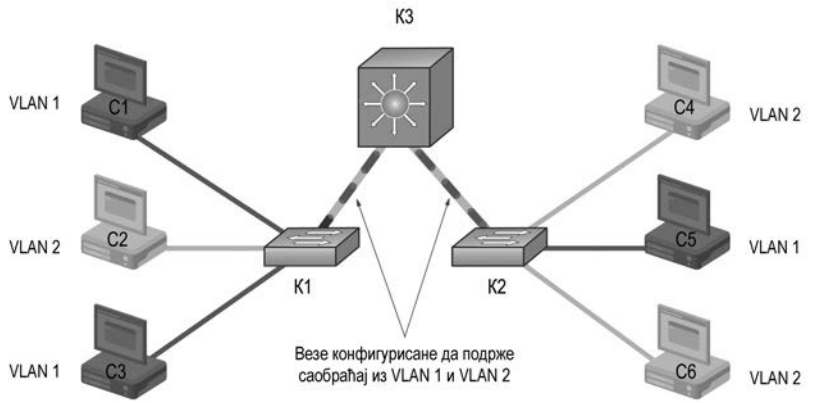
Посматрајмо станицу C1 на виртуелној мрежи VLAN1 која жели да комуницира са станицом C4 на виртуелној мрежи VLAN2 (слика 2.15). Треба уочити да је рутер (тј. његови интерфејси) повезан на два порта комутатора K3: *int1* припада виртуелној мрежи VLAN1 (подмрежа са адресом 172.17.10.0/24) а *int2* припада виртуелној мрежи VLAN2 (подмрежа са адресом 172.17.20.0/24). Види се да су виртуелне мреже

VLAN1 и VLAN2 засебне IP мреже. Адреса *int1* итерфејса представља подразумевани међумрежни[1] пролаз за станице у VLAN1 мрежи а адреса *int2* итерфејса представља подразумевани међумрежни пролаз за станице у виртуелној мрежи VLAN2. Пошто су виртуелне мреже засебни бродкаст домени станице које припадају различитим виртуелним мрежама могу међусобно да комуницирају само посредством рутера или комутатора 3. слоја који повезује те две мреже.

Процес се одвија на следећи начин:

1. Станица C1 на виртуелној мрежи VLAN1 жели да комуницира са станицом C4 на виртуелној мрежи VLAN2. Шаље *ARP request* рам да би добила MAC адресу подразумеваног међумрежног пролаза рутера P (одговарајући интерфејс рутера - *int1*);
2. Рутер P1 као одговор шаље *ARP reply* рам са свог *int1* интерфејса. Сви комутатори прослеђују *ARP reply* рам и станица C1 га прима. Рам садржи MAC адресу *int1* интерфејса;
3. Станица C1 затим формира етернет рам у коме је као одредишна адреса постављена MAC адреса подразумеваног међумережног пролаза интерфејса рутера P1. Тај рам комутатор K1 шаље ка комутатору K3;
4. Рутер P прихвата рам, распакује га, анализира одредишну IP адресу и на основу своје табеле рутирања одређује да IP пакет треба да се испоручи преко свог другог интерфејса (*int2*), оног који се налази у виртуелној мрежи VLAN2;
5. Рутер P шаље *ARP request* рам преко виртуелне мреже VLAN2 да би одредио MAC адресу станице C4. Комутатори K1, K2, K3 шаљу ARP request рам на све портове конфигурисане тако да припадају виртуелној мрежи VLAN2. Станица C4 на виртуелној мрежи VLAN2 прима *ARP request* рам;
6. Станица C4 на виртуелној мрежи VLAN2 шаље *ARP reply* рам ка комутатору K2. Комутатори K2 и K3 прусмеравају *ARP reply* рам ка рутеру P са одредишном MAC адресом интерфејса *int2* рутера P1 који је на виртуелној мрежи VLAN2;
7. На основу информација из примљеног *ARP reply* рам рутер

[1] *Default gateway*



Слика 2.16 Две виртуелне мреже међусобно повезане комутаторима 2. и 3. слоја

сазнаје одредишну MAC адресу станице C4 и IP пакет који је добио од станице C1 уграђује (укалупљује) у етернет рам са одредишном MAC адресом станице C4;

8. Комутатори К3 и К2 прослеђују етернет рам ка станици C4.

Анализираћемо пример представљен на слици 2.16. У овом примеру К3 је комутатор слоја 3 и постављен је да омогући комуникацију између станица виртуелних мрежа VLAN1 и VLAN2. Да би комутатор К3 обезбедио комуникацију између VLAN1 и VLAN3 мрежа потребно је поставити (конфигурисати) комутирани виртуелни интерфејс[1] (*swi1* и *swi2*) за сваку виртуелну мрежу. Интерфејси *swi1* и *swi2* су интерфејси 3. слоја као и физички интерфејси рутера (*int1* и *int2*) у примеру са слике 2.15. Ови интерфејси имају додељене IP адресе и омогућавају рутирање између виртуелних мрежа VLAN1 и VLAN2.

2.7 Обележавање виртуелних локалних мрежа

Уређаји који се користе у виртуелним локалним рачунарским мрежама могу бити различитих облика, величина и начина конфигурисања. Организација IEEE донела је стандард IEEE802.1Q који дефинише обележавање (означавања) рамова[2] који се користе у

[1] *Switch Virtual Interface*

[2] *Frame tagging*

виртуелним локалним рачунарским мрежама. Суштина обележавања је да се омогући да различите виртуелне локалне мреже користе исту мрежну инфраструктуру (везе између комутатора) а да се при томе очува раздвојеност. Тиме се омогућава да станице које припадају истој виртуелној локалној мрежи, а нису на истом комутатору могу да комуницирају а да при томе не морају да се воде независне везе међу комутаторима за сваку посебну виртуелну локалну мрежу.

Виртуелне локалне рачунарске мреже нису мрежни концепт већ корисна особеност коју уређаји могу да подрже. Једна од кључних карактеристика виртуелне локалне рачунарске мреже је способност да постави портове комутатора у виртуелни бродкаст домен.

Друга кључна карактеристика је могућност означавања[1] етернет рамова са идентификаторима виртуелне локалне рачунарске мреже[2] на основу којих се могу разликовати бродкаст домени. Више ознака се може сместити унутар етернет рамова

На слици 2.17 приказани су формати рамова по стандардима IEEE802.3 и IEEE802.3ac[3]. Измена у односу на етернет/IEEE802.3 рам састоји се у додавању поља од четири бајта између изворишне адресе и поља врста/дужина[4]. На тај начин је максимална величина рама која је била дефинисана за етернет/IEEE802.3 рамове повећана са 1518 бајтова на 1522 бајта.

Прво поље је ознака протокола[5] и дужине је два бајта. Оно најчешће има вредност 8100_{16}. Будући да је овај број већи од 1500 (максимална дужина података у етернет/IEEE802.3 раму), све интерфејсне картице интерпретирају га као тип протокола а не као дужину рама.

Друго поље је дужине 3 бита и омогућава доделу приоритета означеним рамовима (дефинисано у спецификацији IEEE802.1p[6]). Дефинисано је осам класа услуга.

Треће поље (величине једног бита) је канонични идентификатор

[1] *Tag*

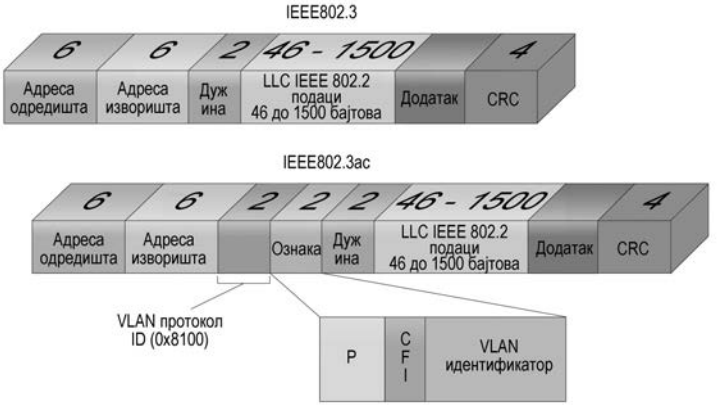
[2] *VLAN identifier*

[3] Стандард IEEE802.1Q је дефинисао концепт виртуелних мрежа са ознакама. Амандманом IEEE802.3ac стандарда дефинисан је формат рама.

[4] Врста (*Type*) је поље код етернет рамова а дужина (*Length*) је поље код IEEE802.3 врсте рамова.

[5] *Protocol ID*

[6] Позната и као спецификација IEEE 802.1D-1998.



Слика 2.17 Формат рамова по стандардима IEEE802.3 и IEEE 802.3ac

формата (CFI[1]) и намена му је да обезбеди пренос рамова IEEE802.5 локалних мрежа (прстен са жетоном) преко етернет/IEEE802.3 локалних мрежа.

Четврто поље је идентификатор виртуелне локалне рачунарске мреже и величине је 12 битова чиме је обезбеђен опсег вредности од 0 до 4095 односно 4096 виртуелних мрежа. Пошто комутатор дода ознаку од четири бајта он поново израчунава контролни збир и убацује га у рам.

Употреба виртуелних локалних рачунарских мрежа са двоструким (и вишеструким ознакама) дефинисана је стандардом IEEE802.1ad[2]. Намена је да се простор идентификатора виртуелних локалних мрежа прошири додавањем нове ознаке на већ означене рамове. Проширивање простора идентификатора виртуелних мрежа омогућава да корисници имају своје сопствене виртуелне локалне мреже у оквиру виртуелне мреже коју им додељују интернет посредници.

Предности у примени виртуелних локалних мрежа су вишеструке. Навешћемо неке од њих:

- Сигурност. Групе корисника које имају податке који су поверљиви изоловани су од остатка мреже смањујући могућност откривања поверљивих података;

[1] *Canonical Format Identificator*

[2] Среће се и као IEEE802.1QinQ, а назива се и премошћавање добављача интернет услуга (*provider bridging*).

- Смањивање трошкова. Настаје као последица тога да што се смањује потреба за скупом надоградњом мреже и што се ефикасније користи постојећи капацитет мреже;
- Високе перформансе. Деобом равне мреже која ради на 2. слоју у више логичких радних група умањује се непотребан саобраћај у мрежи и побољшавају перформансе мреже;
- Ублажавање бродкаст олује. Деобом мреже у више мањих виртуелних мрежа смањује се број уређаја који учествују у бродкаст олуји[1];
- Побољшава ефикасност особља које администрира мрежу. Виртуелне мреже олакшавају надзор мреже пошто корисници са истим мрежним захтевима користе исту виртуелну мрежу.

IEEE стандард	**Година доношења стандарда**	**Опис**
Експериментални Етернет	1972.	2.94Mb/s преко коаксијалног кабла 50Ω
Етернет II (DIX v2.0)	1982.	10Mb/s преко танког коаксијалног кабла[2] - рамови садрже поље тип и користи у скупу протокола на Интернету[3]
IEEE 802.3	1983.	10BASE5 10Mb/s преко дебелог коаксијалног кабла истог Етернет II али је поље тип замењено сапољем дужине а 802.2 LLC заглавље следи 802.3 заглавље
802.3a	1985	10BASE2 10Mb/s преко танког коаксијалног кабла
802.3b	1985	Одбачени стандард чија је идеја била преношење 10Mb/s етернет сигнала кроз 75Ω на растојању 3600m
802.3c	1985	10Mb/s спецификација за обнављиваче
802.3d	1987	FOIRL[4] оригинални стандард за Етернет преко оптичког влакна.

[1] *Broadcast Storming*
[2] *Thinnet, cheapernet*
[3] *Internet protocol suite*
[4] *Fiber-Optic Inter-Repeater Link*

IEEE стандард	Година доношења стандарда	Опис
802.3e	1987	StarLAN је прва имплемнтација етернет мреже са телефонским упреденим парицама и брзином 1Mb/s. Ознака је: 1BASE5 или StarLAN
802.3i	1990	10BASE-T, 10Mb/s преко упредених парица
802.3j	1993	10BASE-F, 10Mb/s преко оптичких влакана
802.3u	1995	100BASE-TX, 100BASE-T4, 100BASE-FX брзи Етенет 100Mb/s. Међусобни договор[1] око брзина и дуплексности
802.3x	1997	Потпуни дуплекс и контрола тока
802.3y	1998	100BASE-T2 100Mb/s преко упредених парица лошијег квалитета
802.3z	1998	1000BASE-X Gb/s Етернета брзине 1Gb/s преко оптичких влакана.
802.3-1998	1998	Ревизије основног стандарда које укључују претходне амандмане и исправке
802.3ab	1999	1000BASE-T Gb/s Етернет преко упредених парица брзином 1Gb/s
802.3ac	1998	Максимална величина рама је проширена на 1522 бајта да би обезбедила Q-ознаку[2] која укључује 802.1Q VLAN информације и 802.1p информације о приоритетима
802.3ad	2000	Агрегација (припајање) паралелних линкова, пребачен је у IEEE802.1AX
802.3-2002	2002	Ревизије основног стандарда које укључују претходне амандмане и исправке
802.3ae	2003	10Gb/s Етернет преко оптичких влакана: 10GBASE-SR, 10GBASE-LR, 10GBASE-ER, 10GBASE-SW, 10GBASE-LW, 10GBASE-EW
802.3af	2003	Енергетско напајање преко Етернета (PoE[3])

[1] *Autonegotiation*
[2] *Q-tag*
[3] *Power over Ethernet*

IEEE стандард	Година доношења стандарда	Опис
802.3ah	2004	Етернет у приступној мрежи – првом километру[1]
802.3ak	2004	10GBASE-CX4 10Gbs, први станадрд за бакарне каблове брзине 10Gbs који су лошијих карактеристика од 10GBASE-T
802.3-2005	2005	Ревизије основног стандарда које укључују претходне амандмане и исправке
802.3an	2006	10GBASE-T 10G/s (1,250 MB/s) Етернет преко неоклопљених упредених парица (UTP[2])
802.3ap	2007	Плоча са међусобно повезаним етернет конекторима[3] брзине 1Gb/s и 10Gb/s преко штампаних плоча[4]
802.3aq	2006	10GBASE-LRM 10Gb/s – Етернет преко вишеугаоног оптичког влакна[5]
802.3as	2006	Проширење рама
802.3at	2009	Унапређење енергетског напајање преко Етернета
802.3au	2006	Изолациони захтеви за PoE (802.3-2005 Cor 1)
802.3av	2009	10Gb/s етернет пасивна оптичка мрежа (EPON[6])
802.3-2008	2008	Верзија основног стандарда која укључује 802.3an/ap/aq/as амандман са исправкама штампарских грешкама и изменама. Агрегација линка је пребачена у 802.1AX
P802.3az	2010	Енергетски ефикасан Етернет[7]

[1] *Ethernet in the First Mile*
[2] *Unshielded twisted pair*
[3] *Backplane*
[4] *Printed circuit boards*
[5] *Multimode fiber*
[6] *Ethernet Passive Optical Network*
[7] *Energy Efficient Ethernet*

IEEE стандард	Година доношења стандарда	Опис
802.3ba	2010	Етернет брзине 40Gb/s и 100Gb/s. Код етернет мреже брзине 40Gb/s линк је дужине 1m (плоче са међусобно повезаним етернет конекторима), 10m за бакарне каблове и 100m за оптичка влакна. Спецификација за брзину 100Gb/s односи се на линкове дужине 10km и 40km преко једноугаоног оптичког влакна[1]
802.3-2008/Cor 1	2009	Увећава кашњење које је било недовољно за10G/s (назив радне групе је био 802.3bb)
802.3bc	2009	Иновира Етернет TLV[2], претходно специфициран у Annex F, IEEE 802.1AB (LLDP[3])
802.3bd	2010	Контрола тока заснована на приоритетима[4].
802.3be	2011	Креира IEEE 802.3.1 MIB (Management Information Base) дефиниције за Етернет које укључују различите IETF RFC документе и 802.1AB анекс F у један главни документ
802.3bf	2011	Обезбеђује прецизну индикацију преноса и пријема иницијалног времена одређених пакета као што се захтева да би се подржао стандард IEEE P802.1AS
802.3bg	2011	Обезбеђује 40Gb/s PMD[5] који одговара постојећем једноугаоним оптичким влакнима (SMF[6]) 40Gb/s са клијентским интерфејсима (OTU[7]/STM[8] -256/OC[9] -768/40G POS[10])

Табела 2.1 Преглед IEEE802.3 серије стандарда

[1] *Single-mode fiber*
[2] *Type, Length, Values*
[3] *Link Layer Discovery Protocol*
[4] *Priority-based Flow Control*
[5] *Physical Medium Dependent*
[6] *Single-Mode optical Fiber*
[7] *Optical Transport Network*
[8] *Packet over SONET/SDH*
[9] *Optical Carrier*
[10] *Packet over SONET/SDH*

2.8 Питања и задаци

1. Описати однос архитектура IEEE802 и OSI.
2. Набројати све функције које реализује слој везе.
3. Који се рамови размењују код вишеструког приступа са избегавањем колизије (CSMA/CA)? Описати начин размене.
4. Која поља су идентична а која различита у форматима Етернет II и IEEE802.3 рамова?
5. Набројати и описати протоколе 2. слоја који се данас најчешће користе.
6. Навести где се све користи PPP протокол. Објаснити.
7. Који се протоколи из скупа PPP протокола највише користе?
8. Који протоколи за аутентификацију се користе код PPP протокола?
9. Зашто се врши сегментирање рачунарских мрежа?
10. Описати и међусобно упоредити основне карактеристике уређаја за међусобно повезивање у рачунарским мрежама.
11. Који уређај је потребан да би се повезале две виртуелне локалне рачунарске мреже?
12. Посматрајући слику 2.16 описати комуникацију станица C3 и C5 виртуелне мреже VLAN1.
13. Посматрајући слику 2.15 описати комуникацију станице C3 виртуелне мреже VLAN1 и станице C6 виртуелне мреже VLAN2.
14. Који IEEE стандард дефинише обележавање (означавања) рамова?
15. Описати намене четири поља стандарда IEEE802.3ac за обележавање виртуелних мрежа.
16. Како се реализује механизам приоритета код виртуелних локалних мрежа?
17. Којим стандардом су дефинисани приоритети у локалним мрежама?
18. На ком се месту у виртуелној локалној мрежи додаје ознака?
19. Како се користе вишеструке ознаке у виртуелним локалним мрежама?
20. Које су све предности у примени виртуелних локалних мрежа?

3. Мрежни слој на Интернету

Мрежни слој задужен је за пребацивање пакета и одабир оптималних путева од извора до одредишта. Према томе, мрежни слој је најнижи слој који се бави преносом с једног краја мреже на други крај. Да би постигао додељени му задатак мрежни слој треба да буде упознат са:

- топологијом подмреже,
- распоредом и везама свих уређаја који повезују подмреже.

У следећим одељцима анализираћемо неке од карактеристика мрежног слоја које су битне за пројектовање рачунарских мрежа и услуге које он треба да обезбеди транспортном слоју.

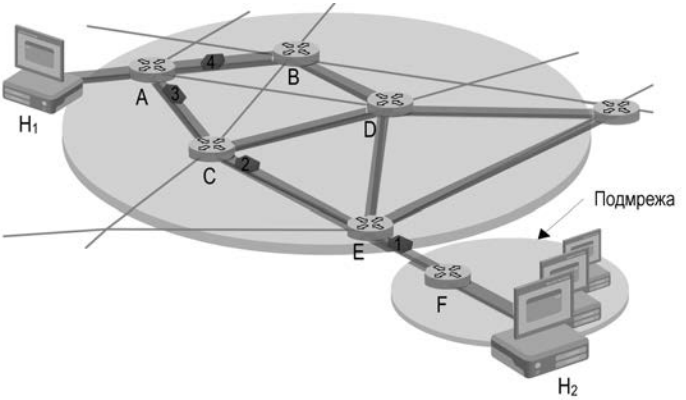
3.1 Комутација пакета

Систем у коме је имплементиран протокол мрежног слоја представљен је на слици 3.1. Главне компоненте система који називамо подмрежа јесу:

- чворови, тј. комутациони уређаји - рутери,
- трансмисионе линије које повезују чворове приказане унутар осенчене елипсе.

Уређаји који желе међусобно да комуницирају називају се крајње станице (хостови). Станица H_1 директно је везана са рутером A подмреже преко изнајмљене линије. С друге стране, станица H_2 налази се на локалној рачунарској мрежи корисника која је са рутером F повезана на подмрежу. Овај рутер такође користи изнајмљену линију као везу до подмреже. Описани систем: подмрежа, станице, локална рачунарска мрежа и чворови (рутери) користе се на следећи начин:

- крајња станица која има пакет за слање шаље га најближем чвору (рутеру) или на својој мрежи или преко тачка-тачка везе до подмреже,
- чвор (рутер) прихвата пакет, смешта га у локалну меморију и проверава да ли је са грешком,



Слика 3.1 Окружење протокола мрежног слоја

- уколико је пакет без грешке шаље се следећем чвору све док не дође до одредишне станице.

Описани процес назива се комутација пакета „смести и проследи"[1].

3.1.1 Услуга која се пружа транспортном слоју

Мрежни слој пружа услугу транспортном слоју на интерфејсу између мрежног слоја и транспортног слоја. Услуга мрежног слоја пројектована је да задовољи следеће циљеве:

- услуга треба да је независна од начина на који се пакети преусмеравају,
- транспортни слој треба да је заштићен од броја, типа и топологије чворова мреже,
- мрежне адресе доступне транспортном слоју треба да користе јединствени (униформни) бројни систем, без обзира да ли се везе остварују преко локалних рачунарских мрежа или мрежа ширег градског подручја.

С овим циљевима на уму пројектанти мрежног слоја имају доста слободе у спецификацији услуга које се нуде транспортном слоју. Основни проблем своди се на то да ли мрежни слој треба да нуди услугу без успоставе везе[2] или услугу са успоставом везе[3].

[1] *Store and Forward*
[2] *Connectionless*
[3] *Connection-oriented*

Интернет заједница и пројектанти на њу ослоњени тврде да је једини задатак чворова (рутера) да преусмере (комутирају) пакете. Из њиховог дугогодишњег искуства подмрежа је непоуздана без обзира како је пројектована. Зато крајње станице треба да имају то на уму и раде проверу (детекцију и корекцију) грешака и контролу тока.

Ово гледиште наводи на закључак да би услуга мрежног слоја (мрежна услуга) требало да буде без успоставе везе. Вођење рачуна о контроли тока и редоследу пакета сувишно је пошто то већ и крајње станице раде. Претпоставка је да сваки пакет мора да носи пуну адресу одредишта јер се шаље независно од својих претходника.

Друга групација (ослоњена на искуства оператера фиксне телефоније) залаже се за то да подмрежа треба да обезбеди поуздану услугу са успоставом везе. Тврде да је 100 година успешног искуства светске јавне телефонске мреже за такво опредељење изванредан узор. По њима је квалитет услуге (QoS[1]) важан чинилац а тешко га је остварити без успоставе веза. Квалитет услуге нарочито је битан за интерактивни саобраћај у реалном времену, као што су пренос гласа и видео садржаја.

Ова два међусобно супротстављена правца оличена су у Интернет и ATM[2] мрежама. Интернет нуди транспортном слоју услугу без успоставе везе а ATM нуди услугу са успоставом везе. Без озира на опредељења гаранција квалитета услуге постаје незаобилазан параметар о коме све мреже морају да воде рачуна па и Интернет.

С обзиром на тип услуге који се нуди, две различите организације су могуће. Ако је у питању услуга без успоставе везе, пакети се шаљу у подмрежу појединачно и преусмеравају (рутирају) независно један од другог. Не постоји унапред дефинисано правило или стратегија. Пакети се називају датаграми[3] (аналогија са телеграмима) а подмрежа се назива подмрежа са датаграмима[4]. Ако се користи услуга са успоставом везе, мора се успоставити виртуелна путања од изворишног до одредишног рутера пре слања пакета. Ова веза означава се као виртуелно коло[5] у аналогији

[1] *Quality of Service*
[2] *Asynchronous Transfer Mode*
[3] *Datagram*
[4] *Datagram subnet*
[5] *Virtual circuit*

са колом које се успоставља у телефонском систему. Одговарајућа подмрежа назива се подмрежа са виртуелним колима[1].

3.1.2 Услуга без успоставе везе

У овом делу анализираћемо како ради подмрежа без успоставе везе - са датаграмима (слика 3.2). Претпоставићемо да процес P_1 на станица 1 (H_1) има дугачку поруку за процес P_2 на станици 2 (H_2). Процес P_1 предаје поруку свом транспортном слоју са успоставом везе да би је он доставио процесу P_2. Покреће се извршни кôд транспортног слоја на станици 1 (H_1), који је обично део оперативног система. Додаје се заглавље транспортног слоја испред поруке[2] и као целина предаје се мрежном слоју[3].

Ако је порука четири пута дужа од максималне величине пакета, мрежни слој мора да је разбије у четири пакета и пошаље сваки од њих један за другим рутеру А користећи неки од протокола за тачка-тачка везе (нпр. PPP[4] протокол). Сваки рутер има своју табелу записа на основу којих усмерава даље пакете на путу до одредишта. Запис у табели састоји се од адресе одредишта и излазне линије која се користи да би се стигло до одредишта. Могу се користити само директне везе. На пример рутер А (слика 3.2) има само две излазне линије: ка рутеру В и ка рутеру С. То значи да сваки пакет који стигне мора бити послат једном од ова два рутера, чак и ако је крајње одредиште неки други рутер. Почетна табела за усмеравање рутера А на слици је представљена ознаком „почетак".

Када дођу у чвор (рутер) А пакети са редним бројевима 1, 2 и 3 само се кратко задржавају (колико је потребно да се израчуна контролни збир). Затим се усмеравају ка рутеру С, као што је у табели рутера А и записано. Пакет са редним бројем 1 се прослеђује до рутера Е и затим до рутера F. У рутеру F пакет са редним бројем 1 укалупљује се у рам слоја везе[5] и преко локалне рачунарске мреже шаље се станици H_2. Пакети са редним бројевима 2 и 3 следе исту путању.

Када пакет са редним бројем 4 доспе до рутера А, он тај пакет усмерава ка рутеру В без обзира што је одредиште непромењено. Из

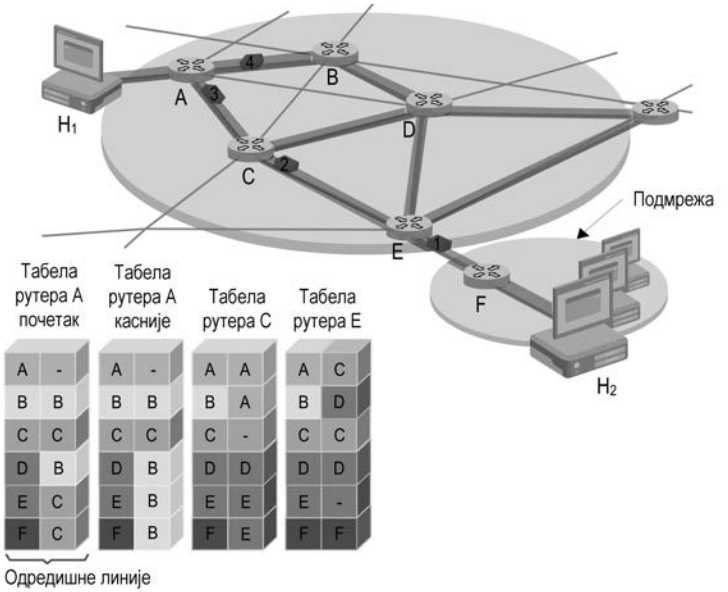
[1] *Virtual circuit subnet*

[2] Погледати поглавље о OSI референтном моделу.

[3] Још једна процедура унутар оперативног система.

[4] *Point-to-Point Protocol*

[5] Формира се Етернет (или IEEE802.3) рам.



Слика 3.2 Табеле рутирања пакета (датаграма) у подмрежи

неког разлога рутер A је одлучио да усмери пакет са редним бројем 4 различитом путањом од оне на коју су усмерена претходна три пакета. Највероватнији разлог да рутер A промени своју табелу рутирања је обавештење о загушењу у саобраћају негде дуж путање ACE. Нова путања је на слици 3.2 означена са „касније".

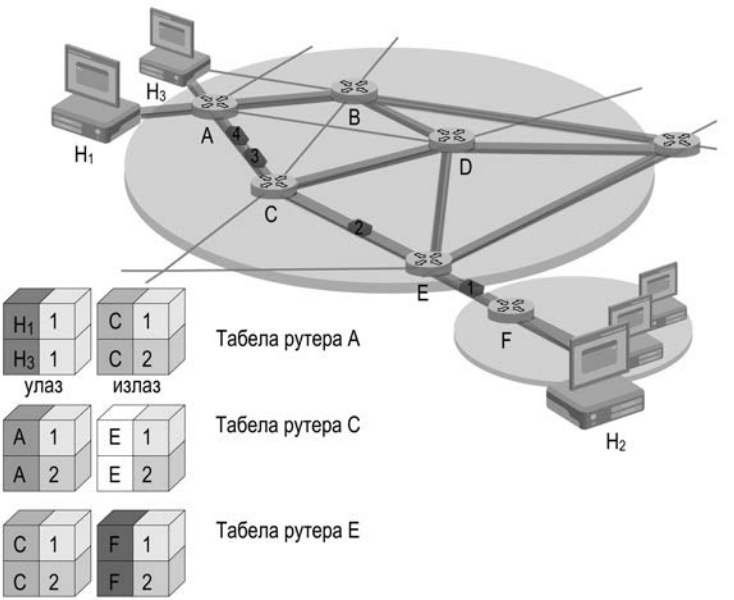
Управљање табелама и доношење одлуке о рутирању назива се алгоритам за рутирање.

3.1.3 Услуга са успоставом везе

За услугу са успоставом везе потребна је подмрежа са виртуелним колима. Концепт виртуелног кола заснован је на идеји да код преусмеравања пакета не треба бирати путању за сваки пакет посебно. Путању од изворишне до одредишне станице (виртуелну везу) треба одабрати у току успостављања везе и записати је у табелама рутера. Она се користи за размену свих пакета (саобраћај) те везе. Принцип је идентичан као у телефонском систему. Кад се веза прекине, виртуелно коло се прекида. Сваки пакет те везе носи идентификатор виртуелне везе (кола) којој припада.

Као пример сведочи ситуација на слици 3.3. Станица H_1 је успоставила везу са станицом H_2. Први запис у свакој појединачној табели рутера представља ознаку тог виртуелног кола. Прва линија табеле рутера А јасно упућује да пакет са идентификатором везе 1, који шаље H_1, треба проследити рутеру С, задржавајући исти идентификатор везе. Слично, први запис у табели рутера С прослеђује пакет до Е, такође са идентификатором 1.

Шта се дешава ако станица H_3 такође жели да успостави везу са станицом H_2? Изворишна станица опет бира идентификатор 1 и тражи од подмреже да успостави виртуелно коло. Овом приликом посматра се други запис у табелама. Јавља се неусаглашеност. Рутер А лако може да разликује пакете који стижу преко везе 1 које шаље станица H_1 од пакета везе који стижу преко везе 1 а које шаље H_3. Рутер С то не може. Због тога рутер А додељује други идентификатор везе излазном саобраћају. Рутери морају имати могућност замене идентификатора везе у излазним пакетима. Понегде се ово зове замена ознаке[1].



Слика 3.3 Рутирање у подмрежи са виртуелним колом

[1] *Label switching*

3.1.4 Упоређивање услуге са успоставом и без успоставе везе

У овом одељку анализираћемо предности и мане услуге са успоставом везе и услуге без успоставе везе. Сигурно је да ниједна од услуга не може да задовољи све захтеве. Посматрајмо случај када се преносе пакети мале величине: комплетна одредишна адреса која се налази у сваком пакету који се преноси представља значајну количину некорисних података (премашење[1]) у односу на количину корисних података[2]. Последица је пропусни опсег који се троши бескорисно.

У подмрежи са виртуелним колима пакети садрже идентификаторе везе уместо пуне адресе одредишта. Цена којом се плаћа ова погодност је меморијски простор потребан за смештање записа о вези у табели сваког рутера дуж путање виртуелног кола.

Виртуелна кола захтевају фазу успоставе и раскидања везе што одузима време и троши ресурсе. С друге стране одлука о томе шта урадити с пристиглим пакетом у подмрежи са виртуелним колом је једноставна и брза: рутер, користећи идентификатор виртуелног кола, проналази у својој табели одговарајући запис о излазној линији преко које усмерава (шаље) даље пакет. У подмрежи са датаграмима потребна процедура претраживања да би се пронашао (лоцирао) одговарајући запис сложена је и траје знатно дуже.

Још један податак је критеријум за упоређивање: величина меморије која је потребна рутеру за смештање табела. Подмрежа са датаграмом захтева да постоји запис о путањи до сваког одредишта. Подмрежа са виртуелним колом има само запис о сваком виртуелном колу. Ова предност је ипак помало варљива с обзиром на то да се пакети, који се размењују у току успоставе везе, такође рутирају. Они садрже пуну изворишну и одредишну адресу као и датаграми.

Виртуелна кола имају значајну предност у гарантовању квалитета услуга и избегавању загушења у саобраћају пошто се ресурси као што су меморијски простор, пропусни опсег и сл. могу унапред, приликом успостављања везе, резервисати. Једном кад пакети почну да пристижу преко успостављене везе неопходни пропусни опсег и капацитет рутера

[1] *Overhead*
[2] *Payload*

биће им на располагању. Код подмреже са датаграмом избегавање загушења велики је проблем.

У системима обрада трансакција губитак времена потребног да се успостави и раскине виртуелно коло може лако да умањи предност коју даје поузданост употребљеног кола. Ако се очекује да ће већи део саобраћаја бити оваквог типа употреба виртуелног кола у подмрежи нема смисла.

С друге стране перманентна виртуелна кола која се ручно успостављају и трају месецима или годинама могла би у овом случају да буду од користи.

Виртуелна кола имају проблем рањивости. Ако се рутер поквари избришу се сви записи из меморије. Чак и ако рутер брзо настави са радом, записи о свим виртуелним колима која пролазе кроз њега изгубљени су.

Такође, када је комуникациона линија у прекиду сва виртуелна кола која користе ту линију су прекинута. Да би се наставила размена пакета између изворишта и одредишта мора се веза поново успоставити.

С друге стране ако се поквари рутер у подмрежи са датаграмима на губитку су само они корисници чији су пакети тада и на том рутеру чекали у реду за обраду. Током слања дугачке секвенце пакета, пошто могу да мењају путању, рутери кроз подмрежу са датаграмима могу и да равномерно расподељују саобраћај.

3.2 Мрежни слој на Интернету

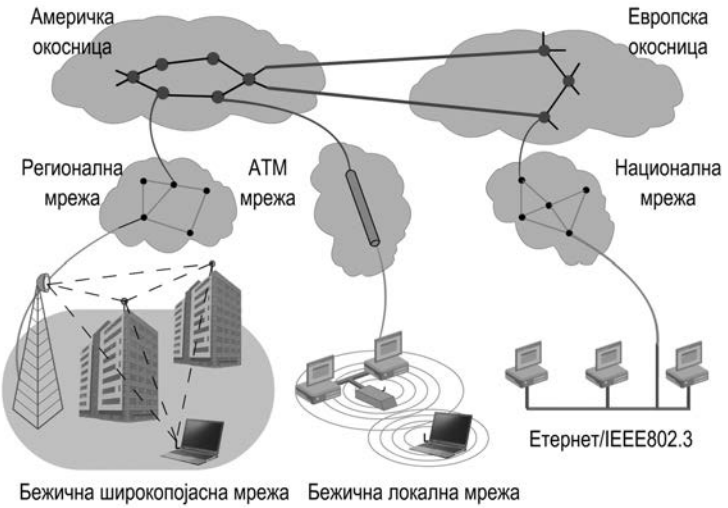
Протокол мрежног слоја који се највише користи је интернет протокол (IP[1] протокол). Због тога ће се у овом поглављу сва пажња посветити детаљној анализи IP протокола (актуелних верзија IPv4 и IPv6) као пример и *de facto* стандард за протокол мрежног слоја.

На мрежном слоју Интернет се може видети као колекција подмрежа међусобно повезаних рутерима. Не постоји стварна структура мреже али постоји неколико окосница[2].

Оне се састоје од веза (линкова) великог пропусног опсега и брзих рутера. На главне везе прикључене су регионалне мреже а на њих локалне

[1] *Internet Protocol*

[2] Кичма, окосница (*backbone*)



Слика 3.4 Интернет је скуп међусобно повезаних мрежа

мреже већих институција, универзитета, компанија и добављача интернет услуга (слика 3.4). Главни везивни елеменат Интернета је IP протокол.

3.2.1 Основна обележја IP протокола

Протокол IP обезбеђује услугу без успоставе везе између крајњих система. Постоје бројне предности и недостаци оваквог приступа који су анализирани у претходном одељку. Додаћемо само још неке који су битни за Интернет као глобалну рачунарску мрежу:

- Интернет може да подржи различите врсте мрежа од којих су неке без успоставе везе,
- интернет услуга без успоставе везе је најпогоднија за транспортне протоколе без успоставе везе.

Слика 3.5 приказује:

- уобичајен пример употребе IP протокола: две локалне рачунарске мреже међусобно повезане[1] са мрежом са штафетним[2] преносом (FR);

[1] Архитектура протокола IEEE802 састоји се од физичког слоја; слоја контроле везе (MAC) који се односи на адресирање и контролу грешака и слоја контроле логичке везе (LLC), који се односи на логичке везе и идентификовање корисника LLC подслоја.

[2] *Frame relay*. Среће се и под називом мрежа за преусмеравање рамова. Спада у технологију мрежа ширих подручја WAN (*Wide Area Network*).

- поступак IP протокола за размену података између станице A из једне локалне рачунарске мреже L_1 и станице B из друге локалне рачунарске мреже L_2 преко мреже са штафетним преносом;
- архитектуру протокола и формат јединице података у свакој фази. Крајњи системи и рутери морају имати исти IP протокол. Такође, крајњи системи морају имати исте протоколе изнад IP протокола. Посредни рутери треба да имплементирају протоколе све до IP протокола.

Са виших слојева целина IP крајњег система A прима блок података који треба да се пошаље до крајњег система B. Додаје заглавље података (у тренутку t_1) које, између осталог, садржи и глобалну интернет адресу крајњег система B. Та адреса се логички дели на два дела: идентификатор мреже и идентификатор крајњег система. Комбинација IP заглавља и података са вишег слоја је јединица података IP протокола или датаграм.

Датаграми се пакују у јединице података протокола локалне рачунарске мреже L_1 која се састоји од:

- заглавља LLC подслоја у тренутку t_2,
- заглавља и краја рама (приколица) MAC подслоја у тренутку t_3.

Затим се шаљу рутеру који:

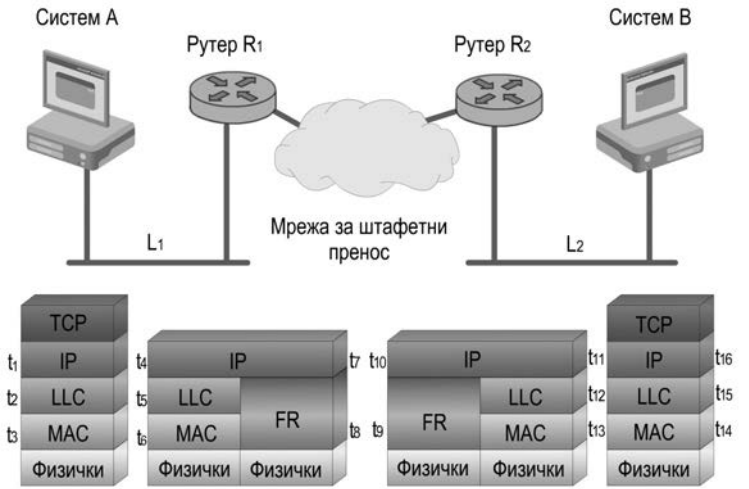
- скида заглавље локалне рачунарске мреже L_1, тј. подслојева LLC и MAC како би прочитао IP заглавље у тренутку t_6,
- поново пакује датаграм са заглављем рама протокола за штафетни пренос у тренутку t_8 и
- прослеђује га преко мреже ширег подручја до другог рутера.

Рутер на другом крају мреже:

- скида заглавље рама протокола за штафетни пренос у тренутку t_{10},
- поново пакује у јединицу података протокола локалне рачунарске мреже L_2 која се састоји од:
  - заглавља LLC подслоја у тренутку t_{12},
  - заглавља и краја рама (приколица) MAC подслоја у тренутку t_{13}.
- шаље је крајњем систему B.

Крајњи систем B:

- скида заглавља локалне рачунарске мреже L_2, тј. подслојева LLC и MAC како би прочитао IP заглавље у тренутку t_{16},



Слика 3.5 Начин рада мрежног протокола на Интернету: IP протокола

Погледајмо овај пример детаљније. Крајњи систем - станица А има пакет који треба да пренесе крајњем систему - станица В; пакет носи IP адресу станице В. Станица А има IP модул који препознаје да је одредиште - станица В у другој мрежи. Први корак је да станица А пошаље податке рутеру, у овом случају рутеру R_1. Да би се то урадило, IP слој прослеђује пакет нижем слоју (у овом случају LLC подслоју) налажући му да га пренесе рутеру R_1. Подслој LLC преноси ове информације MAC подслоју који у своје заглавље на месту одредишне адресе убацује адресу MAC подслоја рутера R_1. Сада се рам, који се преноси локалном рачунарском мрежом L_1, састоји од: података изнад TCP слоја и заглавља свих слојева кроз које пролази: TCP заглавље, IP заглавље, LLC заглавље, MAC заглавље и приколице[1] (слика 3.5). Пакет се на даље преноси локалном рачунарском мрежом L_1 све до рутера R_1. Рутер уклања заглавља MAC и LLC подслојева и анализира IP заглавље да би утврдио одредиште података, у овом случају станица В. Рутер R_1 мора донети одлуку о рутирању. Постоје три могућности:

1. Одредишна станица В (крајњи систем) директно је повезана са једном од мрежа за коју је рутер везан. Ако је тако, рутер шаље пакет директно до одредишта;

[1] Сваки слој везе додаје и почетак (заглавље) рама и крај (*trailer*). У даљем тексту једноставности ради употребљаваћемо само термин заглавље али треба имати на уму да се подразумевају оба дела.

2. Да би дошао до одредишта пакет мора да пређе преко једног или више рутера. Значи да рутер R_1 мора донети одлуку о рутирању: ком рутеру треба послати пакет? У случајевима 1 и 2 IP слој рутера шаље пакет нижем слоју са IP адресом одредишта - станице B (слика 3.6);
3. Рутер не зна одредишну адресу. У овом случају рутер шаље поруку о грешци[1] изворишту пакета којом указује да је дошло до грешке.

У овом примеру подаци морају да пређу преко рутера R_2 пре него што дођу до одредишта. То значи да рутер R_1 прави нови рам у коме је:

- заглавље рама заглавље слоја везе мреже за штафетни пренос,
- подаци су пакет- јединица података мрежног слоја (NPDU[2]).

Заглавље мреже са штафетним преносом указује на виртуелну везу са рутером P_2. Кад овај рам стигне до рутера P_2 заглавље и крај рама се уклањају. Рутер проналази да је ова IP јединица податка мрежног слоја упућена станици B која је директно повезана са мрежом за коју је овај рутер везан. Рутер зато прави рам у коме је изворишна адреса другог слоја његова адреса, одредишна адреса другог слоја је адреса станице B а рам се шаље на локалну рачунарску мрежу L_2. Подаци коначно стижу до станице B која уклања заглавља другог и трећег слоја (LLC, MAC и IP заглавља).

Пре него што се подаци проследе даље, рутер ће можда морати да пакет подели у мање целине (јединицу података) да би је прилагодио мањој максималној величини пакета одредишне мреже. Свака од новонасталих јединица постаје независна јединица података која се пакује у рам нижег слоја и ставља у ред чекања за даљи пренос. На мрежном слоју те мање целине називају се фрагменти а процес деобе назива се фрагментација.

Рутер може да ограничи листу чекања за сваку мрежу за коју је везан како би избегао да спорија мрежа успорава бржу мрежу. Када више нема места у реду за чекање, односно када се капацитет попуни,

[1] *Error message*. Постоје специјалне врсте порука о грешци.

[2] *Network Protocol Data Unit* или *IP Protocol Data Unit* (за интернет протокол)

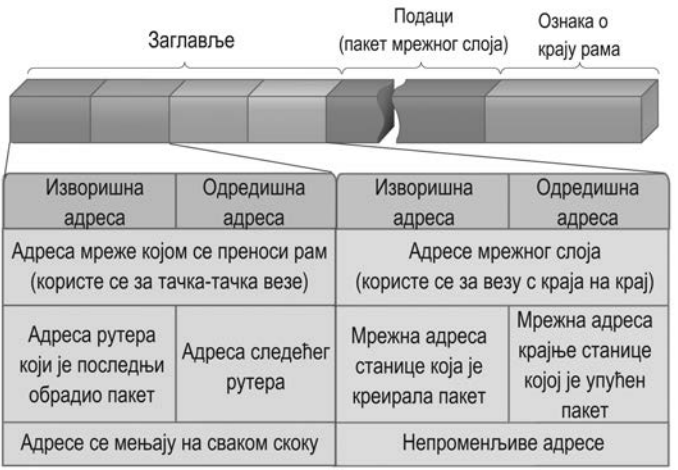
новопристигли пакети се једноставно одбацују. Овај процес се понавља у онолико рутера колико их је потребно прећи да би пакет података стигао до одредишта. На исти начин као што то рутер ради одредишни крајњи систем (крајња станица или хост) преузима IP пакет са мреже. Ако је на путањи извршена деоба (фрагментација) пакета IP слој одредишног система сакупља пристигле делове (фрагменте) и прослеђује их вишем слоју тек када све делове споји у једну целину.

Услуга коју обезбеђује IP протокол сматра се непоузданом[1] зато што се не гарантује да ће подаци стићи до одредишта или да ће стићи истим редоследом којим су и послати. Задатак је вишег слоја (транспортног) да води рачуна да крајњи системи отклони све грешке настале у преносу[2].

3.3 Интернет протокол верзије 4

У овом делу анализираћемо интернет протокол верзије 4 који се означава са IPv4. Протокол IPv4 дефинисан је документима RFC791, RFC950, RFC919, RFC922, и RFC2474.

Коначно се интернет протокол верзије 4 (IPv4) замењује интернет протоколом верзије[3] 6 (IPv6). Интернет протокол (IP протокол) део је



Слика 3.6 Структура рама који се преноси кроз мрежу

[1] У литератури се среће и појам најбољуа могућа услуга (*best effort*).

[2] Користи се и термин опоравак од грешке (*error recovery*).

[3] Постоји и интернет протокол верзије 5 (IPv5) али није у јавној употреби.

скупа TCP/IP протокола[1] и најшире коришћени протокол за међусобно повезивање рачунарских мрежа. У овом поглављу везано за IP протокол анализираћемо:

- интерфејс на транспортном слоју описујући услуге које IP обезбеђује,
- проблеме у пројектовању,
- формат и механизам протокола.

Слој IP обезбеђује две примитиве услуга ка вишем слојевима[2] (слика 3.7). То су:

- примитива за слање - *Send*, која се користи када корисник захтева од IP слоја да пренесе податке.,
- примитива за испоруку - *Deliver*, која се користи да би IP слој обавестио кориснике да су подаци стигли.

Чиниоци који утичу на пројектовање система са IP протоколима су следећи:

- време трајања (животни век) пакета ,
- деоба пакета (фрагментовање) и поновно повезивање,
- контрола грешке,
- контрола тока,
- адресирање и
- рутирање.

3.3.1 Време трајања пакета

Ако се користи динамичко или алтернативно рутирање постоји могућност да пакет бесконачно дуго кружи Интернетом. Ово је недопустиво из два разлога. Прво, бескрајно дуго кружење пакета троши ресурсе. Друго, транспортни протокол може да зависи од ограничења у трајању пакета. Да би се избегли ови проблеми сваком пакету се додељује време трајања TTL[3] (животни век). Када прође то време пакет бива одбачен. У верзији IPv6 протокола ово поље се назива ограничење скока[4].

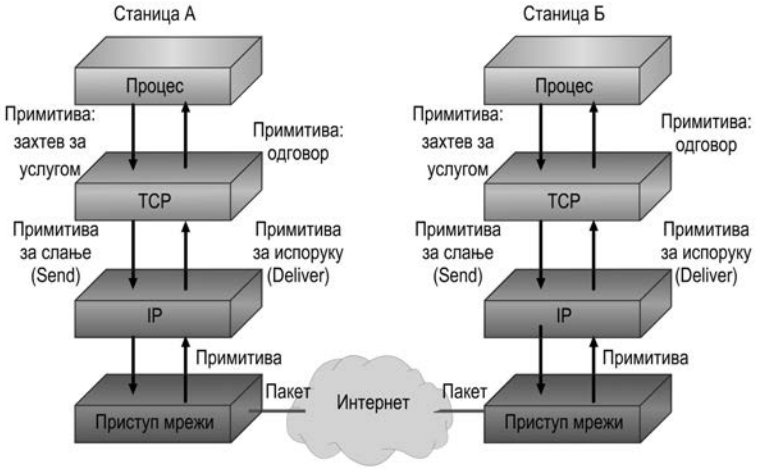
---

[1] *TCP/IP Protocol Suite*

[2] Примитиве услуга описане су детаљно у 1. поглављу.

[3] *Time To Live*

[4] *Hop limit*



Слика 3.7 Примитиве услуга IP слоја: слање (*Send*) и испорука (*Deliver*)

Најједноставнији начин да се реализује трајање пакета је помоћу бројача[1]. Сваки пут када пакет прође кроз рутер бројач се декрементира (умањује за један). Такође, трајање пакета може да означава време. Ово захтева од рутера да зна колико је времена прошло од када је пакет или део тог пакета последњи пут прошао кроз рутер, и да за толико умањи вредност поља TTL. Да би то било могуће потребна је глобална временска синхронизација. Предност коришћења стварне временске величине је што се може користити у алгоритму за поновно повезивање пакета.

3.3.2 Деоба и поновно спајање делова пакета

Појединачне мреже у Интернету могу назначити различите максималне величине пакета. Не би било препоручљиво условљавати исту величину пакета кроз различите мреже. Према томе рутер мора пристигле пакете, пре него што их проследи следећој мрежи, да дели у мање целине које се називају сегменти или фрагменти.

Ако се пакети могу делити на свом путу кроз мрежу (по могућству више од једном) поставља се питање где ће се они поново спајати. Најлакше решење је да се поновно спајање делова одради у самом одредишту.

[1] Назива се и бројач скокова (*hops*).

Ако би се дозволило поновно спајање у рутерима на међупутањи, могло би да дође до следећих неповољних околности:

- потреба за великим меморијским простором у рутеру. Постоји ризик да се цео меморијски простор заузме за складиштење делова пакета;
- сви делови пакета морају да прођу кроз исти рутер, што спречава коришћење динамичког рутирања.

На слоју мреже на Интернету (IP слоју) спајње делова пакета обавља се у одредишном систему. Код деобе IPv4 пакета из његовог заглавља користе се следеће информације:

- идентификатор пакета (ID) чија је намена да одреди крајње системе између којих се размењује пакет. Састоји се од изворишне и одредишне IPv4 адресе и броја који указује протокол ког слоја је генерисао податак (нпр. TCP протокол);
- дужина података која представља дужину корисничких података, исказана у октетима (бајтовима);
- одступање које указује на позицију дела корисничких података у пољу података оригиналног пакета исказано као умножак од 64 бита;
- ознака M[1] показује да ли постоји (или не) још делова оригиналног пакета.

Изворишни систем формира пакет[2] такав да:

- вредност поља о дужини података одговара целој дужини података,
- одступање има вредност нула,
- ознака M постављена је на вредност нула.

Када IPv4 целина рутера дели пакет велике дужине у два дела она:

- прави два нова пакета и копира заглавље оригиналног пакета у оба дела;
- дели долазеће корисничке податке на два приближно једнака дела водећи рачуна о 64-битној граници. Добијене делове смешта у нове пакете. Први део мора бити умножак од 64 бита;
- поље дужине података првог новог пакета поставља се на вредност која је једнака дужини уметнутих података. Бит M се

[1] *More*

[2] То је пакет (датаграм) који називамо оригинални пакет.

поставља на вредност 1. Поље одступања остаје непромењено;

- поље дужине података другог новог пакета поставља се на вредност једнаку дужини уметнутих података. Пољу одступања додаје се дужина уметнутих података првог дела података подељеног са 8. Бит М постављен је на вредност нула.

Слика 3.8 описује претходно наведени пример. Описану процедуру је једноставно проширити на уопштену поделу на *n* делова.

Да би се делови пакета поново повезали у једну целину треба, на месту где се то обавља, да постоји довољно простора у меморији. Како пакети са истим пољем ID пристижу тако се њихови подаци смештају у одговарајући део у меморији, све док се цело поље са подацима не повеже у једну целину. Цео процес започиње када у групи пристиглих пакета први од пакета има вредност поља „одступање" једнаку нули а последњи од пакета има вредност поља „још" (бит М) постављену на нулу.

Једна од могућности са којом се морамо суочити јесте да један или више делова можда неће стићи до одредишта: IP услуга не гарантује испоруку. Због тога је, како би се ослободило место у меморији, потребан метод којим се одређује када треба одустати од повезивања пакета. Тренутно, користе се два приступа:

- Први метод код кога се дефинише време трајања повезивања података када стигне први од делова пакета. Реализује се помоћу локалног часовника реалног времена који је додељен

Слика 3.8 Пример деобе пакета (фрагментовања)

процесу повезивања података. Како се делови пакета смештају у меморију часовник се декрементира. Ако време на које је часовник постављен прође пре него што се заврши поновно спајање делова, примљени делови се одбацују;

- Други метод користи вредност поља TTL које је део заглавља сваког од пристиглих пакета. Процес за склапање података умањује (декрементира) величину овог поља. Као и код првог метода, када поље TTL достигне вредност нула (истекне време трајања пакета) а сви пакети се не повежу у оригинални пакет, примљени делови се одбацују.

3.3.3 Контрола грешке

На Интернету се не гарантује успешна испорука пакета. Када неки од рутера одбаци пакет, требало би да покуша да пошаље неку информацију извориште пакета. Изворишни протокол може да искористи ове информације да би променио стратегију слања или можда да обавести више слојеве.

Пакет може бити одбачен из много разлога, укључујући истицање времена трајања, загушење и грешку на коју указује контролни збир. Када на грешку указује контролни збир, онда није могуће послати информацију о грешци пошто је можда баш поље са изворишном адресом оштећено.

3.3.4 Контрола тока

Контрола тока на Интернету дозвољава рутерима и/или одредишним станицама да поставе ограничење у брзини пријема података. За услугу без успоставе везе која се у овом поглављу анализира, механизам контроле протока је ограничен. Најбољи приступ био би да се другим рутерима и изворишним станицама пошаље пакет за управљање протоком који би захтевао његово смањење. Овакав или сличан пример биће описан касније, када буде анализиран ICMP протокол[1].

3.3.5 Адресирање

Свим системима на крајевима мреже и унутар ње (у међусистемима) додељује се јединствена адреса. У случају TCP/IP архитектуре, ово се односи на IP адресу или једноставно, интернет адресу. IP адреса се

[1] *Internet Control Message Protocol*

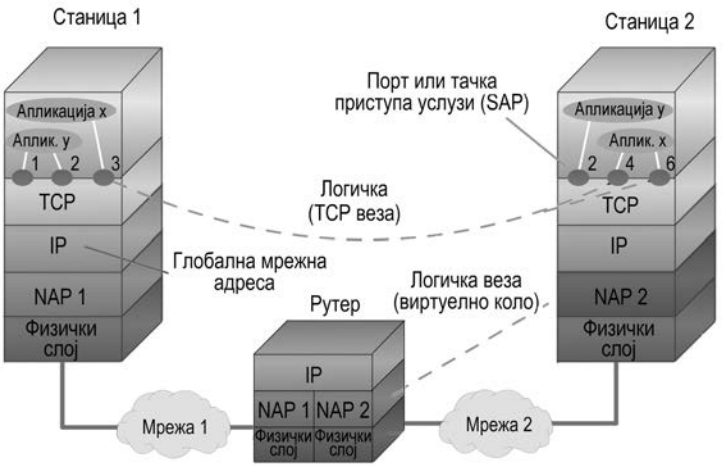
користи за рутирање јединице података мрежног слоја (NPDU[1]) кроз мрежу или мреже до система на који указује адреса у заглављу јединице података.

Адресирање на нижим слојевима потребно је за било коју рачунарску мрежу која има више система који су у њој међусобно повезани као што је нпр. мрежа са штафетним преносом или ATM мреже. Тада се сваком уређају повезаном на ове мреже додељује јединствена адреса.

Када подаци стигну до одредишног система, они морају бити усмерени ка неком процесу или апликацији тог система. Он ће подржавати више апликација а апликација може подржавати више корисника.

Свакој апликацији или сваком кориснику апликације, додељује се јединствени идентификатор, који представља порт у TCP/IP архитектури или приступну тачку услуге (SAP[2]) у OSI архитектури[3] (слика 3.9).

Свако од поља изворишне и одредишне адресе у IPv4 заглављу садржи 32-битну глобалну интернет адресу[4] (IPv4 адресу). Она се састоји од дела који означава мрежу - број мреже[5] и дела који означава



Слика 3.9 Концепт TCP/IP

[1] *Network Protocol Data Unit* или *IP Protocol Data Unit*. Уобичајено је да се означавају као датаграми или пакети..

[2] *Service Access Point*

[3] Детаљно је описано у 1. поглављу.

[4] Описано је у документу RFC1166.

[5] *Network number* се користи као термин у документу RFC1166. У литератури се среће и термин ознака (идентификација) мреже - *NetID* (*Network Identification*).

крајњу станицу (хост) у тој мрежи - број станице (хоста)[1]. Комбинација је јединствена: не постоје два уређаја са истом IPv4 адресом. IP слој прослеђује пакете слоју који користи протокол за приступ мрежи (NAP[2]), који може да буде на пример Етернет, са циљем да пакет проследи рутеру.

Део IP адресе који се односи на мрежу администрира (додељује) једна од три регионалне организације[3]:

- организација ARIN[4] је одговорна за регистрацију и администрацију IP бројева за Северну и Јужну Америку,
- организација RIPE[5] је одговорна за регистрацију и администрацију IP бројева за Европу, Средњи исток и део Африке,
- организација APNIC[6] је одговорна за регистрацију и администрацију IP бројева за део Азије у области Пацифика.

Интернет адреса, која је 32-битни број, обично се пише у децималној нотацији са тачкама. У овом формату сваки од 4 бајта написан је као децималан број у опсегу од 0_{10} до 255_{10}. Најнижа IPv4 адреса је 0.0.0.0 а највиша 255.255.255.255. На пример 128.2.7.9 је IPv4 адреса код које је 128.2 број мреже а 7.9 број станице. Правило по коме се IPv4 адреса дели на број мреже и број станице биће објашњено у следећем одељку. Бинарни формат IPv4 адресе је:

10000000 00000010 00000111 00001001

IPv4 адресе које користи IPv4 протокол јединствено идентификују станицу на Интернету (или уопштено било коју групу међусобно повезаних мрежа). Прецизнија дефиниција би била: IPv4 адреса идентификује интерфејс који је у стању да прима и шаље IPv4 датаграме. Један систем[7] може да има више интерфејса којима је прикључен на више мрежа и да

---

[1] *Host number* користи се као термин у документу RFC1166. У литератури се среће и термин ознака (идентификација) крајње станице - *HostID* (*Host Identification*).

[2] *Network Acess Layer Protocol*

[3] Услуга за регистрацију на Интернету (као што су IP адреса и DNS домен) раније је била у надлежности организације NIC (*Network Information Centre*). Од 1993. год. формирана је организација InterNIC (*Internet Network Information Center*) за све кориснике осим за DDN (*Defense Data Network*).

[4] *American Registry for Internet Number.*

[5] *Reseaux IP Europeens*

[6] *Asia Pacific Network Information Centre*

[7] *Multi-homed*

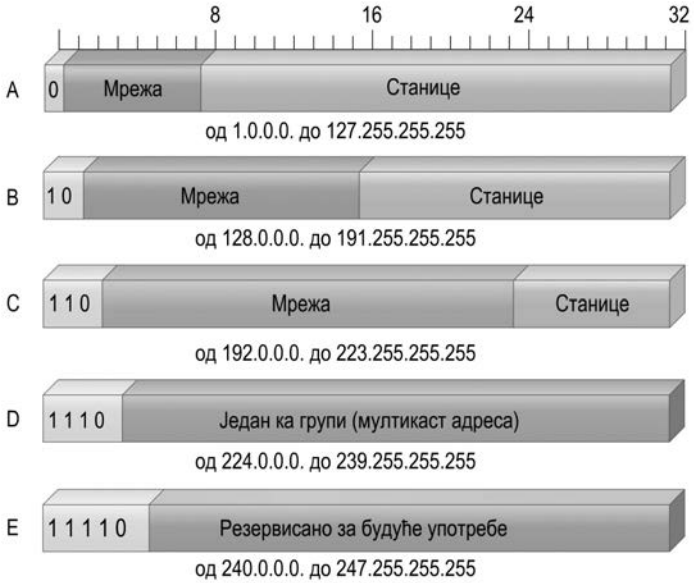
користи више IPv4 адреса (за сваку мрежу). Рутер и крајња станица морају да имају бар по једну IPv4 адресу.

3.3.6 IPv4 адресе засноване на класама

Постоји пет класа IPv4 адреса. Уместо равног адресног простора (нпр. 1, 2, 3,...) користи се структура приказана на слици 3.10. Адреса је кодирана да би се дозволиле различите комбинације за специфицирање мреже или станице. Први битови IPv4 адресе указују како ће остали део адресе бити расподељен између дела за мрежу и дела за станицу.

На слици 3.10 види се да:

- Класа A адреса:
  - користи 7 битова за адресу мреже и 24 бита за адресе рачунара (станица). На овај начин може се адресирати 2^7 - 2 (126) мрежа. У свакој мрежи може да буде 2^{24} - 2 (16777214) рачунара;
  - опсег адреса је од 0. 0. 0. 0 до 127. 255. 255. 255;
  - одговара мрежама са изузетно великим бројем рачунара.
- Класа B адреса:
  - користи 14 битова за адресу мреже и 16 битова за адресе рачунара (станица). На овај начин може се адресирати 2^{16} - 2 (16382) мрежа. У свакој мрежи може да буде 2^{24} - 2 (65534) рачунара;
  - опсег адреса класе B је од 128. 0. 0. 0 до 191. 255. 255. 255;
  - одговара мрежама са релативно великим бројем рачунара.
- Класа C адреса:
  - користи 21 бит за адресу мреже и 8 битова за адресе рачунара (станица). На овај начин може се адресирати 2^{21} - 2 (2097150) мрежа. У свакој мрежи може да буде 2^8 - 2 (254) рачунара;
  - опсег адреса класе C је од 192. 0. 0. 0 до 223. 255. 255. 255;
  - одговара мрежама са малим бројем рачунара.
- Класа D адреса:
  - користи се за мултикаст (врста бродкаста, али за ограничене области и једино за рачунаре који користе исту класу D адреса;
  - опсег адреса је од 224. 0. 0. 0 до 239. 255. 255. 255.
- Класа E адреса:
  - резервисана је за примене у будућности;



Слика 3.10 Класе IPv4 адреса

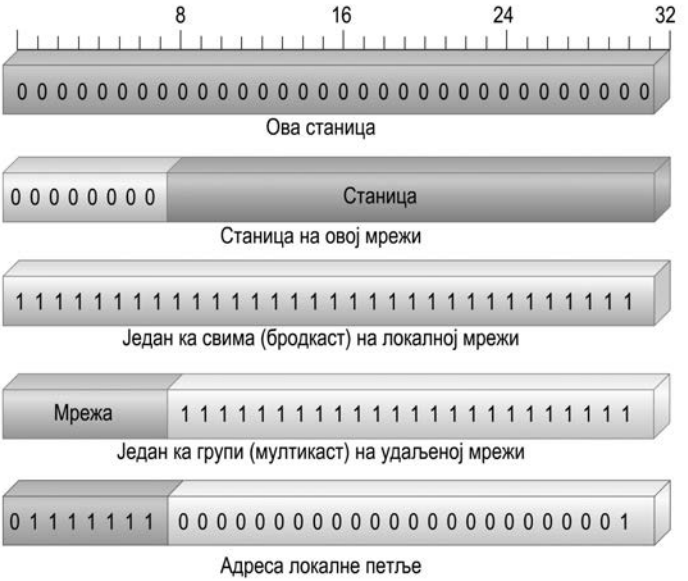
- опсег адреса класе E је од 240. 0. 0. 0 до 247. 255. 255. 255.

Резервисане IPv4 адресе

Посебне (резервисане) IPv4 адресе добијају се постављањем свих битова на 0 или свих битова на 1 и значења су следећа:

- сви битови 0:
  - у адресном делу рачунара интерпретира се као „овај рачунар“,
  - у адресном делу мреже интерпретира се као „ова мрежа“. Када рачунар хоће да комуницира преко мреже, али још увек не зна мрежни део IPv4 адресе, може да пошаље пакет са <netID>=0 који значи „ова мрежа“. Одговори садрже важећу адресу мреже[1];
- сви битови 1:
  - адреса са свим битовима постављеним на један интерпретира се као све мреже или сви рачунари. На пример адреса 128.2.255.255 значи сви рачунари на мрежи 128.2 (класа B);

[1] *Fully qualified network adress*



Слика 3.11 Неке од посебно резервисаних IPv4 адреса

- затворена петља[1]:
  - адреса 127.0.0.1 у класи A је дефинисана као адреса затворене петље. Додељује се интерфејсу за обраду на локалном систему и нема приступ физичкој мрежи.

Станица са више интерфејсних модула имаће више IPv4 адреса - за сваки интерфејс по једну. Додељивање адреса рачунарима (станицама) надлежност је администратора система.

Приватне IPv4 адресе

Један од метода да се сачува IPv4 адресни простор је додељивање адреса у приватним мрежама - интранету[2]. Правило да је IPv4 адреса глобална адресе на тај начин је прекршено. За мреже које не захтевају повезивање на Интернет резервисан је део адресног простора који оне користе. Уобичајено је да је администрација интранета у надлежности једне организације. Опсег адреса је:

- 10.0.0.0 једна мрежа A класе,

[1] *Loopback*

[2] Описан у документу RFC1918 (*Address Allocation for Private Internets*).

- 172.16.0.0 до 172.31.0.0 - укупно 16 узастопних мрежа са адресама из B класе,
- 192.168.0.0 до 192.168.255.0 - укупно 256 узастопних мрежа са адресама из C класе.

Било која организација може да користи било коју од адреса из горе поменутих опсега. Ове адресе се не размењују са рутерима ван приватних мрежа (екстерним рутерима).

3.3.7 Прављење подмрежа

Као последица наглог развоја Интернета принцип додељивања IPv4 адресе постао је превише сложен да би омогућио брзе промене конфигурација локалних мрежа. До промене може да дође када се:

- нови тип физичке мреже инсталира на тој локацији,
- значајно повећа број станица па се захтева деоба локалне мреже у две или више одвојених подмрежа,
- повећа раздаљина која захтева деобу мреже у више мањих подмрежа које су повезане уређајима за међусобно повезивање (гејтвеја).

Да би се спречило додавање мрежа са новим IPv4 адресама од свих рачунара захтева се да подржавају адресирање подмрежа[1]. Додељивање адреса је локално. Мрежа са подмрежама[2] је споља виђена и даље као једна IPv4 мрежа. Уместо да се посматра само адреса мреже и адреса станице (у оквиру IPv4 адреса) део адресе станице дели се у два дела: у адресу подмреже и адресу станице.

Ова идеја има смисла пошто адресе у класи A и класи B имају превише битова додељених адресама рачунара (станица): 2^{24}-2 (A), 2^{16}-2 (B). У пракси није уобичајено да се толико рачунара повезује у једну мрежу.

По добијању IPv4 адресе за мрежу од овлашћене организације, ствар је локалног администратора система да ли ће да прави подмреже или не. Уколико прави подмреже онда је питање колико битова се додељује за адресирање подмреже а колико за адресирање рачунара (станице). Да би се одредило који део IPv4 адресе представља адресу мреже а који

[1] Дефинисано је документом RFC950.

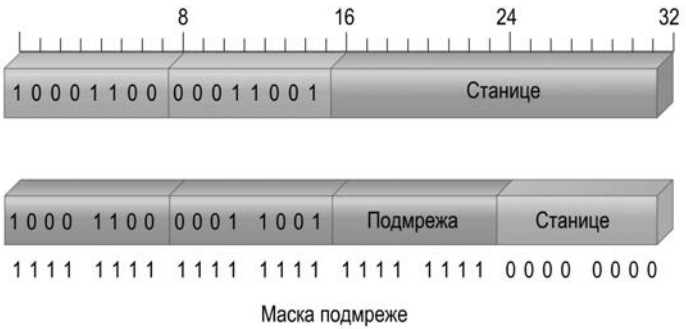
[2] Овај појам је у конфликту са појмом „подмрежа", што значи скуп рутера и комуникационих линија у мрежи (као што смо у уводу овог поглавља поменули). Из контекста се може видети на који појам се односи.

адресу станице користи се маска подмреже[1]. Маска је тридесет двобитна вредност у којој су (слика 3.12):

- на 1 постављене све позиције у мрежном делу адресе[2],
- на 1 постављене све позиције у подмрежном делу адресе[3],
- на 0 постављене све позиције у делу адресе за рачунар[4].

Унутар подељене мреже локални рутери морају рутирати на бази проширеног мрежног броја који се састоји од мрежног дела IPv4 адресе и броја подмреже. Коришћење маске подмреже помаже станици да утврди да ли је пакет намењен станици на истој локалној мрежи (шаље се директно) или на другој локалној мрежи (шаље се рутеру). Усвојено је да се неки други начини (нпр. ручна конфигурација) користе да би се маска подмреже направила и да би била позната локалним рутерима.

Комбинација броја подмреже и броја станице назива се и локална адреса или локални део IPv4 адресе. Подмрежавање се примењује тако да је невидљиво (нетранспарентно) за удаљене мреже (слика 3.13). Структура подмреже рачунару у оквиру мреже, која је подмрежена, је позната. Рачунару у другој мрежи је структура подмреже непозната. Он и даље третира локални део IPv4 адресе као адресу рачунара. Деобу локалног дела IPv4 адресе на адресу подмреже и адресу станице бира локални администратор. Било који битови у локалном делу могу се користити за прављење подмрежа.



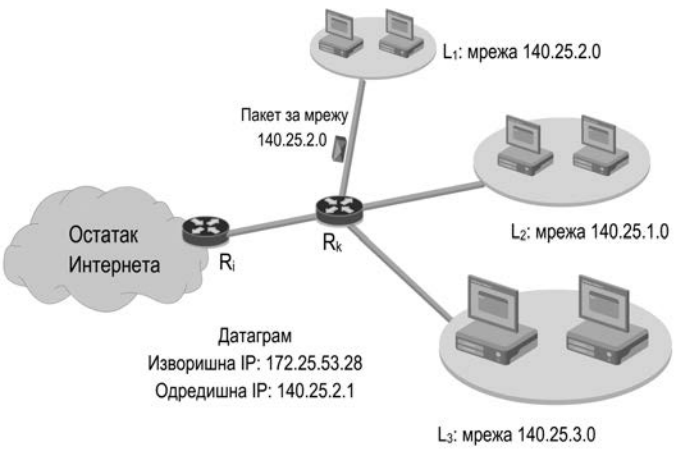
Слика 3.12 Подмрежа и маска подмреже

[1] *Subnet mask*
[2] *NetID*
[3] *Subnet ID*
[4] *HostID*

Слика 3.13 показује пример коришћења подмрежа. Приказан је локални комплекс који се састоји се од три локалне рачунарске мреже L_1, L_2 и L_3 и два рутера R_I и R_K. Остатак Интернета овај комплекс (везан преко рутера R_I) види само као мрежу класе B са мрежном адресом 140.25.x.x где су лева два октета број мреже а десна два октета број станице. Рутер R_K који дели мрежу на подмреже конфигурисан је маском подмреже која има вредност 255.255.255.0. На пример, ако датаграм са одредишном адресом 140.25.2.1 стигне у рутер R_K са остатка Интернета, рутер користи маску подмреже да би утврдио да се ова адреса односи на подмрежу L_1 и онда прослеђује пакет тој подмрежи где свака станица (хост) онда одређује да ли је њој намењен пакет.

Постоје два начина подмрежавања:

- Статичко које подразумева да све подмреже добијене деобом исте мреже користе исту маску подмреже. Овај начин је једноставан за имплементацију и одржавање. Недостатак је што у случају малих мрежа неке од адреса остају неискоришћене. Узмимо за пример мрежу са четири станице која користи маску подмреже 255.255.255.0. Оваквом доделом остаје 250 IPv4 адреса неискоришћено. Од свих станица и рутера очекује се да подржавају статичко подмрежавање.



Слика 3.13 Пример коришћења подмрежа

- Код подмрежавања променљиве дужине, подмреже добијене деобом исте мреже могу да користе маске подмреже које се међусобно разликују. Подмрежа са малим бројем станица може да користи маску подмреже одговарајуће величине. Подмреже са великим бројем станица захтевају другачију маску. Могућност да се доделе маске подмреже према потребама појединих подмрежа доприноси да се IPv4 адресе беспотребно не расипају (ефикасније користе).

Употреба маске подмреже

Када се користи подмрежавање сама класа IPv4 адресе не може да одреди који део адресе представља адресу мреже а који адресу станице.

На слици 3.14 приказане су маске подмреже у класи B када је за подмрежу одвојено 8 и 10 битова.

Иако се IPv4 адреса нормално означава у децималној нотацији, маска подмреже често се пише у хексадецималној, нарочито када границе нису границе бајтова.

Када рачунар поседује IPv4 адресу и маску подмреже, он може да закључи да ли је пакет упућен ка:

- рачунару у истој мрежи у којој је и он,
- рачунару на другој подмрежи у оквиру исте мреже,
- рачунару на другој мрежи.

	16 битова	8 битова	8 битова
Класа B	Адреса мреже	Адреса подмреже	Адреса станице
Маска подмреже	1111111111111111	11111111	00000000
Децимално	255.255.255.0		
Хексадецимално	FFFFFF00		

	16 битова	10 битова	6 битова
Класа B	Адреса мреже	Адреса подмреже	Адреса станице
Маска подмреже	1111111111111111	1111111111	000000
Децимално	255.255.255.192		
Хексадецимално	FFFFFFC0		

Слика 3.14 Подмреже у класи B са подмрежама од 8 и 10 битова

Познавање сопствене IPv4 адресе указује да ли је у питању класа A, B или C (на основу битова највеће тежине) тј. где је граница између мрежног и подмрежног дела адресе. Маска подмреже указује где се налази граница између дела адресе подмреже и дела адресе рачунара. У примерима који следе (3.1, 3.2 и 3.3) прецизно је представљена употреба маски подмреже. Размотрићемо три различита случаја одредишне IPv4 адресе:

- Уколико је одредишна IPv4 адреса 140.25.1.5 видимо да је мрежни део адресе исти (140.25), али је део за подмрежу различит (1 и 4). Значи да се рачунари налазе на различитим подмрежама и рутер R_K ће пакет усмерити на одговарајућу подмрежу (слика 3.13). На слици 3.15 приказано је како се употребом маске подмреже обавља упоређивање две IPv4 адресе;
- Уколико је одредишна IPv4 адреса 140.25.1.22 видимо да је мрежни део адресе исти (140.25) и део адресе за подмрежу је исти (1). Део адресе за рачунар је наравно различит (5 и 22). Значи да се рачунари (изворишни и одредишни) налазе на истим мрежама па рутер R_K неће пакет преусмеравати (слика 3.13);
- Уколико је одредишна IPv4 адреса 192.168.1.225 (адреса из класе C) мрежни делови адреса су различити. Одредишни рачунар не припада ниједној од подмрежа које су у надлежности рутера R_K. Рутер R_K ће овај пакет да проследи рутеру R_I који ће на основу табеле рутирања да усмерава овај пакет даље кроз Интернет.

Пример 3.1: Претпоставимо да је изворишна IPv4 адреса рачунара 142.25.1.1 (класа B) и маска подмреже 255.255.255.0 (8 битова за адресу подмреже и 8 битова за адресу рачунара).

Пример 3.2: Станица са адресом 192.168.1.65 и маском подмреже 255.255.255.224 треба да пошаље пакет на адресу 192.168.1.91.

Као што се види на слици 3.16 одредиште је на истој мрежи као и рачунар (станица) који шаље пакет.

Пример 3.3: Станица са адресом 192.168.1.65 и маском подмреже 255.255.255.224 треба да пошаље пакет на адресу 192.168.1.97.

На основу слике 3.17 види се да одредиште није на истој мрежи као и рачунар (станица) који шаље пакет.

	Адресе мрежа су исте		Адресе подмрежа нису исте	
	16 битова		8 битова	8 битова
Класа B	140	25	1	5
Маска подмреже	11111111	11111111	11111111	00000000
Децимално	255.255.255.0			
Класа B	140	25	4	22

	Адресе мрежа су исте		Адресе подмрежа су исте	
	16 битова		8 битова	8 битова
Класа B	140	25	1	5
Маска подмреже	11111111	11111111	11111111	00000000
Децимално	255.255.255.0			
Класа B	140	25	1	22

Слика 3.15 Пример 3.1: Поређење две адресе класе B

Важно је истаћи да је број станица у подмрежама који се на овај начин добија исти. Подмрежавање мреже класе C, са сегментима од по 30 станица шематски је представљено на слици 3.18.

3.3.8 Подмрежа са променљивим делом за подмрежу

Оригинална имплементација подмрежа подразумева да се мрежа

Моја адреса	11000000	10101000	00000001	01000001
Маска	11111111	11111111	11111111	11100000
Логичко И (AND)				
Моја мрежа	**11000000**	**10101000**	**00000001**	**010**00000

Одредишна адреса	11000000	10101000	00000001	01011011
Маска	11111111	11111111	11111111	11100000
Логичко И (AND)				
Одредишна мрежа	**11000000**	**10101000**	**00000001**	**010**00000

Слика 3.16 Пример 3.2: Употреба маске подмреже

Моја адреса	11000000	10101000	00000001	01000001
Маска	11111111	11111111	11111111	11100000
Логичко И (AND)				
Моја мрежа	**11000000**	**10101000**	**00000001**	**010**00000

Одредишна адреса	11000000	10101000	00000001	01111011
Маска	11111111	11111111	11111111	11100000
Логичко И (AND)				
Одредишна мрежа	**11000000**	**10101000**	**00000001**	**011**00000

Слика 3.17 Пример 3.3: Употребе маске подмреже

подели на сегменте исте величине. Овакав начин поделе адресног простора не одговара увек стварним потребама. Подмрежа са променљивим делом за подмрежу (VLSM[1]) омогућава креирање сегмената са различитим бројем станица. Систем VLSM треба да буде подржан од стране протокола за динамичко рутирање. Већина нових рутера има подршку за VLSM подмреже. Уобичајено је да рачунари чувају маску подмреже у конфигурационој датотеци.

Да би се јасније представила потребе за VLSM системом посматрајмо корпорацијску мрежу којој је додељена адреса из класе C: 165.214.32.0.

Слика 3.18 Подмрежавање мреже класе C у сегменте са по 30 станица

[1] *Variable Length Subnetting*

Слика 3.19 Подмрежа са променљивим делом за подмрежу
(сегменти са различитим бројем станица)

Захтева се да се адресни простор раздвоји у пет независних подмрежа таквих да (слика 3.19):

- подмрежа број 1 има 120 станица,
- подмрежа број 2 има 60 станица,
- подмрежа број 3 има 14 станица,
- подмрежа број 4 има 14 станица,
- подмрежа број 5 има 14 станица и
- подмрежа број 6 има 14 станица.

Овај задатак не може се разрешити статичким подмрежавањем. У овом примеру статичко прављење подмрежа дало би:

- две подмреже (2^1) од којих свака има 126 ($2^7 – 2$) станица, или
- четири подмреже (2^2) од којих свака може да има 62 ($2^6 – 2$) станице, или
- осам подмрежа (2^3) од којих би свака имала по 30 ($2^5 – 2$) станице, или
- шеснаест подмрежа (2^4) од којих би свака имала по 14 ($2^4 – 2$) станица.

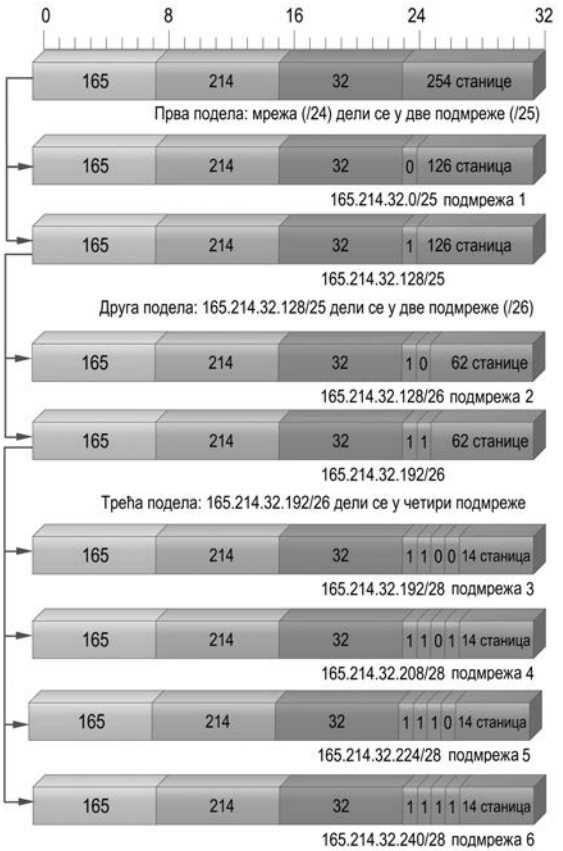
Ниједно од понуђених решења не задовољава постављене захтеве. Да би се мрежа поделила у шест подмрежа треба дефинисати три различите маске подмрежа (слика 3.20):

- маском 255.255.255.128 (/25) мрежа се може поделити у две подмреже са по 126 рачунара,

- друга подмрежа се дели у две подмреже са по 62 рачунара користећи маску 255.255.255.192 (/26) и
- четврта подмрежа се дели у четири подмреже са по 14 рачунара користећи маску 255.255.255.240 (/28).

Добијене су: једна подмрежа са 126 станица, једна подмрежа са 62 станице и четири подмреже са по 14 станица. Ово решење задовољава постављене захтеве.

Може се закључити да је техника са променљивим делом за подмрежу начин у коме се подмрежавање извршава итеративно више



Слика 3.20 Пример подмреже са променљивим делом за подмрежу

пута. Тако се мрежа може поделити у хијерархијске подмреже које су променљиве величине.

3.3.9 Директно и индиректно рутирање

Важна функција мрежног слоја и IPv4 слоја као његовог најраспрострањенијег представника је рутирање. Један од начина поделе рутирања је на основу међусобне позиције изворишне и одредишне станице. Постоје две врсте рутирања: директно и индиректно.

Уколико је одредишна станица повезана на исту физичку мрежу као изворишна станица, IPv4 пакети могу се директно размењивати. Рутирање се реализује паковањем IPv4 пакета у рам физичке мреже за коју су обе станице везане[1]. Користе се и термини директна испорука или директно рутирање.

Индиректно рутирање обавља се када одредишна станица није повезана на исту физичку мрежу на коју је повезана изворишна станица[2]. Једини начин да једна станица стигне до друге станице је преко рутера (међумрежних пролаза[3])[4]. Адреса првог рутера у алгоритмима за рутирање IP пакета назива се индиректна рута. Адреса првог рутера једина је информација која је потребна изворишној станици да би послала пакет ка одредишној станици.

За комуникацију пара станица када постоји више подмрежа у оквиру исте физичке мреже користи се индиректно рутирање. Рутер је неопходан за преусмеравање саобраћаја између подмрежа (слика 3.21).

Директне путање (руте) проналазе се на основу листе локалних интерфејса. Она се аутоматски формира у процесу иницијализације IP рутирања. Поред тога листа мрежа и придружених им рутера (за индиректно рутирање) може се и конфигурисати. Свака станица садржи податке о вези између:

- одредишне IP адресе и
- путање до следећег рутера.

Информације су смештене у табели која се назива табела рутирања.

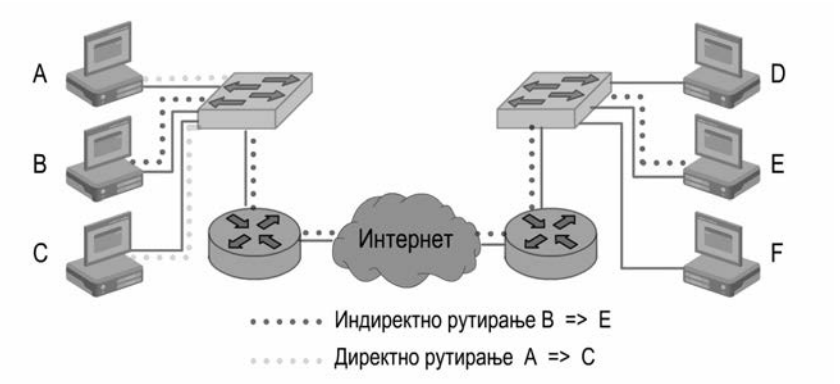
---

[1] Погледати објашњење уз слику 3.5.

[2] Станице A и B на слици 3.5.

[3] *Gateway*

[4] Треба уочити да се у TCP/IP окружењу рутер и мрежни излаз или међумрежни пролаз (*gateway*) користе у истом контексту.



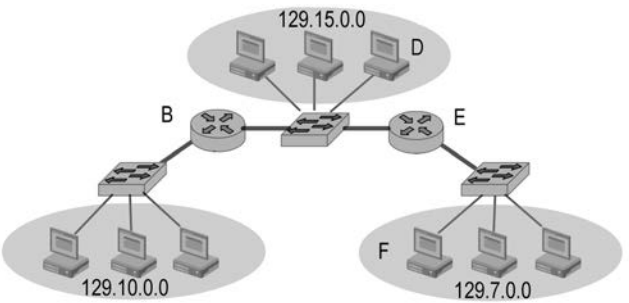
Слика 3.21 Директно и индиректно рутирање

Могу се срести три врсте међусобних веза (мапирања):

- директне путање за интерфејсе који су повезани на исту физичку мрежу,
- индиректне путање које указују на мреже до којих се може доћи само преко једног или више рутера,
- подразумевана путања која садржи (директну или индиректну) путању ако одредишна мрежа није пронађена на један од претходна два начина.

На слици 3.22 приказан је пример мреже. Табеле рутирања станица D и E могу да садрже симболичне записе као што се може видети у табелама 3.3 и 3.4.

Пошто је станица D директно повезана на мрежу 128.15.0.0 она има директну путању ка овој мрежи. Да би станица D дошла до мреже



Слика 3.22 Сценарио за рутирање помоћу табеле рутирања

Одредиште	Путања	Интерфејс
129.7.0.0	E	I0
128.15.0.0	D (директна)	I0
129.10.0.0	B	I0
Подразумевано	B	I0
127.0.0.1	петља	10

Табела 3.3 Садржај табеле рутирања станице D

Одредиште	Путања	Интерфејс
129.7.0.0	F (директна)	I1
128.15.0.0	E	I1
129.10.0.0	E	I1
Подразумевано	E	I1
127.0.0.1	Петља	10

Табела 3.4 Садржај табеле рутирања станице E

129.7.0.0 и 129.10.0.0 мора да користи индиректну путању преко рутера E и B пошто за те мреже станица D није директно везана.

Доношење одлуке о рутирању може се шематски приказати као на слици 3.23. Када је потребно донети одлуку о рутирању пакета између подмрежа онда се то може шематски приказати као на слици 3.24.

Последице измена алгоритма представљеног на слици 3.23 алгоритмом представљеним на слици 3.24 су следеће:

- Алгоритам на слици 3.24 представља генерално промену алгоритма за рутирање. То значи да рутери (међумрежни излази) морају да раде са новим алгоритмом;
- Рутирање IPv4 пакета примењује се у свакој станици (а не само у рутерима). Све станице у подмрежи морају да имају:
  - алгоритме за рутирање који подржавају подмрежавање,
  - исту маску подмреже;
- Уколико станица не подржава подмреже онда ће она моћи да комуницира са свим станицама те подмреже али неће моћи да комуницира са рачунарима других подмрежа те мреже. Разлог је што станица види све као једну мрежу и не може да направи

Слика 3.23 Рутирање без подмрежа

Слика 3.24 Рутирање са подмрежавањем

разлику између станице подмреже и мрежног излаза (рутера) те подмреже[1].

Табеле рутирања могу бити статичке или динамичке. Статичка табела може да садржи алтернативне путање у случају да одређени рутер није доступан. Динамичка табела је флексибилнија у случајевима када дође до нагомилавања пакета или грешака у преносу. Узмимо на пример Интернет: ако један рутер прекине са радом сви његови суседи шаљу статусни извештај који онда омогућава рутерима и станицама да

[1] Уколико већи број станица не подржава IP адресе са подмрежама постоји додатно решење - *proxy* ARP.

ажурирају своје табеле рутирања. Сличан план може се користити и за управљање загушењима у саобраћају (нагомилавања пакета).

Табеле рутирања могу се користити и као помоћ осталим услугама међусобно повезаних мрежа као што су сигурност и приоритети. На пример, појединачне мреже могу служити за преузимање података до одређене сигурносне границе. Механизам рутирања мора обезбедити забрану пролаза подацима траженог сигурносног нивоа кроз мреже које нису овлашћене да преузимају такве податке.

Једна од техника рутирања је и тзв. изворишно рутирање[1]. Изворишна станица одређује путању тако што у пакету који шаље наводи низ узастопних рутера преко којих пакет треба да пређе. Ова техника може бити корисна када се захтевају сигурност и одређени приоритети.

Једна од услуга која је везана за рутирање је и записивање путање (руте). Да би се записала путања сваки од рутера преко кога пакет пређе може да у одговарајућу листу дода и своју IPv4 адресу. Услуга се може употребити код тестирања или отклањања грешака у преносу.

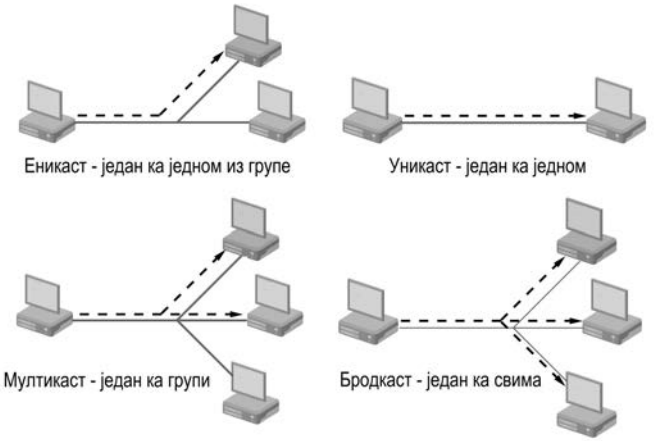
3.3.10 Методе испоруке пакета

Највећи број IP адреса користи се за једног примаоца или метод испоруке ка једној станици (уникаст[2]). Поред овог метода који представља комуникацију једног изворишта са једним одредиштем постоје још три метода испоруке[3] када се IP адреса користи за адресирање више станица:

- IP адреса упућена свима (бродкаст адреса[4]),
- IP адреса упућена групи станица (мултикаст адреса[5]) и
- IP адреса упућена једној од станица из групе станица (еникаст адреса[6]).

На слици 3.25 приказана су поменута четири метода испоруке пакета и одговарајуће врсте адреса. На пример један управљачки центар у мрежи треба да обавести све кориснике да има проблема у раду (нпр. мрежа пада). Адреса за више прималаца може бити бродкаст (намењена

[1] *Source routing*
[2] *Unicast address*
[3] Користи се и термин тип или врста адреса.
[4] *Broadcast address*
[5] *Multicast address*
[6] *Anycast*



Слика 3.25 Методе испоруке IP пакета

свим станицам одређеног домена) или мултикаст (намењена станицама одређене подмреже или некој групи станица).

Адресе један ка свима (бродкаст) никада се не могу употребити као изворишне адресе. Постоје различите врсте ових адреса:

- адресу ограниченог дејства (децимално је представљена 255.255.255.255 тј. све јединице) препознаје свака станица а рутери не преусмеравају пакете са овом адресом[1];
- адреса један ка свима одређене мреже је адреса чији мрежни део адресе одговара мрежи у којој се размењује пакети а део за крајње станице је постављен на све јединице (на пример 128.2.255.255). Ова адреса се односи на све станице у специфицираној мрежи. Рутери би требало да преусмеравају ове пакете[2].

Постоји потреба да се пакет испоручи само одређеном броју станица. У ту сврху користи се класа D IPv4 адреса и метод испоруке који се означава као емитовање групи (мултикастинг). Станице којима се испоручује пакет означавају се као група станица[3]. Пакети са мултикаст адресама испоручују се само члановима групе.

[1] Изузетак представља правило BOOTP *forwarding* (преусмеравање). Протокол BOOTP користи адресу 255.255.255.255 да би станицама без диска омогућио успоставу везе са сервером. Када рутери не би преусмеравали ове пакете било би потребно да се у сваку од мрежа постави по један сервер.

[2] Односи се на ARP захтев. Биће обрађено у делу о ARP протоколу.

[3] *Host group*

IP адреса упућена једној од станица из групе станица може се описати на следећи начин: једна IP адреса (еникаст) додељује се групи станица. Пакет који се пошаље на ту адресу усмерава се до најближе станице која има ту еникаст адресу. Протокол за рутирање по својим критеријумима одређује најближу станицу.

Понекад једна IP услугу може да понуди више станица. На пример корисник жели да учита податке користећи услугу преноса датотека (FTP услуга). Подаци се налазе на више FTP сервера и могу се скинути са сваког од њих. Све станице које могу да обезбеде FTP услугу поседују еникаст адресу. Веза се успоставља са станицом из групе станица са еникаст адресом која прва одговори. На тај начин се гарантује да је добијена услуга из најбоље везе станице и пријемника.

3.3.11 Рутирање између домена са бескласним IPv4 адресама

Стандардно IPv4 рутирање познаје само адресе класа A, B и C. Највише је искоришћен адресни простор класе B. Као следећи најпогоднији показао се адресни простор класе C.Тако је генерисан нови проблем: превелики број записа у табелама рутирања потребних за сваку од мрежа из класе C.

На пример, за 3000 станица класе B потребан је само један запис у табели рутера окоснице Интернета. За исти број станица класе C потребно је 16 записа.

Проблем се може решити рутирањем на основу IPv4 адреса које нису засноване на класама - CIDR[1] адресирањем. У литератури се овај концепт среће и под називом надмрежавање (супер умрежавање[2]) а описан је у документима RFC1518, RFC1519 и RFC1467.

Основна идеја CIDR адресирања је у додели више IPv4 адреса таквих да обезбеђују здруживање у мањи број записа у табелама рутирања. На пример ако је једној локацији (месту[3]) додељено 16 адреса мрежа из класе C и њих 16 је на таквом месту да се могу здружити онда се на сваку од њих може односити само један запис у табели рутирања рутера на Интернету.

---

[1] CIDR (*Classless Inter Domain Routing*) - рутирање између домена на основу IPv4 адреса које нису засноване на класама.

[2] *Supernetting*

[3] *Site*

Такође, ако је 8 различитих локација повезано преко једне прикључне тачке на истог добављача интернет услуга[1] и свакој од локација је додељено 8 различитих IPv4 адреса које се могу здружити онда је потребан само један запис у табелама рутирања за све њих.

Да би се могло извршити здруживање потребне су три ствари:

- IPv4 адресе које треба здружити морају имати исте битове највеће тежине;
- табеле рутирања и алгоритми за рутирање морају своје одлуке доносити на основу 32-битне IPv4 адресе и 32-битне маске;
- протоколи за рутирање који се користе морају размењивати поред адресе и 32-битну маску.

Суштински CIDR механизам обезбеђује централној организацији за доделу адреса флексибилност тако да може да додели групе адреса различитих величина организацијама у складу са њиховим потребама. Када је CIDR адресирање развијено померај је направљен у начину додела јавних адреса.

Метод у коме је цео свет потраживао адресе од једне организације није најбоље решење. То је било неопходно за адресе засноване на класама пошто је хијерархија имала само два нивоа: организација IANA је додељивала корисницима мрежну адресу, идентификатор (ИД) мреже а они су додељивали адресе станица (ИД хоста) директно или подмрежавањем.

Користећи CIDR систем добија се велики број хијерархијских нивоа: деобом великог блока адреса у мање блокове, затим у још мање, итд... Шта се дешава када IANA/ICANN[2] дели адресе у велике блокове и прослеђује их регионалним организацијама (RIR[3]): APNIC[4], ARIN[5], LACNIC[6] и RIPE[7]? Оне затим даље деле адресне блокове и дистрибуирају их нижим нивоима хијерархије: националним организацијама (NIR[8]),

[1] Интернет провајдер

[2] *Internet Assigned Numbers Authority/ Internet Corporation for Assigned Names and Numbers*

[3] *Regional Internet Registries*

[4] *Asia Pacific Network Information Center*

[5] *American Registry for Internet Numbers*

[6] *Latin America and Caribbean Network Information Centre*

[7] *Réseaux IP Européens* - европске IP мреже

[8] *National Internet Registries*

локалним организацијама (LIR[1]) и добављачима интернет услуга (ISP[2]).

Добављачи интернет услуга (интернет посредници) деле блокове адреса у мање и додељују својим корисницима. Некада су ти корисници мањи интернет посредници, који такође додељују их крајњим корисницима адресе. Колико пута се операција деобе може извршити зависи од тога колико је адреса у почетном (оригиналном блоку) адреса.

Свака CIDR табела рутирања садржи 32-битни IPv4 адресу и 32-битну мрежну маску које заједно дају дужину и вредност IP префикса. Ово се представља као пар <IP адреса, мрежна маска>. На пример за адресирање блока од осам адреса из класе C са једном једином табелом рутирања добијамо: <192.32.136.0, 255.255.248.0>. Са гледишта окоснице мреже ово се може реферисати као једна мрежа класе C у опсегу 192.32.136.0 до 192.32.143.0, као што је приказано на слици 3.26.

CIDR метод омогућава да се 64 мреже класе C повежу у један запис у табели рутирања 192.168.64/18 (као што је приказано на слици 3.27). Суштински суперумрежавање[3] помера природну маску улево док је подмрежавање помера удесно.

Претпоставимо да имамо интернет посредника који започиње доделу почевши од блока 71.94.0.0/15. Првих 15 битова је идентификатор мреже а преосталих 17 битова је идентификатор станице. Наравно овај

192.32.136.0 Класа C	11000000	00100000	10001000	00000000
225.255.248.0 Маска	11111111	11111111	11111000	00000000
Логичко И (AND)				
192.32.136.0 IP префикс	**11000000**	**00100000**	**10001**000	00000000

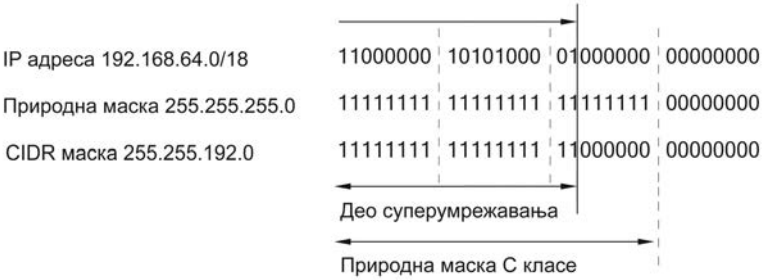
192.32.143.0 Класа C	11000000	00100000	10001111	00000000
225.255.248.0 Маска	11111111	11111111	11111000	00000000
Логичко И (AND)				
192.32.136.0 IP префикс	**11000000**	**00100000**	**10001**000	00000000

Слика 3.26 Пример CIDR супер умрежавања

[1] *Local Internet Registries*

[2] *Internet Service Providers*

[3] *Supernetting*



Слика 3.27 Пример CIDR адресирања

блок адреса добијен је од већег добављача интернет услуга. Интернет посредник овај велики блок адреса (131070 станица) може хијерархијски да подели. У том случају он прво дели на два дела блокове са по 16 битова за идентификатор мреже (/16). На пример један је резервисан а други се деле у четири блока са по 18 битова за идентификатор мреже (/18). Сваки од њих даље се може поделити у блокове различите величине и доделити организацијама које захтевају до 62 (/26), 126 (/25), 254 (/24) и 510 (/23) станица.

Може се констатовати да као што се VLSM механизам користи за деобу мреже на подмреже CIDR механизам се користи за деобу целокупног Интернета.

3.3.12 Формат IPv4 пакета

Протокол између IP целина може се најбоље описати посматрањем формата IP датаграма (пакета). Састоји се од два дела: заглавља и дела за податке. Заглавље чине:

- део дужине 20 бајтова тачно дефинисаних садржаја поља и
- део променљиве дужине које је опциони.

Формат заглавља је приказан на слици 3.28. Он се шаље редоследом слева надесно а на првом месту је бит највеће тежине поља „верзија"[1].

Верзија (дужине 4 бита) носи информацију којој верзији протокола пакет припада. Укључујући верзију у сваки пакет постаје могуће обезбедити прелаз између рачунара који раде са старим верзијама протокола и другим које раде са новим. Вредност овог поља је 4 код IPv4 а 6 код IPv6.

[1] *Version*

Слика 3.28 Формат заглавља IPv4

Дужина заглавља IHL[1] (дужине 4 бита) није константна тако да ово поље обезбеђује информацију колико је заглавље дугачко изражено у 32-битским речима. Минимална вредност је 5 за минималну дужину заглавља од 20 бајтова која се примењује када опције не постоје у заглављу. Максимална вредност овог 4-битског поља је 15. Тиме је ограничено заглавље на 60 бајтова што значи да је максимална величина поља опција 40 бајтова. За неке опције, као што су оне које праве запис о рути (путањи) коју пакет пређе, 40 бајтова је мало.

Поље диференцирана услуга/експлицитно обавештење о загушењу (DS/ECN[2]) подељено је на следећи начин:

- првих 6 битова (DS) одређују различите нивое квалитета услуге (QoS). Могуће је дефинисати 64 (2^6) класе саобраћаја (RFC2474),
- друга 2 бита односе се на експлицитно обавештавање о загушењу у саобраћају[3] (ECN).

Диференциране услуге замениле су раније коришћено поље тип (врсту) услуге (TOS[4]) дужине 8 бита: које дозвољава станици да укаже подмрежи коју врсту услуге (тј. који квалитет услуге) жели за овај пакет. Могуће су различите комбинације поузданости и брзине. За дигитализовани говор,

[1] *Internet Header Length*

[2] *Differentiated Services /Explicit Congestion Notification*

[3] Детаљније ће бити описано у поглављу о TCP протоколу (6. поглавље).

[4] TOS - *Type Of Service*

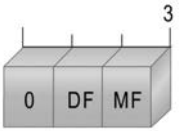
Слика 3.29 Врста услуге

брза испорука пакета је важнија од тачне испоруке. За пренос датотека важнији је пренос без грешке него брз пренос. Поље садржи селедеће информације (слика 3.29):

- Предност[1] (3 бита) је поље које указује какав је пакет и какав приоритет му треба доделити од вредности 0 (за нормалан пакет) до вредности 7 (за мрежни управљачки пакет). Постоје следеће могућности:
  - 000 уобичајени приоритет,
  - 001 - 101 различите категорије приоритета,
  - управљање је у надлежности међумреже и
  - управљање је у надлежности мреже;
- Четири бита која дозвољавају станици да специфицира шта јој је најважније за подешавање: што мање кашњење, што већа пропусна моћ, што већа поузданост или што нижа цена. Само један од битова треба да буде постављен на један. Уколико су сва четири постављена на нулу у питању је нормална услуга. По теорији ово поље дозвољава рутерима да направе избор између, на пример сателитског линка са великом пропусношћу и великим кашњењем и изнајмљених линија са малом пропусношћу и малим кашњењем (RFC1340, RFC1349). Постоје следеће могућности:
  - 1000 да смањи кашњење на најмању могућу вредност,
  - 0100 да повећа пропусну моћ што је више могуће,
  - 0010 да омогући што већу поузданост,
  - 0001 да обезбеди најмање трошкове,
  - 0000 нормална услуга;
- Један бит[2] (бит 7) резервисан је за будућу употребу.

[1] *Precedence*

[2] MBZ (*Must Be Zero*) - мора бити нула



Слика 3.30 Ознаке (флегови) у IPv4 датаграму

Укупна дужина[1] (16 битова) укључује све у пакету и заглавље и податке. Максимална дужина је 65535 бајтова. За сада је горња граница прихватљива, али са будућим гигабитским мрежама биће потребни већи пакети;

Идентификација[2] (дужине 16 бита) је потребна да би омогућила удаљеној станици да одреди ком пакету управо доспели део припада. Сви делови добијени деобом једног пакета садрже исту идентификациону вредност;

Ознаке (флегови) (дужине 3 бита) садржи ознаке за управљање од којих су само две тренутно дефинисане (слика 3.30):

- Први бит за сада мора да буде 0;
- Други бит означен са DF[3]:
  - ако је постављен на вредност 1 означава да се не дозвољава деоба пакета. То је команда рутеру да не дели пакет пошто одредиште није у могућности да делове спаја поново у једну целину. Предност пакета са постављеним DF битом је та што пошиљалац зна да ће он стићи на одредиште као једна целина. Недостатак је што се може десити да се пакет одбаци уколико на својој путањи премаши максималну величину пакета коју нека од успутних мрежа може да подржи. Да се ово не би десило било би добро користити изворишно рутирање[4] да би се заобишле мреже које прихватају само мале пакете,
  - ако је постављен на вредност 0 дозвољава се деоба;
- Трећи бит означен са MF[5]:

[1] *Total Length*
[2] *Identification*
[3] *Don't Fragment* – не фрагментирај (не дели)
[4] *Source routing* - унапред, на предајној страни (извориште) одреди путању (руту)
[5] *More Fragments* - још делова (фрагмената)

- постављен на вредност 0 указује да је ово последњи део у пакету,
- постављен на вредност 1 указује да постоји још делова.

Одступање фрагмената (делова[1]) (дужине 13 бита) користи се као испомоћ у повезивању делова у оригинални пакет. Указује где фрагмент припада у текућем пакету. Сви фрагменти осим последњег морају бити мултипл (умножак) од 8 бајтова, што је елементарна јединица фрагмента. Пошто је обезбеђено 13 битова постоји максимално 8192 фрагмената по пакету, па је максимална дужина пакета 65536, за један већа од поља укупне дужине[2].

Време трајања пакета TTL[3] (дужине 8 бита) указује колико дуго (у секундама) је дозвољено пакету да „путује" по мрежи. Теоретски сваки рутер који обрађује пакет требало би да од вредности овог поља одузме време обраде пакета. Практично сваки рутер обради пакет за мање од 1 секунде. Зато рутери само умањују за један вредности овог поља. Тако време трајања TTL постаје мера броја прелазака (скокова)[4] а не мера времена.

Када вредност TTL поља достигну вредност нула онда се претпоставља да се пакет креће у затвореној петљи и зато се одбацује. Иницијалну вредност треба да поставе протоколи виших слојева који и формирају пакет и иницирају његов пренос.

Ознака протокола[5] (дужине 8 бита) указује ком протоколу вишег слоја треба податке из овог пакета испоручити. Неке од вредности су дате у табели 3.5. Нумерисање протокола је глобално кроз цео Интернет и дефинисано је документом RFC1700.

Контролна збир (сума) заглавља[6] (16 битова) израчунава се само за заглавље. Израчунавање се обавља тако што се прво у поље за контролни збир упишу све нуле. Затим се израчуна први комплемент од 16-то битских целина (као да се заглавље састоји од 16-то битских речи).

---

[1] *Fragment Offset*
[2] *Total length Field*
[3] *Time To Live*
[4] *Hop metric*
[5] *Protocol Number*
[6] *Header Checksum*

Децимална вредност поља	Протокол
0	Резервисано
1	Управљачке поруке на Интернету - ICMP[1] протокол
2	Протокол за управљање групама- IGMP[2] протокол
3	Протокол између мрежних пролаза - GGP[3] протокол
4	IPv4 (IPv4 укалупљивање)
5	Ток[4]
6	TCP протокол
8	Спољашњи протоколи за рутирање - EGRP[5] протокол
9	Протокол за рутирање унутар мрежа - IGRP[6] протокол
17	UDP протокол
41	IPv6 протокол
50	Сигурносни протокол за IPv6 - ESP[7] протокол
51	Заглавље за аутентификацију за IPv6 - AH[8] заглавље
89	Протокол за рутирање најкраћом путањом - OSPF[9] протокол

Табела 3.5 Листа додељених бројева на Интернету[10]

Од добијеног збира (16-то битски број) направи се први комплемент и упише у поље за контролни збир.

На пријему се понови операција над целим заглављем. Ако у преносу није дошло до грешке добиће се 16-то битски број са свим јединицама. Ако је дошло до грешке пакет се једноставно одбацује. Протоколи IP, ICMP, IGMP, UDP и TCP користе исти алгоритам за прављење контролног збира дефинисан у документу RFC1071. Уочимо

[1] *Internet Control Message Protocol*
[2] *Internet Group Management Protocol*
[3] *Gateway to Gateway Protocol*
[4] *Stream*
[5] *Exterior Gateway Protocol*
[6] *Interior Gateway Protocol*
[7] *Encap Security Payload for IPv6*
[8] *Authentication Header for IPv6*
[9] *Open Shortest Path First*
[10] Комплетна листа додељених бројева може се наћи у STD2.

да се збир мора израчунавати на сваком рутеру пошто се мења најмање једно од поља (нпр. TTL). Постоји и ефикаснији начин и описан је у документу RFC1141.

Адреса изворишта[1] (32 бита) указује на адресу мреже и адресу станице.

Адреса одредишта[2] (32 бита) указује на адресу мреже и адресу станице.

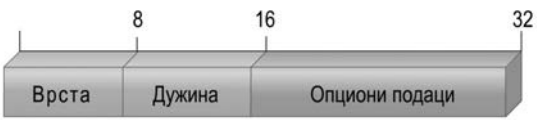
Опције[3] (променљиве дужине) имају формат приказан на слици 3.31 (поља врста, дужина[4] и опциони подаци[5]) или само 1 бајт (врста[6]);.

У оба случаја поље врста има структуру приказану на слици 3.32 где је:

- O_K[7] је поље које указује да ли ће се (ако је вредност поља 1) или се неће (ако је вредност поља 0) копирати поље опције приликом деобе (фрагментације) пакета,
- класа је поље које садржи 2- битски број без знака:
  - 0 - управљање,
  - 1 - резервисано,
  - 2 - откривање грешке и мерење,
  - 3 – резервисано;
- дужина је поље које садржи број бајтова опције узимајући у обзир и поља врста и дужина;

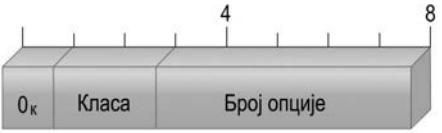
Описаћемо само неке од опција:

- 2 је вредност поља „број опције“ и односи се на сигурност[8]. Класа је 0, O_K бит је постављен на вредност 0, поље „дужина” има



Слика 3.31 Формат поља опције

[1] *Source Address*
[2] *Destination Address*
[3] *Options*
[4] *Length*
[5] *Option Data*
[6] *Type*
[7] *Flag copy*
[8] Описано у документу RFC1108.



Слика 3.32 Структура поља врста

вредност 11 и 8 бајтова за податке. Користи се за безбедносне информације које су потребне организацији DoD[1];

- 3 је вредност поља „број опције“ и односи се на незаобилазно рутирање[2]. Класа је 0, O_K бит је постављен на вредност 1, и променљиве је дужине поље за податке. Изворište специфицира листу рутера које не треба заобићи;
- 4 је вредност поља „број опције“ и односи се на означавање времена[3]. Класа је 2, ОК бит је постављен на вредност 0 и променљиве је дужине поље за податке. Дозвољава сваком рутеру да додаје своју адресу и време када је рутиран;
- 7 је вредност поља „број опције“ и односи на прављење записа о рути[4]. Класа је 0, ОК бит је постављен на вредност 0 и променљиве је дужине поље за податке. Дозвољава сваком рутеру да додаје своју адресу;
- 9 је вредност поља „број опције“ и односи се на стриктно рутирање изворишта[5]. Класа је 0, ОК бит је постављен на вредност 1 и променљиве је дужине поље за податке. Даје комплетан пут који треба пакет да следи;

Опционе податке сачињавају подаци који се односе на ту опцију.

Додатак[6] (променљиве дужине) је део који се додаје пакету уколико дужина пакета (датаграма) није целобројни умножак од 32. Састоји се од свих нула.

Подаци (променљиве дужине) прослеђују се вишим слојевима. Максимална дужина пакета (поља података и заглавља) је 65535 бајтова.

---

[1] *U.S. Department of Defense*
[2] *Loose source routing*
[3] *Timestamp*
[4] *Record route*
[5] *Strict source routing*
[6] *Padding*

3.4 Интернет протокол верзије 6

Број рачунара који је повезан на Интернет расте изузетно великом брзином. У јулу 2005. год. Интернет је користило скоро 350000000 станица[1]. Са IPv4 величином адресног поља могло би да се обезбеди 2^{32} различитих адреса што је десет пута више него број станица који тренутно користи Интернет.

Има више разлога због којих је IPv4 адресни простор постао неодговарајући. Навешћемо неке од њих:

- Неефикасно коришћење адресног простора:
  - двослојна структура IPv4 адреса (број мреже и број станице) је добро решење, али је и непотребно трошење адресног простора. Када се број мреже једном додели одређеној мрежи, све станице на тој мрежи аутоматски добијају свој број (адресу). Адресни простор те мреже може бити неискоришћен, али ако се посматра ефекат на IPv4 адресни простор онда када се искористи број мреже искоришћене су и све адресе за станице у оквиру те мреже;
  - адресни простор верзије IPv4 подељен је у мреже класа A, B и C различитих величина. Потребно је посебно анализирати адресни простор сваке од њих;
  - модел IPv4 адресирања генерално захтева да јединствен мрежни број буде додељен свакој IPv4 мрежи без обзира да ли јесте (или није) повезана на Интернет;
  - као последица неефикасности у подмрежавању сложено је искористити све адресе у блоку. Густина крајњих станица[2] (дефинисана у документу RFC3194) мера је за искоришћеност IPv4 адресних блокова;
- Број рачунарских мрежа расте великом брзином. Многе организације без икакве потребе користе уместо једне више локалних рачунарских мрежа;
- Бежичне мреже добијају на значају;
- Проширивање области у којима се користе TCP/IP протоколи као

[1] *ISC Internet Domain Survey, http://www. isic.org*

[2] *Host-density ratio*

што су POS[1] терминали, мобилни уређаји[2], кабловска телевизија доводе до повећања потражње за IP адресама;

- Стални приступ - некада је доминантан начин приступа био коришћењем јавне телефонске мреже са комунтираним телефонским линијама. Од 2007. године, широкопојасни приступ премашује 50% свеукупног приступа Интернету. Широкопојасни приступ се данас уобичајено користи. И међумрежни уређаји (рутери, широкопојасни модеми) веома се ретко искључују. На тај начин се број адреса које интернет посредници користе стално повећава;
- Интернет демографија. 90-тих година прошлог века мали број домаћинстава имао је приступ Интернету. Само 15 година касније скоро половина поседује стални, широкопојасни, приступ Интернету;
- Виртуелизација - напретком перформанси хардвера и способношћу обраде серверских система постало је могуће инсталирати више оперативних система на једном рачунару. Сваки од тих система може да захтева јавну IP адресу.
- Предвиђања су била да ће се скуп IPv4 адреса са којима организација IANA располаже испразнити у 2011. години, а да ће IPv4 адресе код различитих регионалних организација „нестати“ у 2012.години.

Видимо да су потребе за већим адресним простором довеле до потребе за новом верзијом IP протокола. Протокол IPv4 је дуго у употреби а у међувремену су се појавили нови захтеви везани за конфигурацију адреса, рутирања и механизмима подршке у управљању саобраћајем. Проблеме које IPv4 протокол није могао да разреши су и:

- У чворовима (рутерима) Интернета (нарочито у окосницама) табеле рутирања су постале превелике да би се могле одржавати, без обзира на примену CIDR[3] рутирања;
- Апликације у реалном времену (које се све више користе) захтевају да постоје класе услуга и механизми приоритета. Код

[1] *Point Of Sale*

[2] Спецификација за 4G уређаје захтева IPv6 адресирање.

[3] *Classless Inter Domain Routing*

верзије IPv4 механизми који обезбеђују квалитет услуге нису били јасно дефинисани и веома су се мало користили;

Организација IETF је као одговор на постојеће проблеме[1] 1992. год. започела процедуру за нову верзију интернет протокола. До данас су објављени бројни документи међу којима су најважнији: RFC1752[2], RFC2460[3], RFC2373[4], RFC2374[5] и унапређене верзије (2003. год.) RFC3513[6] и RFC3587[7].

Разлике између верзија IPv6 и верзије IPv4 су следеће:

- Дужина IPv6 заглавља повећана је на 40 бајтова (са 20 бајтова IPv4) и садржи две адресе (изворишта и одредишта) од по 16 бајтова. Заглавље IPv4 (слика 3.22) садржи две адресе од по 4 бајта иза којих је 12 бајтова управљачких информација и обично опциони подаци. Проширење адресног простора је за фактор 2^{96}. Израчунато је да ово омогућава $6x10^{23}$ јединствених адреса по квадратном метру површине Земље. Како год да се адресе додељују, овај адресни простор је довољно велики. Предвиђања су да ће бити довољан за следећих 30 година;
- Намера пројектаната верзије IPv6 је да се смањи време обраде пакета у рутерима смањивањем броја управљачких информација и измештање опционих података из заглавља. Опције IPv6 смештене су у посебна заглавља која су постављена између заглавља IPv6 и заглавља транспортног слоја. Највећи број додатних заглавља рутери, преко којих пракети прелазе, не анализирају;
- Повећана флексибилност адресирања: IPv6 укључује и концепт адресе један ка једном од групе (еникаст). Пакет се испоручује само једним путем. Перформансе рутирања један ка групи (мултикаст) су унапређене тако што је додат опсег поља за мултикаст адресе;

---

[1] *IPv6 Global Unicast Address Format*

[2] *IP Address Exhaustion Problem*

[3] Сматра се прекретницом у дефинисању новог протокола означеног са IPng. (*Recommendation for the IP Next Generation*).

[4] Односи се на свеобухватну спецификацију протокола IPv6.

[5] Односи се на структуру адресирања IPv6.

[6] *An IPv6 Agregatable Global Unicast Format*

[7] *Internet Protocol Version 6 Addressing Architecture*

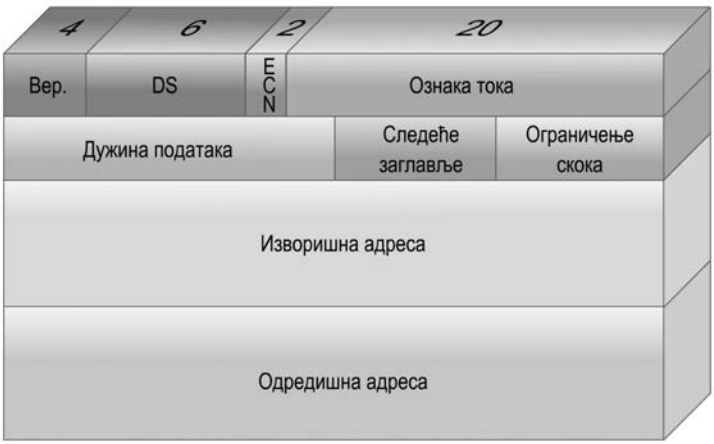
- Подршка у додели ресурса: IPv6 омогућава означавање пакета за различите брзине протока ако то пошиљалац тражи. На располагању су и додатне услуге у саобраћају када се преноси видео сигнал[1].

3.4.1 Формат заглавља IPv6 пакета

Заглавље IPv6 пакета је поједностављено у односу на заглавље IPv4 пакета. Формат заглавља IPv6 пакета представљен је на слици 3.33.

Састоји се од следећих поља:

- Верзија (дужине 4 бита) - вредност је 6 за IPv6 протокол;
- DS/ECN[2] (дужине 8 бита) - првих шест бита поља односи се на поље диференцираних услуга (DS[3]), а остала 2 бита резервисана су за експлицитно обавештења о загушењу (ECN[4]). Рутери обавештавају одредиште да „прети“ загушење у саобраћају. Одредиште обавештава извориште користећи протокол транспортног слоја. Обавештење о загушењу



Слика 3.33 Формат заглавља IPv6 пакета

[1] *Real-time video*

[2] У првим документима ово поље означавано је као „класе саобраћаја“ (енг. *traffic class*). Његова употреба је резервисана за почетне чворове и/или прослеђујуће рутере да би се направила разлика између различитих класа приоритета IPv6 пакета.

[3] *Differentiated Services*

[4] *Explicit Congestion Notification*

разликује се од механизама које користи TCP протокол и оно се означава као имплицитно обавештење о загушењу. TCP протокол користи индиректне механизме којима закључује да је дошло до загушења у мрежи[1].

- Ознака тока (дужине 20 бита) - може користити станица да обележи оне пакете са којима рутери треба да поступају на посебан начин у оквиру мреже;
- Дужина података (дужине 16 бита) - дужина остатка IPv6 пакета који прати заглавље, у октетима. Другим речима, ово је комплетна дужина свих заглавља и јединице података транспортног слоја (TPDU[2]);
- Следеће заглавље (дужине 8 бита) - идентификује тип заглавља које прати IPv6 заглавље. Ово може бити неко од IPv6 додатних заглавља или заглавље вишег слоја, као што су заглавља TCP или UDP протокола. Поред оних вредности које су идентичне са верзијом IPv4 ово поље може да има следеће вредности:
  - 41- IPv6 заглавље,
  - 45 - међудоменски протокол за рутирање[3],
  - 58 -ICMP пакет протокола IPv6,
  - 46 - протокол за резервисање ресурса[4].

  Вредности за додатна заглавља су следеће:
  - 0 - заглавље опције корак по корак ,
  - 43 - заглавље IPv6 рутирања,
  - 44 - заглавље IPv6 делова,
  - 50 - заглавље сигурности податка,
  - 51 - заглавље о IPv6 аутентификацији,
  - 59- нема следећег заглавља[5],
  - 60 - заглавље одредишних опција.
- Ограничење скока (дужине 8 бита) - колико је још скокова за овај пакет дозвољено. Ограничење скока поставља извориште

[1] Детаљније у поглављу о TCP протоколу.
[2] *Transport Protocol Data Unit*
[3] *Interdomain Routing Protocol*
[4] *Resource Reservation Protocol*
[5] *No Next Header*

на одговарајућу вредност. Декрементира се у сваком чвору који прослеђује пакет. Пакет се одбацује[1] када вредност овог поља постане нула[2];

- Изворишна адреса (дужине 128 бита) - адреса оног који пакет формира;
- Одредишна адреса (дужине 128 бита) - адреса оног који пакет прима. Ово не мора да буде крајња одредишна адреса ако је присутно заглавље рутирања;
- Контролни збир - протокол IPv6 не израчунава контролни збир зато што транспортни протоколи[3] то већ раде и зато што постоји опција за аутентификацију која може да обезбеди интегритет података.

Иако је IPv6 заглавље дуже него обавезни део IPv4 заглавља (40 према 20 бајтова) број поља је мањи (8 према 12). То значи да ће рутери имати мање посла око обраде заглавља чиме је процес рутирања значајно убрзан.

3.4.2 Структура IPv6 пакета

Пакет IPv6 има општу структуру приказану сликом 3.34. Сваки IPv6 пакет започиње са основним заглављем. У многим случајевима ово ће бити и једино заглавље које је потребно да би се пакет испоручио одредишту. Некада је потребно пренети и још неке податке[4] одредишту или рутерима на путу до одредишта. Додатна заглавља се користе у ту сврху а постављају се одмах иза IPv6 заглавља. Свако додатно заглавље (осим оног које носи број 59) има своје поље[5] које означава који тип (врста) заглавља следи иза њега. Оваква структура омогућава уланчавање различитих додатних заглавља (проширења).

Дефинисана су следећа додатна заглавља (проширења):

- заглавље опције корак по корак[6]: дефинише специјалне опције које захтевају обраду корак по корак,

---

[1] *Rejects*

[2] Ово је поједностављен поступак у односу на поступак који би требало да се уради са TTL пољем код IPv4. Рад са временским интервалима код верзије IPv4 није донео никакву предност. У ствари рутери са IPv4 протоколом третирали су TTL поље само као ограничење у броју скокова.

[3] Код израчунавања контролне суме TCP и UDP користе псеудозаглавље у коме је садржана и IP адреса.

[4] Код верзије IPv4 тај задатак обављало је поље опција.

[5] *Next Header Field*

[6] *Hop-by-Hop Options header*

Слика 3.34 Форма пакета IPv6

- заглавље IPv6 рутирања[1] - обезбеђује проширење рутирања, слично изворишном рутирању у IPv4,
- заглавље IPv6 делова[2] (фрагмента) пакета - садржи информације о дељењу и поновном повезивању пакета,
- заглавље о IPv6 аутентификацији[3] - обезбеђује интегритет и аутентичност сваког пакета,
- заглавље о сигурности податка[4] - обезбеђује заштиту пакета,
- заглавље одредишних опција[5] - садржи опционе информације које ће одредишни чвор анализирати.

У случају када се користи више додатних заглавља IPv6 препоручује да се заглавља појављују следећим редоследом:

- заглавље IPv6 (обавезно мора да буде прво),
- заглавље опције корак по корак,
- заглавље одредишних опција - за опције које ће бити обрађене од стране првог одредишта које се појављује у пољу IPv6. Одредишне адресе и накнадна одредишта која су набројана у заглављу рутирања,
- заглавље IPv6 рутирања,
- заглавље IPv6 делова,
- заглавље аутентификације,
- заглавље сигурности података,
- заглавље одредишних опција - за опције које ће бити обрађене само у крајњем одредишту пакета.

Слика 3.35 приказује пример једног IPv6 пакета који укључује део сваког заглавља осим оних која су повезана са сигурношћу. IPv6 заглавље

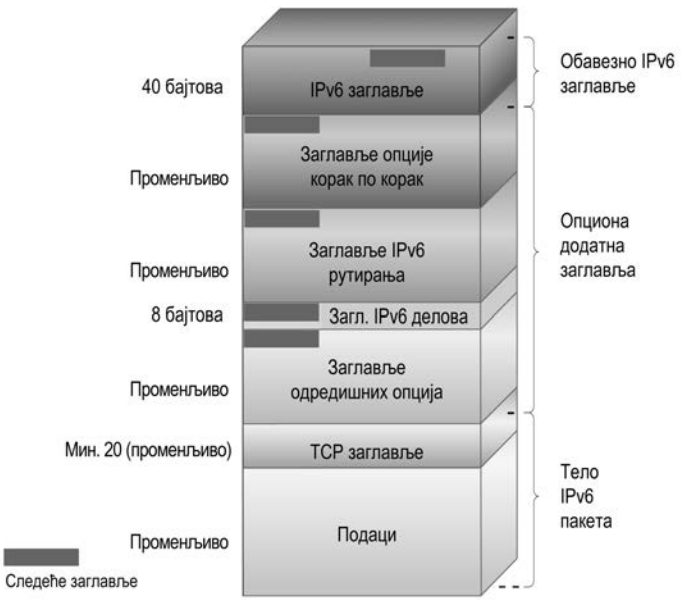
[1] *Routing header*
[2] *Fragment header*
[3] *Authentication header*
[4] *Encapsulating Security Payload header*
[5] *Destination Option header*

и свако продужено заглавље садрже 8-битно поље „следеће заглавље". Ово поље идентификује тип следећег заглавља. Ако је следеће заглавље једно од продужених заглавља, онда ово поље садржи идентификатор типа тог заглавља. У противном, ово поље садржи идентификатор протокола вишег слоја који користи IPv6 (обично протокол транспортног слоја).

На слици 3.29 као протокол вишег слоја постављен је TCP протокол. Подаци (виших слојева) које носи IPv6 пакет садрже заглавља TCP протокола и податке апликације. Пример уланчавања додатних заглавља дат је на слици 3.36.

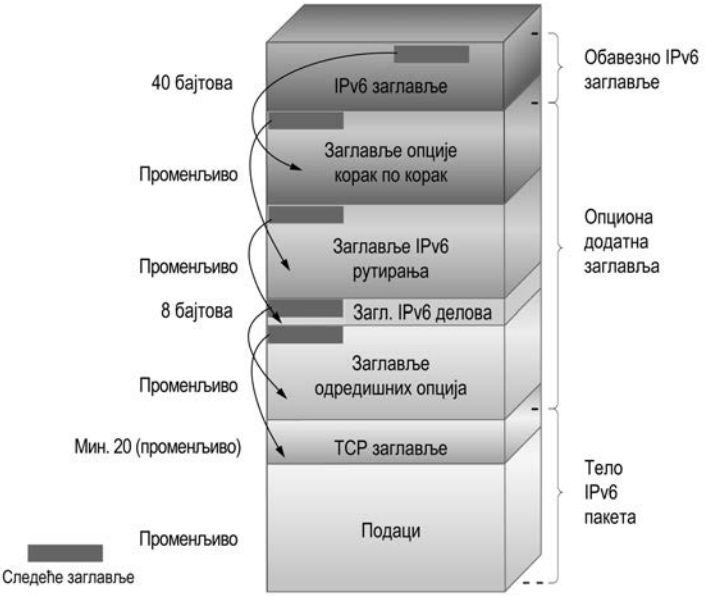
3.4.3 Ознака тока

IPv6 стандард дефинише ток (проток) као низ пакета који се шаљу од одређеног изворишта ка одређеном одредишту[1] и за које изворište захтева од рутера да на посебан начин поступа. Проток је јединствено дефинисан тако што се комбинују изворишна и одредишна адреса са



Слика 3.35 IPv6 пакет са продуженим заглављима

[1] Уникаст (*unicast*) или мултикаст (*multicast*).



Слика 3.36 Пример уланчавања

ознаком тока која је 20-битни број различит од нуле. Извориште додељује исту ознаку тока[1] свим пакетима који су део једног тока.

Посматрано са стране извориште ток је низ пакета које је генерисала апликација извориште а који захтева исту транспортну услугу. Ток може да обухвати једну или више транспортних (TCP) веза. Једна апликација може да генерише један или више токова. Пример два тока је мултимедијални састанак коме је додељен један ток за звук а други за слику. Сваки од токова има различите захтеве код преноса података, кашњења или одступања у кашњењу.

Ако се погледа из угла рутера, ток је низ пакета који има исте атрибуте који утичу на то како ће се рутер понашати према пакетима тог тока. Ту спадају: одабрана путања, додељени ресурси, критеријуми за одбацивања, алгоритам заштите итд. Рутер може да се понаша другачије према пакетима који нису из истог тока. На пример: доделом различитог меморијског простора (бафера), придруживањем различитог приоритета

[1] *Flow label*

код прослеђивања пакета или различитим захтевом за квалитетом услуге који се од мреже очекује.

Не постоји посебно значење било које ознаке тока. Уместо тога, да би се обезбедио специјални поступак ток података мора се обележити на неки други начин. Примери специјалног поступка су и захтеви за нестандардним квалитетом услуге или неком од услуга у реалном времену.

У начелу, сви кориснички захтеви за одређеним током могу бити дефинисани у додатним заглављима и укључени у сваки од пакета. Ако се жели оставити концепт тока потпуно отвореним за мноштво захтева, резултат би могао бити превелико заглавље пакета. Друга могућност, која је усвојена код протокола IPv6, је ознака тока код које су захтеви тока дефинисани на самом почетку а јединствена ознака тока додељена је сваком току. У овом случају рутер мора да чува информације о захтевима сваког тока.

Следећа правила важе за ознаке тока:

- Станица, или рутер који не подржава поље „ознака тока“ мора да када формира пакет постави ово поље на нулту вредност, када преусмерава пакет да не мења његову вредност и на пријему да га игнорише;
- Сви пакети који су направљени у истом изворишту морају имати исту: вредност поља „ознака протока“, одредишну адресу, изворишну адресу, садржај заглавља опције „корак по корак“ (ако постоји) и садржај заглавља „рутирање“ (ако постоји). Рутер може да обрађује пакет тако што ће једноставно погледати ознаку тока у табели без испитивања осталих заглавља;
- Извориште додељује току ознаку тока. Нова ознака тока мора бити изабрана као случајни број из опсега 1 до 2^{20}-1. Услов је да извориште не може да додели ознаку тока новом току ако се она користи за ток који је још активан. Вредност нула указује да се ниједна од ознака тока не користи.

3.4.4 Додатна заглавља IPv6 пакета

У додатним заглављима преносе се подаци намењени одредишту а постављена су иза IPv6 заглавља.

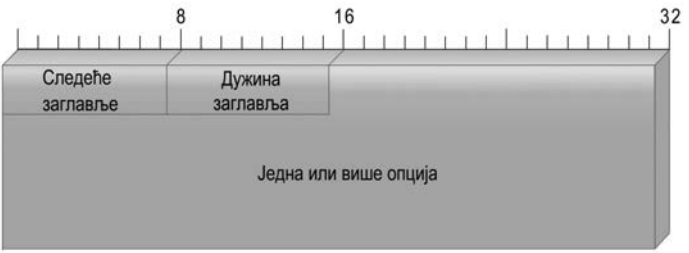
Заглавље опција корак по корак

Заглавље опција корак по корак носи опционе информације које, уколико је присутно, мора да анализира сваки рутер који се налази на путањи тог пакета. Састоји се (слика 3.37) од следећих поља:

- Следеће заглавље[1] (дужине 8 бита): одређује тип заглавља које се налази иза овог заглавља;
- Дужина заглавља[2] (дужине 8 бита) означава дужину тог заглавља исказаног у јединицама од по 64 бита, не укључујући прва 64 бита;
- Опције[3]: поље променљиве дужине које се састоји од поља која дефинишу опцију (означава се као TLV[4] формат):
  - тип опције (дужине 8 бита) који одређује опцију,
  - дужина опције (дужине 8 бита) која одређује дужину поља опције података у бајтовима и
  - подаци опције (променљиве дужине).

Да би се описала нека одређена опција у пракси се из поља „тип опције“ користи пет битова најмање тежине. Два бита највеће тежине указују на активност коју чвор треба да одради уколико не препознаје тип опција:

- 00 – прескочи ову опцију и настави са обрадом заглавља,
- 01 – одбаци пакет,
- 10 – одбаци пакет и пошаљи ICMP поруку о проблему са



Слика 3.37 Опције корак по корак

[1] *Next header*
[2] *Header extension length*
[3] *Options*
[4] *Type-Length-Value*

параметрима[1] изворишној адреси пакета, што указује на немогућност препознавања типа опција,

- 11 – одбаци пакет само под условом да одредишна адреса није мултикаст адреса, пошаљи ICMP поруку о проблему са параметрима изворишној адреси пакета, указујући на непрепознатљив тип опција.

Трећи бит највеће тежине одређује да ли се на путу од изворишта до одредишта поље „опција података“ не може (вредност 0) или може (вредност 1) мењати.

Конвенције које се користе код поља „тип опција“ односе се и на заглавље „одредишне опције“.

За сада су дефинисане четири опције корак по корак :

- Додатак 1[2] - користи се да би се уметнуо један бајт (додатак) у делу заглавља за опције;
- Додатак N - користи се да би се уметнуло N (N≥2) бајтова (додатка) у делу заглавља за опције;
- Велики део за податке[3] - користи се да би се послао IPv6 пакет са корисним подацима већим од 65535 бајтова. Поље „опција података“ је дужине 32 бита и указује на дужину пакета изражену у бајтовима изузимајући IPv6 заглавља. За ове пакете вредност поља „дужина података“ у IPv6 заглављу мора бити постављена на нулу и не сме бити присутно заглавље фрагмената. Са овом опцијом протокол IPv6 подржава величину пакета до 4 милиона бајтова. На тај начин омогућен је пренос великих пакета са видео садржајем и омогућава се IPv6 протоколу да на најбољи могући начин искористи капацитет трансмисионих медијума;
- Упозорење рутера[4] - информише рутер да је садржај пакета користан за рутер и да треба да поведе рачуна о управљачким подацима који су у њему. Изостанак ове опције у IPv6 пакету знак је рутеру да пакет не садржи информације које су њему од значаја. Због тога рутер ове пакете може проследити даље

[1] *Internet Control Message Protocol (ICMP) Parameter Problem*
[2] *Pad*
[3] *Jumbo payload*
[4] *Router alert*

без рашчлањавања и анализе појединих делова. Изворишне станице IPv6 пакета треба ову опцију да користи само под одређеним условима. Опција обезбеђује ефикасну помоћ протоколима као што је нпр. протокол за резервацију ресурса (RSVP[1]) који генерише пакете које анализирају међурутери у циљу управљања саобраћајем. Уместо да се од међурутера захтева детаљна анализа додатних заглавља пакета ова опција опомиње рутер када је то заиста потребно.

Заглавље дела (фрагмента) пакета

Код протокола IPv6 дозвољено је да деобу уради само изворишне станице а не и рутери преко којих пакет прелази на свом путу до одредишта. Да би се искористиле комплетне предности умрежавања изворишни чвор мора применити алгоритам за откривање путање који јој омогућава да сазна која је то најмања максимална јединица за пренос (MTU[2]) коју рачунарске мрежа на путу пакета од изворишта до одредишта подржавају[3]. Када сазна величину MTU јединице изворишна станица може, узимајући тај податак у обзир, пакет да подели. Уколико изворište нема податак о MTU јединици онда оно мора да величину пакета ограничи на 1280 бајтова, колика је минимална MTU јединица коју свака рачунарска мрежа мора да подржи.

Заглавље дела пакета састоји се од следећих поља (слика 3.36):

- Следеће заглавље[4] (дужине 8 бита) - одређује тип заглавља које се налази иза овог заглавља;
- Резервисано[5] (дужине 8 бита) - резервисано за будућу употребу;
- Одступање дела пакета[6] (дужине 13 бита) - указује где је у оригиналном пакету место корисних података[7] фрагмента. Мери се у јединицама од по 64 бита. Ово значи да део пакета (осим

[1] *Resource Reservation*, дефинисан документом RFC2205. Иновиран и проширен у бројним документима ка о што су нпр. RFC2750 и RFC4495.

[2] *Maximum Transmission Unit*

[3] Објашњено је у документу RFC 1191 (*Path MTU discovery*).

[4] *Next Header*

[5] *Reserved*

[6] *Fragment Offset*

[7] *Payload*

Слика 3.38 Заглавље дела (фрагмента) пакета

последњег) мора да садржи поље података које је умножак од 64 бита;

- Резервисано (дужине 2 бита) - резервисано за будућу употребу;
- M[1] ознака (дужине 1 бит) - када је M=1 значи да има још делова а када је M=0 значи да је то последњи део оригиналног пакета;
- Идентификација[2] (дужине 32 бита) - служи за јединствену идентификацију оригиналног пакета. Идентификатор мора бити јединствен за изворишну и одредишну адресу пакета за време док је он на Интернету. Сви делови пакета са истим идентификатором, изворишном и одредишном адресом поново се повезују како би се добио оригинални пакет.

Начин деобе (фрагментације) је исти као што је описано у делу о деоби IPv4 пакета.

Заглавље дела за рутирања

Заглавље дела за рутирања користи се за усмеравање пакета ка једном или више међучворова пре него што се пошаље ка свом крајњем одредишту. Због тога заглавље дела за рутирање и садржи листу једног или више чворова преко којих пакет треба да пређе на свом путу до одредишта. Заглавље дела за рутирања почиње са 32- битским делом који се састоји од четири 8-битских поља, иза којих следи поље са подацима за рутирања који су специфични за одређени тип рутирања (слика 3.39). Четири 8-битна поља су следећа:

- Следеће заглавље[3] - одређује тип заглавља које се налази иза овог заглавља;

[1] *M flag*
[2] *Identification*
[3] *Next Header*

- Дужина заглавља[1] - означава дужину тог заглавља исказаног у јединицама од по 64 бита, не укључујући прва 64 бита;
- Тип рутирања[2] - има вредност 0, 1 или 2:
  - тип 0 је из сигурносних разлога (опасности од DOS[3] напада) одбачен[4]. Препоручује се рутерима и крајњим станицама да игноришу ово заглавље;
  - тип 1 користи организација DARPA[5];
  - тип 2 се користи код мобилног IPv6 протокола;

  Ако рутер не препозна вредност типа рутирања, мора да одбаци пакет;
- Преостали сегменти[6] (број преосталих сегмената рутирања) - указује на број преосталих чворова (од оних који су експлицитно наведени) преко којих пакет још треба да пређе на путу до одредишта;

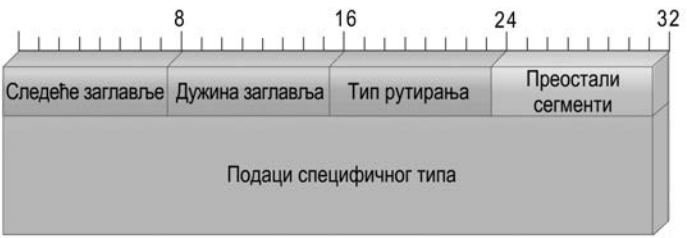
Заглавље рутирања се не испитује док пакет не стигне до одредишне адресе која је постављена у IPv6 заглављу.

Заглавље одредишних опција

Заглавље одредишних опција носи опционе информације које, уколико је заглавље присутно, испитује само одредиште пакета. Формат овог заглавља исти је као и формат заглавља опција корак по корак.

Заглавље аутентификације

Ово заглавље се користи да би потврдило да примљени пакет



Слика 3.39 Заглавље рутирања

[1] *Header Extension Length*
[2] *Routing Type*
[3] *Denial-Of-Service*
[4] Описано у документу RFC5095.
[5] *Defense Advanced Research Projects Agency*
[6] *Segments Left*

није измењен у преносу и да заиста потиче од пошиљаоца за кога се представља. Заглавље аутентификације је одређено вредношћу 51 у пољу следеће опције.

Упаковани заштићени корисни подаци

Упаковани заштићени корисни подаци (ESP[1]) представљају специјално додатно заглавље које се може наћи било где у пакету између основног заглавља и заглавља неког од виших протокола које пакет преноси. Сви подаци који се налазе иза ESP заглавља су шифрирани.

3.4.5 Модел IPv6 адреса

Комбинација адреса велике дужине (128 бита) и више адреса додељених интерфејсу омогућава ефикасније рутирање у поређењу са верзијом IPv4. Код IPv4 протокола адреса није направљена тако да потпомаже процес рутирања. Због тога је рутер морао да води рачуна о великим табелама путања (рутирања).

Дуже интернет адресе омогућавају повезивање већег броја адреса према: хијерархијама рачунарских мрежа, добављачима интернет услуга преко којих се приступа Интернету, географској локацији, итд. Такав број требало би да учини да табеле рутирања буду мање, а преглед табела бржи. Претплатнику је омогућено да приступа Интернету преко више добављача интернет услуга а да за то користи исти интерфејс.

Могуће су три врсте IPv6 адреса[2]:

- Уникаст адреса је идентификатор једног интерфејса. Пакет послат уникаст адресом испоручује се интерфејсу који је одређен (идентификован) том адресом;
- Еникаст адреса је идентификатор за више интерфејса који припадају различитим чворовима. Пакет послат еникаст адресом испоручује се неком од интерфејса који је одређен том адресом (најближем, гледајући по протоколу за рутирање);
- Мултикаст адреса је идентификатор за групу интерфејса који, обично, припадају различитим чворовима. Пакет послат мултикаст адресом испоручује се свим интерфејсима који су одређени том адресом.

---

[1] *Encapsulated Security Payload*

[2] Детаљно је описано у документима RFC3513, RFC3585 и RFC3587.

Бродкаст адресе нису дефинисане у протоколу IPv6, а њихова функција је замењена мултикаст адресама.

Адресе IPv6 додељују се сваком од интерфејса чвора[1] а не самом чвору. Сваки интерфејс може да има више јединствених уникаст адреса. Једна уникаст IPv6 адреса односи се на један интерфејс. Пошто један интерфејс припада једном чвору било која уникаст адреса чвора може се искористити за идентификацију чвора. Један интерфејс може да има више IPv6 адреса било које врсте уникаст, еникаст, мултикаст.

Уникаст адреса или скуп уникаст адреса може се доделити за више физичких интерфејса уколико се у примени посматрају као један интерфејс када се „представља“ интернет слоју. Ово је корисно код распоређивања оптерећења преко више физичких интерфејса.У текућој употреби је принцип преузет из верзије 4 а то је да се префикс подмреже додељује једном линку. Више префикса подмрежа могу се доделити истом линку.

3.4.6 Начин представљања IPv6 адреса

У рачунарима сви су бројеви представљени у бинарном бројном систему. Људи су навикли да користе децимални бројни систем па су због тога и IPv4 адресе представљене у децималном бројном систему осим када (као што је случај са подмрежавањем) није потребно користити бинарни бројни систем. Величина IPv6 адреса значајно је већа од IPv4 адреса па би се коришћењем децималне нотације добило:

254.220.186.152.118.84.50.16.254.220.186.152.118.84.50.16.
16.128.0.0.0.0.0.0.8.8.0.32.12.65.122
FEDC:BA98:7654:3210:FEDC:BA98:7654:3210
1080:0000:0000:0000:0008:0800:200C:417A

Због своје величине IPv6 адресе представљене су са 8 група од по четири хексадецималне цифре. Користе се и два друга начина представљања: са смањењем броја нула и са уклањањем нула.

Постоји и мешовити начин представљања који се користи када су IPv4 адресе угњеждене у IPv6 адресе[2]. У том случају погодно је да је IPv4 адреса представљена у децималној нотацији:

---

[1] Код протокола IPv6 термин чвор односи се и на рутер и на крајњу станицу; односно на сваки уређај код кога је имплементиран IPv6 протокол.

[2] *IPv6 that embed IPv4*

0:0:0:0:0:0:13.1.68.3

0:0:0:0:0:FFFF:129.144.52.38

IPv6 адресе подељене су у два дела (као код IPv4): мрежни део адресе и део адресе крајње станице (хоста). Мрежни део адресе означава се као префикс а број који указује колике је величине је дужина префикса. Префикс (у примеру који следи је величине 48) користи се на исти начин као код бескласног CIDR IPv4 адресирања:

805B:2D9D:DC28:FC57:D4C8:1FFF/48

Уколико се у адреси јави дуги низ нула оне се замењују са „::“ и означава се као уклањање или компресија нула: на пример 805B:2D9D:DC28:0:0:0:0:0/48 може се заменити са 805B:2D9D:DC28::/48.

Пример правилног представљања адреса 12AB00000000CD316 са 60-битним префиксом:

- 12AB:0000:0000:CD30:0000:0000:0000:0000/60
- 12AB::CD30:0:0:0:0/60
- 12AB:0:0:CD30::/60

Пример неправилног представљања адреса:

- 12AB:0:0:CD3/60 – могу се изоставити нуле на почетку али не на крају 16-битних делова адресе
- 12AB::CD30/60 адреса лево од „/“ проширена је на 12AB:0000:000 0:0000:0000:000:0000:CD30
- 12AB::CD3/60 адреса лево од од „/“ проширена је на 12AB:0000:0 000:0000:0000:000:0000:0CD3

Адреса чвора и адреса префикса тог чвора (тј. префикс подмреже чвора) могу се комбиновати на следећи начин: адреса чвора 12AB:0:0: CD30:123:4567:89AB:CDEF и његов број подмреже 12AB:0:0:CD30::/60. Скраћено се може написати као: 12AB:0:0:CD30:123:4567:89AB:CDEF/60

3.4.7 Идентификатори врсте IPv6 адреса

Као што је био случај и са IPv4 адресама основ за одлуку о подели IPv6 адресног простора заснован је на начину доделе адреса и рутирању. Пројектанти IPv6 концепта желели су да структурирају адресни простор тако да је интернет посредницима, организацијама и појединцима додела адреса што једноставнија.

Битови највеће тежине у IPv6 адреси указују на неке њене особености (тип адресе) и означавају се као префикс који је описан у документу RFC4291[1]. Формат префикса концепцијски је идентичан битовима 1 до 4 који се користе у IPv4 адресама са класама[2] али је променљиве дужине од три до десет битова. Без обзира што IPv6 адресе нису засноване на класама постоје различите врсте IPv6 адреса које су одређене форматом њеног префикса. Постоје резервисане IPv6 адресе за затворене петље[3] (::1), мултикаст (која почиње са FF) итд. [4567]

Такође постоје неспецифициране адресе[8] које се састоје од свих нула (0:0:0:0:0:0:0:0, компримоване као ::) и које се могу користити као изворишна адреса IPv6 уређаја коме још није додељена IPv6 адреса.

Као што се види сви ови префикси представљени су у хексадецималном бројном систему. Адреса IPv6 која започиње са 1101 значи 0001 0001 0000 0001 а може се представити и као: 11::1. Мултикаст IPv6 започиње са FF_{16} и значи 1111 1111:1111 1111 (табела 3.6).

О IPv6 адресном простору својих корисника (као и код верзије 4) води рачуна интернет посредник обично почевши од вредности 200x. Постоји и начин да се доделе променљиве дужине поља за идентификатор организације[9], идентификатор интернет посредника, идентификатор

Врста адресе	Бинарни префикс	IPv6 нотација
Неспецифицирана[4]	00...0 (128 битова)	::/128
Затворена петља[5]	00...1 (128 битова)	::1/128
Мултикаст	11111111	FF00::/8
Уникаст локалног линка[6]	1111111010	FE80::/10
Глобална уникаст[7]	Све остало	

Табела 3.6 Врсте IPv6 адреса[8]

[1] Документи који су неважећи (*obsolete*) су RFC3513, RFC2373.
[2] *Classful*
[3] *Loopback*
[4] *Unspecified*
[5] *Loopback*
[6] *Link-Local Unicast Addresses*
[7] *Global Unicast*
[8] *Unspecified address*
[9] *Registry identifier* – организације које су задужене да додељују IPv6 адресни простор интернет посредницима.

корисника[1], идентификатор подмреже[2] и идентификатор интерфејса[3] (као што је MAC адреса). У ствари већина интернет посредника доделиће IPv6 адресе на исти начин како су то радили са IPv4 адресама: простор мрежних адреса и префикс. О адресама које су независне од интернет посредника[4] (на шта и сам назив указује) интернет посредници не воде рачуна.

3.4.8 Уникаст адресе

Сматра се да ће се IPv6 уникаст адресе користити за највећи део саобраћаја на Интернету, као што је био случај са IPv4 адресирањем. То је разлог што је највећи блок IPv6 адресног простора додељен уникаст адресама. Наравно, даље се поставља питање како се преосталих 125 битова адресног простора користи.

Пројектанти IPv6 адреса су користећи искуство IPv4 адресирања закључили да постоје велике предности уколико се уникаст адресе структурирају тако да пресликавају целокупну топологију Интернета. То укључује:

- једноставнију доделу адресних блокова на различитим нивоима тополошке хијерархије Интернета;
- IP мрежне адресе које аутоматски пресликавају хијерархију којом рутери прослеђују информације преко Интернета, дозвољавајући да се једноставно међусобно повезују адресе у циљу ефикаснијег рутирања;
- флексибилност за организације као што су интернет посредници у подели адресних блокова својим корисницима;
- флексибилност организација, крајњих корисника[5] да даље деле адресне блокове тако да то одговара њиховим интерним мрежама, слично као подмрежавање код IPv4 адресирања.
- више информација у IP адресама. Уместо да су само низ од 128 битова без икакве структуре могуће је погледати адресу и на основу тога знати нешто више о тој адреси.

[1] *Subscriber identifier*
[2] *Subnet identifier*
[3] *Interface identifier*
[4] *Provider independent*
[5] *End user organization*

IPv6 уникаст адресе могу се повезивати користећи префиксе произвољне дужине, слично начину на који се IPv4 адресе користе код CIDR рутирања. Постоји неколико врста уникаст адреса:

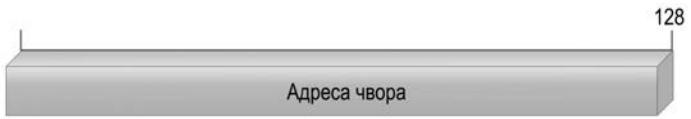
- глобалне уникаст адресе[1],
- уникаст адресе локалног места[2] и
- уникаст адресе локалног линка[3].

Специјалне подврсте глобалне уникаст адресе су IPv6 адресе са угњежденим IPv4 адресама (слика 3.45). Додатне врсте или подврсте могу бити у будућности дефинисане.

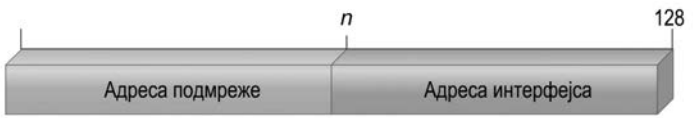
IPv6 чворови могу да имају значајно много или мало сазнања о интерној структури IPv6 адресе у зависности од улоге коју чвор има: нпр. да ли је он крајња станица или рутер. Најједноставнији случај је када чвор сматра да уникаст адреса (укључујући његову) нема интерну структуру (слика 3.40)

Мало сложенија крајња станица (али и даље веома једноставна) може да познаје адресу (префикс) подмреже[4] за линкове на које је повезана. Различите адресе могу да имају различите вредности за број *n* којим се расподељује број битова између префикса подмреже и адресе (идентификатора) интерфејса[5] (слика 3.41).

Без обзира на чињеницу да веома једноставан рутер не познаје



Слика 3.40 Адреса чвора без интерне структуре



Слика 3.41 Адреса чвора са једноставном интерном структуром

[1] *Global unicast*

[2] *Site-local unicast*. Проглашене застарелим према RFC3879 и RFC4291.

[3] *Link-local unicast*

[4] *Subnet prefix*

[5] *Interface ID*

интерну структуру уникаст адресе, рутери ће генерално знати за једну или више хијерархијских граница када користе протоколе за рутирање. Границе ће се мењати од рутера до рутера, у зависности од позиције коју рутери заузимају у хијерархији рутирања.

Идентификатори интерфејса

Идентификатори интерфејса код IPv6 уникаст адреса користе се за идентификацију интерфејса на линковима. Захтева се да су јединствени у оквиру једне подмреже (тј. одређене префиксом подмреже). Препоручује се да се не додељују исти идентификатори интерфејса различитим чворовима на једном линку. Могу се користити исти идентификатори интерфејса под условом да се налазе на различитим подмрежама.

Треба уочити да је јединственост идентификатора интерфејса независна од јединствености IPv6 адреса. На пример глобална уникаст адреса може се направити са локалним опсегом идентификатора интерфејса а адреса локалног линка може се направити са универзалним опсегом идентификатора интерфејса.

За све уникаст адресе осим оних које почињу са бинарном вредношћу 000 захтева се да интерфејсни идентификатор буде дугачак 64 бита и представљен у модификованом EUI-64 формату, при чему се он најчешће изводи на основу јединственог идентификатора који интерфејс већ има, на пример MAC адресе.

На пример 48-битна IEEE MAC адреса представљена је на слици 3.42

На сликама 3.42 (и 3.43) су са „x“ означени битови додељени организацији - произвођачу (OUI[1]), са „u“ универзални/локални бит (могуће вредности су 0/1) и са „g“ индивидуално/групни бит. Део адресе (24 бита) који произвођач додељује интерфејсној картици означен је са „z“.

Идентификатор интерфејса (модификовани EUI-64) добија се:

- уметањем вредности $FFFE_{16}$ (1111111111111110_2) између дела додељеног произвођачу и дела који произвођач додељује интерфејсној картици и
- променом вредности универзалног/локалног бита са вредности 0 на вредност 1 (слика 3.43).

[1] *Organizationally Unique Identifier*

Слика 3.42 Формат MAC адреса

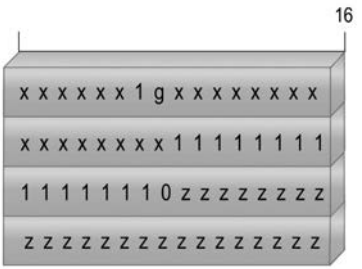
Неспецифицирана адреса

Адреса 0:0:0:0:0:0:0:0 назива се неспецифицирана адреса. Она се не сме доделити ниједном чвору. Указује на одсуство адресе. Један пример њене примене је вредност уписана у поље изворишне адресе IPv6 пакета који је послао иницијатор пре него што је добио (научио) своју адресу.

Неспецифицирана адреса се не сме користити као одредишна IPv6 адреса пакета или IPv6 рутирајуће заглавље. IPv6 пакет са неспецифицираном изворишном адресом не сме преусмеравати ниједан рутер.

Адреса затворене петље

Уникаст адреса 0:0:0:0:0:0:0:1 назива се адреса затворене петље. Може је користити чвор за слање IPv6 пакета самом себи. Не сме никад бити додељена било ком физичком интерфејсу. Посматра се као да је из опсега адреса локалног линка. Може се сматрати и као уникаст адреса виртуелног интерфејса имагинарног линка који не води никуда. Адреса затворене петље не сме се користити као изворишна адреса IPv6 пакета. Пакет са IPv6 одредишном адресом затворене петље рутери никада не преусмеравају.



Слика 3.43 Модификовани EUI-64 формат

Глобална уникаст адреса

Генерални формат IPv6 глобалне уникаст адресе састоји се од глобалног префикса рутирања[1], идентификатора (ИД) подмреже и идентификатора интерфејса (слика 3.44).

Глобална уникаст адреса састоји се од:

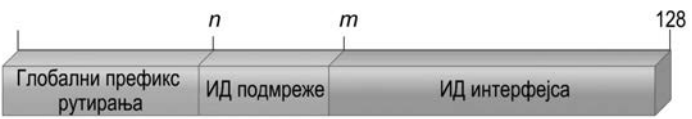
- глобалног префикса рутирања (обично хијерархијски структурираног) који представља вредност додељену месту (кластер[2], подмреже/линкови),
- ИД подмреже који је идентификатор линка у оквиру места и
- ИД интерфејса који одређује интерфејс на начи описан у претходном тексту.

Захтева се да све глобалне уникаст адресе, осим оних које почињу са 000_2 имају идентификатор интерфејса дужине 64 бита (тј. $n + m = 64$) који је направљен у модификованом EUI-64 формату. За глобалну уникаст адресу која започиње са 000 не важе ограничења у величини или структури поља идентификатора интерфејса. Пример глобалне уникаст адресе на чијем почетку је 000 је адреса са угњежденим[3] IPv4 адресама.

Пример глобалне адресе која започиње бинарном вредношћу која није 000 (и има 64-битно поље идентификатора интерфејса) може се наћи у документу RFC3587.

Глобални префикс рутирања хијерархијски је подељен за коришћење на Интернету слично CIDR методу. Постоји 45 битова (48 битова умањено за прва три бита која имају вредност 001). Структура уникаст адреса (објављена у документу RFC2374) предвиђала је поделу 45 битова на два хијерархијска нивоа:

- агрегатори највећег нивоа (TLA[4]) - највеће интернет организације



Слика 3.44 Адреса чвора са једноставном интерном структуром

[1] *Global routing prefix*
[2] *Cluster*
[3] *Embedded*
[4] *Top-Level Agregators*

којима су додељени велики блокови IPv6 адреса од организација за регистрацију и

- агрегатори нижег нивоа (NLA[1]) - ове организације добијају блокове адреса од TLA организација и деле их организацијама - крајњим корисницима (местима).

Августа 2003. године објављен је документ RFC3587 који предвиђа да регионалне организације за доделу адреса RIR[2] (APNIC, ARIN, LACNIC RIPE) саме одлуче како ће користити 45 битова. Значи концепт TLA/NLA је одбачен али се још увек може у литератури срести.

Документом RFC2374 дефинисан је префикс формата 001 (2000::/3). Проглашавањем RFC2374 документа застарелим формално би требало да је и овај префикс одбачен. Ипак за сада је организација IANA делегирала само префикс 2000::/3 што не значи да он има посебан значај. У будућности организација IANA ће можда делегирати недодељене делове IPv6 адресног простора такође за глобалне уникаст адресе.

Када се користи аутоконфигурација адреса (SLAAC[3]) подмрежа захтева /64 адресни блок као што је дефинисано у документу RFC4291. Локалним организацијама за регистрацију (LIR[4]) додељује се најмање /32 блока које оне расподељују између интернет посредника. Застарели документ RFC3177 препоручује доделу /48 крајњим корисницима. Документ којим је замењен (RFC6177) препоручује да се крајњим корисницима додели више од /64 али не и да им се додели /48. Посебно је разматрана могућност доделе блока /56. Видеће се како ће интернет посредници прихватити ову препоруку.

IPv6 адресе са угњежденим IPv4 адресама

Дефинисане су две врсте IPv6 адреса код којих најнижа 32 бита представљају IPv4 адресу:

- IPv4 адреса – усаглашена са IPv6 адресом[5] (њихово даље коришћење је одбачено[6]),

[1] *Next-Level Agregators*
[2] *Regional Internet Registries*
[3] *Stateless Address Autoconfiguration*
[4] *Loal Internet Registries*
[5] *IPv4 - compatible IPv6 address*
[6] Документ RFC4291.

Слика 3.45 Формат IPv4 адресе мапиране у IPv6 адресу

- IPv4 адреса мапирана у IPv6 адресу[1] (формат представљен на слици 3.45)

IPv4 адресе усаглашене са IPv6 су својевремено биле замишљене као механизам за постепен прелаз мрежа на нову верзију протокола. Овај метод данас се сматра застарелим јер су развијени други методи.

IPv4 адреса мапирана у IPv6 адресу такође има везе са преласком мреже и апликација на нову верзију протокола. Ове адресе омогућавају да мрежне апликације које подржавају обе верзије протокола, и које су покренуте на хостовима који подржавају оба протокола, буду доступне хостовима који подржавају само IPv4 протокол. Овим адресама је омогућено апликацији креирање утичнице са IPv6 адресом која се мапира на IPv4 адресу станице.

Механизам за прелазак на IPv6 протокол

Механизам за прелазак на IPv6 протоклол[2] дефинише начине како да крајње станице и рутери динамички пренесу IPv6 пакете користећи мреже са IPv4 рутирањем. Развијене су специјалне врсте метода који омогућавају међусобни рад као што су:

- уређаји са двоструким скупом протокола[3] - рутери и друге врсте уређаја програмирани су тако да могу да користе и IPv6 и IPv4 адресе и на тај начин могу да комуницирају са станицама које користе једну или другу врсту адресирања;
- IPv4/IPv6 превођење[4] - уређаји са двоструким скупом протокола могу да буду пројектовани да прихвате захтеве од IPv6 станица, претворе их у IPv4 пакете, и пошаљу ка IPv4 одредишту. Такође прихватају пакете од IPv4 станица претварају их у IPv6 пакете и шаљу IPv6 станицама;

[1] *IPv4-mapped IPv6 address*
[2] RFC2893 (*Transition Mechanisms for IPv6 Hosts and Routers*)
[3] *Dual stack devices*
[4] *IPv4/IPv6 translation*

- IPv4 тунел за IPv6 - IPv6 уређаји између којих не постоји путања која се састоји од IPv6 рутера[1] могу међусобно да комуницирају пакујући[2] IPv6 пакет у IPv4 пакет. Суштински станице које користе IPv6 протокол понашају се као било који протокол вишег слоја у односу на IPv4 слој (нпр. TCP и UDP протоколи). Тако упакован IPv6 пакет усмеравају IPv4 рутери.

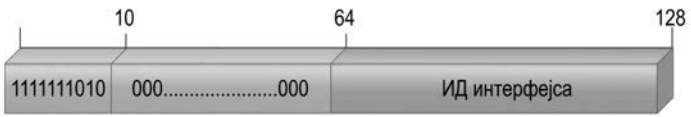
Локалне IPv6 уникаст адресе

На самом почетку дефинисане су две врсте уникаст адреса које се локално користе[3]: за линк[4] и за место[5]. Локална IPv6 уникаст адреса линка има формат као што је приказан на слици 3.44 и односи се само на одређени физички линк (физичку мрежу).

Локална IPv6 уникаст адреса линка пројектована је за адресирање на једном линку и може се користити само у следећим случајевима: аутоматско конфигурисање адреса, откривање суседа[6], или када не постоје рутери. Рутери не смеју да преусмеравају пакете када је адреса локалног линка изворишна или одредишна адреса. Оне су предвиђене за локалну комуникацију на одређеном физичком сегменту мреже.

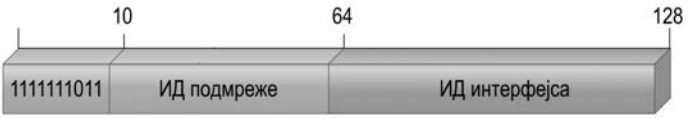
Локалне IPv6 уникаст адресе места[7] имају формат као што је приказан на слици 3.47. Оне су пројектоване да се користе за адресирање унутар једног места без потребе за глобалним префиксом.

Специјална намена овог префикса дефинисана у документу RFC3513 не би требало да буде подржана у новим имплементацијама. Нове



Слика 3.46 Формат локалне IPv6 уникаст адресе линка

[1] *IPv6 capable router*
[2] *Encapsulate*
[3] *Local-use unicast addresses*
[4] *Link-Local*
[5] *Site-Local*
[6] *Neighbor Discovery*
[7] За сада се сматра обезвређеним (*deprecated*) израз који се користи на Интернету за више него застарело (*more than obsolete*).



Слика 3.47 Формат IPv6 локалне адресе места

имплементације треба да третирају овај префикс као глобални уникаст. Постојеће имплементације могу да наставе са коришћењем овог префикса.

Замена за локалне IPv6 уникаст адресе места су глобално јединствене локалне адресе. Јединствена локална IPv6 уникаст адреса[1] је термин који је дефинисан у документу RFC4193 а одговара IPv4 приватним адресама. Подсетимо се да су се приватне IPv4 адресе обично користиле када нису могле да се добију јавне адресе, често у комбинацији са технологијом превођења адреса (NAT[2]). За IPv6 адресирање превођење адреса (NAT) се не захтева већ се користе локалне адресе за комуникацију која је предвиђена искључиво између локалних уређаја.

На пример за функцију откривања суседа коришћењем ND[3] протокола користе се искључиво локалне адресе. Намена локалних уникаст адреса је за локалну комуникацију. Не очекује се да се користе за рутирање на Интернету. Могу се користити за рутирање у оквиру ограничене области (места) или у оквиру ограничене групе места.

3.4.9 Еникаст IPv6 адресе

Еникаст IPv6 адресе додељују се већем броју интерфејса (обично који припадају различитим чворовима) са особеношћу да се пакет послат ка адреси било ког чвора рутира ка „најближем[4]“ интерфејсу који има ту адресу. Када се уникаст адреса додели за више интерфејса и на тај начин постане еникаст адреса, чворови којима је адреса додељена морају експлицитно да буду конфигурисани тако да знају да је то еникаст адреса.

Еникаст IPv6 адресе додељују се из уникаст адресног простора користећи дефинисани уникаст формат. За сваку додељену еникаст адресу постоји префикс P те адресе који идентификује тополошку регију

[1] *Unique Local IPv6 Unicast Addresses*
[2] *Network Address Translation*
[3] *IPV6 Neighbor Discovery*
[4] Појам најближи одређује протокол за рутирање. Детаљније у 4. поглављу овог уџбеника.

у којој су смештени сви интерфејси са том еникаст адресом. У оквиру одређене регије, одређене префиксом P за еникаст адресе, одржавају се посебни записи у табели рутирања (обично се називају „путања крајње станице“). Ван регије одређене идентификатором P еникаст адресе могу се повезати[1] у један запис користећи префикс P.

Једна од очекиваних примена еникаст адреса је да се идентификује скуп рутера који припадају организацији која обезбеђује интернет услугу. Таква адреса може се користити као међуадреса у IPv6 заглављу за рутирање и она би довела до тога да се пакет испоручи преко одређеног интернет посредника или групе интернет посредника. Друга могућа употреба еникаст адресе је да се идентификује скуп рутера који обезбеђују пролаз ка одређеном домену рутирања.

Као и мултикаст адресирање еникаст концепт више упошљава рутере и сложенији је од уникаст адресирања. Практично што су уређаји који имају исту еникаст адресу удаљенији један од другог то га је сложеније применити.

За сада (док се не стекне више искуства) за еникаст адресе постављена су следећа ограничења[2]:

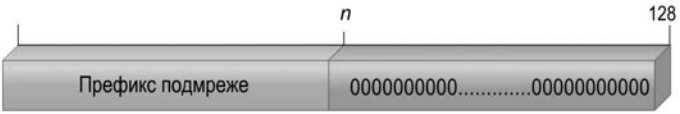
- не може се употребити као изворишна адреса IPv6 пакета,
- не може се доделити IPv6 крајњој станици (хосту),
- може се доделити само рутерима.

Захтеване еникаст адресе[3]

Еникаст адреса рутера подмреже[4] дефинисана је унапред. Формат је представљен на слици 3.48.

Префикс подмреже[5] у еникаст адреси идентификује одређени линк. Иста је синтакса као код уникаст адресе за интерфејс линка у којој је идентификатор интерфејса постављен на нулу. Пакет послат ка еникаст адреси рутера подмреже биће испоручен једном рутеру подмреже. Од свих рутера захтева се да подрже еникаст адресу рутера подмрежа са којима су повезане (имају интерфејсе). Предвиђа се да се адреса

[1] *Aggregated*
[2] RFC3513
[3] *Required Anycast Address*
[4] *Subnet-Router anycast address*
[5] *Subnet prefix*



Слика 3.48 Формат еникаст адреса рутера подмреже

рутера подмрежа користи у свим апликацијама где чворови треба да комуницирају са било којим од скупа рутера.

Мултикаст IPv6 адресе

Мултикаст адресе се користе да би омогућиле једном уређају да пошаље пакете (датаграме) групи пријемника. Код IPv4 адреса класа D је коришћена за слање групи станица. У IPv6 адресирању мултикаст адресама је додељен мултикаст блок. Он представља 1/256. део адресног простора који чине адресе које почињу са вредношћу 11111111_2. Тако свака адреса која започиње са FF_{16} је IPv6 мултикаст адреса.

Преосталих 120 битова адресног простора довољни су да обезбеде изузетно велики број мултикаст адреса (2^{120}). Као што су уникаст адресе расподељене коришћењем посебних формата исти принцип је примењен и код мултикаст адреса.

Мултикаст IPv6 адреса је идентификатор групе интерфејса, обично на различитим чворовима. Формат мултикаст адресе приказан је на слици 3.49.

Бинарна вредност 111111111 на почетку адресе указује да је реч о мултикаст адреси.

Ознаке (флегови) су 4-битни скуп чија је вредност 000T. Вредност T=1 указује на привремено[1] додељену мултикаст адресу, а вредност T=0 на сталну мултикаст адресу (добро познату[2]).

Опсег је 4-битна вредност која ограничава област мултикаст групе.

Идентификатор групе одређује мултикаст групу било сталну било привремену у оквиру задатог опсега.

Мултикаст адресе се не смеју користити као изворишне адресе IPv6 пакета и не могу се појавити у било ком заглављу рутирања. Рутери не

[1] *Transient*
[2] *Well known*

Слика 3.49 Формат IPv6 мултикаст адресе

смеју да усмеравају мултикаст адресе ван опсега на који указује поље опсега у одредишној мултикаст адреси.

Мултикаст опсези

Обележавање опсега мултикаст адреса веома је значајно. Опсег глобалних мултикаст адреса мора бити јединствен на целом Интернету, али опсег локалних адреса мора бити јединствен само у оквиру организације. Овај концепт пружа велику флексибилност пошто се сваки тип мултикаст адреса појављује у ствари у неколико верзија: у једној за чвор, другој за локални линк (локалну мрежу), трећој за локално место итд. Опсег такође омогућава рутерима да одмах одреде колико далеко могу да усмеравају мултикаст пакете да би побољшали ефикасност и отклонили проблеме при слању ван планиране области (слика 3.50).

На слици 3.50 приказано је како опсег омогућава IPv6 мултикаст адресама да су ограничене на одређене сфере утицаја. Најмањи опсег је опсег локалног чвора са вредношћу 1 за идентификатор опсега. Што се вредност идентификатора опсега увећава, опсег се шири и обухвата локалну мрежу, место, организацију и на крају целокупан Интернет.

Унапред дефинисане мултикаст адресе

Ознака T дозвољава да се експлицитно одреди које од мултикаст адреса су доступне за нормалну употребу у поређењу са оним адресама које су унапред дефинисане.

Добро познате, унапред дефинисане мултикаст адресе су:

1. Резервисане мултикаст адресе:

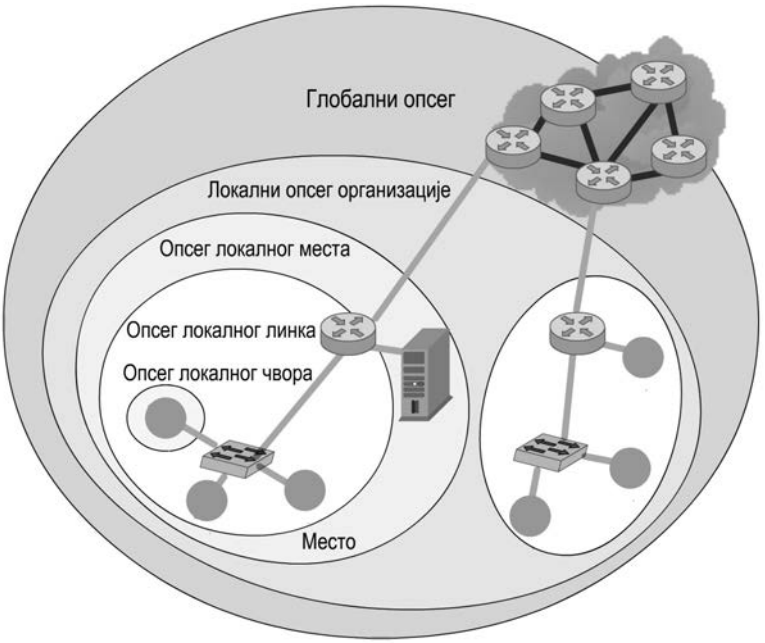
FF00:0:0:0:0:0:0:0

FF01:0:0:0:0:0:0:0

.....

FF0F:0:0:0:0:0:0:0

не смеју никад бити додељене некој мултикаст групи.



Слика 3.50 Опсег IPv6 мултикаст адреса

2. Адресе свих чворова:

   FF01:0:0:0:0:0:0:1

   FF02:0:0:0:0:0:0:1

   идентификују групу IPv6 чворова са опсегом 1 (локални интерфејс[1]) или 2 (локални линк[2]).

3. Адресе свих рутера:

   FF01:0:0:0:0:0:0:2

   FF02:0:0:0:0:0:0:2

   FF05:0:0:0:0:0:0:2

   идентификују групу свих IPv6 рутера са опсегом 1 (локални интерфејс) или 2 (локални линк) или 5 (локално местс[3]).

[1] *Interface- local*
[2] *Link- local*
[3] *Site-local*

4. Тражена адреса чвора[1]:

FF02:0:0:0:0:1:FFXX:XXXX

Израчунава се као функција уникаст и еникаст адресе чвора. Тражена мултикаст адреса чвора формира се тако што се нижа 24-бита адресе (уникаст или еникаст) додају на префикс FF02:0:0:0:0:1:FF00::/104 чиме се добија мултикаст адреса у опсегу FF02:0:0:0:0:1:FF00:0000 до FF02:0:

Ова адреса се користи у оквиру механизма откривања MAC адреса суседа, што је у IPv6 мрежама замена за ARP протокол.

Види се да се мултикаст адреса користи за истовремено слање података већем броју уређаја у мрежи. Могу се специфицирати за различите опсеге и на тај начин циљати широк или узак круг пријемних уређаја

3.4.10 Захтевана адреса чвор

Од станице (хоста) захтева се да препозна следеће адресе којима себе идентификује:

- адресе локалног линка за сваки од интерфејса,
- било какве додатне уникаст или еникаст адресе које су конфигурисане за интефејс чвора (ручно или аутоматски),
- адреса затворене петље,
- мултикаст адреса свих чворова,
- тражена мултикаст адреса за сваку од његових уникаст и еникаст адреса и
- мултикаст адреса за све групе којима чвор припада.

Од рутера се захтева да препознаје све адресе које крајња станица треба да препозна и још додатне адресе којима себе идентификује:

- еникаст адреса рутера подмреже за све интерфејсе за које је конфигурисана као рутер,
- све друге еникаст адресе са којима је рутер конфигурисан и
- све мултикаст адресе рутера.

Иницијална подела (додела) IPv6 адресног простора дата је у табели 3.7.

[1] *Solicited-Node Address*

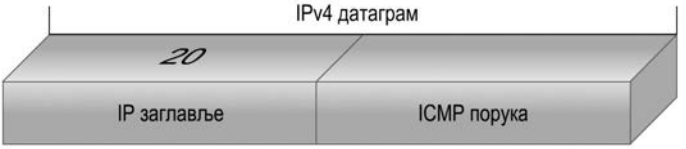
Додела	Префикс (бинарни)	Део адресног простора[1]
Недодељене адресе[2]	0000 0000	1/256
Недодељене адресе	0000 0001	1/256
Резервисане за NSAP	0000 001	1/128 [RFC1888]
Недодељене адресе	0000 01	1/64
Недодељене адресе	0000	1 1/32
Недодељене адресе	0001	1/16
Глобалне уникаст[3]	001	1/8 [RFC2374]
Недодељене адресе	010	1/8
Недодељене адресе	011	1/8
Недодељене адресе	100	1/8
Недодељене адресе	101	1/8
Недодељене адресе	110	1/8
Недодељене адресе	1110	1/16
Недодељене адресе	1111 0	1/32
Недодељене адресе	1111 10	1/64
Недодељене адресе	1111 110	1/128
Недодељене адресе	1111 1110 0	1/512
Уникаст адресе локалног линка	1111 1110 10	1/1024
Уникаст адресе локалног места	1111 1110 11	1/1024
Мултикаст адресе	1111 1111	1/256

Табела 3.7 Подела (додела) IPv6 адресног простора

3.5 Управљачке поруке на Интернету [123]

Протокол који обезбеђује управљачке поруке на Интернету је ICMP[4] протокол. Најчешће се сматра делом мрежног слоја Интернета. Протокол ICMP користи рутер или одредишни рачунар када треба да обавести

[1] *Fraction of Address Space*
[2] *Unassigned*
[3] *Global Unicast*
[4] *Internet Control Message Protocol*



Слика 3.51 ICMPv4 порука је укалупљена у IPV4 датаграм

изворишни рачунар о грешкама у обради IP датаграма. Постоји верзија ICMP протокола за IPv4 протокол (ICMPv4)[1] и верзија за IPv6 протокол (ICMPv6)[2].

3.5.1 Протокол ICMP верзије 4

ICMPv4 поруке се шаљу у оквиру IP датаграма као што је приказано на слици 3.51. Када IPv4 пакет садржи ICMPv4 поруку:

- у заглављу IPv4 пакета поље „протокол“ постављено је на вредност 1,
- у заглављу IPv4 пакета поље „врста услуге“ постављено је на вредност 0,
- поље података IPv4 пакета је ICMPv4 порука.

На слици 3.52 приказан је формат ICMPv4 поруке. Прва четири бајта су истог формата за све поруке али остатак се разликује од поруке до поруке.

Поље „тип“ има петнаест различитих вредности које одређују врсту ICMPv4 поруке. Неке ICMPv4 поруке користе поље кôд[3] за додатне спецификације. Поље контролног збира[4] покрива ICMPv4 поруку.

Слика 3.52 Формат ICMPv4 поруке

[1] Описан у документу RFC792.
[2] Описан у документу RFC4443
[3] *Code*
[4] *Checksum*

Алгоритам који се користи је исти као алгоритам који се користи за контролни збир IP заглавља[1].

3.5.2 Врсте ICMPv4 порука

Табела 3.8 садржи различите врсте ICMPv4 порука. Међусобно се разликују по вредностима у пољима „тип“ и „кôд“. ICMPv4 протокол може се окарактерисати на следећи начин:[23]

Тип	Кôд	Опис	Упит	Грешка
0	0	Ехо одговор (*Ping* одговор)	√	
3		Одредиште недоступно		
	0	Мрежа је недоступна		√
	1	Рачунар је недоступан		√
	2	Протокол је недоступан		√
	3	Порт је недоступан		√
	4	Потребна је фрагментација али је DF[2] =1		√
	5	Грешка у изворишној рути		√
	6	Непозната одредишна мрежа		√
	7	Непознат одредишни рачунар		√
	8	Изворишни рачунар је изолован		√
	9	Приступ одредишној мрежи је административно забрањен		√
	10	Приступ одредишном рачунару је административно забрањен		√
	11	Мрежа је недоступна за врсту услуге (TOS)		√
	12	Рачунар је недоступан за врсту услуге TOS		√
	13	Комуникација је административно забрањена филтрирањем		√
	14	Првенство рачунара (хоста) је нарушено		√
	15	Прекид првенства је у току		√
4		Угасити извориште[3] – контрола тока		√
5		Преусмеравање		

[1] Прво се постави вредност поља *Checksum* на 0. Затим се израчуна први комплемент заглавља. Добијена вредност се упише у поље *Checksum*.

[2] *Don't Fragment*

[3] *Source quench*

Тип	Кôд	Опис	Упит	Грешка
	0	Преусмеравње за мрежу		√
	1	Преусмеравање за рачунар (хост)		√
	2	Преусмеравање за тип сервиса и мрежу		√
	3	Преусмеравање за тип сервиса и хост		√
8	0	Ехо захтев	√	
9	0	Оглашавање рутера	√	
10	0	Тражење рутера[1]	√	
11		Време је истекло		
	0	Време трајања пакета једнако је 0 за време преноса[2]		√
	1	Време трајања пакета једнако је 0 за време поновног спајања		√
12		Проблем са параметрима		
	0	Заглавље IP пакета није у реду		√
	1	Недостају захтеване опције		√
13	0	Захтев о тачном времену[3]	√	
14	0	Одговор о тачном времену[4]	√	
15	0	Информације – захтев (застарело)[5]	√	
16	0	Информације – одговор (застарело)	√	
17	0	Захтев за маском адресе[6]	√	
18	0	Одговор о маски адресе	√	
30		*Traceroute*	√	
31		Грешка у конверзији датаграма	√	
32		Преусмеравање мобилног хоста	√	
33		Тражење IPv6	√	
34		Одговор IPv6[7]	√	

1234567

[1] *Router solicitation*
[2] *Traceroute*
[3] *Timestamp request*
[4] *Timestamp reply*
[5] *Information request (obsolete)*
[6] *IPv6 Where - Are - You*
[7] *IPv6 I - Am - Here*

Тип	Кôд	Опис	Упит	Грешка
35		Захтев за регистрацијом мобилног корисника	√	
36		Одговор мобилног корисника	√	
37		Захтев за именом домена	√	
38		Одговор имена домена	√	

Табела 3.8 Врсте ICMv4 порука

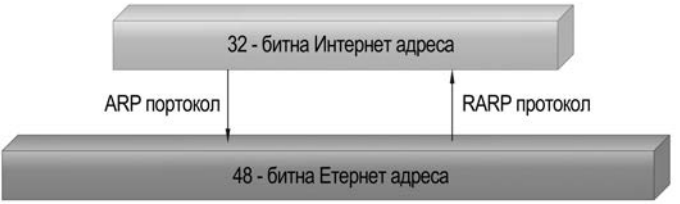
- ICMP се користи да извести о грешкама а не да обезбеди поуздан IP протокол. Поузданост треба да обезбеде протоколи виших слојева;
- ICMPv4 порука о грешци не може се користити да би се послале било какве информације о ICMPv4 протоколу. На тај начин је спречен проблем неограниченог понављања. ICMPv4 одговор може се послати само као одговор на ICMPv4 упит (ICMPv4 тип 0, 8, 9, 10 и 13 до 18);
- ICMPv4 извештај о грешци никада се не шаље за датаграм који је упућен свима или групи станица;
- за фрагментиране датаграме ICMPv4 извештај о грешци може се послати само за први фрагмент;
- ICMPv4 извештај о грешци никада се не шаље за датаграм чија изворишна адреса не дефинише један рачунар. То значи да изворишна адреса не може да буде нулта адреса, адреса затворене петље, адреса упућена свим станицама или групи станица.

Ова правила су уведена да би се спречила олуја[1] као последица слања свим станицама. Раније су се ICMPv4 поруке о грешци слале као одговор на пакете упућене свима.

3.6 Протокол за разрешавање адреса

Проблем који се јавља је следећи: IP адресе су препознатљиве само скупу TCP/IP протокола[2]. Протоколи на другом слоју као што су IEEE 802.3 (Етернет) и IEEE802.11 имају свој систем адресирања. Мрежу

[1] *Broadcast storm*
[2] *TCP/IP protocol suit*



Слика 3.53 Разрешавање адреса протокола ARP и RARP

Етернет типа могу да користе истовремено различити мрежни слојеви. На пример група рачунара са TCP/IP протоколима и група рачунара са неким другим мрежним софтвером могу да буду на истој локалној мрежи.

Када се Етернет рам шаље од једног до другог рачунара на истој локалној мрежи онда адреса од 48 бита одређује ком интерфејсу је рам упућен. Софтвер[1] никада не анализира одредишну IP адресу у IP датаграму.

Разрешавање адреса је процес повезивања[2] између различитих врста адреса: IPv4 адресе од 32 бита и адресе коју користи други слој. RFC826 даје спецификацију протокола за разрешавање адреса (ARP[3] протокол). На слици 3.53 скицирана су два протокола: протокол за разрешавање адреса (ARP протокол) и инверзни протокол за разрешавање адреса (RARP[4] протокол).

ARP протокол обезбеђује динамичко повезивање IPv4 адресе у одговарајућу хардверску (Етернет, MAC) адресу. Користи се термин динамички пошто се одвија аутоматски и није у надлежности ни апликације ни администратора система. RARP протокол најчешће користе системи који немају диск.

3.7 Питања и задаци

1. Наведите неке од области примене рачунара којима највише одговара услуга са успоставом везе и друге за које је најбоља услуга без успоставе директне везе.

[1] *Device driver*
[2] *Mapping*
[3] *Address Resolution Protocol*
[4] *Reverse Adress resolution Protocol*

2. Набројати класе IPv4 адреса.
3. Описати каква је намена TTL времена.
4. Описати поступак поновног спајања делова IPv4 пакета на одредишту.
5. Хексадецимално задату IP адресу C22F1582 напишите користећи децималну нотацију с тачком.
6. Мрежа на Интернету има маску подмреже 255.255.240.0. Колико највише рачунара може она да подржи?
7. Које адресе од понуђених су приватне IP адресе?
   а) 12.0.0.1
   б) 168.172.19.39
   в) 172.20.14.36
   г) 172.33.194.30
   д) 192.168.42.34
8. Почев од адресе 198.16.0.0, на располагању је велики број узастопних адреса. Претпоставимо да четири организације: O1, O2, O3 и O4 захтевају, редом, 4000, 2000, 4000 адреса. Одредити прву и последњу адресу која се свакој од њих додељује, као и маску у нотацији *w.x.y.z/s.*
9. Која од понуђених поља су део IPv4 заглавља?
   а) изворишна адреса
   б) број одредишног порта
   в) TTL поље
   г) IHL поље
   д) DF поље
   ђ) контрола тока
10. Одредите којим класама IP адреса (A, B, C, D, E) одговарају следеће адресе:
    а) 64.36.29.87
    б) 242.242.42.24
    в) 224.0.0.9
    г) 190.168.23.57
    д) 192.168.26.57
11. Које станице са слике 3.21 користе индиректно рутирање за међусобну комуникацију?
12. Описати који проблем и на који начин подмрежа са променљивим делом за подмрежу (VLSM) разрешава.

13. Адреса станице је 160.10.10.145/28. Којој подмрежи припада ова станица?
14. Адреса станице је 89.8.30.188/27. Којој подмрежи припада ова станица?
15. Када се додељују адресе за тачка-тачка линкове која маска подмреже је оптимална?
16. Колика је величина адресног поља IPv6 протокола?
17. Набројати и описати додатна заглавља IPv6 пакета.
18. Како је дефинисан и подржан ток у IPv6 стандарду?
19. Описати идентификаторе врсте IPv6 адреса.
20. Описати намену и врсте уникаст адреса код IPv6 протокола.
21. Каква је функција идентификатора интерфејса код IPv6 протокола?
22. Описати механизме за прелазак са IPv4 на IPv6 протокол.
23. Које све врсте адресирања подржава IPv6 протокол?
24. Који протокол реализује управљачке поруке на Интернету? Описати га.
25. Описати протоколе за разрешавање адресе.

4. Алгоритми за рутирање

Алгоритам за рутирање је софтверски део мрежног слоја одговоран за одлуку на коју од излазних линија рутера ће пристигли пакет бити послат. Када подмрежа користи датаграм (без успоставе везе[1]) одлука о преусмеравању доноси се за сваки пристигли пакет пошто се оптимална путања у међувремену (можда) променила.

Када подмрежа користи виртуелно коло (са успоставом везе[2]) одлуке о рутирању доносе се при успостављању везе и пакети се шаљу том утврђеном путањом.

Корисно је направити разлику између преусмеравања (рутирања) и прослеђивања. Рутер има два процеса у себи:

- процес који анализира доспели пакет и проналази у табели рутирања излазну комуникациону линију (везу, линк), на коју ће пакет бити послат. Овај процес назива се прослеђивање[3];
- процес који је одговоран за попуњавање и иновирање табела рутирања. Овај процес назива се преусмеравање (рутирање). Кључну улогу у овом процесу заузима алгоритам за рутирање.

Пре покушаја изналажења решења задатак алгоритма за рутирање јесте да донесе одлуку шта је то што треба оптимизовати. На пример: смањење средњег кашњења пакета добар је избор, али је такође добар избор и повећање укупног саобраћаја (протока) кроз мрежу. Ова два циља су у међусобној супротности пошто сваки систем са великим саобраћајем повлачи и дуже време чекања у редовима за обраду пакета. Као компромис покушало се са смањењем броја рутера[4], јер се тако смањује кашњење али и искоришћеност пропусног опсега што даље погоршава проток.

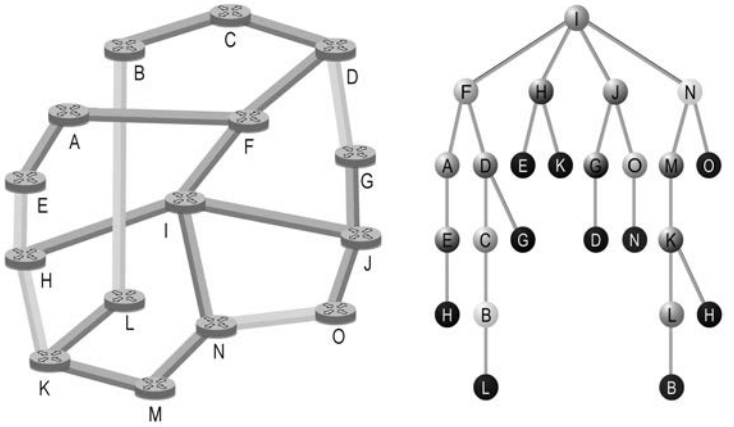
---

[1] *Connectionless*

[2] *Connection oriented*

[3] *Forwarding*

[4] Уобичајен термин је да се пролазак пакета кроз рутер назива скок (*hop*).



Слика 4.1 а) Подмрежа б) Понируће стабло за рутер I

Постоје два начина формирања табеле за рутирање, па у складу с тим и алгоритми за рутирање могу се поделити у две главне групе: неприлагодљиви (статичко рутирање) и прилагодљиви (динамичко рутирање).

Пре анализе конкретних алгоритама потребно је дефинисати принцип изналажења оптималне путање који не зависи од топологије и саобраћаја мреже. У литератури се овај принцип среће под називом принцип оптималности и може се описати на следећи начин: „Ако се рутер J налази на оптималној путањи од рутера I до рутера K, онда је оптимална путања од J до K део тог пута[1]“.

Директна последица принципа оптималности јесте да скуп оптималних путања из свих извора ка задатом одредишту сачињава стабло с кореном у одредишту. Такво стабло зове се понирућe стабло[2]. На слици 4.1 приказана је подмрежа (слика 4.1а) и понирућe стабло за одредиште I (слика 4.1б), где је као мера растојања узет број скокова. Циљ свих алгоритама за рутирање је да открију и користе понирућа стабла за све рутере.

Понирућe стабло не садржи петље, па се сваки пакет преноси са коначним бројем скокова. У пракси, то није тако једноставно. Везе

[1] Доказ се може наћи у литератури [13].

[2] *Sink tree*

и рутери могу да откажу и наставе касније са радом тако да различити рутери могу да имају различите представе о тренутној топологији мреже. У сваком случају принцип оптималности и понирање стабло представљају стандардну подлогу за процену других алгоритама за рутирање.

4.1 Рутирање најкраћим путем

Алгоритам који се назива рутирање најкраћим путем[1] широко је распрострањен. Идеја се састоји у томе да се направи граф подмреже; сваки чвор у графу представља рутер, а свака линија у графу комуникациону линију[2]. За одабир путање између два задата рутера, алгоритам проналази најкраћи пут (путању) на графу.

Анализираћемо концепт најкраће путање. Један начин за мерење дужине пута је број скокова. Користећи овај начин мерења раздаљине[3] (удаљености), путеви ABC и ABE (слика 4.2) су једнаке дужине. Други начин мерења раздаљине може бити географска удаљеност. У том случају ABC је очигледно знатно краћа од ABE.

Поред поменутих користе се и други начини мерења удаљености: нпр. свака веза може се обележити средњим временом чекања или средњим кашњењем у преносу, а вредности би се проверавале сваког сата[4]. У овом случају најкраћи пут је онај који је најбржи а не онај који је географски најкраћи.

У општем случају, ознаке на линијама графа могу бити израчунате као функција следећих критеријума: удаљености, пропусног опсега, просечног саобраћаја, трошкова комуникације, средњег времена чекања у реду, средњег кашњења... Алгоритам израчунава најкраћи пут на основу једног или на основу комбинације више критеријума.

Познато је неколико алгоритама за рачунање најкраће путање између два чвора. У следећим одељцима биће анализирана два најчешће коришћена статичка алгоритма: алгоритам Дикстра[5] и алгоритам плављења[6].

---

[1] *Shortest Path Routing*

[2] Линк, веза или линија

[3] Користи се термин метрика.

[4] Код оваквих мерења користи се унапред дефинисана величина пакета (тест пакет).

[5] Детаљније описан у раду [3].

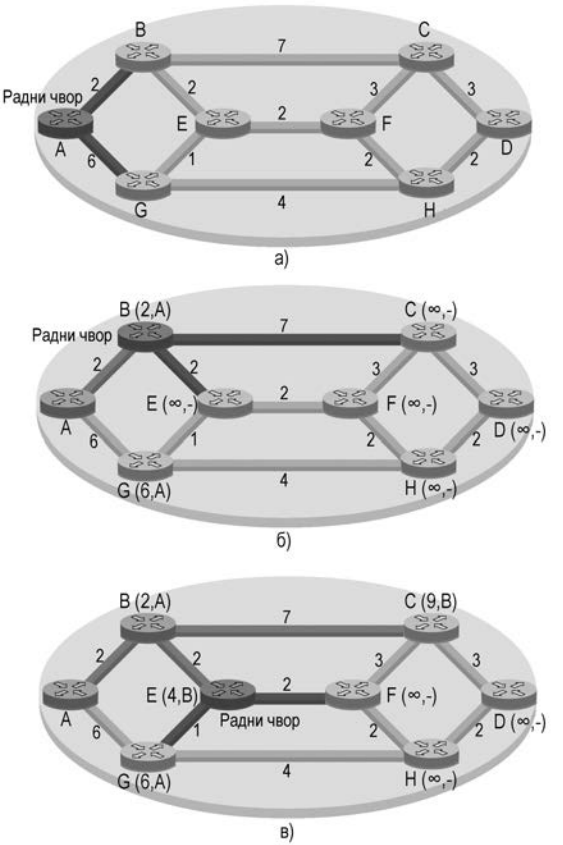
[6] *Flooding*

4.1.1 Алгоритам Дикстра

Сваки чвор обележен је (на сликама 4.2 и 4.3 ознаке у заградама):

- удаљеношћу од изворног чвора дуж најбоље познате путање,
- ознаком чвора из кога се врши испитивање.

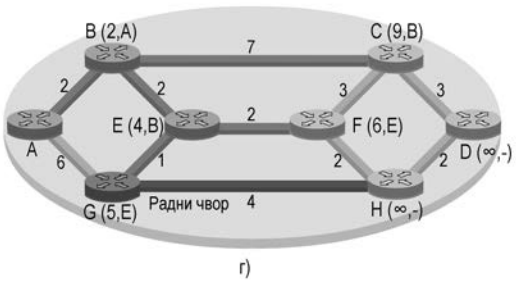
У почетку, путање су непознате, па су сви чворови обележени ознаком бесконачно „∞“. У току примене алгоритма и изналажења путање ознаке чворова се мењају указујући на боље путање. Ознаке чворова могу бити променљиве и сталне. У самом почетку све ознаке чворова су променљиве. Кад се докаже да одређена ознака чвора представља



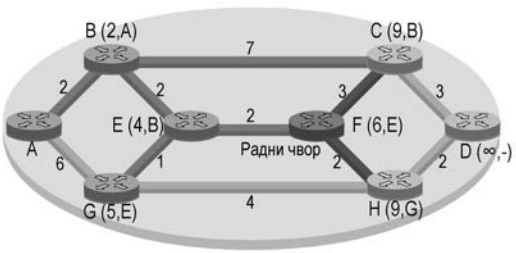
Слика 4.2 Прва 3 корака Дикстра алгоритма у израчунавању најкраћег пута од чвора A до чвора D

најкраћу путању од извора до одговарајућег чвора, она постаје стална ознака и више се никада не мења.

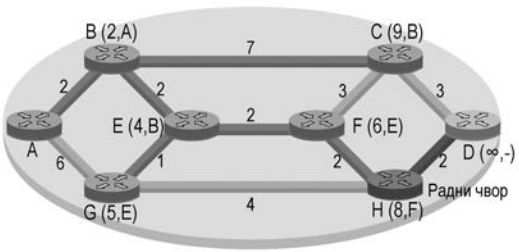
Као илустрација начина рада овог алгоритма послужиће нам обележени неусмерени граф представљен на сликама 4.2 и 4.3. Ознаке на линијама графа представљају удаљености. Тражи се најбољи пут од A до D. Започећемо анализу проглашавајући чвор A сталним (слика 4.2а). Затим се испитују сви суседни чворови чвору A, један за другим, и означаваја се удаљеношћу до чвора A (радног чвора). Кад год је чвор „радни“ обележава се чвором из кога је испитиван, тако да се касније може



г)



д)



ђ)

Слика 4.3 Друга 3 корака Дикстра алгоритма у израчунавању најкраћег пута од чвора A до чвора D

реконструисати коначна путања. Испитивањем свих суседних чворова чвору A, проверене су променљиве ознаке чворова целог графа. Чвор са најмањом вредношћу добија сталну ознаку и постаје нови радни чвор. На слици 4.2б то је чвор B.

У следећем кораку испитују се су седни чворови чвора B. Ако је збир вредности ознака чвора B и удаљености од чвора B до чвора који се посматра мањи од ознаке тог чвора, значи да постоји краћи пут, па се посматрани чвор означава новом вредношћу. Пошто се сви суседни чворови радног чвора испитају и променљиве ознаке промене вредност (уколико је то могуће) цео граф се анализира у потрази за чвором који има променљиву ознаку најмање вредности. Изабрани чвор добија сталну ознаку и постаје радни чвор за следећи корак реализације алгоритма. На сликама 4.2 и 4.3 представљени су првих шест корака алгоритма.

Доказаћемо да алгоритам исправно ради. На слици 4.2б чвор E је постао стални чвор. Ако претпоставимо да постоји краћи пут од ABE, рецимо AXYZE, постоје две могућности: или је чвор Z већ постао сталан, или није. Ако јесте, онда је чвор E већ испитан (у кораку који је следио кораку у ком је Z постао сталан), тако да путања AXYZE не може бити краћа путања.

Сада размотримо случај кад чвор Z још увек има променљиву ознаку. Ако је ознака чвора Z већа или једнака ознаци чвора E онда путања AXYZE није краћа од путање ABE. Ако је ознака чвора Z мања од ознаке чвора E то значи да је чвор Z а не чвор E први постао радни чвор дозвољавајући да се чвор E проверава из чвора Z.

4.1.2 Алгоритам плављења

Принцип алгоритма плављења је следећи: сваки пакет који рутер прима (долазећи пакет) прослеђује на све излазне линије рутера осим на оне на којој је примљен. Да би се заштитили од великог умножавања пакета (које у крајњем случају може бити неограничено) у заглавље сваког пакета постављају се бројачи скокова. Бројач умањује своју вредност за један (декрементира се) при сваком проласку кроз рутер (тј. при сваком скоку). Пакет се одбацује кад вредност бројача постане нула. Најбоље би било да се вредност бројача постави на вредност једнаку дужини пута од извориштa до одредиштa. Уколико пошиљалац не зна дужину пута,

вредност бројача може се поставити на највећу вредност једнаку величини мреже (или подмреже).

Други начин за избегавање „плављења“ пакета је вођење евиденције о пакетима који су плављењем послати да би се спречило њихово поновно слање. Она се може реализовати тако да изворишни рутер додаје редни број сваком пакету који прими од изворишне станице. Сваки од рутера (дуж путање) треба да има листу редних бројева пакета које је примио са одређеног изворишног рутера и које је већ плављењем проследио. Уколико је долазећи пакет на тој листи он неће плављењем поново бити послат.

У циљу спречавања неограниченог раста листе, додаје се бројач *k*. Вредност овог бројача означава да су сви пакети са редним бројевима мањим или једнаким овој вредности већ послати. Такви пакети се одбацују.

Мало практичнија варијанта алгоритма плављења је алгоритам селективног плављења. Када користе овај алгоритам рутери не прослеђују пакете на све линије већ само на оне које на први поглед иду у правом смеру.

Алгоритам плављења је за највећи број апликација неприменљив. Користи се у неким случајевима као што су:

- војне апликације где постоји стална опасност да велики број рутера престане с радом,
- дистрибуиране базе података код којих је потребно истовремено ажурирање свих база података,
- бежичне мреже код којих су све поруке које шаље једна станица на располагању другим станицама унутар домета радио-сигнала,
- употреба овог алгоритма као референце за процену других алгоритама.

Алгоритам плављења бира увек најкраћу путању пошто испитује сваку од могућих путања.

4.2 Рутирање на основу вектора удаљености

У савременим рачунарским мрежама даје се предност динамичким над статичким[1] алгоритмима за рутирање пошто статички алгоритми не

[1] *Dijkstra, Flooding*

узимају у обзир тренутну оптерећеност мреже. Два динамичка алгоритма која се највише користе и која ћемо анализирати у следећим одељцима су:

- рутирање на основу вектора удаљености[1] и
- рутирање на основу стања везе[2].

Код алгоритама за рутирање на основу вектора удаљености[3] сваки рутер одржава табелу (тј. вектор) чије записе чине удаљеност (метрика) најбоље познате путање ка одредишту и веза (линија) која се користи за ту путању. Табеле се ажурирају разменом информација са суседима. Метрика може бити број скокова, кашњење у милисекундама, укупни број пакета који чекају дуж путање итд.

Претпоставља се да рутер зна удаљеност до свих својих суседа. Ако је:

- удаљеност (метрика) скок, удаљеност је један скок,
- удаљеност (метрика) дужина реда[4], рутер прегледа сваки ред,
- удаљеност (метрика) кашњење, тада рутер шаље специјалну врсту пакета[5] у које се на пријему уписује временска ознака[6] и враћа што је могуће пре натраг.

На пример: претпостављамо да је за метрику узето кашњење а да рутер зна кашњење до својих суседа. Једном сваких *Tms* сваки рутер шаље суседима листу са својом проценом кашњења до сваког одредишта. Такође он прима листе (табеле) од својих суседа са њиховим проценама. Посматрамо ситуацију у којој је једна од табела управо стигла од суседа *x* са записима x_i који представљају време потребно да се од рутера *x* стигне до рутера *i*. Ако рутер зна да кашњење до његовог суседа *x* износи *y* милисекунди, он тада зна и да до рутера *i* може да стигне преко рутера *x* за $x_i + y$ милисекунди. Уколико рутер израчуна кашњење до сваког суседа, на основу тога може да одреди која је путања најбоља и упише је у нову табелу рутирања Овај процес ажурирања илустрован је на слици 4.4. Са

[1] *Distance vector routing*

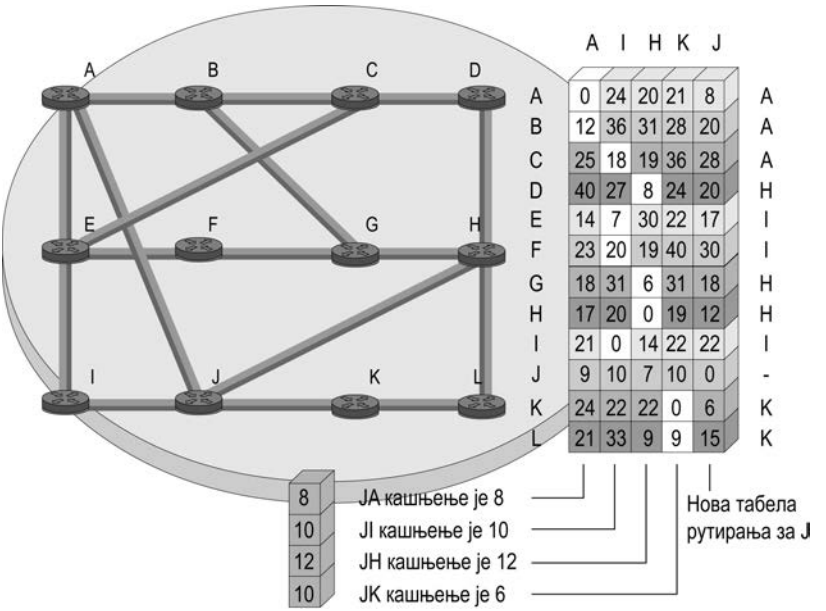
[2] *Link state routing*

[3] Некада се ови алгоритми називају именима истраживача који су их развили: *Bellman-Ford* или F*ord-Fulkerson* алгоритми. Овај алгоритам коришћен је у ARPANET мрежи и на Интернету под називом RIP протокол (*Routing Information Protocol*).

[4] Под редом (*queue*) подразумева се ред у коме пакети чекају да их рутер прихвати и обради.

[5] ECHO

[6] *Timestamp*



Слика 4.4 Подмрежа, вектори из рутера A, I, H, K и нова табела рутирања за рутер J

леве стране слике приказана је подмрежа. Прве четири колоне у десном делу ове слике приказују вредности вектора кашњења које је рутер J примио од својих суседа рутера A, I, H и K. Посматрајући вектор кашњења за рутер A види се да је:

- 12 ms кашњење до рутера B,
- 25 ms кашњење до рутера C,
- 40 ms до рутера D, итд.

Претпоставимо да је рутер J измерио и проценио да је кашњење до својих суседа:

- 8 ms кашњење до рутера A,
- 10 ms кашњење до рутера I,
- 12 ms кашњење до рутера H и
- 6 ms кашњење до рутера K.

Како рутер J рачуна своју нову путању до рутера G? Зна да стиже до рутера A за 8 ms, а рутер A тврди да може да стигне до рутера G

за 18 ms. То значи да рутер J зна да може да рачуна на кашњење од 26 ms, ако пошаље пакете ка рутеру G преко рутера A. Слично, рутер J израчунава кашњење до рутера G преко рутера I, H и K и добија вредности 41ms следећим редоследом: (31ms +10ms), 18ms (6ms+12ms) и 37ms (31ms+6ms). Најмања израчуната вредност је 18ms па се у табелу рутирања за рутер J уноси кашњење од 18ms и путања преко рутера H коју треба користити.

На исти начин врши се прорачун за сва остала одредишта. Нова табела рутирања за рутер J приказана је у последњој колони на слици 4.4.

4.2.1 Проблем бројања до бесконачности

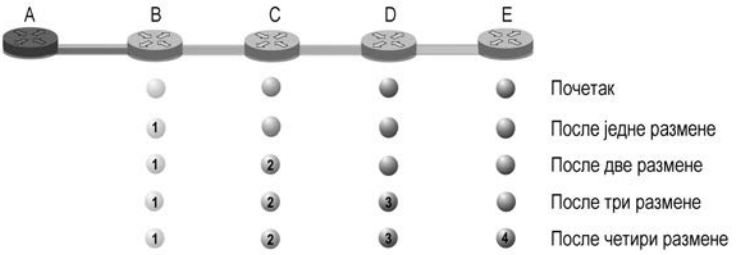
Алгоритам рутирања помоћу вектора удаљености теоријски детаљно је разрађен али се у самој реализацији јављају многи проблеми од којих ћемо поменути следеће:

- без обзира што се постижу добри резултати, често је тај процес спор (тзв. спора конвергенција),
- алгоритам брзо реагује на добре, а споро на лоше вести.

Посматрајмо рутер чија је најбоља путања до одредишта *x* веома дуга. Ако у следећој размени информација о рутирању сусед рутера A извести о бољој метрици до рутера *x*, рутер ће одмах почети да користи комуникациону линију до рутера A и да преко њега шаље пакете ка рутеру *x*. Каже се да се добре вести брзо простиру (обрађују).

Да бисмо видели како се брзо шире добре вести, размотримо линеарну подмрежу од 5 чворова на слици 4.5, где се за метрику користи број скокова. Претпоставимо да је рутер A у квару од самог почетка и да сви остали рутери у мрежи то знају. Другим речима, свима је метрика до рутера A бесконачна (∞).

Кад се рутер A поново укључи у мрежу, други рутери ће то сазнати разменом вектора. Због једноставности претпоставићемо да се размена вектора врши синхронизовано у свим рутерима. Код прве размене вектора рутер B сазнаје да је рутер A сада активан. Рутер B уписује у своју табелу рутирања да је рутер A на растојању од једног скока посматрано улево. Сви остали рутери и даље мисле да је рутер A у прекиду. У следећој размени вектора рутер C сазнаје даје је рутер B на растојању од једног скока од рутера A. Иновира своју табелу растојањем до рутера A која има

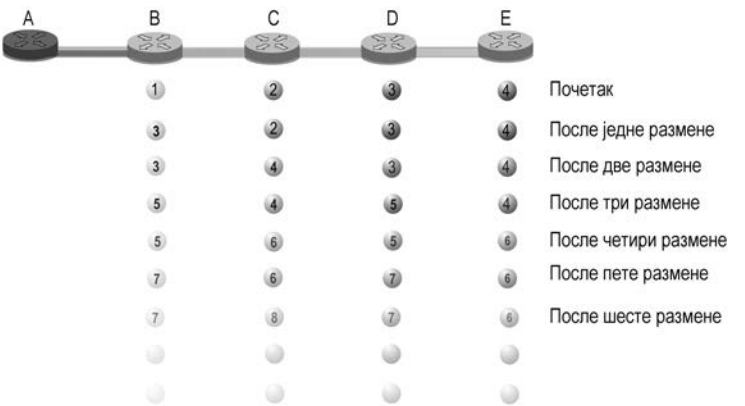


Слика 4.5 Ширење добрих вести кроз подмрежу

вредност 2. У овој фази до рутера D и E још увек нису допрле добре вести. Као што се види добра вест се прошири за један скок у току једне размене табела рутирања. У подмрежи чија је најдужа путања састављена од N скокова, кроз N размена сви рутери ће сазнати о поново успостављеним комуникационим линијама и активним рутерима.

Сад размотримо ситуацију на слици 4.6, где су све линије и рутери иницијално исправни. Рутери B, C, D и E су удаљености од рутера A за 1, 2, 3 и 4 скока. Изненада рутер A престаје с радом или долази до прекида линије између рутера A и B (што је за рутер B исто).

У првој размени пакета рутер B не успоставља комуникацију са рутером A. Рутер C решава проблем: појављује се са информацијом да има путању до рутера A с метриком 2. Рутер B не зна да рутер C у ствари комуницира са рутером A преко њега. Као резултат, рутер B сазнаје да



Слика 4.6 Ширење лоших вести кроз подмрежу

може да стигне до рутера A преко рутера C, са метриком 3. У првој размени вектора рутери D и E не ажурирају своје записе о рутеру A.

У другој размени, рутер C примећује да сваки од његових суседа тврди да има путању до рутера A са метриком 3. Бира једну од њих, случајним избором, и ажурира своју удаљеност на вредност 4.

Са слике 4.6 јасно се види зашто лоше вести „споро путују". Највећа вредност записа о удаљености до неког рутера (нпр. рутера A) само је за један већа од минималне вредности записа о удаљености до тог рутера коју имају његови суседи. Постепено, вредности записа у табелама рутирања о удаљености до рутера A се неограничено повећавају. Овај проблем познат је као „бројање до бесконачности[1]".

Да би се проблем решио треба подесити „бесконачност" на вредност најдуже путање увећане за један. Суштина је у томе да када рутер x саопшти рутеру y да има путању ка рутеру *z* не постоји начин да рутер y утврди да ли се и он сам налази на тој путањи. Анализирано је више начина да се овај проблем разреши (нпр. подела хоризонта[2]).

4.3 Рутирање на основу стања комуникационе везе

На самом почетку се у ARPANET мрежи користило рутирање помоћу вектора удаљености а од 1979. год. рутирање на основу стања комуникационе везе. Два најважнија недостака рутирања на основу вектора удаљености (растојања) су:

- алгоритам не узима у обзир пропусни опсег комуникационе везе (линка) приликом избора путања. У почетку су везе биле брзине 56kb/s, па је нормално било водити рачуна само о броју линија које сачињавају путању до одредишта. Кад су почеле да се користе линије већих брзина од 1,5 Mb/s; 2Mb/s до 10Gb/s[3] не разматрати пропусни опсег постало је неприхватљиво;
- проблем бројања до бесконачности.

Из ових разлога, уведен је потпуно нови алгоритам, назван рутирање

[1] Као што се види појам бесконачности је симболичан: уколико не би спречили процес увећавања метрике путање она би могла да порасте у недоглед.

[2] Описана у документу RFC1058.

[3] Етернет стандард. У време писања овог уџбеника ради се и на стандарду за брзине од 100Gb/s.

на основу стања комуникационих веза[1] (линка). Разне варијанте овог алгоритма широко су и данас распрострањене. Основна замисао је једноставна и може се представити у пет целина. Сваки рутер мора да :

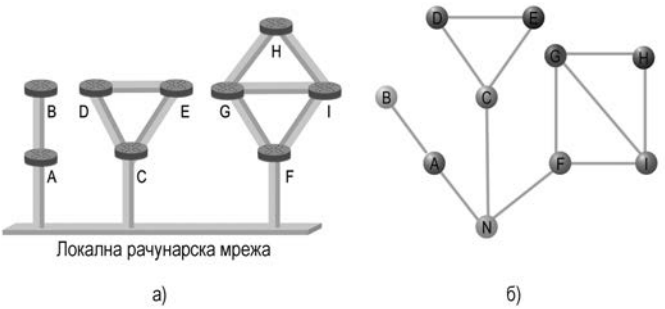
- открије своје суседе и сазна њихове мрежне адресе,
- измери кашњење, односно метрику до сваког суседа,
- формира пакет који садржи информације које је сазнао,
- пошаље овај пакет свим другим рутерима,
- израчуна најкраћу путању ка сваком рутеру.

4.3.1 Откривање суседа

Када рутер почне с радом, његов први задатак је да сазна ко су му суседи. То рутер остварује слањем специјалног HELLO пакета (поздрава) на сваку од тачка-тачка комуникационих веза (линкова). Од рутера на другом крају линка очекује се да пошаље одговор са својим именом. Ова имена морају бити глобално јединствена. На пример ако удаљени рутер добије информацију да су нека три рутера повезана са рутером F, веома је битно да може да се утврди да ли сва три мисле на исти рутер F.

Ако је више рутера спојено преко локалне рачунарске мреже ситуација је мало сложенија. Слика 4.7 приказује локалну рачунарску мрежу са којом су директно повезана три рутера: A, C и F. Сваки од ових рутера повезан је и са још неким рутерима (једним или више).

На слици 4.7б локална рачунарска мрежа представљена је као нови, вештачки чвор N, с којим су спојени рутери A, C и F. Могућност преласка из рутера A у рутер C преко локалне мреже приказано је путањом ANC.



Слика 4.7 а) Девет рутера и локална рачунарска мрежа б) Граф модел слике

[1] *Link State Routing*

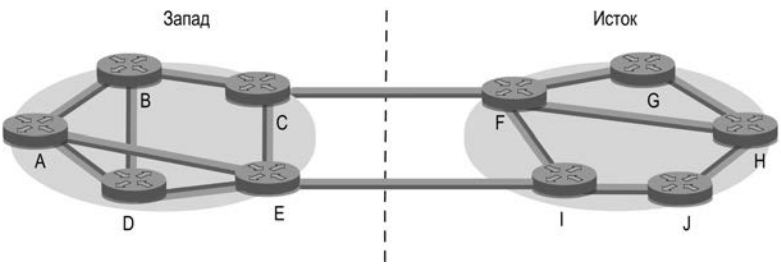
4.3.2 Мерење кашњења

Потребно је да сваки рутер зна кашњење или бар да има реалну процену кашњења до својих суседа. Непосредан начин да се одреди ово кашњење је слањем специјалног ECHO пакета (одговора), који друга страна треба одмах да пошаље натраг. Када рутер време (RTT[1]) које је протекло од тренутка слања HELLO пакета до тренутка приспећа ECHO пакета подели са два добије своју процену кашњења[2]. У овом методу мерења подразумева се да су кашњења симетрична, тј. иста у оба смера, што у реалним условима и не мора да буде тачно.

Поставља се питање да ли треба приликом мерења кашњења урачунати и оптерећење мреже? Ако се урачунава и оптерећење мреже часовник којим се мери време мора се покренути када ECHO пакет „стане“ у ред. Ако се не узима у обзир оптерећење мреже часовник се покреће када ECHO пакет доспе на чело реда.

Ако се узима у обзир и оптерећење мреже рутер ће при избору између две линије са истим пропусним опсегом одабрати ону која је мање оптерећена. На тај начин се побољшавају перформансе мреже.

Понекад овај метод и није тако ефикасан. Обратимо пажњу на подмрежу са слике 4.8 која је подељена у два дела: исток и запад, и који су спојени са две комуникационе везе: CF и EI. Претпоставка је да се већина саобраћаја одвија преко линије CF и као последица тога ова линија је веома оптерећена па су кашњења велика. Ако се при израчунавању оптерећења мреже узме у обзир најкраћа путања, линија



Слика 4.8 Подмрежа у којој су источни и западни део повезани са две комуникационе везе (линије)

[1] *Round Trip Time*

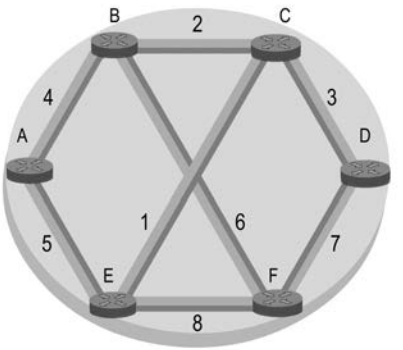
[2] Уколико је потребна тачнија процена може се користити усредњена вредност већег броја мерења.

EI ће постати привлачнија. Пошто су табеле рутирања ажуриране већи део саобраћаја се сада одвија преко линије EI и она ускоро постаје преоптерећена. Као последица тога ће се у следећем ажурирању путања CF појавити као најкраћа. Записи у табели рутирања мењаће наизменично вредности (табеле рутирања ће „осциловати“) изазивајући непредвидиво и непоуздано рутирање. Ако се оптерећење мреже занемари и разматра само пропусни опсег линија до оваквих проблема не може да дође. Да би се избегле брзе промене записа (осцилације) табеле рутирања код избора најбоље путање препоручује се да се саобраћај распореди на више комуникационих линија.

4.3.3 Прављење пакета о стању комуникационих веза

Кад рутер прикупи све информације које треба да размени са својим суседима следећи корак је да направи пакет о стању комуникационих веза. Пакет почиње идентификацијом пошиљаоца, следи редни (секвенцијски број) затим старост пакета и листа суседа. За сваког суседа уписано је кашњење до њега. Пример подмреже дат је на слици 4.9а. Ознаке на линијама представљају кашњења. Одговарајући пакети стања веза за свих шест рутера приказани су на слици 4.9б.

Прављење пакета стања веза је једноставно. Већи проблем је донети одлуку кад пакет треба направити. Једна од могућности је да се пакети периодично праве, тј. у стандардним временским интервалима. Друга



а)

A		B		C		D		E		F	
Р.бр		Р.бр		Р.бр		Р.бр		Р.бр		Р.бр	
Стр.		Стр.		Стр.		Стр.		Стр.		Стр.	
B	4	A	4	B	2	C	3	A	5	B	6
E	5	C	2	D	3	F	8	C	1	D	7
		F	6	E	1			F	8	E	8

Р.бр - редни (секвенцијски) број пакета
Стр. - старост пакета

б)

Слика 4.9 а) Изглед подмреже
б) Пакети стања комуникационих веза једне подмреже

могућност била би да се пакети праве када се догоди неки значајан догађај као што су: завршетак или почетак рада суседа, прекид или поновно успостављање линије итд.

4.3.4 Дистрибуирање пакета о стању комуникационих веза

Најделикатнији део алгоритма је поуздано дистрибуирање пакета о стању комуникационих веза (пакета стања веза). Када се пакети дистрибуирају и информације које они носе унесу у табеле рутирања рутери ће променити своје путање. Последица је да различити рутери могу да користе различите топологије исте мреже што доводи до недоследности, затворених петљи, недоступних рачунара и других проблема.

За дистрибуцију пакета стања веза користи се механизам плављења[1]. Да би се управљало „плављењем“ пакет стања везе садржи редни број који се инкрементира са сваким послатим пакетом. Рутери воде рачуна о пару података (изворишни рутер, редни број) примљених пакета.

Кад доспе нови пакет стања везе, у листи се проверава да ли је он већ виђен. Ако је реч о новом пакету онда се он прослеђује на све линије осим оне са које је примљен. Ако је дупликат онда се он одбацује. Ако је редни број новопристиглог пакета мањи од редног броја већ примљног пакета новопристигли пакет се одбацује као застарели.

Овај алгоритам има неколико (решивих) проблема:

- прво ако дође до поновног коришћења истих редних бројева може да дође до грешке[2]. Решење је користити 32-битне редне бројеве. Ако се у свакој секунди пошаље по један пакет стања везе потребно је 137 година да се поново употреби исти редни број;
- други проблем настаје када рутер престане са радом: губи се траг о редним бројевима. Када рутер настави са радом и почне да шаље пакете почевши од редног броја 0 пакети биће одбачени као дупликати;

[1] *Flooding*

[2] Подсетимо се да је простор редних бројева коначан и зависи од броја битова који се користе за редне бројеве. Уколико се користи n битова простор редних бројева је од 0 до 2^n-1. Када се искористи редни број 22^n-1 поново се покреће нови круг редних бројева од 0 до 2^n-1 (*wrap around*).

- трећи проблем настаје ако дође до грешке у редном броју: на пример примљен је редни број 2056_{10} (100000001000_2) уместо 8_{10} (000000001000_2) грешка у једном биту, пакети од 9 до 2056 биће одбачени као застарели, јер је тренутни редни број (грешком) 2056.

Проблем се може разрешити увођењем још једног поља које се означава као старост пакета и које се декрементира сваке секунде. Кад поље старост пакета достигне нулту вредност, пакет се одбацује. Вредност поља старост пакета се и при иницијалном процесу дистрибуције пакета декрементира у сваком рутеру. Ово је неопходно да би се обезбедило да се пакети не изгубе и евентуално опстану неодређени период времена.

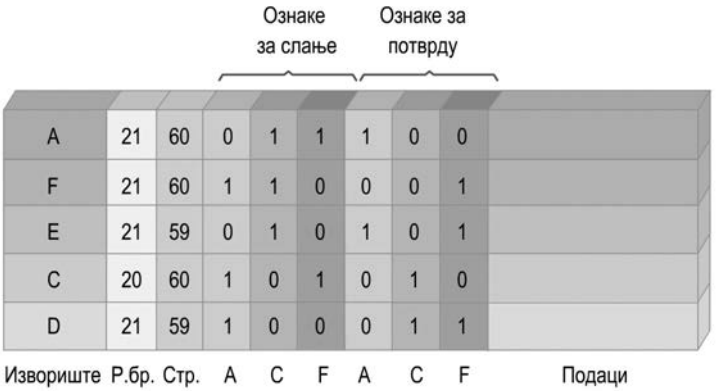
Неке преправке чине овај алгоритам погоднијим за примену. Кад пакет протокола стања везе (LSP[1]) доспе у рутер за дистрибуцију он се не ставља одмах у ред за слање. Уместо тога, задржава се један кратак период времена у меморијском простору (баферу). Ако стигне други LSP пакет из истог извора пре него што се пошаље прводоспели пакет, њихови редни бројеви међусобно се упоређују. Ако су исти новодоспели (дупликат) се одбацује. Ако су различити одбацује се старији пакет. Због заштите од грешака на линијама које повезују рутере сваки послати LSP пакет мора се потврдити.

Структура података коју користи рутер B подмреже са слике 4.9а приказана је на слици 4.10.

Сваки ред табеле одговара скоро пристиглом и не потпуно обрађеном LSP пакету. У табели су дати извориште пакета, редни број, старост, ознаке (флег) слања и ознаке потврда. Ознака слање постављена на вредност 1 указује да је потребно послати пакет на одговарајућу линију. Када је постављена ознака потврде на вредност 1 значи да треба послати пакет потврде.

Пакет стања везе стиже директно из рутера A али га треба послати ка рутерима C и F и послати потврду рутеру A (слика 4.10). Битови ознака постављени су на одговарјућу вредност. Слично, пакет из рутера F треба проследити ка рутеру A и C и послати потврду ка рутеру F. Али ситуација

[1] *Link State Protocol*



Слика 4.10 Бафер за пакете рутера B

са трећим пакетом, који шаље рутер E је другачија. Пакет је стигао два пута: једном преко путање EAB и други пут преко путање EFB. Значи, треба га проследити само рутеру C, али послати потврде рутерима A и F (на шта указују битови ознака).

4.3.5 Израчунавање нових путања

Када рутер прикупи све пакете стања веза, у ситуацији је да конструише читав граф подмреже, јер за сваки линк има податке. Заправо, два податка, за сваки смер по један. Ове две вредности могу се посебно користити или се може наћи средња вредност.

Затим се покреће Дикстра алгоритам у циљу изналажења најкраће путање до свих могућих одредишта. Резултати ових израчунавања уписују се у табеле рутирања и наставља се са радом.

За подмрежу са n рутера, од којих сваки има k суседа, меморија потребна за складиштење улазних података пропорционална је производу kn. За велике подмреже меморијски простор и велики број израчунавања могу да представљају проблем. Без обзира на евидентне проблеме сматра се да за највећи број практичних примена рутирање на основу стања комуникационих веза ради веома добро. Протокол OSPF[1], широко распрострањен на Интернету користи рутирање засновано на стању комуникационих веза.

[1] *Open Shortest Path First*

4.3.6 Поређење алгоритама за рутирање

Коришћење рутирања на основу стања комуникационих веза у већини случајева захтева више меморијског простора за чување података и више времена се потроши на обраду података, него код рутирања на основу вектора удаљености. Администратори мрежа морају обезбедити рутерима могућност резервације неопходних ресурса. Један део опсега мора се искористити за почетну размену пакета стања веза. У почетку сви рутери шаљу пакете стања веза свим другим рутерима, што оптерећује мрежу и привремено смањује ширину опсега предвиђену за саобраћај корисничких података.

Како се повећавају величине рачунарских мрежа, протоколи за рутирања на основу стања комуникационих веза постају све атрактивније решење и то из следећих разлога:

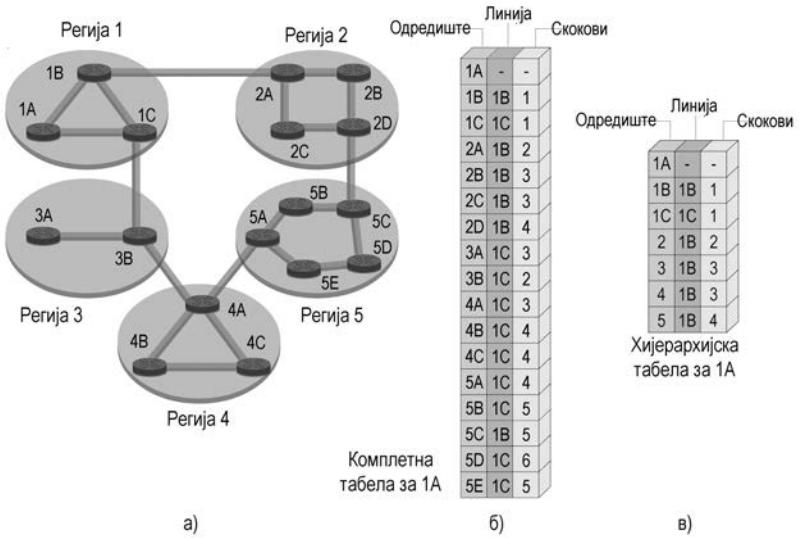
- пакет за ажурирање шаље се само кад се топологија мреже промени,
- периодична ажурирања дешавају се много ређе него код протокола за рутирања на основу вектора удаљености,
- мреже у којима се користе протоколи за рутирања на основу стања комуникационих веза могу да се поделе у хијерархијске области чиме се ограничава домет промена путања,
- мреже у којима се користе ови протоколи подржавају бескласно адресирање.

4.4 Хијерархијско рутирање

Како расту мреже пропорционално расту и табеле рутирања. Не само да се троши меморија рутера на табеле које се непрестано повећавају, већ је потребно и више процесорског времена да се прегледају и више пропусног опсега за слање извештаја о њиховом стању. Када мреже нарасту до одређене тачке више неће бити изводљиво да сваки рутер има запис о сваком другом рутеру, па рутирање треба спровести на хијерархијски начин, као у телефонској мрежи.

У хијерархијском рутирању[1] рутери су расподељени у регије и сваки рутер зна све детаље о рутирању пакета у оквиру своје области (регије),

[1] *Hierarchical Routing*



Слика 4.11 Хијерархијско рутирање

али не зна ништа о унутрашњој структури других области. Када се различите мреже међусобно споје природно је прогласити сваку посебном регијом, како би се рутери једне мреже ослободили потребе познавања топологије других мрежа.

Код великих мрежа двонивовска (двослојна) хијерархија може бити недовољна. Тада је потребно регије поделити у гроздове, гроздове у зоне, зоне у групе, итд. док постоји потреба за даљим дељењем.

Као пример вишенивовске хијерархије, узмимо да се пакет из Крагујевца (Србија) рутира у Беркли (Калифорнија, САД). Рутер у Крагујевцу детаљно познаје топологију мреже у Србији, али саобраћај који је предвиђен да иде ван државе он шаље рутеру у Београд. Рутер у Београду је у могућности да проследи саобраћај другим центрима у регији, Атини, Тирани, Скопљу, Сарајеву, Загребу, итд., но он га прослеђује рутеру у Будимпешти, јер је он одговоран за прекоокеански саобраћај. Рутер у Будимпешти сав саобраћај за САД прослеђује у Њујорк. У Њујорку се саобраћај прослеђује у Лос Анђелес, који је задужен за прослеђивање саобраћаја унутар своје федералне јединице. И на крају пакет стиже у Беркли.

На слици 4.11а приказано је двонивовско хијерархијско рутирање са пет регија. Пуна табела рутирања рутера 1А има 17 записа (слика 4.11б). Ако се рутирање обавља хијерархијски, постоје записи за сваки локални рутер, а остале регије су представљене са по једним рутером, односно једним записом, тако да сав саобраћај за регију 2 иде преко линије 1В-2А, а остатак спољашњег саобраћаја иде преко линије 1С-3В. Хијерархијско рутирање смањило је табелу са 17 на 7 записа (слика 4.11в).

Уштеда у простору може довести до тога да се не одабира најповољнија путања (рута). Нпр. видимо да је најбољи пут између регије 1 и 5 преко регије 2, али је хијерархијским рутирањем одређено да сав саобраћај за регију 5 иде преко регије 3, јер је то најповољније за већину одредишта у регији 5.

Код великих мрежа поставља се питање колико хијерархијских нивоа треба да постоји. Једно од истраживања[1] је показало да је оптималан број нивоа за подмрежу са N рутера, $ln\ N$ а потребно је $e ln\ N$ записа по рутеру. Такође је доказано да је повећање ефективног средњег пута, проузрокованог хијерархијским рутирањем, довољно мало да је најчешће прихватљиво.

4.5 Рутирање емисијом

У неким апликацијама крајње станице треба да пошаљу поруке великом броју станица или чак свима. Слање пакета свим одредиштима истовремено (емисија, дифузија) назива се и слање један ка свима (бродкаст[2]). Постоје разне методе којима се то може учинити. Због једноставности даље у овом поглављима користиће се термин емисија или бродкаст.

Један метод који не захтева специјалне карактеристике подмреже је једноставно слање различитих пакета сваком од одредишта. Овај метод троши велики пропусни опсег и захтева да изворište има листу свих одредишта.

Плављење је још један метод који се може користити. Проблем са плављењем је што такође генерише превише пакета и захтева велики пропусни опсег.

---

[1] Детаљније се може наћи у раду [5].

[2] Дифузна емисија, дифузија (*broadcasting*)

Трећи алгоритам је рутирање ка више одредишта[1]. Сваки пакет садржи листу одредишта или мапу са означеним одредиштима. Када пакет стигне у рутер, рутер проверава сва одредишта како би одредио скуп излазних линија које ће користити. Излазна линија која је одабрана користи се ако је то најбоља путања до бар једног одредишта. Рутер генерише копију пакета за сваку излазну линију коју користи и укључује у сваки пакет само она одредишта која користе ту линију. Дакле, скуп одредишта је подељен међу излазним линијама. Након довољног броја скокова сваки пакет ће имати у себи адресу само једног одредишта и посматраће се као нормалан пакет. Вишеодредишно рутирање је слично одвојено адресираним пакетима, осим што кад неколико пакета следи исту путању за само један пакет рутер одређује путању а сви остали ту путању следе.

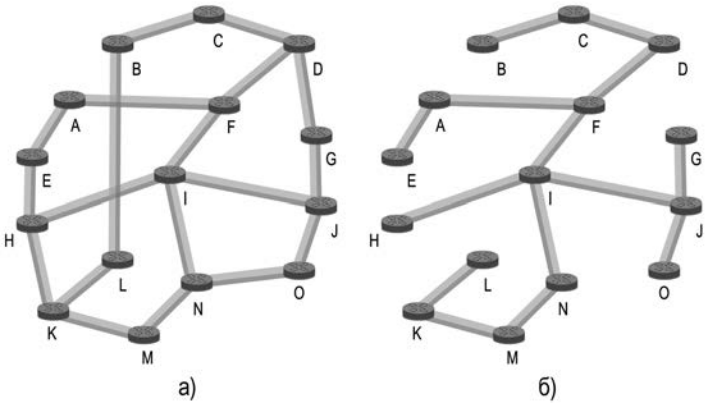
Четврти алгоритам рутирања емисијом[2] користи понирује стабло за рутер који иницира емисију, или неко друго погодно разгранато стабло. Разгранато стабло је подскуп подмреже који укључује све рутере, али не садржи петље. Ако сваки рутер зна које од његових линија припадају разгранатом стаблу, он може да копира пристигли емисиони пакет на све линије разгранатог стабла осим на ону са које је пакет пристигао. Овај метод одлично користи пропусни опсег, генеришући минималан број пакета за реализацију алгоритма. Једини проблем је што сваки рутер мора познавати структуру разгранатог стабла да би метод био применљив. Понекад је ова информација приступачна (нпр. код рутирања на основу стања комуникационих веза) а некад не (нпр. код рутирања користећи вектор удаљености).

Последњи алгоритам је покушај да се процени понашање емисионог алгоритма који користи разгранато стабло, чак и ако рутери не знају баш ништа о разгранатом стаблу. Идеја, названа прослеђивање реверзном путањом[3] веома је једноставна. Када емисиони пакет доспе у рутер, он проверава да ли је пакет стигао линијом која се обично користи за слање пакета изворишту емисије. Ако је тако велике су шансе да је емисиони пакет такође пратио најбољу путању и да је, према томе, прва копија

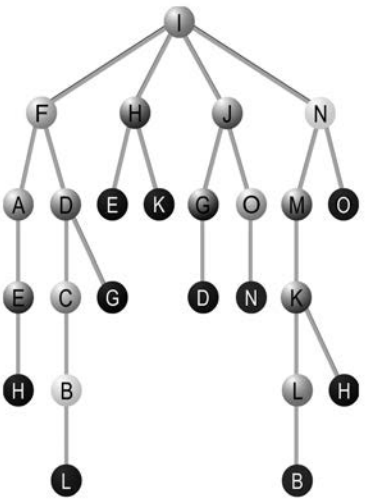
[1] *Multidestination routing*
[2] *Broadcast Routing*
[3] *Reverse path forwarding*

пристигла у рутер. У овом случају рутер прослеђује своје копије на све линије сем на ону са које је пакет пристигао. Ако је, међутим, емисиони пакет стигао другом линијом, одбацује се пошто је највероватније дупликат.

Пример овог алгоритма може се видети на сликама 4.12 и 4.13. Део приказан на слици 4.12а показује подмрежу, 4.12б понирује стабло, а



Слика 4.12 Прослеђивање реверзном путањом: а) Подмрежа б) Понирује стабло



Слика 4.13 Приказ рада алгоритма рутирања емисијом

4.13 функционисање алгоритма. На првом скоку рутер I шаље пакете ка рутерима F, H, J и N. Сваки од ових пакета стиже најбољом путањом до рутера I. На другом скоку осам пакета се генерише, два од сваког рутера који је примио пакет у првом скоку. Како се испоставља, свих осам пакета стижу на досад непосећене рутере, а пет од њих стиже најбољом путањом. Од шест пакета генерисаних у трећем скоку само три пакета стижу најбољом путањом (на рутере C, E и K); остали су дупликати. После пет скокова и 24 пакета емисија се окончава. Четири скока и 14 пакета су пратили понирућe стабло.

Основна предност алгоритма реверзног усмеравања[1] је у томе да је истовремено веома ефикасан и лак за имплементацију. Не захтева се знање рутера о разгранатим стаблима[2], није потребан никакав специјални механизам за заустављање процеса као што је то потребно код плављења.

4.6 Рутирање за мобилне кориснике[3]

Данас милиони људи имају преносне рачунаре и мобилне телефоне. Они обично желе да читају своју е-пошту и приступе својим стандардним системима датотека, где год да су у свету. Ове мобилне станице захтевају посебан приступ. Да би усмерила пакет мобилном кориснику, рачунарска мрежа прво мора да пронађе где се корисник налази.

Модел глобалне мреже који се често користи представљен је на слици 4.14. Чине је мреже ширих подручја, мреже градског подручја, локалне рачунарске мреже и бежичне мреже.

Крајње станице, које никада не мењају место означићемо као непомерљиве (фиксне). Оне су повезане са глобалном мрежом упреденим парицама или оптичким влакнима. Постоје две врсте померљивих станица:

- номадске (миграцијске станице[4]) су станице које се померају са једног места на друго с времена на време, али користе мрежу само кад се физички повежу с њом,

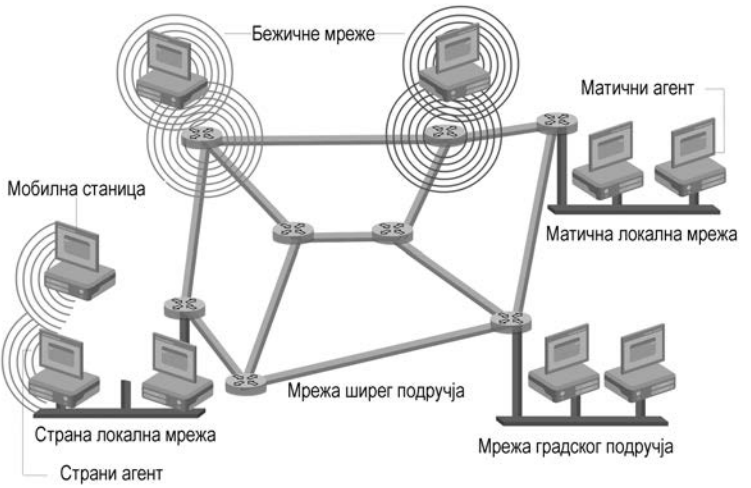
[1] *Reverse path forwarding*
[2] *Spanning tree*
[3] *Routing for Mobile Hosts*
[4] *Migratory hosts*

- роминг станице[1] одражавају везу са мрежом током кретања.

Користиће се израз мобилни корисник[2] за две последње категорије, тј. за све станице које су ван матичног објекта (куће, пословне организације...) а потребна им је веза с мрежом.

Претпоставља се да све станице имају сталну (матичну) локацију, која се никад не мења. Станице имају матичну адресу која се користи за одређивање матичне локације[3]. Циљ рутирања у системима са мобилним корисницима је омогућити слање пакета на матичне адресе и њихово допремање до корисника, ма где се они налазили.

Глобална мрежа (представљена на слици 4.14) подељена је (географски) у мање целине - области. Уобичајено је да је област једна



Слика 4.14 Кретања мобилног корисника по глобалној мрежи

[1] *Roaming hosts*. У бежичним комуникацијама роминг (*roaming*) је генерални термин који се односи на проширење услуге повезивања (*connectivity service*) на локацију која је различита од локације где је услуга регистрована. Роминг осигурава да се бежични уређај држи повезан са мрежом без губитка везе. Термин „роминг" потиче из GSM (*Global System for Mobile Communications*) мреже. Традиционално *GSM Roaming* је дефинисан као могућност за мобилног претплатника да аутоматски обавља и прима говорне позиве, шаље и прима податке или приступа другим услугама када путује ван географске области која је покривена матичном (*home*) мрежом користећи посећену (*visited*) мрежу. Може се обезбедити употребом комуникационог терминала или само коришћењем претплатничког идентитета у гостујућој мрежи (погледати *GSM Association Permanent Reference*).

[2] *Mobile hosts*

[3] Слично као кад број телефона (јавне телефонске мреже – фиксна телефонија) указује којој држави и којој области државе претплатник припада.

локална рачунарска мрежа или бежична ћелија[1]. Свака област има једног или више страних агената (агената за удаљене, стране станице[2]). Страни агенти су процеси који воде рачуна о свим мобилним станицама које посећују ту област. Поред тога, свака област има и свог матичног (домаћег[3]) агента који води рачуна о станицама које припадају тој области, а тренутно су у другој области.

Када нова станица приступи области, она се мора регистровати код страног агента те области. Процедура регистрације обично изгледа овако:

- периодично, сваки страни агент емитује пакет оглашавајући своје постојање и своју адресу. Придошла, мобилна станица може да чека на једну од ових порука, али ако се ниједна не појави довољно брзо, може да емитује пакет тражећи (распитујући се за) страног агента;
- мобилна станица се региструје код страног агента, дајући му своју матичну адресу, тренутну MAC адресу и информације о сигурности;
- страни агент обавештава матичног агента мобилног хоста о присуству његовог корисника у новој области, шаљући му поруку која садржи мрежну адресу страног агента, као и информације о сигурности, чија је улога да убеде матичног агента да је мобилни хост заиста у тој удаљеној мрежи;
- матични агент анализира добијене информације, које садрже и временски печат (доказ да су информације скоро генерисане). Ако је све у реду, матични агент дозвољава страном агенту да настави с радом;
- када страни агент добије потврду од матичног агента, прави запис у својим табелама рутирања и обавештава мобилну станицу о успешној регистрацији.

Добро би било да се мобилна станица одјави када напушта удаљену област. Дакле, кад се пакет шаље мобилној станици, рутира се ка његовој матичној (кућној) адреси што то (неретко) није случај.

[1] *Wireless cell*
[2] *Foreign agent*
[3] *Home agent*

Слика 4.15 Рутирање код мобилних станица

Када се пакет шаље мобилној станици, он се рутира ка његовој матичној адреси (корак 1, слика 4.15). Нпр. пошиљалац у Суботици жели да пошаље податке у Ниш. Пакете са подацима у Нишу прихвата матични агент Ниша, претражује у својој бази нову (привремену) локацију мобилне станице и проналази адресу страног агента у Сегедину коме је пријављена мобилна станица из Ниша.

Матични агент ће затим урадити две ствари:

- прво, укалупити (енкапсулира) пакет у поље података другог пакета који шаље страном агенту у Сегедин (корак 2, слика 4.15). Овај механизам се назива тунеловање; пошто прими пакет, страни агент узима оригинални пакет из поља података, укалупљује га у рам другог слоја и шаље мобилној станици;
- друго, обавестити пошиљаоца да убудуће пакете намењене овој станици ставља у поље података пакета и адресира на страног агента уместо да га шаље на матичну адресу мобилне станице (корак 3, слика 4.15). Сви следећи пакети упућени мобилној станици биће рутирани до ње преко страног агента (корак 4, слика 4.15) заобилазећи потпуно матичну локацију мобилне станице.

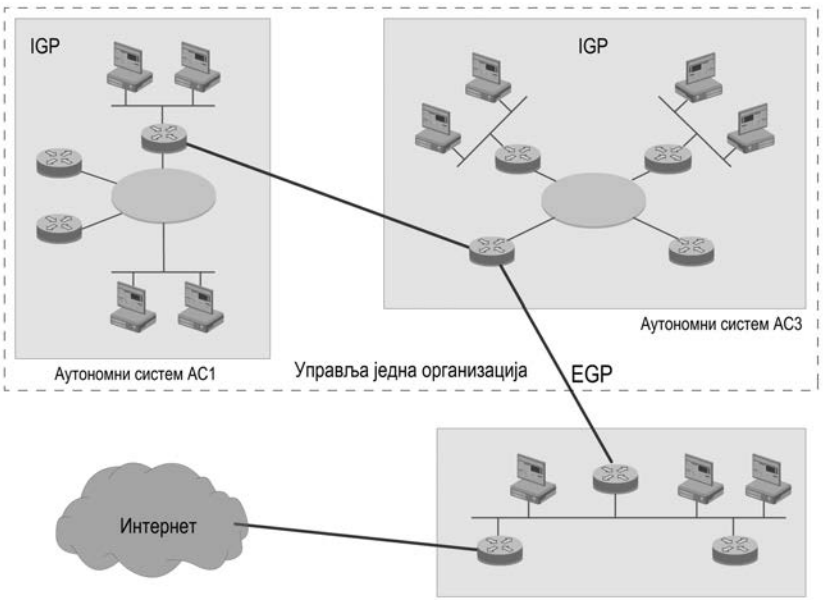
4.7 Протоколи за рутирање на Интернету

Протоколи за рутирање омогућавају да се динамички прикупљају и одржавају информације у табелама рутирања. Како се мења топологија мреже табеле рутирања се иновирају.

Аутономни систем представља целину која се састоји од међусобно повезаних мрежа. Аутономним системом може да управља једна организација и једна организација може да има више аутономних система.

Протоколи за рутирање деле се на:

- унутрашње протоколе за рутирање (IGP[1]) који омогућавају размену информација у оквиру аутономног система. Пример IGP протокола су OSPF[2] и RIP[3];
- спољашње протоколе за рутирање (EGP[4]) који омогућавају



Слика 4.16 Аутономни системи, унутрашњи (IGP) и спољашњи (EGP) протоколи за рутирање

[1] *Interior Gateway Protocol*
[2] *Open Shortest Path First*
[3] *Routing Information Protocol*
[4] *Exterior Gateway Protocols*

размену здружених информација између аутономних система. Пример EGP протокола је BGP[1] протокол.

Као што се на слици 4.16 види аутономни системи могу се међусобно повезивати и повезати са другим приватним или јавним рачунарским мрежама.

4.7.1 RIP протокол

Релативно једноставан протокол интерног рутирања је протокол за размену информација о рутирању RIP[2] протокол. Поред своје једноставности RIP протокол се показује ефикасним у мањим подмрежама и практично је највише коришћени протокол за рутирања.

Кључна чињеница везана за протокол RIP је да он користи алгоритам рутирање заснован на вектору удаљености. Слика 4.17 показује формат RIP пакета. Сваки пакет укључује заглавље са следећим пољима:

- управљачко поље - једно за захтев, друго за одговор. Измене стања рутирања се шаљу као одговору без обзира да ли су захтевани или не. Када чвор иницијализује RIP, он шаље RIP захтева упућен свим станицама (бродкаст). Сваки рутер прима захтев и тренутно шаље одговор;
- верзија - ово поље има вредност 1 за основни RIP протокол а вредност 2 за протокол RIP-2;

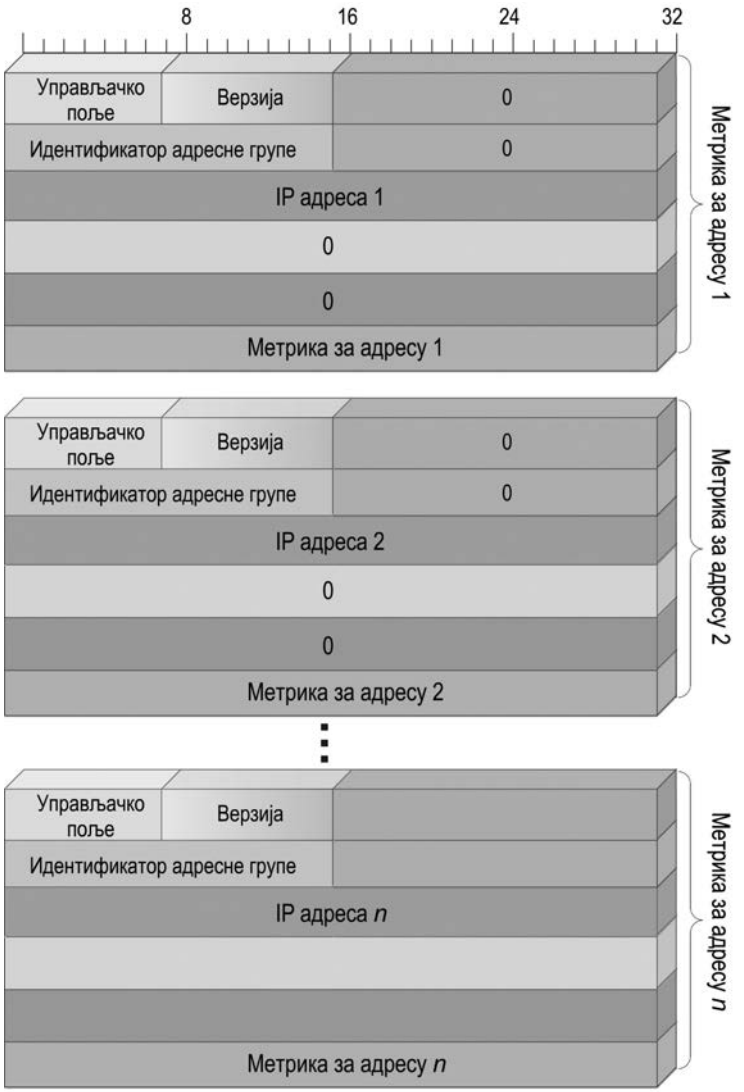
Иза заглавља налази се један или више блокова од којих сваки даје путању до одговарајућих одредишта мреже. То су:

адресна група - има вредност 2 за IP адресе;

- IP адресе - односи се на IP адресе код којих ни адреса мреже ни адреса станице не могу имати нулту вредност. На тај начин су поједине мреже јединствено дефинисане;
- метрика - удаљености рутера од препознате мреже. Обично се употребљава вредност једног линка, тако да се метрика изражава бројањем скокова. Ако су дозвољене веће вредности линка, тада је број скокова који могу бити употребљени неупоредиво мањи.

RIP протокол је популаран протокол за рутирање због своје једноставности и добре прилагођености мањим подмрежама. Има својих

[1] *Border Gateway Protocol*
[2] *Routing Information Protocol*



Слика 4.17 Формат RIP пакета

ограничења и споменућемо само неке од њих:

- како подмрежа расте, одредишта која захтевају метрику већу од 15 постају недоступна. Због тога RIP протокол не одговара већим мрежама;

- Једноставност метрике води до подоптималних табела рутирања. Резултат је да се пакети шаљу преко споријих (или на други начин захтевних) веза иако су доступне боље путање;
- Уређаји који не подржавају RIP протокол прихватиће RIP измене са било ког уређаја. Ово омогућава погрешно конфигурисаном уређају да лако поремети целу конфигурацију.

4.7.2 OSPF протокол

Протокол (отворени) за рутирање најкраћом путањом (OSPF протокол[1]) широко је распрострањен протокол за рутирање унутар аутономних система. Реч отворен у називу протокола указује да је спецификација протокола јавно доступна[2]. Најновије верзије (верзија 2) овог протокола описане су у документима RFC2328 и RFC5340 (за IPv6).

OSPF протокол сматра се наследником RIP протокола. Нуди неколико значајних предности у односу на RIP протокол, укључујући способност да ради у много већим мрежама, а да измене у табелама рутирања имају много мањи утицај на мрежни саобраћај. Главна разлика између протокола за рутирање је у начину или алгоритму којим рутери прикупљају ажурне информације о рутирању.

На пример, RIP протокол користи алгоритам вектора раздаљине[3], који мери само број скокова (максимално 15) до удаљеног рутера, док OSPF протокол користи алгоритам рутирања на основу стања комуникационих веза[4]. OSPF протокол врши одабир између алтернативних путања не само на основу броја скокова, већ и кашњења, капацитета, протока и поузданости линија које повезују рутере. И што је можда још важније, за одржавање ажурних табела рутирања OSPF протокол захтева много мањи проток.

OSPF области

Многи аутономни системи на Интернету су велики и веома сложени за одржавање. OSPF протокол дозвољава да се аутономни системи

[1] *Open Shortest Path First*

[2] Протокол компаније Cisco EIGRP (*Enhanced Interior Gateway Routing Protocol* - унапређени интерни протокол рутирања мрежног пролаза) у власништву је компаније *Cisco* и извршава се искључиво на *Cisco* рутерима.

[3] *Distance vector*

[4] *Linkstate*

поделе у области (регионе) које се састоје од скупа међусобно суседних мрежа. Области се не преклапају, али је могуће и да постоје рутери који не припадају ниједној области.

Сваки аутономни систем има област окоснице[1] и означава се као област 0. Све друге области повезане су на окосницу (највероватније преко тунела). На тај начин се из једне области у другу област може доћи преко окоснице. Рутер који је повезан на две или више области је део окоснице. Као што је случај и са другим областима топологија окоснице је видљива само унутар окоснице (невидљива је ван ње).

У оквиру једне области рутери користе исту базу података са стањем линкова и покрећу исти алгоритам најкраће путање. Основни задатак овог алгоритма је да пронађе најкраћу путању од рутера до било ког другог рутера у области укључујући и рутер који је повезан са окосницом (ABR[2] рутер). Рутер који повезује две области мора да поседује базу података за обе области и да примени алгоритам најкраће путање посебно за сваку од области. OSPF протокол разликује четири класе рутера (слика 4.18):

- интерни рутери – припадају само једној области,
- гранични рутери области – повезују две или више области,
- рутери окоснице – припадају области окоснице и
- гранични рутери аутономног система (ASBR[3] рутер)

Слика 4.18 Повезивање OSPF региона (области)

[1] *Backbone*

[2] *Area Border Router*

[3] *Autonomous System Boundary Router*

Ове класе рутера могу се преклапати. На пример сви гранични рутери аутоматски су део и области окоснице. Такође, рутер који припада области окоснице је интерни рутер.

OSPF поруке

Користећи *Hello* поруке протокол OSPF провера исправности рада својих линкова. Када се рутер укључи он шаље *Hello* поруке на све своје тачка-тачка везе и шаље поруке ка групи рутера (мултикаст) на локалној рачунарској мрежи. За мреже ширег подручја, да би послао *Hello* поруке, потребне су додатне информације о конфигурацијама. На основу одговора рутери закључују (уче[1]) ко су њихови суседи. Рутери на једној локалној рачунарској мрежи су сви међусобни суседи[2].

OSPF протокол се примењује тако што се информације размењују између оближњих[3] рутера што није исто као код размене између суседних рутера. Нарочито је неефикасно да сваки рутер на локалној рачунарској мрежи „разговара“ са сваким рутером на тој мрежи. Да би се оваква ситуација избегла један од рутера се бира за именованог[4] рутера. Каже се за њега да је он оближњи рутер свим осталим рутерима на локалној рачунарској мрежи и да размењује информације са њима. Суседни рутери који нису оближњи рутери међусобно не размењују информације. Води се рачуна да резервни[5] именовани рутер има све потребне податке тако да у случају квара примарни именовани рутер може одмах да се замени.

У току нормалног рада сваки рутер периодично шаље (преплављује) поруке о иновирању стања (*Link state update*) својим оближњим рутерима. Ова порука садржи податке о стању тог рутера и цене које су биле коришћене у бази података топологије. Да би се обезбедила поузданост поруке се потврђују. Свака порука има редни број па рутер може да види да ли је долазећа порука старија или новија од оне коју он већ има.

Порука *Database description* носи податке о свим редним бројевима записима о стањима комуникационих веза (линкова) које пошиљалац

[1] *Learns*
[2] *Neighbors*
[3] *Adjacent*
[4] *Designated*
[5] *Backup*

има. Упоређивањем сопствених података са подацима добијеним од предајника (пошиљаоца) пријемник може да одреди ко има најновије податке. Ови подаци се користе када се линије укључују.

Да би рутер добио податке о стању линкова од другог рутера шаље поруку *Link state request*. Резултат овог алгоритма је да сваки пар оближњих рутера проверава ко има најновије податке и они се дистрибуирају по области.

На основу претходног описа може се сумирати начин рада OSPF протокола. Користећи плављење рутер обавештава све остале рутере у својој области о својим суседима и ценама линкова. Ова информација омогућава сваком рутеру да направи граф своје области и прерачуна најкраће путање (односи се и на област окоснице). Поред тога рутери окоснице прихватају податке од граничних рутера области (ABR рутер) са циљем изналажења најбоље путање од рутера окоснице до сваког другог рутера. Ова информација се прослеђује натраг до ABR рутера који је објављују у својој области.

Користећи ове податке рутерима је омогућено да рутер који планира да пошаље пакет ван области може да одабере излазни рутер на најбољој путањи до окоснице.

Формат OSPF пакета

Сви OSPF пакети користе исто 24-битно заглавље (слика 4.19) које садржи следећа поља:

- верзија[1] - број 2 је текућа верзија;
- тип[2] (врста) - одређује једну од пет врста (типова) пакета;
- дужина пакета[3] - дужина овог пакета је у бајтима (октетима) укључујући OSPF заглавље;
- идентификатор рутера[4] - одређује извориште пакета. Сваком рутеру у области додељује се јединствен 32-битни идентификатор;
- идентификатор области[5] - одређује област којој изворишни рутер припада. Сви OSPF пакети су везани за једну област;

[1] *Version number*
[2] *Type*
[3] *Packet Length*
[4] *Router ID*
[5] *Area ID*

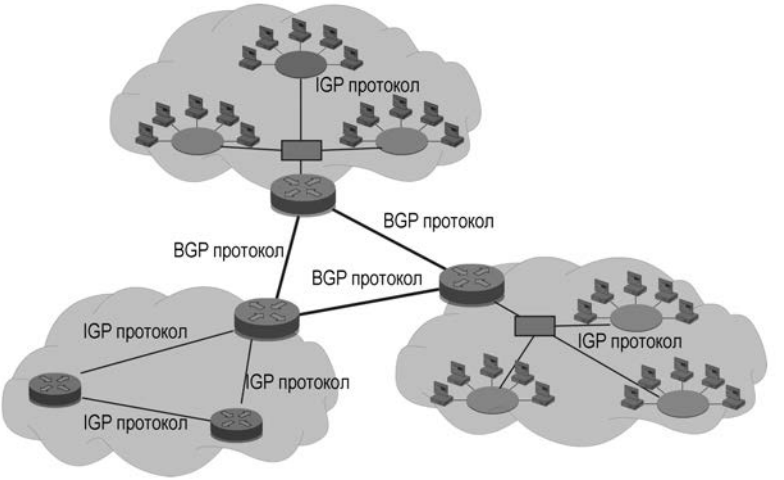
Слика 4.19 Заглавље OSPF пакета

- контролни збир[1] - стандардни контролни збир садржаја пакета почевши од заглавља OSPF пакета;
- врста аутентификације[2] - указује на процедуру аутентификације која се користи за пакет. Процедура може да буде нула (нема аутентификације), једноставна шифра или криптографска;
- подаци за аутентификацију[3] - 64-битно поље које користи процедуру за аутентификацију.

4.8 Спољашњи протоколи за рутирање

Намена протокола за рутирање између аутономних система или протокола граничног међумрежног пролаза (BGP[4] протокол) је да обезбеди комуникацију између рутера у различитим аутономним системима (слика 4.20). Протокол BGP је замена за протокол EGP[5] који се користио у ARPANET мрежи. Актуелна верзија овог протокола (верзија 4) представља *de facto* стандард на данашњем Интернету. Описан је у документима RFC4271 (такође и у RFC4274 и RFC4276). Систем са BGP протоколом размењује информације о мрежи са другим системима који користе BGP протокол. Ове информације омогућавају формирање графа

[1] *Checksum*
[2] *Authentication Type*
[3] *Authentication Data*
[4] *Border Gateway Protocol*
[5] *Exterior Gateway Protocol*



Слика 4.20 Аутономни системи и спољашњи протоколи за рутирање

повезаности аутономног система. Затворене петље у путањама могу се уклонити и може се применити нека од метода одабира путања.

У аутономном систему пренос IP пакета може се посматрати као локални саобраћај или пролазак (транзит). Локални саобраћај може да буде изворишни или одредишни (започиње или се завршава у аутономном систему). Ако је саобраћај локални онда је или изворишна IP адреса или одредишна IP адреса адреса станице у том аутономном систему. Све остало сматра се пролазећим (транзитним) саобраћајем. Најважнији допринос употреби BGP протокола је у смањењу транзитног саобраћаја. Аутономни системи могу се категоризовати на следећи начин:

- аутономни систем који има само једну везу са другим аутономним системима[1] и може да преноси само локални саобраћај;
- аутономни систем повезан је са више од једне везе са другим аутономним системима[2] али не прихвата транзитни саобраћај;
- транзитни аутономни систем повезан је са више аутономних система и може уз одређена ограничења да прихвата и локални и транзитни саобраћај.

[1] *Stub*
[2] Multihomed

Свеукупна топологија Интернета је таква да се састоји од све три врсте аутономних система[1].

Протокол BGP разликује се од RIP и OSPF протокола по томе што BGP протокол користи TCP протокол (порт 179). Стриктно говорећи, BGP је протокол вектора путање[2] што значи да рутери који користе BGP протокол (BGP рутери) међусобно размењују информације о путањи. Аутономни систем одређен је 16-битним бројем. Правила рутирања код BGP протокола могу се врло прецизно поставити изменом BGP атрибута и подешавањем филтрирања рута.

Пошто BGP протокол користи TCP протокол за испоруку информација о векторима путања сматра се поузданим протоколом. Протокол BGP користи принцип суседних аутономних система. Када се открије суседни аутономни систем специјалне поруке[3] се непрестано размењују да би се осигурала видљивост оглашених путања. На тај начин BGP рутери знају да је дошло до отказа суседног рутера и да су информације о вези постале неважеће. Сваки BGP рутер у аутономном систему конфигурише се да оглашава скуп мрежа унутар тог аутономног система, и да дефинише са којим ће се рутерима на суседним аутономним системима размењивати информације о векторима путања. Различита тежина може се доделити појединим путањама тако да се једна путања учини атрактивнијом од друге.

У случају аутономних система статичко рутирање може бити пожељније од динамичког. Пошто се информације о рутирању не деле преко веза које повезују различите аутономне системе, знатно се смањује опасност да један лоше подешен рутер из другог аутономног система негативно утиче на рутере датог аутономног система.

4.9 Питања и задаци

1. Која су два најчешће коришћена статичка алгоритма за рутирање?
2. Који подаци су придружени сваком од чворова када се користи Дикстра алгоритам?

---

[1] *Stub, multihomed, transit*

[2] *Path vector protocol*

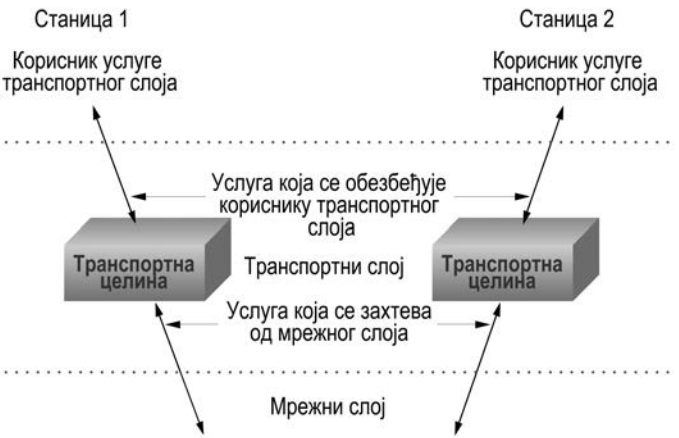
[3] *Keepalive* - порука којом указује да је у радном стању.

3. Метрика за линеарну мрежу на слици 4.5 је број скокова. Када се рутер A укључи колика је метрика према рутеру A у осталим рутерима после три размене података из њихових табела рутирања?
4. На који начин долази до проблема „бројање до бесконачности“?
5. Који су недостаци рутирања на основу вектора удаљености?
6. Који су задаци рутера који примењује алгоритам рутирања на основу стања комуникационих веза?
7. Међусобно упоредити алгоритме за рутирање.
8. Описати принцип рутирања за мобилне кориснике.
9. Описати појам и намену аутономног система.
10. Набројати унутрашње и спољашње протоколе за рутирање.
11. Који алгоритам користи RIP протокол?
12. Нацртати и описати намене поља RIP пакета
13. Који протокол користи алгоритам на основу стања веза?
14. RIP протокол бира путање на основу броја скокова. На основу којих параметара OSPF протокол врши одабир између алтернативних путања?
15. Ако се узима у обзир величина мреже који протокол за рутирање се користи код мањих а који код већих мрежа?
16. OSPF протокол разликује четири класе рутера. Набројати их и описати.
17. Како OSPF протокол користи *Hello* поруке?
18. Када OSPF протокол шаље поруке *Link state* update својим суседима?
19. Нацртати и описати формат OSPF пакета.
20. Описати намену спољашњих протокола за рутирање.

5. Транспортни слој

Транспортни слој је кључна карика у целокупном концепту архитектуре протокола. Протоколи нижих слојева су лакши за разумевање и мање сложени од транспортног протокола. Транспортни протокол обезбеђује услугу преноса података између крајњих корисника (с краја на крај). Свака апликација се може пројектовати тако да користи директно услугу транспортног слоја без посредства слоја сесије и презентације.

Најважнији циљ транспортног слоја је да обезбеди ефикасну, поуздану и економичну услугу својим корисницима, обично процесима на апликационом слоју[1]. Да би достигао овај циљ транспортни слој користи услуге које му обезбеђује мрежни слој. Хардвер и/или софтвер у оквиру транспортног слоја који обавља те задатке назива се транспортна целина. Транспортна целина може бити реализована



Слика 5.1 Веза мрежног, транспортног и виших слојева

[1] Или слоју сесије ако је реч о OSI референтном моделу.

као део језгра оперативног система, као посебан кориснички процес у библиотеци повезаној са мрежном апликацијом или на мрежној интерфејсној картици. Повезаност (логичка) мрежног, транспортног и виших слојева приказана је на слици 5.1.

Услуга коју пружа транспорни слој веома је слична услузи коју пружа мрежни слој[1]. Поставља се питање: ако је то тако зашто постоје и користе се два различита слоја? Зашто није довољан само један слој?

Разлог је једноставан. Софтвер који реализује услуге транспортног слоја налази се на рачунару самог корисника, док се софтвер који реализује услуге мрежног слоја најчешће налази на рутерима који су код рачунарских мрежа ширих географских подручја[2] део опреме оператера. Тиме је изнад мрежног слоја постављен још један слој – транспортни, који може да побољша квалитет услуге мрежног слоја на месту и према захтеву крајњег корисника (апликације).

5.1 Услуге које пружа транспортни слој

Услуге које обезбеђује транспортни слој јесу пренос података с краја на крај на такав начин да штите кориснике од детаља везаних за комуникациони систем који користе. Категорије услуга које су корисне у описивању услуга транспортног слоја (транспортна услуга) су:

- тип услуге,
- степен услуге,
- пренос података,
- кориснички интерфејси,
- надзор везе,
- убрзана испорука података,
- извештај о статусу,
- безбедност.

5.1.1 Тип транспортне услуге

Могућа су два основна типа услуге: са успоставом везе[3] и без

[1] Користићемо термин транспортна услуга за услугу коју пружа транспортни слој и мрежна услуга за услугу коју пружа мрежни слој.

[2] WAN - *Wide Area Network*

[3] *Connection oriented*

успоставе везе[1]. Услуга са успоставом везе обезбеђује успостављање, одржавање и раскидање логичке везе између транспортних корисника. Оваква услуга генерално је поуздана и садржи механизме као што су: управљање протоком, управљање грешкама и испорука у правилном редоследу.

5.1.2 Степен транспортне услуге

Протокол транспортне целине дозвољава кориснику да специфицира степен услуге или квалитет услуге који очекује. Транспортна целина ће покушати да оптимизира употребу мреже и везе. Примери услуге који се могу захтевати су:

- прихватљив ниво погрешних и изгубљених података,
- пожељно и максимално прихватљиво кашњење,
- пожељна и минимална пропусност[2],
- нивои приоритета.

Јасно је да су могућности транспортне целине да обезбеди захтеване услуге ограничене могућностима услуга које нуде нижи слојеви. Навешћемо неке од апликација које захтевају специфични степен услуге:

- протокол за пренос датотека FTP[3] може да захтева велику пропусну моћ система и високу поузданост да би спречио поновно слање,
- трансакциони протоколи (упити база података) могу да захтевају мала кашњења,
- електронска пошта може да захтева више нивоа приоритета.

Постоје четири категорије транспортног протокола:

- поуздан протокол са успоставом везе,
- мање поуздан протокол без успоставе везе,
- протокол за пренос говора који захтева благовремени пренос и пренос исправног редоследа,
- протокол за пренос у реалном времену који захтева високу поузданост и минимална кашњења.

---

[1] *Connectionless*
[2] *Throughput*
[3] *File Transfer Protocol*

5.1.3 Пренос података у транспортној вези

Намена транспортног протокола је да обезбеди пренос података између транспортних целина. Подаци и управљачке информације могу се преносити истим или различитим каналом. Могуће је: потпуни дуплекс, полудуплекс и симплекс начин рада.

5.1.4 Интерфејс ка кориснику транспортне услуге

Прецизно дефинисан интерфејс ка кориснику[1] нема потребе да буде стандардизован. Боље је да се у реализацији прилагоди специфичностима окружења у коме се реализује. Веза између транспортне целине и корисника посматра се преко примитива (операција) и параметара који се размењују. Ове примитиве налажу целини да изврши неку активност, или да извести о некој активности коју су парњак целине обавиле. Један од начина да се класификују примитиве је да их групишемо у четири класе (табела 5.1). Примитиве које се користе су:

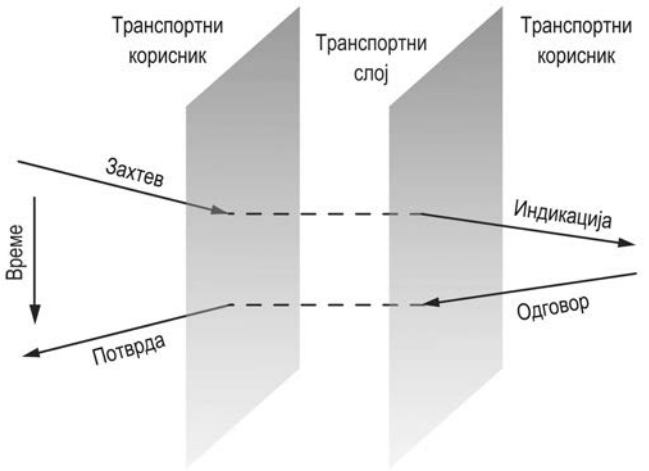
- захтев који иницира транспортни корисник да би покренуо одређену услугу,
- индикација коју обезбеђује транспортна целина да би обавестила о покретању активности за остваривање одређене услуге,
- одговор који обезбеђује траспортни корисник као одзив на примитиву индикација,
- потврда која се враћа транспортном кориснику по реализацији тражене услуге.

На слици 5.2 приказан је временски редослед ових догађаја.

Примитиве	Значење
TS *Request* -TS захтев	Захтев корисника транспортној целини за одређеном услугом
TS *Indication* – TS индикација	Индикација транспортне целине о покретању активности за реализацију услуге
TS *Response* - TS одговор	Одговор корисника траснпортној целини на примитиву индикације коју је она упутила.
TS *Confirm* - TS потврда	Потврда кориснику (иницијатору) на захтев за услугом о реализацији одређене услуге

Табела 5.1 Четири класе примитива транпортних услуга

[1] Користи се и термин кориснички интерфејс.



Слика 5.2 Временски редослед примитива

5.1.5 Надзор транспортне везе

Када је обезбеђена услуга са успоставом везе транспортна целина је одговорна за успостављање и раскидање (окончање) везе. Када је симетрична процедура обезбеђена корисници са оба краја везе могу да иницирају комуникацију. Асиметричне процедуре омогућавају реализацију симплекс веза.

Раскид везе може да буде тренутан или одложен. Када је тренутан раскид везе онда подаци који су послати, а још нису стигли на одредиште, могу бити изгубљени. Одложен раскид спречава било коју страну да прекине везу док сви послати подаци не стигну на своје одредиште.

5.1.6 Убрзана испорука података у транспортној вези

Представља услугу која одговара механизму прекида и користи се за пренос повремено хитних података (нпр. аларм). Разликује се од услуге са приоритетом која додељује виши приоритет одређеним ресурсима. Подаци са тих ресурса се прослеђују брже од осталих података.

5.1.7 Извештај о стању транспортне везе

Извештај о стању и параметрима је услуга која даје могућност транспортном кориснику да обезбеди или да сам добије информације које се односе на услове транспортне целине или везе.

Примери статусних информација су:

- карактеристике везе (нпр. пропусна моћ, средње кашњење...),
- адресе (мрежна, транспортна),
- класа протокола која је у употреби,
- текуће стање часовника,
- стање протокола (у контексту дијаграма стања),
- деградација у односу на захтевани степен услуге.

5.1.8 Сигурност на транспортном слоју

Транспортна целина може да обезбеди различите услуге везане за сигурност као што су:

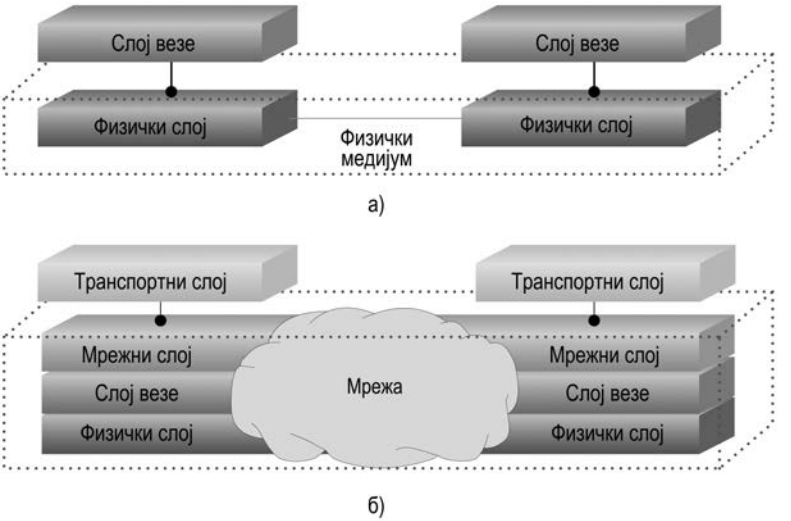
- управљање приступом које се реализује на локалном и удаљеном крају,
- шифровање/дешифровање података,
- рутирање преко поузданих веза и чворова уколико су на располагању.

5.2 Елементи транспортног протокола

Транспортна услуга имеђу две транспортне целине реализује се помоћу транспортног протокола.Транспортни протокол сличан је протоколу слоја везе. Код оба протокола, поред осталог, имплементирани су механизми за контролу грешке, секвенционирање и контролу тока. Постоје и значајне разлике које ћемо објаснити у овом поглављу (слика 5.3).

Разлике које су на први поглед видљиве су следеће:

- два слоја везе комуницирају преко физичког канала док транспортни слојеви комуницирају преко подмрежа,
- успостављање везе преко физичког канала је једноставно, док је успостављање везе преко подмрежа знатно сложеније,
- подмрежа може да задржава податке и да их испоручи са значајним кашњењем,
- контрола тока и додељивање меморијског простора за чување пакета (баферовање) присутно је и код транспортног и код слоја везе. На слоју везе обично се резервише меморијски простор одређене величине за сваку од веза. Када рам стигне,



Слика 5.3 Аналогија протокола: а) Слоја везе б) Транспортног слоја

меморијски простор за његово смештање је расположив. Пошто на транспортном слоју постоји велики број веза које треба услужити принцип са наменским меморијским простором је практично неприхватљив.

Сложеност транспортног протокола зависи у значајној мери од мрежне услуге. ISO је дефинисао три типа мрежна услуга:

- тип А за мрежне везе са прихватљивим степеном погрешних пакета и прихватљивим степеном сигнализираних грешака,
- тип В за мрежне везе са прихватљивим степеном погрешних пакета и неприхватљивим степеном сигнализираних грешака,
- тип С за мрежне везе са неприхватљивим степеном погрешних пакета.

Под погрешним пакетима подразумевају се изгубљени или дуплицирани пакети – (NPDU[1]). Уколико грешку уочи и исправи мрежна целина на такав начин да је невидљива за транспортног корисника проблем је успешно разрешен. Уколико мрежна целина детектује грешку, не може да је отклони него обавести (сигнализира) транспортну целину,

[1] *Network Protocol Data Unit* – јединице података мрежног слоја

онда се то назива сигнализирана грешка. У овом поглављу анализираћемо транспортни протокол за случај поуздане мрежне услуге и за случај непоуздане мрежне услуге.

5.3 Транспортни протокол и поуздана мрежна услуга

Претпоставка је да мрежна услуга прихвата поруке произвољне дужине и да их са вероватноћом од 100% испоручује у правилном редоследу. Ова претпоставка доприноси примени једноставног транспортног протокола. Примери комуникационих мрежа које обезбеђују овакву услугу су:

- X.25 мрежа са комутацијом пакета високе поузданости,
- мрежа са штафетним преносом која користи LAPF[1] протокол,
- локалне рачунарске мреже по стандарду IEEE802.3 и LLC услугом са успоставом везе.

Анализираћемо:

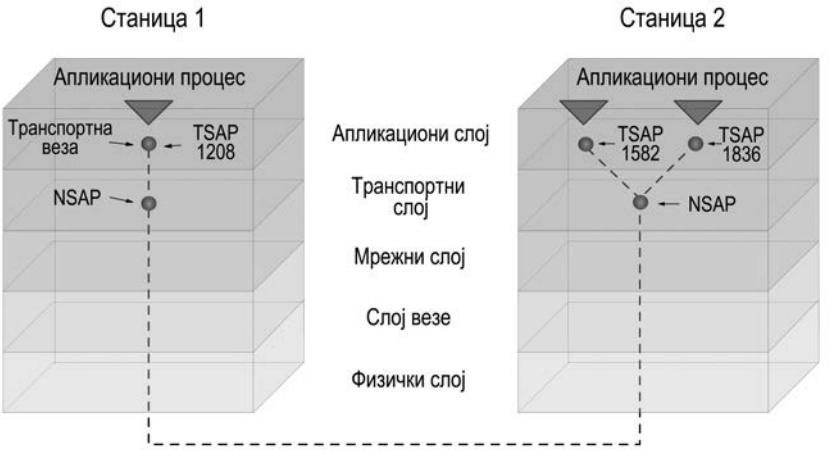
- адресирање,
- мултиплексирање,
- успоставу и раскидање везе и
- контролу тока.

5.3.1 Адресирање на транспортном слоју

Када апликациони процес (тј. корисник) жели да успостави везу са другим апликационим процесом потребно је да означи са којим то процесом, тј. потребно је да дефинише његову транспортну адресу. Термин који се користи[2] је тачка приступа транспортној услузи (TSAP[3]). На Интернету се ове крајње тачке -TSAP називају портови. На мрежном слоју се ове тачке (адресе мрежног слоја) називају тачке приступа мрежној услузи (NSAP[4]). Пример тачке приступа мрежној услузи (NSAP) су IP адресе. На слици 5.4 приказана је веза између тачака приступа транспортној и мрежној услузи и транспортне везе.

Апликациони процеси на страни клијента и сервера могу да се прикључе на тачку приступа транспортној услузи да би успоставили

[1] *Link Access Procedure for Frame Mode Bearer Services*
[2] Дефинисан је у 1. поглављу овог уџбеника.
[3] *Transport Service Access Point*
[4] *Network Service Access Point*



Слика 5.4 Транспортна веза, TSAP и NSAP

везу са удаљеном тачком приступа транспортној услузи. Ове везе реализују се преко тачака приступа мрежној услузи на страни сваке крајње станице (хоста).

Значи адреса корисника је одређена паром (станица, порт). Порт је променљив и представља одређеног корисника. Идентификација транспортне целине није потребна пошто обично постоји само једна целина за одређени тип протокола. Касније ћемо видети да адреса треба да укључи тип транспортног протокола (нпр. TCP, UDP).

У случају једне, одвојене, мреже станица представља уређај повезан на рачунарску мрежу. Када је реч о Интернету станица представља глобалну интернет адресу.

Рутирање није у надлежности транспортног слоја, тако да он само прослеђује део адресе која се односи на станицу мрежној целини. Порт се налази у транспортном заглављу и на одредишту га може употребити одредишна транспортна целина.

Поставља се питање како иницијатор транспортне везе зна адресу одредишног транспортног корисника? Одговор се може категоризовати на следећи начин:

- Корисник зна унапред одредишну адресу. У основи то је функција конфигурисања система.

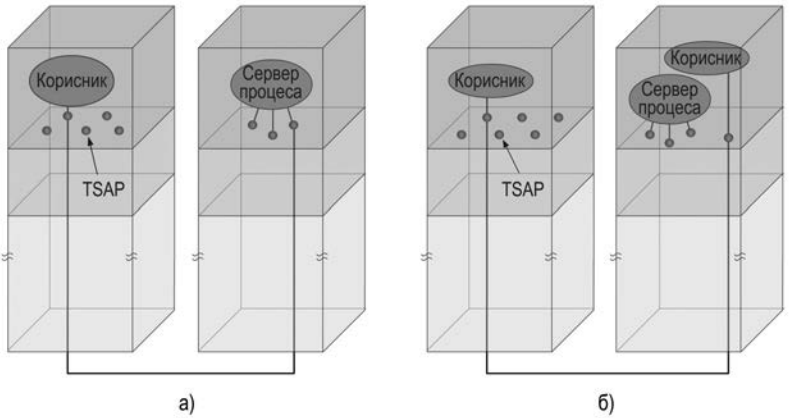
- Неке од општекоришћених услуга проглашене су „познатим адресама”[1] (табела 5.2). На пример, то су серверске стране протокола FTP, SMTP и неки други стандардни протоколи у скупу TCP/IP протокола.
- Обезбеђени су сервери имена. Корисник захтева услугу користећи неко генеричко или глобално име. Захтев се шаље неком од сервера имена који садржи базу адреса. На основу имена проналази тражену адресу и шаље је кориснику. Добивши тражену адресу транспортна целина успоставља везу са одредиштем. Ова услуга је погодна за апликације које се често користе али повремено мењају своју локацију. На пример неки процес може се пребацити са једног рачунара на други у оквиру локалне рачунарске мреже да би се обезбедило равномерно оптерећење.
- У неким случајевима одредишни корисник је процес који се по захтеву треба покренути, односно изазвати[2]. Корисник иницијатор шаље захтев ка познатим адресама. Процес – сервер на тим адресама покреће нове процесе и шаље иницијатору адресу новопокренутог процеса. На слици 5.5а приказана је комуникација са сервером процеса, на сл. 5.5б комуникација са траженим корисником (изазваним процесом).

Порт	Протокол	Намена
21	FTP	Пренос фајлова
23	Telnet	Виртуелни терминал
25	SMTP	Електронска пошта
80	HTTP	Веб
110	POP-3	Удаљени приступ електронске поште
53	DNS	Сервер имена

Табела 5.2 Најчешће коришћени портови у скупу TCP/IP протокола

[1] *Well-known address.* Листа се може наћи на *www.iana.org.*

[2] *Spawned*



Слика 5.5 Како корисник успоставља везу са:
а) Сервером имена (процеса) б) Покренутим процесом

5.3.2 Мултиплексирање на транспортном слоју

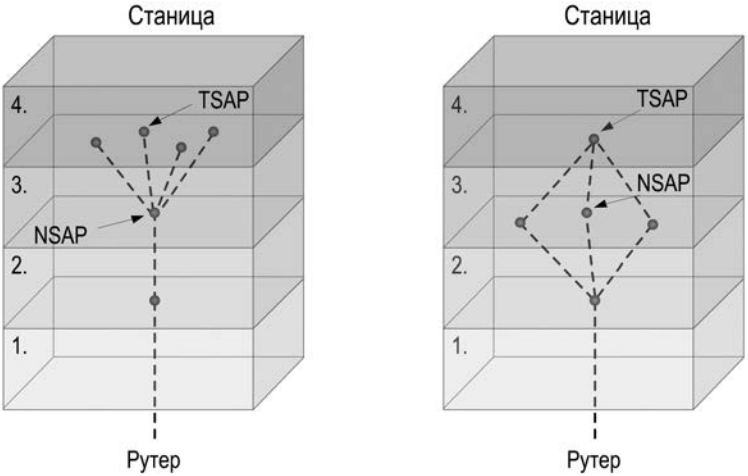
На транспортном слоју мултиплексирање је приказано на слици 5.6 и дефинишемо га као:

- мултиплексирање нагоре[1] уколико више транспортних корисника дели једну виртуелну везу успостављену између мрежних целина,
- мултиплексирање надоле[2] уколико једна транспортна веза користи више веза на мрежном слоју.

Навешћемо ситуације у којима се мултиплексирање користи. Узмимо за пример да је станици (хосту) на располагању само једна мрежна адреса тако да све транспортне везе на тој станици морају ту адресу да користе.

Претпоставимо да подмрежа користи виртуелне везе и намеће ограничење у максималној брзини података по једној вези. Уколико је кориснику потребна већа брзина од брзине једног виртуелног кола може да отвори више мрежних веза и подели саобраћај између њих. Уобичајени пример мултиплексирања надоле је коришћење две ISDN везе од по 64kb/s за повезивање са посредником интернет услуга брзином од 128 kb/s.

[1] *Upward multiplexing*
[2] *Downward multiplexing*



Слика 5.6 Мултиплексирање: а) Нагоре б) Надоле

5.3.3 Контрола тока на транспортном слоју

Ово је једноставан механизам на слоју везе а сложен механизам на транспортном слоју из следећих разлога:

- контрола тока на транспортном слоју укључује међусобну интеракцију транспортних корисника, транспортних целина и услуга мрежног слоја;
- кашњење у преносу између транспортних целина може бити велико. То значи да постоји прилично кашњење у размени информација везаних за контролу тока;
- кашњење у преносу пакета између транспортних целина је у општем случају знатно дуже од времена потребног за његово слање;
- с обзиром на то да транспортни слој функционише преко нижих слојева, кашњење може да буде веома променљиво. Ово доприноси томе да је прилично отежано коришћење часовника за поновно слање изгубљених података.

Основна сличност контроле тока на слоју везе (на пример код HDLC протокола), на мрежном слоју (на пример X.25 протокол) и на транспортном

слоју (на пример TCP протокол) настаје из потребе да се спречи брзи пошиљалац да загуши спори пријемник. Код слоја везе пошиљалац чува рамове (баферује их) и на пријемној и на предајној страни. Када подмрежа обезбеђује непоуздану услугу транспортна целина - пошиљалац чува све јединице података (TPDU[1]) док не добије потврду за њих. Пријемна транспортна целина може на пример да резервише меморијски простор[2] за све везе заједно а не за сваку појединачно.

Свака транспортна целина има одређену количину простора за чување пристиглих података, односно величину бафера. Јединице података (у даљем тексту сегменти[3]) који пристижу смештају се у бафер. Сваки сегмент се анализира, нпр. TPDU заглавље, и по завршетку анализе шаље се одређеном транспортном кориснику. Ако пријемна транспортна целина не може да одржи корак са дотоком сегмената, бафери бивају препуњени[4] и може да дође до губитка података. Да би се спречила појава „препуњених бафера", пријемна транспортна целина мора да предузме кораке за заустављање или успоравање дотока сегмената да би могла да обради већ пристигле сегменте. Овај задатак није једноставно реализовати с обзиром на могућу удаљеност (временску и просторну) транспортних целина. Изложићемо четири начина за разрешавање захтева који се постављају пред механизмом за контролу тока. Пријемна транспортна целина може:

- да не реагује ни на који начин,
- да одбије да надаље прима сегменте (TPDU) од мрежног слоја,
- да користи протокол клизајућег прозора са фиксним отвором прозора,
- да користи механизам кредита[5].

Прва варијанта значи да ће пријемна целина одбацити пакете у недостатку простора за њихово чување. Пошто не добије потврду за њих, пошиљалац их поново шаље. Овај приступ није пожељно користити, јер

---

[1] *Transport Protocol Data Unit*

[2] *Buffer pool*

[3] Јединице података PDU (*Protocol Data Unit*) које међусобно размењују транспортне целине називају се сегменти.

[4] *Buffer overflow*

[5] Може се сматрати као променљиви отвор прозора.

долази до ретрансмисије, а предност мреже са успоставом везе је та да не би требало да постоји ретрансмисија (или бар да се сведе на минимум). Друга варијанта је она која је примењена код протокола HDLC[1]. Кључни параметри су:

- свака јединица података добија редни број,
- отвор прозора је фиксан,
- користи се потврда која доводи до померања (клизања) прозора.

Са поузданом услугом мрежног слоја техника клизајућег прозора ради сасвим добро.

У трећој варијанти пријемна транспортна целина поред потврде о пакетима које је успешно примила шаље нови параметар - кредит. Он представља отвор прозора али може да се мења у току трајања везе.

5.3.4 Успостава и завршетак транспортне везе

Да бисмо боље објаснили како се операције (примитиве) могу употребити у фазама успоставе и раскида везе посматраћемо апликацију са сервером и већим бројем удаљених клијената. У табели 5.3 дат је преглед примитива и њихово значење.

На почетку сервер извршава примитиву *Passive open* обично системским позивом који блокира сервер док се не појави захтев клијента.

Када клијент жели да комуницира са сервером извршава примитиву

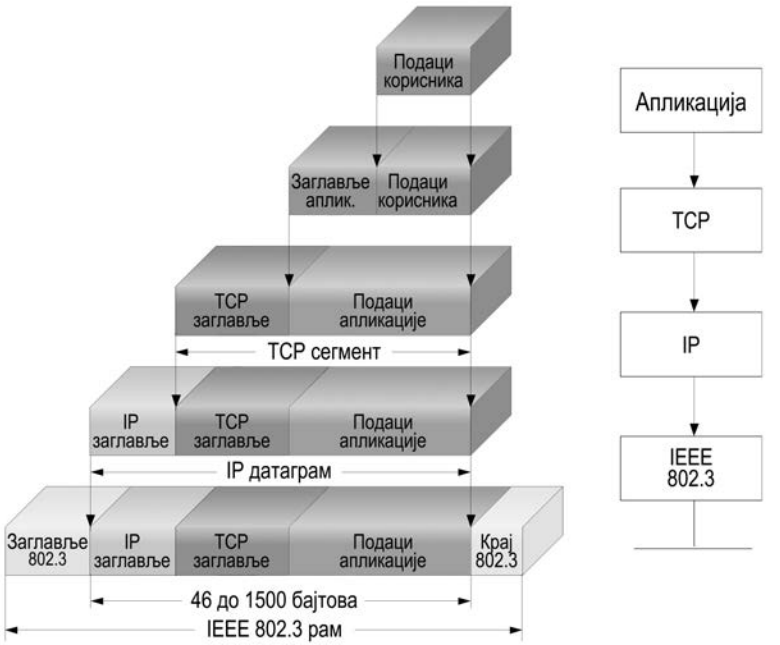
Примитива	Послата транспортна јединица података (TPDU)	Значење
Passive open	Не шаље се ништа	Блокиран је док неки од процеса не покуша да се повеже са њим
Active open	Шаље захтев за успоставу везе	Активно покушава да успостави везу
Send	Подаци	Шаљу се информације
Receive	Не шаље се ништа	Блокиран је док не пристигну *Data* пакети.
Close	Шаље захтев за раскид везе	Ова страна жели да оконча везу
53	DNS	Сервер имена

Табела 5.3 Примитиве и њихово значење

[1] Детаљно је описано је у 11. поглављу уџбеника [1].

Active open. Транспортна целина извршава примитиву тако што блокира позивајућег корисника и пошаље серверу јединицу података „захтев за успоставу везе"[1]. Порука транспортног слоја клијента упућена транспортном слоју сервера спакована (учаурена)[2] је у део за податке[3] пакета мрежног слоја. Пакет мрежног слоја спакован је у рам слоја везе. Када рам дође до сервера одвија се обрнут процес - распакивање на сваком слоју. Повезивање јединица података код TCP/IP референтног модела приказано је на слици 5.7.

За реализацију услуге са успоставом везе неопходне су процедуре за успоставу и завршетак везе.



Слика 5.7 Повезивање јединица података (PDU[4]) на транспортном, мрежном и слоју везе код TCP/IP референтног модела

[1] *Connection Request TPDU*
[2] *Encapsulated*
[3] *Payload*
[4] *Protocol Data Unit*

Три битне функције успоставе везе су:

- омогућавање сваком учеснику да се увери да је други учесник присутан,
- дозвољавање договора о опционим параметрима као што су на пример: максимална величина сегмента, максимална величина отвора прозора, квалитет услуге,
- покретање додељивања ресурса потребних транспортној целини као што су меморијски простор за смештање података (бафер) или формирање табела.

Успостава везе је међусобни договор корисника и може се остварити разменом различитих команди и управљачким сегментима, као што је приказано на дијаграму стања на слици 5.8.

На почетку корисник је у неактивном стању - CLOSE[1], нема успостављену ниједну везу. Корисник командом (примитивом) *Passive Open* чека на захтев за успоставом везе. После издавања ове команде транспортна целина ствара (гради) објекат везе (нпр. улаз у табелу) који се налази у стању LISTEN[2]. Ако се транспортни корисник у међувремену предомислио и жели да се врати у CLOSE стање, може то једноставно да реализује извршавањем команде *Close*.

Уколико стигне сегмент *Syn* (Cr[3]) који је адресиран на корисника који је у стању пасивног чекања, веза је отворена и транспортна целина ће:

- сигнализирати кориснику да је веза отворена,
- послати *Syn_Ack* (*Cc*[4]) сегмент као потврду удаљеној транспортној целини,
- поставити објекат везе у стање ESTABLISHED.

Корисник може да успоставља (отвара) везу користећи команду *Active Open*[5] која покреће транспортну целину да пошаље сегмент *Syn*.

---

[1] Указује да је веза затворена. Стања ће бити означена великим словима (нпр. CLOSE), примитиве великим и малим словима (нпр. *Passive Open*) а сегменти које размењују транспортне целине великим и малим словима али италик (нпр. *Syn*).

[2] Указује да је процес на страни сервера у стању ослушкивања позива клијента

[3] TPDU сегмент - захтев за успоставу везе *Cr* (*Connection request*). Код TCP протокола означава се са *Syn*

[4] Користи се и *Cc* (*Connection confirm*). Уобичајено је да се код TCP протокола исти TPDU користи и као захтев за успоставу везе и као потврда о успостави везе и означава као *Syn_ Ack*. Биће детаљније обрађено у 6. поглављу овог уџбеника.

[5] Назива се и захтев за успоставу везе (*Connect*).

Пријем одговарајућег *Syn* сегмента успоставља везу.

Када страна која је иницирала отварање везе прими *Syn_Ack* сегмент, може да пређе у ESTABLISHED стање. У случају да било који транспортни корисник изврши *Close* команду, веза се одмах прекида.

Било која страна може да иницира успостављање везе. У случају да обе стране иницирају везу у исто време, веза се без икаквих проблема успоставља. Разлог је то што се *Syn* сегменти посматрају и као захтеви за успоставу везе и као потврда о успостављеној вези. Веза се превремено прекида када локални корисник пошаље *Close* команду или када удаљена транспортна целина одбије везу слањем транспортне јединице података за окончање везе *Fin_Ack* (*Dr*[1]).

Шта се дешава уколико је корисник који би требало да се налази у стању пасивног чекања (LISTEN) неактиван (CLOSE)? Може да се деси:

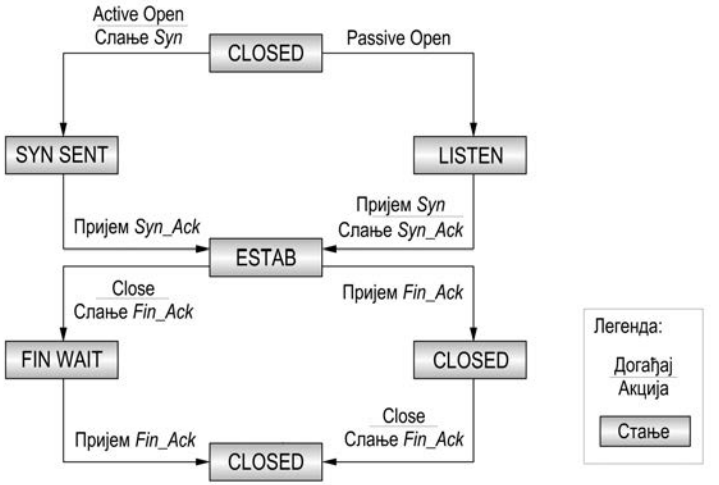
- да транспортни слој одбаци захтев слањем *Fin_Ack* сегемента,
- да се захтев ставља у ред за чекање док корисник не пошаље команду *Active open*,
- да транспортна целина сигнализира кориснику да постоји нерешен захтев за успоставу везе са њим.

У случају да транспортна целина реагује на трећи од наведених начина, Activ open команда није неопходна и може бити замењена командом Accept, која представља сигнал који шаље транспортни корисник транспортној целини да прихвати захтев за успоставу везе.

Раскид везе извршава се на сличан начин. Било која страна, или обе, могу да иницирају окончање, тј. затварање (*Close*). Веза се затвара обостраном сагласношћу.

Постоје два начина да се прекине веза а то су: изненадан прекид, када једна страна покрене иницијативу за прекид везе не сачекавши да то друга страна одобри, или прекид везе који настаје као резултат договора обе стране. Ако се веза изненада прекине, подаци који се у том тренутку налазе на путу између две станице могу бити изгубљени. Прекид везе уз међусобни договор обавезује обе стране да не прекидају везу док се сви подаци не испоруче. Да би се то постигло, веза у стању FIN WAIT треба да настави да прима податке све док не добије *Fin_Ack* сегмент.

[1] Користи се *Dr* (*Disconnect request*) а код TCP протокола означава се са *Fin_Ack*



Слика 5.8 Поједностављен дијаграм стања везе

Слика 5.8 илуструје процедуру правилног прекида везе. Прво је неопходно уочити страну која иницира прекид везе:

- Као одговор на примитиву *Close* корисника, транспортна целина шаље *Fin_Ack* сегмент другој страни, захтевајући прекид везе;
- Када пошаље *Fin_Ack* сегмент, транспортна целина ставља везу у FIN WAIT стање. Када је веза у овом стању, транспортна целина мора да настави да прихвата податке који стижу са друге стране и да испоручи податке кориснику;
- Када *Fin_Ack* сегмент стигне као одговор транспортна целина затвара везу прелази у стање CLOSE и о томе обавештава корисника.

Са тачке гледишта стране која није иницирала прекид, ситуација је следећа:

- Када страна која није иницирала прекид прими *Fin_Ack* сегмент, транспортна целина информише корисника да је иницран прекид везе и ставља везу у стање CLOSE WAIT. Када се веза налази у овом стању, транспортна целина мора да настави да прима податке од свог корисника и да их прослеђује ка другој страни;

- Када корисник изврши *Close* примитиву, транспортна целина шаље одговарајући *Fin_Ack* сегмент другој страни и затвара везу.

Ова процедура обезбеђује да се обе стране сложе око затварња везе и да приме податке који се налазе на линији, пре самог затварања везе, да не би дошло до губљења података.

5.4 Транспортни протокол и непоуздана мрежна услуга

Сложенији случај транспортног протокола је када је услуга мрежног слоја непоуздана. Примери мрежа које користе овакве услуге су:

- мреже које користе интернет протокол (IP протокол),
- мреже са штафетним преносом рамова (FR) које користе само LAPF протокол,
- локалне рачунарске мреже етернет типа[1] које на слоју везе користе услугу без успоставе везе и без потврде.

Проблеми су:

- повремени губитак сегмената,
- њихово пристизање ван послатог редоследа као последица променљивог кашњења кроз мрежу.

Потребно је анализирати:

- стратегију поновног слања,
- откривање (детекцију) дуплих пакета,
- контролу тока,
- успоставу везе,
- окончање везе,
- опоравак после испада из рада (отказа).

5.4.1 Стратегија поновног слања сегмената

Два догађаја указују да је потребна стратегија за поновно слање сегмената. Прво, сегмент може да буде оштећен при проласку кроз мрежу али и поред тога да стигне на одредиште. Уколико сегмент садржи контролу грешке пријемна транспортна целина може да детектује грешку и одбаци сегмент.

[1] Стандард IEEE802.3

Друго, сегмент може да не стигне на одредиште. Транспортна целина која је послала сегмент не зна да је слање неуспешно. Да би се овакви проблеми разрешили уводи се потврда (*Ack* сегмент). Пријемник мора да потврди сваки успешно примљени сегмент. Због ефикасности код слања података не користи се потврда за сваки примљени сегмент већ кумулативна потврда – један *Ack* сегмент за више примљених сегмената. Значи, на пријемној страни стижу сегменти података са редним бројевима 0, 1 и 2. Као потвда о успешном пријему натраг се шаље само сегмент *Ack3*. То значи да пријемник очекује да као први следећи сегмент података добије сегмент са редним бројем 3.

Уколико сегмент није успешно примљен, неће бити послата потврда и извориште ће поново послати непотврђен сегмент. Поновно слање (ретрансмисија) може бити селективно (поново се шаљу само неисправни пакети) или се шаљу сви пакети, од неисправног па до оних који су послати до пријема негативне потврде[1].

За свако слање сегмената мора да постоји часовник[2] који се поставља на одређено време[3]. Ако то време истекне, а сегмент није потврђен пошиљалац мора поново да шаље пакет.

Увођење часовника донекле разрешава проблем. Поставља се питање на коју вредност треба поставити часовник? Уколико је вредност превише мала постојаће много беспотребних ретрансмисија. Уколико је вредност превелика протокол ће бити инертан у реаговању на изгубљене сегменте. Часовник треба да буде постављен на вредност мало већу од укупног кашњења кроз мрежу[4] које је једнако збиру времена потребног да сегмент стигне на одредиште и да се врати његова потврда. Кашњење је променљиво и код мрежа са константним саобраћајем а нарочито код мрежа променљивих карактеристика.

Постоје два метода: са тачном (фиксном) вредношћу и са променљивом (адаптивном, самоподешавајућом) вредношћу. Тачно одређена вредност на коју се часовник подешава заснована је на

---

[1] Негативне потврде (*NACK- Negative ACKnowledge*) или поновљена потврда убрзавају поступак ретрансмисије – не чека се истицање времена на које се поставља часовник.

[2] *Timer*

[3] *Time out*

[4] *Round trip delay*

Часовник	Намена
За ретрансмисију[1]	Поново шаље сегмент који није потврђен
За поновну успоставу везе[2]	Минимално време између затварања једне везе и отварања друге везе са истом одредишном адресом
За отвор клизајућег прозора[3]	Максимално време узмеђу два *Ack* сегмента
За ретрансмисију *Syn* сегмента[4]	Време између два покушаја да се веза успостави (отвори)
За прекид[5]	Прекида везу пошто ниједан сегмент није потврђен

Табела 5.4 Часовници који се користе на транспортном слоју

уобичајеном стању мреже. Проблем са овим методом је што не одговара мрежама чије се стање мења. Уколико је вредност превише велика услуга ће увек бити инертна. Уколико је вредност превише ниска јавља се позитивна повратна спрега – загушење мреже које доводи до више ретрансмисија које повећавају саобраћај у мрежи и још веће загушење.

Адаптивна шема има својих ограничења. Претпоставимо да транспортни слој прати колико времена је потребно да би се потврдио сегмент и подешава часовник према праћеној усредњеној вредности. Може се десити да:

- одредиште не шаље потврду одмах (нпр. користи се кумулативна потврда),
- сегмент се шаље поново али пошиљалац не зна да ли је примљена потврда (*Ack*) за првобитно слање или за ретрансмисију,
- услови у мрежи могу се нагло променити.

Сваки од ових проблема се може разрешити на неки начин али свако од решења оставља одређени степен неодређености. Часовник за ретрансмисију веома је важан за исправан рад транспортног протокола. У табели 5.4 описани су сви часовници које транспортни протокол користи.

[1] *Retransmission timer*
[2] *Reconnection timer*
[3] *Window timer*
[4] *Retransmit Syn timer*
[5] *Give up timer*

5.4.2 Откривање дупликата транспортне везе

Уколико се деси да је сегмент изгубљен и поново послат не може да дође до грешке. Уколико је *Ack* изгубљен један или више сегмената биће поново послати и уколико стигну успешно биће дупликати претходно послатих сегмената. Потребно је пронаћи начин да пријемник препозна дупликате. Постоје два случаја:

- дупликат је примљен пре затварања везе,
- дупликат је примљен по затварању везе.

Може да се деси да дупликат стигне пре оригиналног сегмента. Уколико се деси да дупликат стигне пре затварања везе онда:

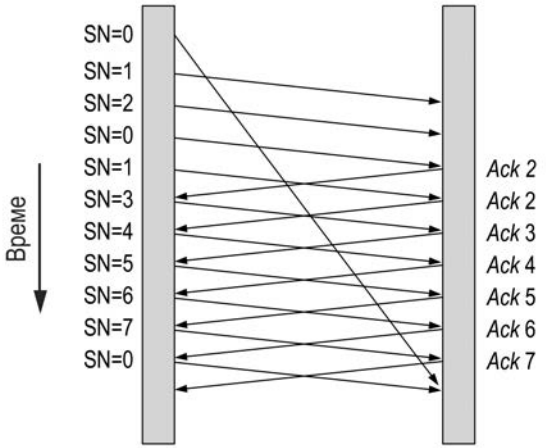
- пријемник треба да претпостави да је потврда изгубљена и да је потребно да пошаље потврду за дупликат,
- простор редних бројева (број редних бројева[1]) треба подесити да је довољно велики.

Слика 5.9 приказује разлог за захтеве који следе. У овом примеру простор редних бројева је 8. Због једноставности претпостављамо да је отвор клизајућег прозора 3. Нека је станица 1 послала сегменте са редним бројевима 0, 1, 2. Станица 2 прима сегменте 1 и 2 али је при проласку кроз мрежу сегмент 0 закашњен. Због тога станица 2 не шаље потврду о њиховом успешном пријему. Оглашава се часовник за ретрансмисију станице 1, пошто је време на које је подешен часовник истекло. Станица 1 поново шаље сегмент 0. Када станица 2 прими дупликат сегмента 0 шаље потврду (сегмент *Ack3*[2].) о успешном пријему сегмената 0, 1 и 2. У међувремену истекло је и време на које је подешен часовник за ретрансмисију за сегмент 1 па га станица 1 поново шаље. Станица 2 га потврђује новим сегментом *Ack3*. Од тог тренутка на даље веза улази у регуларан ток: пријем сегмената и слање потврде. Када је простор редних бројева потпуно искоришћен (послат је сегмент са редним бројем 7)

---

[1] Поруку коју транспортна целина добија од свог корисника (апликационог слоја или слоја сесије) и коју треба да пошаље она може да подели у мање делове и шаље као посебне сегменте. Сваки од тих делова добија свој број. Ти бројеви се у литератури означавају као редни (секвенцни) бројеви. Означавају се са SN (*Send sequence Number*) и AN (*Acknowledge sequence Number*). Простор редних бројева (укупан број расположивих бројева) зависи од броја битова који се користе. На пример ако је додељено поље за редне бројеве величине 3 бита онда је простор редних бројева $2^3 = 8$. То значи да сегменти могу да буду означени редним бројевима 0,1,...7.

[2] Најчешће потврда указује на редни број првог сегмента који се очекује.



Слика 5.9 Пример погрешне детекције дупликата

станица 1 шаље нови сегмент са редним бројем 0. Али у међувремену, пре него што је стигао нови, стиже стари сегмент са редним бројем 0.

Јасно је да пристизање старог сегмента са редним бројем 0 „у невреме" не би изазвало никакве проблеме да није кренуо нови круг редних бројева. Станица 2 нема начина да зна да је стигао стари сегмент а не нови који очекује. Поставља се питање колико треба да буде велики простор редних бројева? Додавањем само једног бита у поље редних бројева може се удвостручити простор редних бројева. Стандардни транспортни протокол омогућава огроман простор редних бројева.

Анализираћемо други проблем: сегмент кружи мрежом по затварању транспортне везе. Ако је следећа веза отворена између истих транспортних целина сегмент из старе везе може да се појави и да буде прихваћен. Такође закаснела потврда може да се појави у новој транспортној вези и изазове проблеме.

Постоји неколико метода који могу да разреше ове проблеме. Анализираћемо неке од њих. Први метод се састоји у следећем: транспортна целина памти последњи редни број који је употребљен у завршеној транспортној вези. Када се нова веза успоставља *Syn* сегмент садржи редни број са којим треба да се започне пренос података и који се разликује од редних бројева сегмената из претходне везе.

Други приступ је увођење посебног идентификатора транспортне везе и употреба новог идентификатора са сваком новом везом.

Описана процедура ради добро докле год не дође до пада система, па се информација о идентификатору последње везе губи. Алтернатива је да се једноставно сачека довољно дуго између две транспортне везе да би осигурали да су сви сегменти испоручени. Уколико дође до пада система на једној страни друга страна може да одбије успостављање везе док време на часовнику за поновну успоставу везе не истекне. Ово наравно може да изазове штетна кашњења.

5.4.3 Успостављање транспортне везе

Успостављање транспортне везе као и остали механизми протокола треба да узму у обзир непоуздану услугу мрежног слоја. Размена *Syn*[1] сегмената некада се назива двострана процедура[2]. Претпоставимо да станица 1 шаље Syn сегмент ка станици 2. Очекује да добије натраг потврду да је веза успостављена. До грешке може да дође ако се деси следеће: сегмент *Syn* који шаље станица 1 је изгубљен, или је сегмент *Syn_Ack* који као одговор шаље станица 2 изгубљен. Оба случаја могу се превазићи часовницима за поновно слање (ретрансмисију).

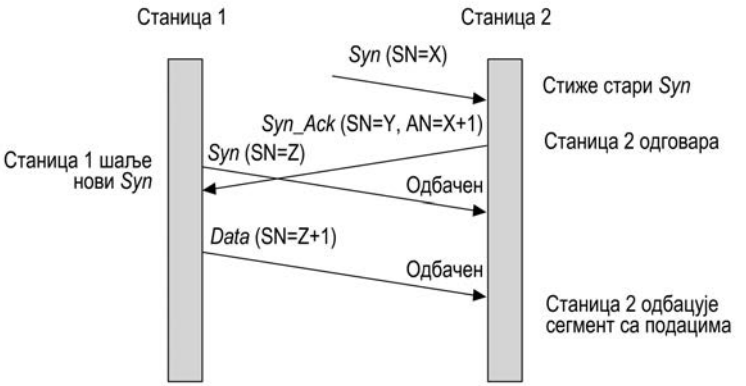
Поновна слања могу довести до дуплирања *Syn* сегмената. Када је веза успостављена станице 1 и 2 могу једноставно да игноришу дупликате.

Шта се дешава уколико дупликати *Syn* сегмента опстану по завршетку везе? На слици 5.10 приказан је проблем који може да се појави. Стари *Syn* сегмент (захтев за успоставу везе, редни број почиње са *X*) стиже до станице 2 пошто је веза окончана. Станица 2 претпоставља да је ово нови захтев и одговара са *Syn_Ack* (*SN=Y, AN=X+1*) (редни број почиње са *Y*). У међувремену станица 1 је одлучила да успостави нову везу са станицом 2 и шаље *Syn* (SN=Z). Станица 2 га одбацује. Станица 1 започиње пренос података сегментом са редним бројем *Dat a* (SN=Z+1). Станица 2 га одбацује као пакет ван очекиваног опсега редних бројева.

---

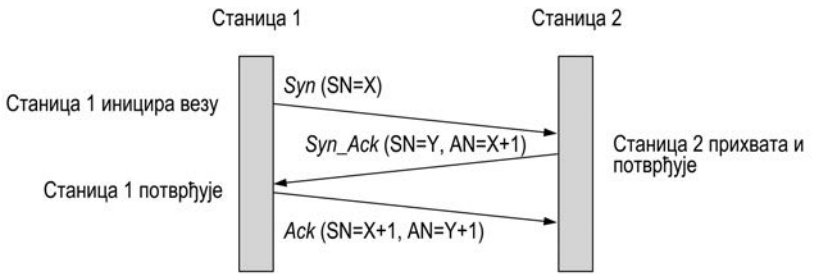
[1] Код TCP протокола означава се са *Syn* сегмент што значи да је постављен SYN бит у заглављу TCP сегмента, *Rst* сегмент значи да је постављен RST бит у заглављу TCP сегмента, *Fin* сегмент што значи да је постављен FIN бит у заглављу TCP сегмента (детаљније у 6. поглављу овог уџбеника). У литератури нпр. [13] користи се и ознака CR TPDU (*Connection Request Transport Protocol Data Unit*).

[2] *Two-way handshake*



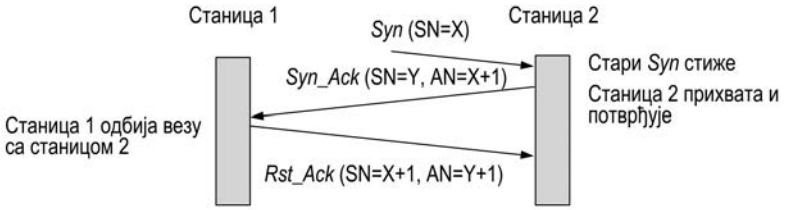
Слика 5.10 Пример двостране процедуре транспортне везе

Начин за превазилажење овог проблема је да свака од страна потврди пријем *Syn* и редног броја друге стране. Процедура се назива тространa[1] (или трострука, троделна, у три корака) а слике 5.11, 5.12 и 5.13 илуструју типични начин рада са тространом процедуром. У нормалним околностима (слика 5.11) станица 1 шаље *Syn* (*SN=X*) који садржи предајни редни број X. Ова вредност представља иницијални редни број станице 1. Станица 2 у *Syn_Ack* (*SN=Y, AN=X+1*) сегменту који шаље као одговор садржи потврду примљеног редног броја увећаног за један и свој иницијални редни број Y. Станица 1 првим сегментом података потврђује пријем *Syn_Ack* сегмента станице 2. На слици 5.12 приказана је ситуација у којој станици 2 стиже стари *Syn*

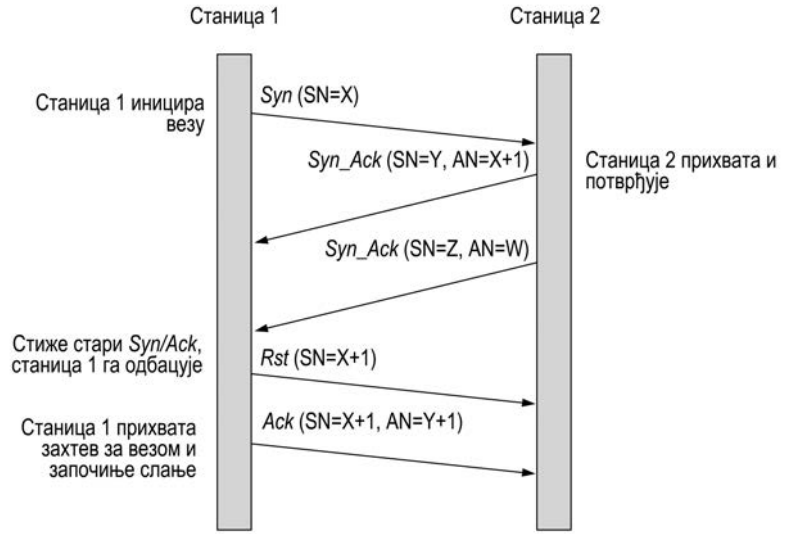


Слика 5.11 Нормалан начин рада

[1] *Three - way handshake*



Слика 5.12 Пример пристизања старог *Syn* сегмента после окончања везе



Слика 5.13 Пример са кашњењем *Syn_Ack* сегмената

(*SN=X*) после окончања везе. Станица 2 претпоставља да је ово захтев за успоставом нове везе и одговара са *Syn_Ack* (*SN=Y, AN=X+1*). Када станица 1 добије овај сегмент, и види да није захтевала ову везу, шаље сегмент *Rst_Ack* (*SN=X+1, AN=Y+1*) којим указује на нерегуларност. Пример на слици 5.13 указује на случај када стари *Syn_Ack* стиже у току успоставе нове везе. Због употребе редних бројева у потврди овај догађај неће изазвати никакав проблем.

5.4.4 Окончање транспортне везе

Као и у случају успоставе везе процедура за окончање везе мора да се бави старим и изгубљеним управљачким сегментима. Да би се

спречила неусаглашеност користе се редни бројеви на следећи начин:

- *Fin_Ack* сегмент садржи редни број последњег сегмента увећан за један,
- *Ack* сегмент садржи редни број примљен у *Fin_Ack* сегменту.

Процедуре које се покрећу када истекне време на часовницима користе се за затварање везе када друга страна везе не ради исправно.

5.4.5 Опоравак транспортне везе после отказа[1]

Када систем на коме ради посматрана транспортна целина престане са радом (падне) и после се поново покрене, информације о стању свих веза су изгубљене. Везе се сматрају полуотвореним пошто друга страна (код које није дошло до грешке) није још сагледала проблем.

Још увек активна страна полуотворене везе може да је затвори користећи часовник за прекид[2]. Часовник мери време које истекне после максималног броја ретрансмисија. По истицању тог времена транспортна целина претпоставља да је друга транспортна целина у квару и обавештава транспортног корисника о нерегуларном окончању везе.

У случају да је транспортна целина кратко време у квару па убрзо поново почне да ради, полуотворена веза може се окончати употребом *Rst* сегмента.

5.5 Контрола тока на транспортном слоју

Контрола тока је на слоју везе једноставан механизам док је на транспортном слоју знатно сложенији[3]. Два разлога доводе до тога да једна транспортна целина ограничава брзину преноса сегмената ка другој транспортној целини:

- корисник пријемне транспортне целине не може да подржи брзину пристизања података.
- пријемна транспортна целина не може да подржи брзину пристизања сегмената.

Постоји више начина које може да примени транспортна целина. У

---

[1] *Crash recovery*

[2] *Give up timer*

[3] Објашњено у поглављу о транспортном слоју TCP/IP референтног модела (6. поглавље овог уџбеника).

овом делу анализираћемо једну од њих - механизам кредитирања који се користи код TCP протокола. Механизам кредитирања користи:

- редне бројеве,
- отвор (ширина) прозора за слање,
- потврду о пријему пакета која доприноси повећању поузданости и служи за подешавање величине прозора.

Код механизма кредитирања сваки индивидуални бајт података има јединствен редни број и као такав се преноси. Претпоставка је да је реч и истовременом, двосмерном преносу података (потпуни дуплекс). У том случају сегменти података који се шаљу у једном смеру могу се користити за потврду о успешном пријему података супротног смера. Такође, сваки сегмент који се шаље у свом заглављу има три поља која се односе на контролу тока:

- редни број означен са *SN*,
- број потврде означен са *AN*,
- отвор прозора или кредит означен са *W*.

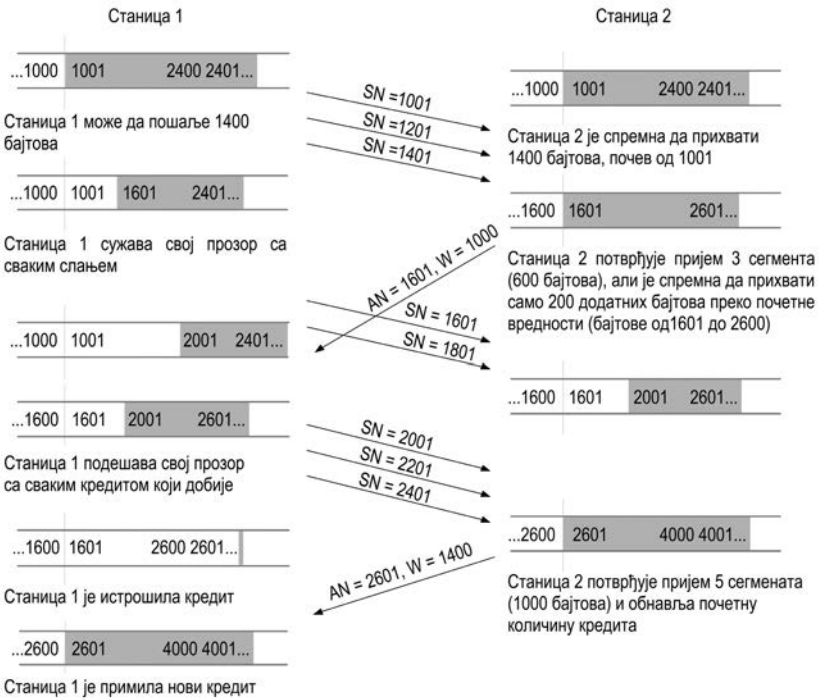
Редни број *SN* означава редни број сегмента датог смера, број потврде *AN* односи се на супротни смер Када транспортна целина шаље сегмент, она у заглавље уписује редни број првог бајта (октета). Пријемна транспортна целина потврђује пријем тако што шаље сегмент код кога су постављене вредности $SN = k$, $AN = I$ што значи:

- сви бајтови закључно са бројем $i-1$ су примљени, и очекује се сегмент чији је први бајт означен редним бројем $SN = i$.
- дозвољава се предајнику да је његов отвор прозора величине $Z = j$ бајтова.

Очекује се да ће редни бројеви бити у опсегу од $SN = i$ до $SN = i + j - 1$.

Слика 5.14 илуструје овај механизам. Због једноставности приказан је проток података у једном смеру и претпостављена је величина сегмента од 200 бајтова. У току трајања процеса успоставе везе пријемни и предајни редни бројеви су синхронизовани.

Рачунар (станица 1) добио је кредит од 1400 бајтова, почев од бајта са редним бројем 1001. После слања 600 бајтова, у по три сегмента, станица 1 сужава свој прозор на 800 бајтова (редни бројеви од 1601 до 2400). Када станица 2 прими ова три сегмента њој остаје кредит од 800 бајтова. Претпоставимо да станица 2 сада може да

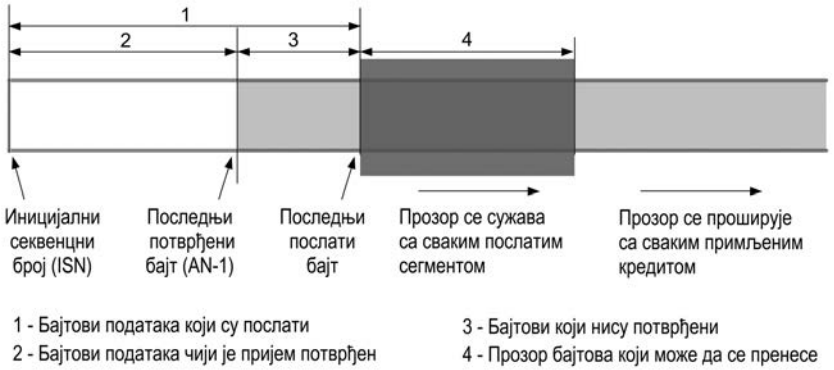


Слика 5.14 Пример механизма за доделу кредита код TCP протокола

прихвати 1000 бајтова са ове везе. Према томе, станица 2 је потврдила пријем свих бајтова до редног броја 1600 и послала кредит од 1000 бајтова. То значи да станица 1 може да пошаље бајтове од 1601 до 2600 (5 сегмената, 1 сегмент = 200 бајтова). За време за које је сегмент станице 2 са додатним кредитом стигао до станице 1, станица 1 је већ послала два сегмента са редним бројевима од 1601 до 2000 (почетна величина прозора). Према томе, станици 1 може да пошаље 600 бајтова (3 сегмента, са редним бројевима од 2000 до 2600).

У току размене података станица 1 смањује величину свог прозора сваки пут када пошаље сегмент, а повећава отвор свог прозора када прими сегмент са додатним кредитом.

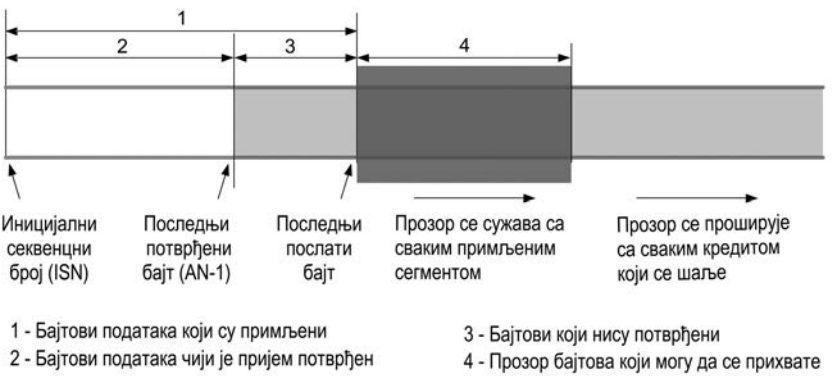
Пријемник је у обавези да усвоји неки скуп правила која ће одређивати колико података пошиљалац може да пошаље примаоцу. Један приступ био би да пошиљалац прима нове сегменте само до



Слика 5.15 Контрола тока при слању сегмената

тренутка док они не попуне меморијски простор (бафер). Ако овај приступ применимо на пример приказан на слици 5.14 онда би први сегмент са кредитом коју пошаље станица 2 указивао да она има на располагању слободан меморијски простор за 1000 бајтова, а други за 1400 бајтова. Слика 5.15 приказује како се механизам клизајућег прозора реализује на предајној а слика 5.16 пријемној страни.

У случају да је пошиљалац бржи од примаоца неке сегменте прималац ће одбацити, што ће довести до потребе за поновним слањем (ретрансмисијом). У комуникационој мрежи није пожељно поновно слање сегмената.



Слика 5.16 Контрола тока на пријему

5.6 Питања и задаци

1. Који је најважнији циљ транспортног протокола?
2. У коју категорију протокола спада транспортни протокол?
3. Зашто постоје и користе се два различита слоја: мрежни и танспортни?
4. Набројати и описати услуге које пружа транспортни слој?
5. Колико категорија примитива транспортних услуга постоји? Описати њихову намену и редослед извршавања.
6. Описати сличност и разлику транспортног слоја и слоја везе.
7. Како се означавају приступне тачке услуге на транспортном и мрежном слоју? Како се оне означавају на Интернету?
8. Који су све начини на које корисник, иницијатор транспортне везе може да сазна адресу одредишног транспортног корисника?
9. Описати мултиплексирање нагоре и мултиплексирање надоле.
10. Како се може описати намена прикључнице.
11. Који портови се означавају као добро познати портови? Навести неке који се често користе.
12. Који све фактори утичу на то да је механизам контроле тока сложенији на транспортном слоју у односу на слој везе.
13. Који се проблеми могу јавити када транспортна целина користи непоуздану мрежну услугу?
14. Објаснити стратегију поновног слања сегмената.
15. Анализирати рад часовника који се користе код поновног слања сегмената.
16. Каква је разлика између тачне вредности и променљиве адаптивне вредности на коју се часовници постављају?
17. Објаснити на који начин пријемник може да детектује дупликате (за објашњење користити слику 5.9).
18. Објаснити шта се дешава када *Syn* сегмент стигне са кашњењем:
    а) после окончања везе,
    б) у току трајања везе.
19. Како се користи сегмент *Rst_Ack*?
20. Уколико закашњени *Syn* сегмент има редни број SN=2002 који редни број ће бити у сегменту *Rst_Ack?*
21. Који редни број ће имати *Fin_Ack* и *Ack* сегменти код окончања везе

уколико је последњи сегмент података који је клијент који окончава везу послао имао редни број SN=1641.

22. Када систем на коме ради посматрана транспортна целина престане са радом (падне) и после се поново покрене шта се дешава са информацијама о вези?
23. Када се веза сматра полуотвореном? Како до тога долази?
24. Навести и објаснити разлоге који доводе до тога да једна TCP целина ограничава брзину преноса сегмената ка другој TCP целини.
25. Који се параметри у заглављу TCP сегмента користе у контроли тока?
26. Претпоставити да је величина транспортног сегмента у примеру са слике 5.14 400 бајтова. Описати овакву модификацију примера.
27. Израчунати после колико послатих сегмената а код величине кредита 2000 може да дође до затварања прозора.
28. Претпоставити да је
    а) иницијални редни број предајника ISN =5000,
    б) величина сегмента 1500 бајтова,
    в) расположиви пријемни меморијски простор величине 12000 бајтова,
    г) први сегмент потврде садржи вредности AN=6001 и W=3000,

    Колико сегмената предајник може још да пошаље и колики је расположиви меморијски простор на пријему?
29. Меморијски простор пријемника је 15000 бајтова. Предајник шаље по три сегмента величине 1500 бајтова а пријемник потврђује исправан пријем сва три сегмента и повећава кредит за 4500 бајтова. Како се мења меморијски простор пријемника?
30. Нацртати и објаснити процес слања и пријема сегмената представљених на сликама 5.14 и 5.15 са следећим параметрима везе:
    а) иницијални редни број ISN =2000,
    б) величина сегмента 400 бајтова,
    в) расположиви пријемни меморијски простор је величине 1200 бајтова,
    г) потврђена су три сегмента једним сегментом потврде којим је повећан пријемни меморијски простор за 1600 бајтова.

6. Транспортни слој на Интернету

У овом поглављу приказано је како су принципи транспортног слоја, описани у претходном поглављу, имплементирани у протоколима на Интернету:

- TCP протокол са успоставом везе и
- UDP протокол без успоставе везе.

Протокол TCP је веома сложен протокол пројектован тако да обезбеди поуздан проток бајтова с краја на крај. Сваки рачунар који подржава TCP протокол поседује TCP целину као библиотечку процедуру, кориснички процес, или део језгра оперативног система. У сваком случају обезбеђује проток TCP сегмената и интерфејс до мрежног слоја.

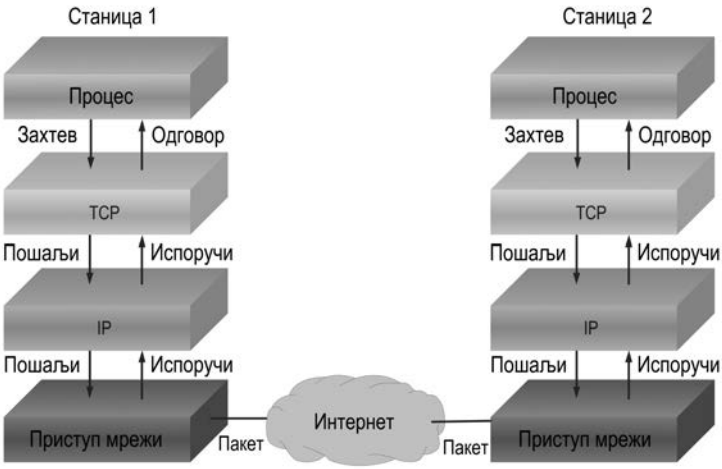
У овом поглављу анализираће се механизам транспортног протокола који се захтева у рачунарским мрежама са непоузданом мрежном услугом као што је Интернет.

6.1 Услуга транспортног слоја

Као што је претходним поглављима поменуто, за сваки од слојева везан је одговарајући скуп примитива и параметара који имају за циљ пренос података и управљачких информација.

Слика 6.1 показује контекст у коме се примитиве TCP услуга користе. Корисници на апликационом слоју размењују са TCP целином примитиве „Захтев" и „Одговор". Многе од њих иницирају слање једног или више сегмената ка удаљеној прикључници (сокету). Ове сегменте прихвата мрежни слој (IP) преко својих примитива „Пошаљи"и дистрибуира их одредишној TCP целини преко примитиве „Испоручи".

Да би се обезбедила услуга транспортног слоја (TCP услуга) морају



Слика 6.1 Примитиве које се размењују код TCP, IP услуга и приступа мрежи

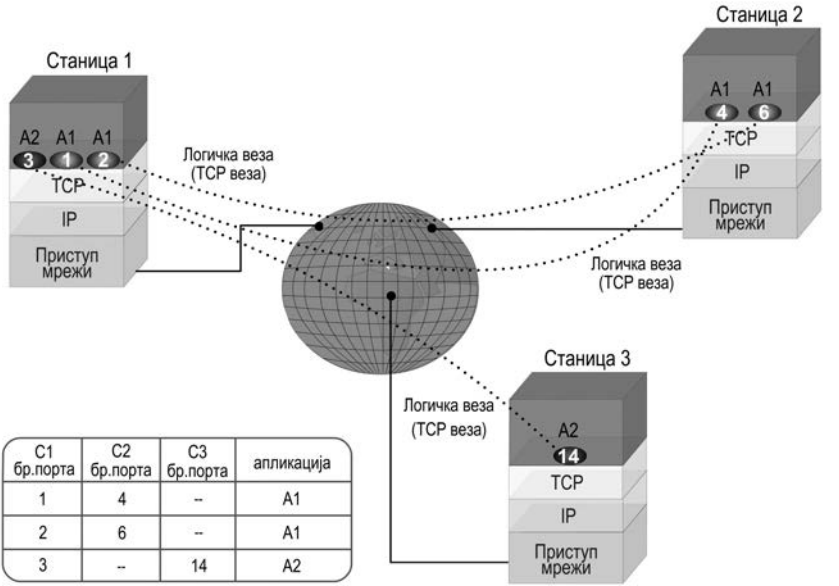
се дефинисати две прикључне тачке[1]. Једна је креирана од стране пошиљаоца, а друга од стране примаоца. Свака прикључна тачка има свој број (адресу) који се састоји од пара (станица, порт). Порт је назив који се користи за приступну тачку услузи транспортног слоја (TSAP[2]). У случају једне рачунарске мреже (нпр. локалне рачунарске мреже) станицу одређује мрежни уређај којим се прикључује на локалну рачунарску мрежу. Када је рачунарска мрежа интернет станица представља глобалну адресу на Интернету. Да би се реализовала TCP услуга мора се успоставити веза између прикључне тачке на рачунару пошиљаоца и прикључне тачке на рачунару примаоца.

Крајње тачке могу се користити за више веза истовремено. Другим речима, две или више веза могу се завршавати на истој крајњој тачки. Везе су одређене идентификаторима (прикључницама) на оба краја који су означени као (*socket1, socket2*).

На слици 6.2 приказане су три станице повезане на Интернет. Свака од станица има јединствену интернет адресу. TCP целина у оквиру станице услужује више портова а свакој од апликација додељен је један или више

[1] *Socket*

[2] *Transport Service Access Point*



C1 бр.порта	C2 бр.порта	C3 бр.порта	апликација
1	4	--	A1
2	6	--	A1
3	--	14	A2

Слика 6.2 Пример мултиплексирања код TCP заснованог на броју порта

портова. У овом примеру постоје две активне TCP везе: између апликације 1 (A_1) на станици 1 (C_1) и апликације 2 (A_2) на станици 2 (C_2). Такође постоји веза између апликације A_1 на станици 1 и апликације A_2 на станици 3 (C_3). TCP целина може да:

- истовремено подржава више веза са апликационим слојем и
- истовремено подржава више веза са истом апликацијом на апликационом слоју.

Портови чији су бројеви мањи од 1024 називају се познати портови и резервисани су за стандардне услуге (табела 6.2). На пример: било који процес који жели да успостави везу са неком станицом и преноси датотеке користећи FTP протокол може да се повеже са портом 21 удаљене станице да би успоставио везу са његовим FTP демоном[1]. Све TCP везе су двосмерне (потпуни дуплекс) и тачка-тачка. Двосмерна веза указује да постоји двосмерни проток података а тачка-тачка да свака веза има само

[1] Процес који се у позадини, невидљиво, извршава.

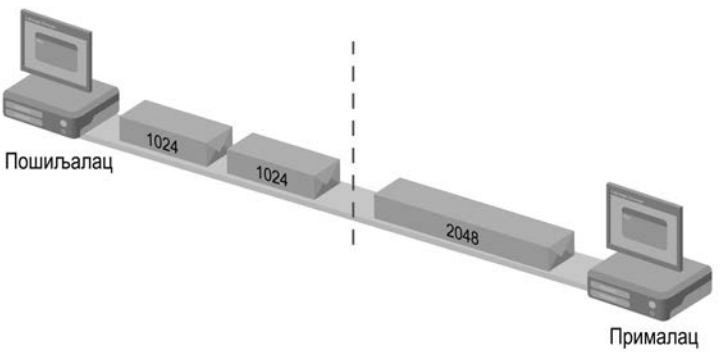
две крајње тачке. TCP протокол не подржава везе „један ка групи"[1] нити „један ка свима"[2].

За TCP услуге ситуација је сложенија када се она ослања на непоуздану мрежну услугу. Пример за то су:

- мреже са IP протоколом,
- локалне рачунарске мреже са IEEE802.3 протоколом и услугом без успоставе везе LLC подслоја.

TCP веза представља проток бајтова, не проток порука. Границе порука нису очуване при преносу са једног на други крај. На пример, ако TCP пошиљалац шаље четири податка од по 512 бајтова они могу бити уручени апликацији примаоца као четири дела од 512 бајтова, два дела од 1024 бајта или као један део од 2048 бајтова. На слици 6.3 приказана је испорука два дела од 1024 бајта као један део од 2048 бајтова. Не постоји начин да прималац сазна у којим јединицама је порука послата. TCP софтвер не зна шта ти бајтови значе и нема интерес да то сазна.

Када апликација проследи податак TCP процесу он га може послати или сместити у прихватну меморију у намери да сакупи већу количину података и пошаље их одједном. Понекад апликација заиста жели да подаци буду одмах послати. На пример, претпоставимо да је корисник



Слика 6.3 Испорука података апликацији: два послата дела од 1024 бајта испоручују се као један део од 2048 бајтова

[1] Мултикастинг

[2] Бродкастинг

пријављен на удаљеном рачунару. Након што се изврши једна командна линија и откуца прелазак у нови ред[1] битно је да се командна линија пошаље што пре удаљеном рачунару и проследи пре него што нова командна линија пристигне. Да би апликација убрзала прослеђивање података она ће указати одговарајућим механизмом TCP целини да не треба да задржава те податке.

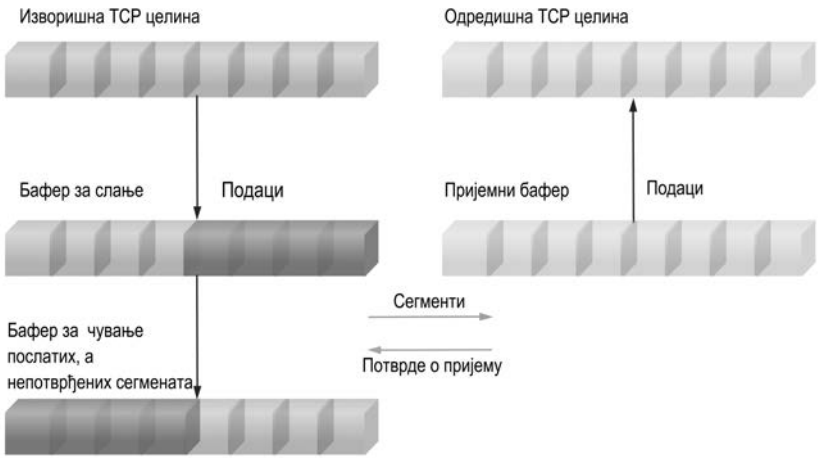
Последња битна примена TCP услуге, коју ћемо поменути, јесте употреба података које хитно треба проследити[2]. Када корисник притисне тастере DEL или CTRL-C да би прекинуо рад на удаљеном рачунару, апликација за слање додаје контролне информације у низ података и обавештава на одговарајући начин TCP процес на пријемној страни да заустави нагомилавање података и одмах корисницима (апликационим процесима) проследи све податке које има за ту везу.

6.2 Преглед TCP протокола

У овом одељку дајемо општи преглед TCP протокола. Слике 6.4 и 6.5 илуструју основне операције које извршава TCP протокол. Корисник шаље податке TCP целини, део по део, помоћу примитива за слање[3] (слика 6.1). Подаци се смештају у резервисани меморијски простор[4] за слање. С времена на време TCP узима податке из меморијског простора и пакује их у сегменте који се даље преносе. Сваки сегмент се прослеђује одредишној TCP целини (и даље кориснику) помоћу услуге мрежног слоја (IP услуге). Када одредишна TCP целина прими податке она уклања заглавље са сегмента а корисничке податке ставља у пријемни меморијски простор. Пријемна TCP целина помоћу примитиве за испоруку[5] повремено обавештава корисника да су подаци стигли и да су спремни за прослеђивање.

Сваки бајт на TCP вези има свој сопствени 32-битни редни број. Он се користи (као што је већ објашњено) код механизма потврде и механизма

[1] *Carriage Return*
[2] *Urgent data*
[3] *Send Primitives*
[4] Бафер
[5] *Deliver*



Слика 6.4 Основне TCP операције

клизајућег прозора. Предајна и пријемна TCP целина[1] размењују податке у облику сегмента. Сегмент се састоји од двадесетобајтног заглавља, опционог дела и (или не) бајтова података. TCP софтвер одлучује колике ће величине бити сегмент. Може сакупити податке у један сегмент или их поделити у неколико сегмената. Постоје ограничења која утичу на величину сегмента.

Један од проблема који може настати при преносу података је да подаци могу стићи на одредиште у другачијем редоследу од онога у којем су послати. На основу редних бројева смештених у TCP заглављу сваког сегмента одредишна TCP целина лако може да уочи прави распоред сегмената, да их потом пресложи и испоручи кориснику.

Други проблем који може да настане је ако се сегмент изгуби при преносу. Овај проблем се превазилази употребом потврда о пријему сегмената, чиме се омогућава да TCP целине могу да поново пошаљу изгубљене сегменте.

Да би TCP целина могала поново да пошаље изгубљене сегменте, мора да поседује копију сваког послатог сегмента. Копија се чува у за то намењеном меморијском простору, све док не добије потврду да је сегмент успешно испоручен (слика 6.4).

[1] Ентитет

Рад са отвором прозора код TCP није директно везан за механизам потврде као што је случај са протоколима слоја везе, што се може уочити у примеру који следи. Претпоставимо да прималац има на располагању меморијски простор од 8192 бајтова, као што је приказано на слици 6.5. Ако се пошаље сегмент од 4096 бајтова[1] и на одредишту исправно прими, прималац ће потврдити сегмент (AN=4096). У пријемном баферу је на располагању меморијски простор од 4096 бајтова све док апликација не учита (уклони) неке податке из њега. Пријемник ће обавестити предајник да је отвор прозора (W[2]), тј. слободан меморијски простор за смештање пријемних података, величине 4096 бајтова. То значи да пријемник не може да пошаље више од 4096 бајтова.

Пошиљалац шаље других 4096 бајтова. Пријемник потврђује успешан пријем тог сегмента и очекује сегмент чији је редни број првог бајта 8192. Отвор прозора (W) сада је нула. То значи да пријемник за сада нема простора да прими нове податке. Пошиљалац мора да прекине слање док процес апликације на страни примаоца не очита неке податке из меморијског простора. Тада се ослобађа меморијски простор за пријем нових података и пријемна TCP целина може послати сегмент којим „отвара“ прозор (W≠0).

Када је отвор прозора (W) једнак нули пошиљалац не може да шаље сегменте осим када је потребно послати:

- хитне податке, или
- сегмент са податком од једног бајта са потврдом и променом величине отвора прозора за супротни смер везе.

TCP стандард дозвољава ову опцију да би спречио потпуни застој[3] који би настао у случају да се обавештење о отвору прозора изгуби.

Од пошиљаоца се не захтева да проследи податке чим их добије од апликације. Нити се од примаоца очекује да одмах пошаље потврду. Као што је приказано на слици 6.5, након доласка првих 4kB[4] података TCP целина на пријему, знајући ди има на располагању величину отвора прозора од 8kB, може да складишти податке док не стигне још 4kB података.

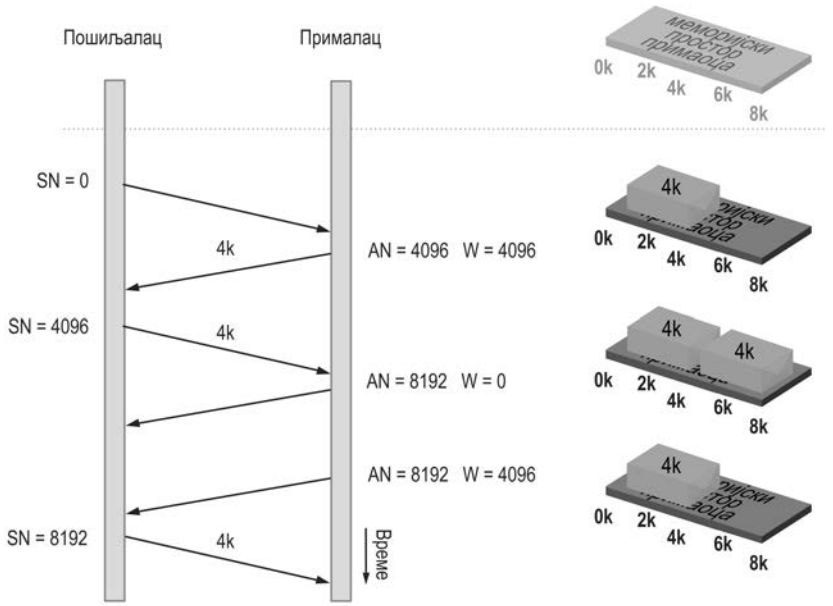
---

[1] Редни бројеви SN=0,…, 4097.

[2] *Window*

[3] D*eadlock*

[4] 4kB = 4*1024 бајтова



Слика 6.5 Меморијски простор пријемника и отвор прозора

6.3 Формат TCP сегмента

Слика 6.6 показује формат TCP сегмента. Сваки сегмент почиње заглављем од 20 бајтова. Заглавље може бити праћено опционим делом. После опционог дела може се послати највише 65515 бајтова података[1]. Сегмент без података могућ је и обично се користи као потврда или управљање. У овом одељку анализираћемо садржај заглавља TCP сегмента.

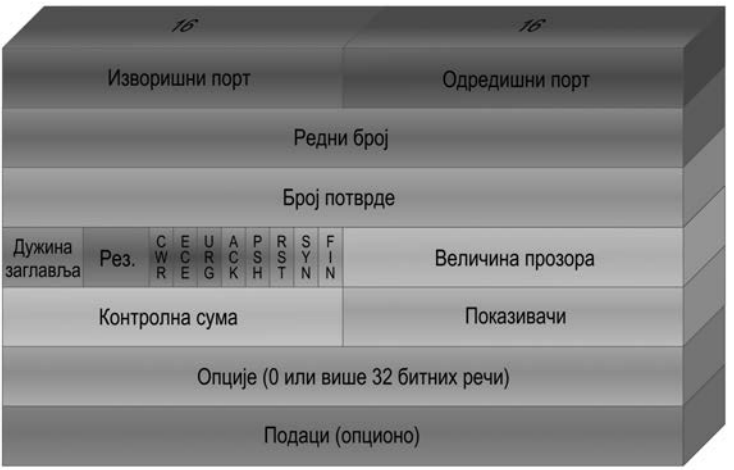
Изворишни и одредишни порт[2] су поља дужине по 16 бита. Ова поља одређују крајње тачке везе. Сваки корисник може одлучити како да распореди сопствене портове почевши од 1024 па навише.

Редни број (SN[3]) је поље величине 32 бита. Представља редни број

[1] 65535-40 = 65515; 20 бајтова (IP заглавље) + 20 бајтова (TCP заглавље без опционог дела) = 40 бајтова.

[2] *Source port, destination port*

[3] *Sequence Number*



Слика 6.6 Заглавље TCP сегмента

првог бајта података у том сегменту осим у случају када је постављен на јединицу SYN бит. Тада поље редни број означава почетни (иницијални) редни број ISN[1] тог правца везе. Првом бајту података у том правцу везе биће додељен редни број ISN+1.

Број потврде (AN[2]) је поље величине 32 бита. Означава редни број података који TCP целина очекује да прими.

Дужина заглавља[3] TCP сегмента је поље које носи информацију колико речи дужине 32 бита садржи заглавље. Ово поље је потребно да би пријемна страна знала величину заглавља пошто је поље опције променљиве величине.

Резервисано поље је величине 4 бита. Намењено је за будућу употребу и тренутно се не користи.

Ознака (флег) је поље величине 8 битова. Сваки бит у овом пољу има одређену функцију:

- CWR[4] (прозор загушења је смањен) постављен на 1 указује да је прозор загушења смањен.

[1] *Initial Sequence Number*
[2] *Acknowledgement Number*
[3] TCP *header lenght*
[4] *Congestion Window Reduced*

- ECE[1] (ехо експлицитног обавештавања о загушењу) заједно са ознаком CWR дефинисани су у документу RFC3168. Користе се за обавештавање о загушењу које ће бити објашњено у следећем поглављу.
- URG[2] постављен на 1 означава да је поље које указује[3] на место „хитних" података од значаја и у употреби. Функција која обезбеђује испоруку хитних података[4] служи да би се указало апликацији да су стигли хитни подаци и да одмах треба да јој се испоруче.
- ACK[5] постављен на 1 означава да је поље број потврде важеће и у употреби. Сегмент са постављеним ACK битом означава се као *Ack* сегмент[6]. Ако је ACK=0 сегмент не садржи потврду. У том случају поље „потврдни број" се игнорише.
- PSH[7] постављен на 1 од TCP целине - примаоца захтева да испоручује податке апликацији одмах по њиховом доласку и да их не складишти (што би он урадио због ефикасности).
- RST[8] постављен на 1 користи се за ресетовање везе. Сегмент са постављеним RST битом означава се као R*st* сегмент. Користи се и да би се одбацио неочекивани сегмент или одбио захтев за отварање нове везе.
- SYN[9] постављен на 1 користи се при успостављању везе. Сегмент са постављеним SYN битом означава се као *Syn* сегмент. Када се шаље захтев за успоставом везе, поље SYN и ACK имају вредности 1 и 0 респективно[10]. Када се шаље потврда на захтев за успоставом везе оба поља[11]имају

[1] *Explicit Congestion Notification - Echo*
[2] *Urgent function*
[3] *Urgent Pointer*
[4] Са постављеним URG битом на вредност 1 и одговарајућом вредношћу у пољу *Urgent Pointer.*
[5] *Acknowledge*
[6] У овом уџбенику се сегменти означавају *Ack,* а ознака (флег) са ACK. У литератури нпр. RFC793 се и сегменти означавају великим словима ACK.
[7] *Push*
[8] *Reset*
[9] *Synchronize*
[10] SYN=1, ACK=0
[11] SYN=1, ACK=1

вредности 1 а одговарајући сегмент означавамо као *Syn_Ack* сегмент.

- FIN[1] постављен на 1 користи се при ослобађању везе. Сегмент са постављеним FIN битом означава се као *Fin* сегмент. Указује да пошиљалац нема више података за слање.

Величина прозора[2] (отвор прозора, кредит) је поље величине 16 битова и користи се за контролу тока. Садржи број бајтова података, почев од редног броја који је наведен у пољу „потврдни број" које прималац може да прихвати. Када је вредност овог поља једнака 0 указује да прималац не може да прихвати више података.

Контролна сума[3] је поље величине 16 битова. Користи се да би се избегао пријем сегмената који су оштећени при преносу. Када пошиљалац шаље сегмент он израчуна контролну суму и упише је у ово поље. Када прималац прими сегмент он такође израчуна контролну суму и ако је вредност коју добије иста као и вредност поља контролне суме, значи да при преносу није дошло до грешке и сегмент се прихвата.

Контролна сума се израчунава као први комплемент суме првих комплемената шеснаестобитних речи псеудозаглавља, заглавља и података TCP сегмента. Псеудозаглавље укључује следећа поља из заглавља IP пакета: изворишну и одредишну IP адресу, поље које означава протокол[4] и поље које означава дужина TCP сегмента (слика 6.7). При израчунавању контролне суме вредност поља контролне

Слика 6.7 Формат псеудозаглавља

[1] *Finish*
[2] *Window size*
[3] *Checksum*
[4] За TCP протокол ово поље има вредност 6.

Слика 6.8 TCP поље опције променљиве дужине

суме поставља се на нулу. Разлог због кога TCP целина укључује и IP псеудозаглавље (само за време израчунавања) је што се на овај начин штити од грешака које могу да настану на мрежном слоју. У случају да се испоручи пакет погрешној станици, TCP целина ће препознати грешку.

Показивач „хитних" података[1] је поље величине 16 битова. Када се ова вредност дода на редни број TCP сегмента добија се вредност последњег бајта у секвенци података које треба хитно испоручити апликацији.

Опције[2] је поље променљиве величине. Користи се за додатне услуге које нису укључене у регуларно заглавље. Једна од опција је, на пример, величина сегмента која се користи при преносу. Требало би користити што веће сегменте[3] јер се на тај начин преноси више података[4] а заглавље од 20 бајтова представља његов мањи део. Као и у случају IP датаграма у пољу опције може бити један или више бајтова променљиве дужине приказане на слици 6.8.

Тренутно је дефинисано седам опција[5] (табела 6.2)

Максимална величина сегмента (MSS[6]) постоји само у сегменту типа[7] *Syn*, односно користи се само за време успостављања везе. Означава максималну дозвољену величину сегмента коју пријемна страна може да прихвати. Уколико један крај везе не добије податак о максималној величини сегмента од друге стране везе он онда претпоставља подразумевану величину од 536 бајтова. Максимална величина сегмента не мора да буде иста у оба правца везе. Уколико се опција (слика 6.9) не

[1] *Urgent Pointer*
[2] *Options*
[3] Онолико велики колико дозвољавају станице које учествују у размени података.
[4] Додатном анализом [8] показује се да није увек боље преносити веће сегменте.
[5] Описане су у документу RFC1072.
[6] *Maximum Segment Size*
[7] Сегменте са постављеним SYN битом означавамо са *Syn*.

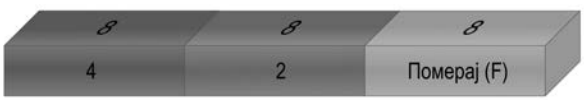
Врста опције	Дужина	Значење
0	-	Крај листе опција
1	-	Нема активности
2	4	Максимална величина сегмента
3	3	Опсег величине прозора
4	2	Селективна потврда се дозвољава
5	X	Селективна потврда
8	10	Временска ознака

Табела 6.2 Опције у TCP заглављу

користи могућа је било која величина сегмента. Свака рачунарска мрежа има своју максималну јединицу за пренос (MTU[1]) и сваки сегмент се мора уклопити у њу[2]. Уколико се сегмент преноси мрежом чија је максимална јединица за пренос мања од величине тог сегмента гранични рутер га дели на два или више мањих сегмената.

Опсег величине прозора (WSO[3]) постоји само у *Syn* сегменту. Обе стране морају да пошаљу WSO опцију у свом *Syn* сегменту да би се омогућила промена величине прозора. У току трајања везе није га могуће мењати. Поље „величина прозора" је број који указује колико бајтова је пријемник спреман да прихвати. У случају да се користи опција[4] „опсег

Слика 6.9 Опција максимална величина сегмената



Слика 6.10 Опција опсег величине прозора

[1] *Maximum Transfer Unit*

[2] На пример: за локалне рачунарске мреже етернет типа је 1460 бајтова, за локалне рачунарске мреже по IEEE802.3 је 1452 бајта, а за неке BSD имплементације 1024 бајта пошто захтевају умножак (мултипл) од 512 бајтова.

[3] *Window Scale Option*

[4] Дефинисана у документу RFC1325.

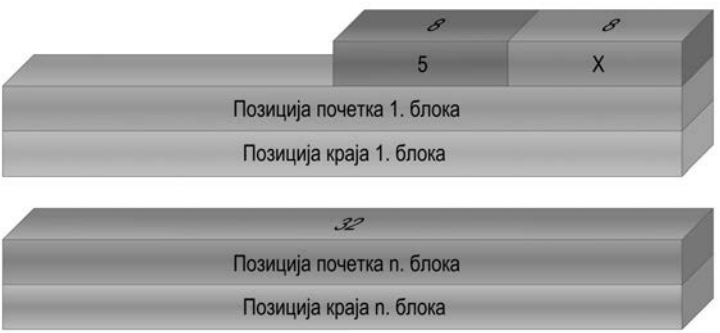
Слика 6.11 Опција селективна потврда се дозвољава

величине прозора" (слика 6.10) онда се вредност из поља „величина прозора" множи са 2^F, где је F померај такав да је $2^F < 2^{14}$.

Поље „селективна потврда се дозвољава" (SACKP[1]) указује да је дозвољена употреба селективних потврда о успешном пријему података (слика 6.11).

Поље „селективна потврда" (SACK[2]) омогућава примаоцу да обавести пошиљаоца о сегментима који су успешно примљени (слика 6.12). Пошиљалац ће поново послати само сегменте који нису успешно примљени[3]. Уколико је број сегмената које треба поново послати велики дужина SACK поља била би превише велика. Због тога се на пријему подаци групишу у целине успешно примљених сегмената. Предајној страни се шаље информација о релативној позицији и величини целине. Обавештење о броју целина за које се може слати SACK опцијом је ограничено.

Временска ознака (TS[4]) је поље које омогућава слање временских ознака у TCP сегментима[5] (слика 6.13). Ова опција садржи два поља



Слика 6.12 Опција селективна потврда

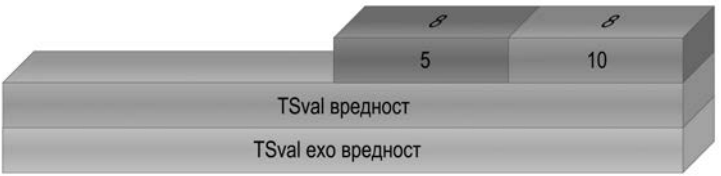
[1] *Selective Acknowledgements Permitted* (SACK *Permitted*)

[2] *Selective* ACK

[3] Опције SACK и SACK *Permitted* дефинисане су у документу RFC2018.

[4] *Timestamp*

[5] Ова опција је дефинисана у документу RFC1325.



Слика 6.13 Опција временске ознаке

величине четири бајта. Поље „величина временске ознаке" (TSval[1]) садржи текући податак о такту предајне TCP целине. Садржај поља „ехо временске ознаке" (TSecr[2]) има значаја само уколико је постављен бит ACK у TCP заглављу.

Употреба опције са временском ознаком омогућава предајнику да у сваки сегмент постави временску ознаку. Пријемник поставља исту вредност у *Ack* сегмент. Када предајна целина прими *Ack* сегмент она може да израчуна укупно време које је потребно да се пошаље пакет и да стигне његова потврда (RTT[3]).

6.4 Успостављање и раскидање TCP везе

Успостављање везе има за циљ да се крајеви TCP везе увере да постоје, да се договоре око параметара као што је нпр. величина прозора и да се доделе ресурси TCP целини (нпр. меморијски простор).

Раскидање TCP везе обавља се сагласношћу обе стране. У зависности од начина како се обави може да дође:

- до губитка података чији је пренос у току код наглог раскида везе или,
- до одлагање раскида везе док се сви подаци чији је пренос у току не испоруче одредишту.

Приступ за контролу тока (описан у претходном поглављу) може ограничити пропусност TCP везе када постоје велика кашњења. Прималац може да повећа пропусност тако што ће пошиљаоцу понудити већи кредит него што је величина слободног меморијског простора. На

[1] *Timestamp value*
[2] *Timestamp Echo Reply*
[3] *Round Trip Time*

пример прималац ће послати пакет са кредитом величине 1000 бајтова иако је његов меморијски просто попуњен. Прималац претпоставља да може да ослободи меморијски простор од 1000 бајтова за време које протекне од тренутка када пошаље сегмент са кредитом и до тренутка када од пошиљаоца добије нове податке.

6.4.1 Успостављање TCP везе

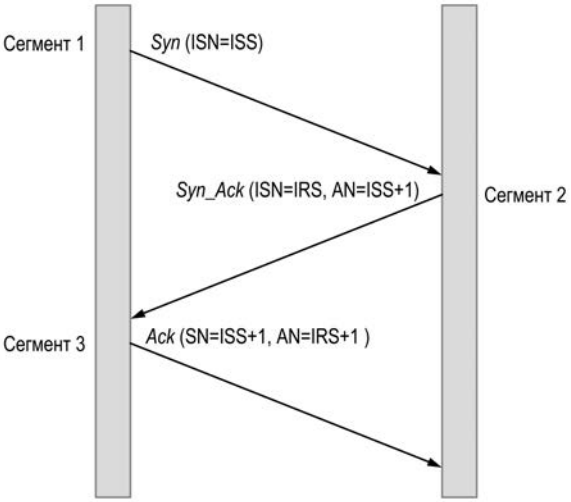
За TCP везу потребна је трострука размена управљачких сегмената. Када је на једној страни везе клијент а на другој страни сервер тада да би се остварила веза:

1. страна која захтева комуникацију (уобичајено клијент) шаље TCP сегмент[1] (*Syn* сегмент):
   - са ознакама постављеним на следеће вредности: SYN=1 и ACK=0,
   - специфицирајући број порта сервера са којим клијент жели да се повеже и
   - са постављеним иницијалним редним бројем (SN=ISS[2]) клијента,
   - на слици 6.14 овај сегмент означен је као сегмент 1;
2. сервер шаље одговор: свој TCP сегмент (*Syn_Ack* сегмент)
   - са ознакама постављеним на следеће вредности: SYN=1 и ACK=1.
   - серверов почетни (иницијални) редни број (SN=IRS[3]),
   - са редним бројем потврде постављеним на почетни редни број клијента увећан за један (AN=ISS+1),
   - на слици 6.14 овај сегмент означен је као сегмент 2;
3. клијент мора да потврди пријем *Syn_Ack* сегмента слањем *Ack* сегмента:
   - са ознакама постављеним на следеће вредности: SYN=0 и ACK=1,
   - са редним бројем (SN=ISS+1),
   - са редним бројем потврде постављеним на почетни редни

[1] Могуће је да истовремено две стране везе пошаљу *Syn* сегмент (одељак 6.7).

[2] *Initial Send Sequence*

[3] *Initial Receive Sequence*



Слика 6.14 Временски редослед размене сегмената код успостављања везе

број сервера увећан за један (AN=IRS+1),

- на слици 6.14 овај сегмент означен је као сегмент 3.

Као што смо видели свака од страна поставља свој почетни (иницијални) редни број за ту везу (ISS, IRS). Почетни редни број мора да се мења у току времена, тако да свака веза има различит. У документу RFC739 специфицирано је да иницијални редни број треба посматрати као бројач са 32 бита који се инкрементира сваких 4μs[1]. Намена овог броја је да се сегменти који пристижу са великим кашњењем погрешно не интерпретирају као део постојеће везе.

Постоје ситуације када веза не може да се успостави. Један од примера био би да сервер није подигнут, или да је кабл којим је клијент повезан на локалну рачунарску мрежу у прекиду. Питање је колико често ће TCP клијент слати *Syn* сегмент да би успоставио везу. На пример други *Syn* сегмент биће послати 5,8s после слања првог *Syn* сегмента а трећи 24s после другог сегмента. Колико дуго ће клијент покушавати да поново шаље *Syn* сегмент пре него што одустане? Већина система заснованих на беркли имплементацији поставља временско ограничење на 75s.

[1] Већина имплентација које су проистекле из беркли дистрибуције BSD (*Berkly Software Distribution*) користи 8μs.

6.4.2 Раскидање TCP везе

За успостављање TCP везе потребно је да клијент и сервер размене три сегмента док је за раскид везе потребно да размене четири сегмента. Разлог је што TCP веза може бити и полузатворена.

Као што је познато TCP веза је истовремено двосмерна (потпуни дуплекс). Да би се боље објаснио раскид везе најбоље је посматрати TCP везу као две једносмерне (симплекс) везе. Свака једносмерна веза раскида се независно. Да би прекинула везу било која страна може послати *Fin* сегмент (са постављеним ознакама FIN=1, ACK=1). То значи да нема више података за слање. По пријему *Fin* сегмента TCP целина мора да обавести апликацију да је друга страна завршила са слањем података. Апликација која хоће да оконча везу иницира слање *Fin* сегмента[1].

Подаци се у другом правцу могу и даље слати без икаквих ограничења. Ову могућност у пракси мали број TCP апликација користи. Нормални сценарио приказан је на слици 6.15. Када су оба правца затворена веза је раскинута.

Страна која иницира затварање везе (она која шаље први *Fin* сегмент) извршава активно затварање[2], а друга страна (која прима *Fin* сегмент) извршава пасивно затварање[3].

Сегмент 4 (слика 6.15) који иницира крај везе шаље се, на пример, када *Telnet* клијент затвара везу откуцавши команду „quit*e*". То доводи клијент страну TCP везе у ситуацију да пошаље *Fin* сегмент окончавајући ток података од клијента ка серверу.

Када сервер прими *Fin* сегмент он шаље натраг *Ack* сегмент (са постављеном ознаком ACK=1) инкрементира предајни редни број за 1 (сегмент 5). *Fin* сегмент користи редне бројеве баш као и *Syn* сегмент. У овој тачки TCP процес на страни сервера прослеђује апликацији информацију о крају[4] датотеке. Сервер затим затвара везу налажући TCP целини да пошаље *Fin* сегмент (сегмент 3) који клијент мора да потврди *Ack* сегментом у коме је примљени редни број увећен за један (сегмент 4).

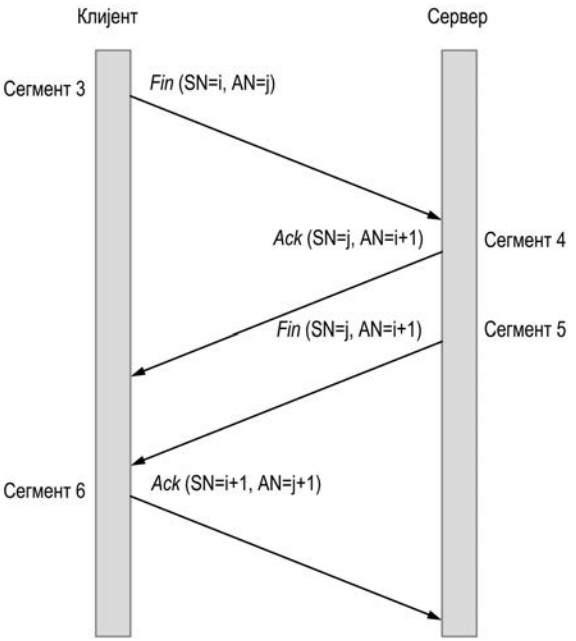
---

[1] Апликација шаље *Close* примитиву, односно захтев за окончање везе. На основу тог захтева TCP целина шаље *Fin* сегмент. Детаљно ће бити објашњено у следећем одељку.

[2] *Active Close*

[3] *Pasive Close*

[4] *End of file* - EOF



Слика 6.15 Временски редослед размене сегмената код раскидања везе

Треба напоменути да слање *Fin* сегмената изазивају апликације на крајевима TCP везе а потврде *Fin* сегмената (*Ack* сегменте) аутоматски генерише TCP софтвер.

На слици 6.15 могле би да се промене ознаке и да се лева страна која иницира затварање везе означи као сервер а десна као клијент. Уобичајено је ипак да процес клијента (као интерактивни корисник) први иницира прекид.

Мало је вероватно да оба краја TCP везе пошаљу *Fin* сегменте истовремено. У том случају оба би била потврђена на уобичајени начин и веза раскинута. У суштини нема никакве разлике у томе да ли је веза раскинута истовремено од обе стране или не.

6.5 Дијаграм стања TCP везе

Одржавање TCP везе захтева памћење неколико променљивих. Сматраћемо да су ове променљиве смештене у запису о вези који се

обележава са TCB[1]. Промењљиве које су смештене у запису о вези су и: број локалне и удаљене прикључнице, сигурност везе, приоритети у вези, показивачи на предајни и пријемни меморијски простор корисника, показивачи на ретрансмисионе листе и текући сегмент. Поред поменутих променљивих које се односе на предајне и пријемне редне бројеве у запису о вези такође се памте[2] и:

- подаци о послатим сегментима:
  - послати и непотврђени сегменти (SND.UNA),
  - шаље се као следећи (SND.NXT[3]),
  - послати отвор прозора (SND.WND[4]),
  - послати показивач хитних података (SND.UP[5]),
  - редни број сегмента који је искоришћен за последње иновирање прозора (SND.WL1[6]),
  - редни број потврђеног сегмента који је искоришћен за последње иновирање прозора (SND.WL2[7]),
  - иницијални предајни редни број (ISS[8]),
- подаци о примљеним сегментима;
  - следећи (RCV.NXT[9]),
  - пријемни прозор (RCV.WND[10]),
  - пријемни показивач хитних података (RCV.UP[11]),
  - иницијални пријемни редни број (IRS[12]).

Објаснили смо бројна правила у иницирању и окончању везе. Ова правила могу се приказати на дијаграмима стања[13] везе који су приказани на слици 6.16.

---

[1] *Transmission Control Block*

[2] Назив и ознаке променљивих преузети су из документа RFC795.

[3] *Send next*

[4] *Send window*

[5] *Send urgent pointer*

[6] *Segment sequence number used for last window update*

[7] *Segment acknowledgment number used for last window update*

[8] *Initial send sequence number*

[9] *Receive next*

[10] *Receive window*

[11] *Receive urgent pointer*

[12] *Initial receive sequence number*

[13] *Finite state machine*

Имена и опис 11 стања која су дефинисана у RFC793 дати су у табели 6.3. Стање CLOSED у ствари је имагинарно стање када не постоји TCB запис а самим тим ни веза. У сваком стању дефинисани су прихватљиви догађаји. Уколико се они појаве предузимају се одређене акције. У противном пријављује се грешка.

Свака веза започиње стањем CLOSED. Из тог стања може се прећи у пасивно отворено LISTEN, или активно отворено CONNECT стање. Уколико друга страна има супротан прелаз веза се остварује и прелази се у ESTABLISHED стање. Раскид везе може да иницира било која страна. Када је он завршен прелази се у CLOSED стање.

Дијаграм стања приказан је на слици 6.16. Активно остваривање везе са пасивним сервером на слици је приказано пуном линијом која иде од клијента ка серверу. Пут пасивног отварања везе (сервер) приказан је на слици 6.16 испрекиданом дебљом линијом. Танке линије представљају неуобичајени след догађаја. Свакој од линија придодат је пар догађај/активност[1].

Догађај може бити:

- системски позив иницијализован од стране корисника - примитиве *Active Open, Pasive Open, Send, Close*,
- долазећи сегмент *Syn*, Sin_*Ack*, *Fin_Ack*, *Ack*, или
- истицање времена на које је постављен неки од часовника.

Активност је слање управљачког сегмента: *Syn*, Sin_*Ack*, *Fin_Ack*, *Ack* или R*st*, или се ништа не предузима.

Дијаграм се најбоље анализира ако се прати пут клијента (дебља пуна линија) а затим пут сервера (испрекидана дебља линија). Тексту који следи детаљно ће бити објашњен дијаграм стања део по део.

Када апликација на страни клијента (слика 6.17) тражи захтев за успоставу везе (*Active Open* примитива) локална TCP целина:

- прави запис о вези TCB,
- означава да је у стању SYN SENT,
- шаље *Syn*[2] сегмент.

Приметимо да више веза за више апликација може бити

[1] *Event/action*

[2] То је сегмент код кога је: SYN=1, ACK=0.

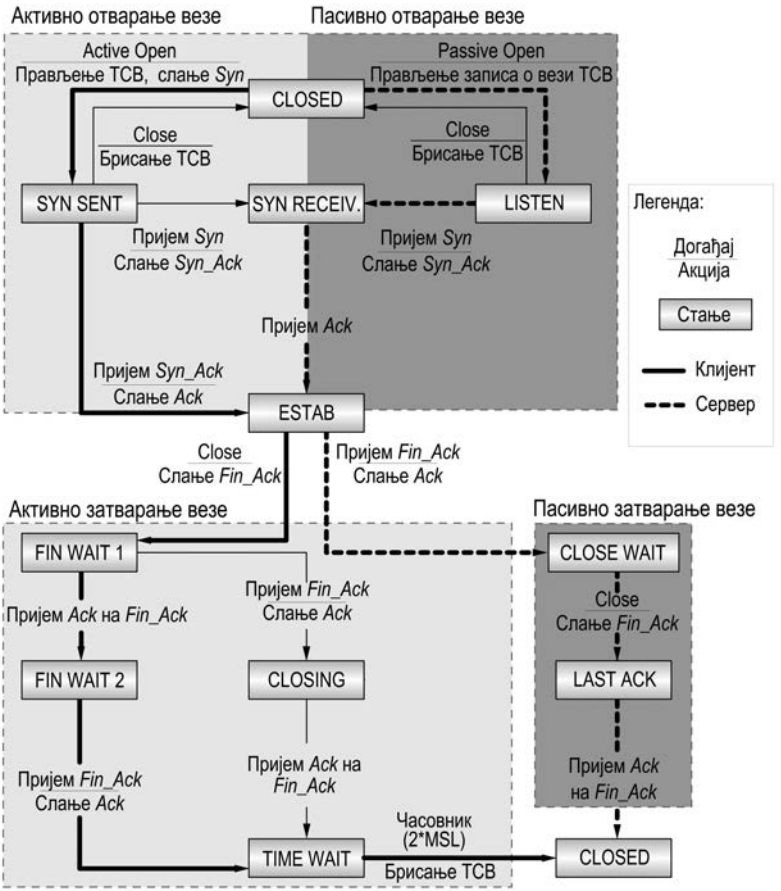
Стање	Опис
CLOSED	Указује да не постоји TCB запис а самим тим ни веза.
LISTEN	Представља ишчекивање (ослушкивање) захтева за успоставу везе од удаљеног TCP процеса или порта.
SYN-SENT	Представља стање ишчекивања одговарајућег захтева за успоставу везе после слања захтева за успоставу везе (Syn сегмента).
SYN-RCVD	Представља стање ишчекивања пријема потврде захтева за успоставу везе (Ack сегмента) после и слања и пријема захтева за успоставу везе .
ESTABLISHED	Представља успостављену везу. Подаци који се добијају могу се испоручити кориснику. Ово је нормално стање за фазу преноса података у вези.
FIN WAIT1	Представља стање ишчекивања захтева за окончање везе од удаљеног TCP процеса, или потврду на захтев за окончање везе који је претходно послат.
FIN WAIT2	Представља ишчекивање захтева за окончање везе од удаљеног TCP процеса.
TIME-WAIT	Представља чекање да протекне довољно времена да би били сигурни да је удаљени TCP процес примио потврду свог захтева за окончање везе.
CLOSING	Представља ишчекивање потврде захтева за окончање везе.
CLOSE WAIT	Представља ишчекивање захтева за окончањем везе од локалног корисника.
LASTACK	Представља ишчекивање потврде захтева за окончање везе претходно послатог удаљеном TCP процесу (што укључује потврду свих његових захтева за окончање везе).

Табела 6.3 Стања везе и њихов опис

успостављено истовремено, тако да се стања односе на сваку појединy везу и уписују се у одговарајући запис о вези (TCB). Када стигне сегмент потврде[1] и захтева за успоставу везе TCP целине сервера (*Syn_Ack*) TCP целина клијента шаље завршну потврду - сегмент потврде *Ack*[2]

[1] То је сегмент код кога је: SYN=1, ACK=1.

[2] То је сегмент код кога је: SYN=0, ACK=1.



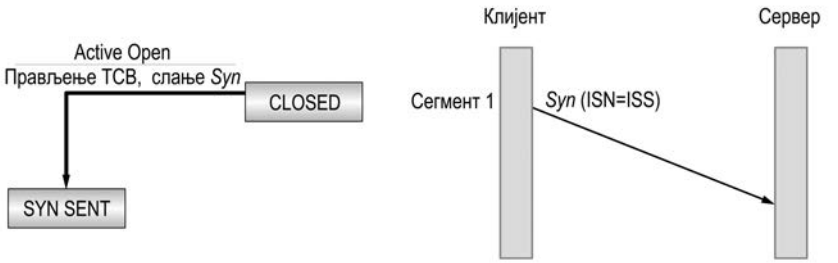
Слика 6.16 Дијаграми прелаза из једног TCP стања у друго

(слика 6.18) троструке размене и прелази у стање ESTABLISHED. Подаци се од тог тренутка надаље могу слати и примати.

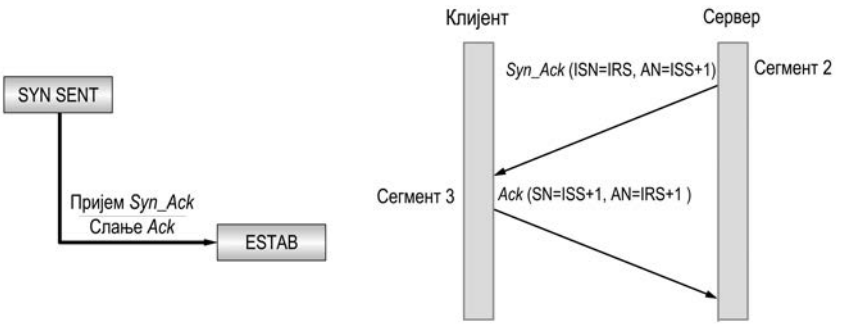
Ако је апликација завршила са радом извршава *Close* примитиву (слика 6.19) која доводи до тога да локална TCP целина пошаље *Fin_Ack*[1] сегмент, прелази у стање FIN WAIT1 и чека потврду *Ack* на *Fin_Ack* сегмент (на слици 6.16 у левом доњем углу, обележено као активно затварање везе[2]).

[1] То је сегмент код кога је: FIN=1, ACK=1.

[2] Важно је истаћи да и страна клијента и страна сервера могу да иницирају затварање везе. У анализи



Слика 6.17 Захтев за успоставу везе на страни клијента



Слика 6.18 Прелазак у стање успостављене везе на страни клијента

Када стигне *Ack* на *Fin_Ack* сегмент прелази се у стање FIN WAIT2 и један правац везе је сада затворен (слика 6.20).

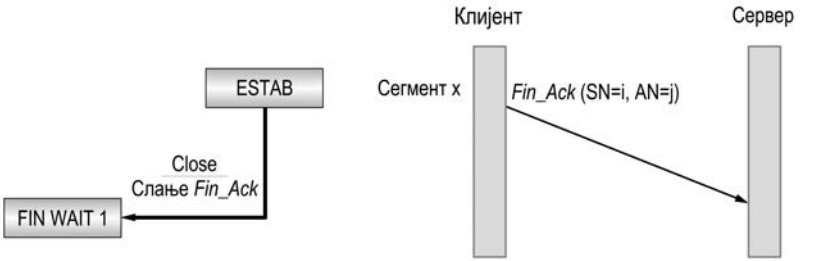
Док је у стању FIN WAIT2 клијент очекује још један сегмент – *Fin_Ack* сегмент који указује да и апликација на страни сервера окончава везу. По пријему тог *Fin_Ack* сегмента клијент га потврђује *Ack* сегментом и прелази у стање TIME -WAIT (слика 6.21). Док је у овом стању клијент може поново да пошаље *Ack* сегмент уколико је претходно послати изгубљен.

Сада су обе стране затвориле везу и TCP чека да истекне одређено време. Након истека овог времена TCP брише запис о вези.

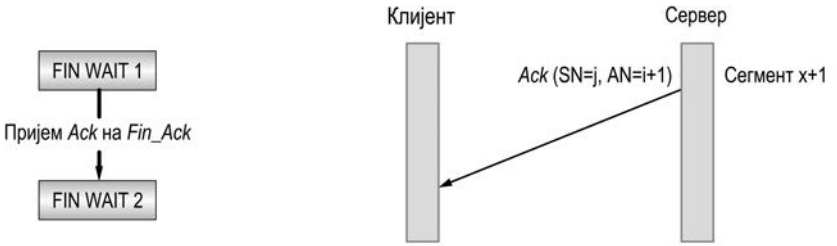
Сада ћемо проучити управљање везом с тачке гледишта сервера. Сервер поставља *Pasive Open* и чека. Када стигне *Syn* сегмент, он бива потврђен и сервер прелази у стање SYN RCVD (слика 6.22).

Када је *Syn* сегмент сервера потврђен завршава се трострука

која је описана у овом поглављу претпостављено је да страна клијента иницира затварање везе.



Слика 6.19 Апликација на страни клијента иницира затварање везе

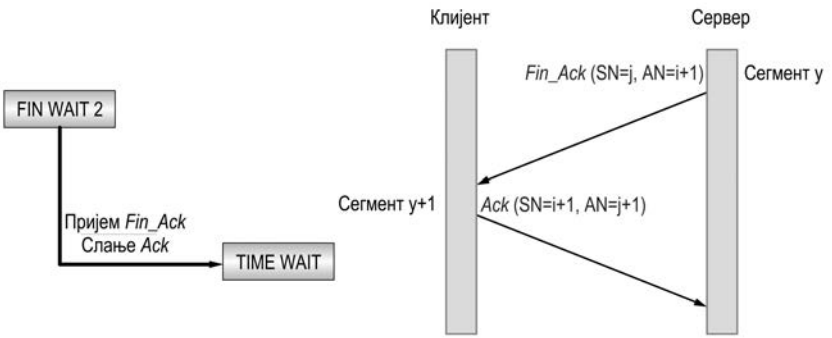


Слика 6.20 Затварање везе смера клијент-сервер

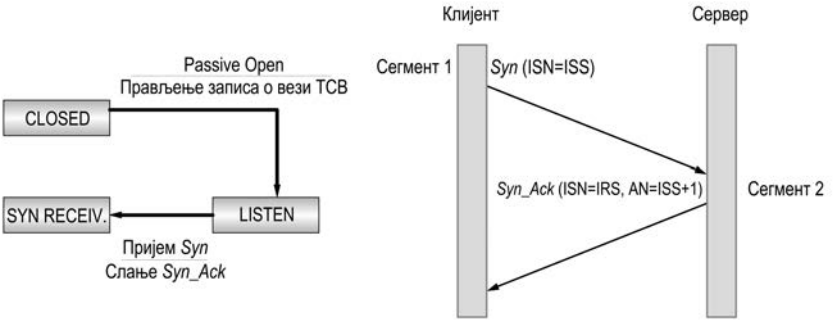
размена и сервер прелази у стање ESTABLISHED (слика 6.23). Може да започне пренос података.

Сада ћемо анализирати како се раскида веза. По пријему *Fin_Ack* сегмента TCP процес обавештава о томе апликацију и шаље потврду о пријему тог *Fin_Ack* сегмента (слика 6.24).

Када апликација сервера пошаље *Close* примитиву, TCP процес на



Слика 6.21 Затварање везе смера сервер-клијент

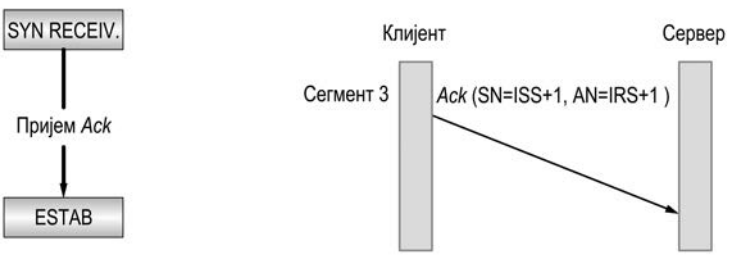


Слика 6.22 Пријем захтева за успоставу везе на страни сервера

страни сервера шаље клијенту *Fin_Ack* сегмент (на слици 6.16 означено као пасивно затварање везе). Када стигне потврда клијента сервер ослобађа везу, брише запис о вези и прелази у стање CLOSED (слика 6.25).

6.5.1 Стање чекања

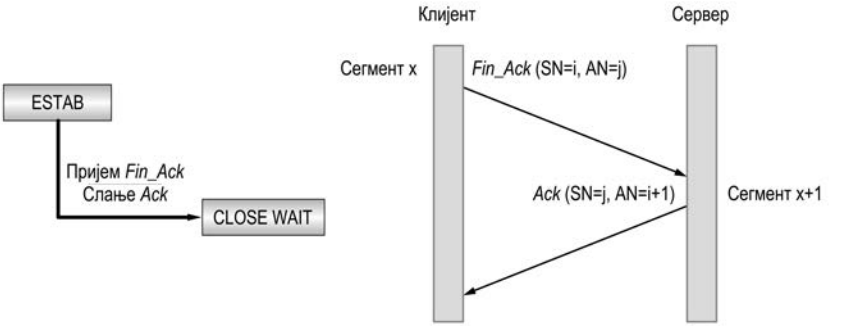
Стање TIME- WAIT означава се и као стање чекања једнако двоструком максималном трајању сегмента 2*MSL[1]. Максимално трајање сегмента дефинише се као време постојања сегмента на мрежи. После истека тог времена сегмент се одбацује. Познато је да је ово време ограничено пошто се TCP сегменти преносе IP пакетима који у свом заглављу имају TTL поље које ограничава време постојања IP пакета[2]. За постављено 2*MSL време важи следеће правило: када TCP целина извршава примитиву активног затварања везе и шаље последњи *Ack* сегмент та веза мора да остане у



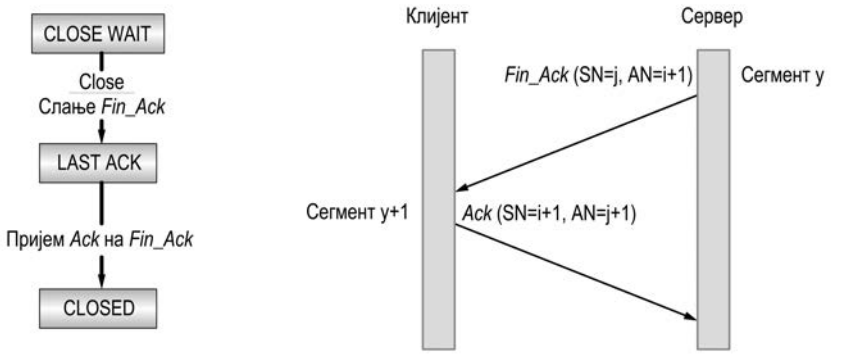
Слика 6.23 Веза сервер-клијент је успостављена

[1] *Maximum Segment Lifetime* - максимално време живота сегмента.

[2] Подсетимо се да TTL (*Time To Live*) поље практично ограничава број рутера преко којих IP пакет прелази а не време колико је IP пакет присутан у мрежи (тј. постоји).



Слика 6.24 Страна сервера по пријему *Fin_Ack* сегмента клијента шаље потврду, обавештава апликацију и прелази у стање CLOSE WAIT



Слика 6.25 Затварање везе на страни сервера

стању TIME WAIT у трајању једнаком двострукој вредности времена MSL. Ово омогућава TCP процесу да ако треба пошаље последњи *Ack* сегмент за случај да се претходно послати изгубио (на удаљеној страни часовник ће сигнализирати да је време истекло и она ће поново да пошаље свој *Fin_Ack* сегмент).

Последица 2*MSL чекања је и та што се пар прикључака[1] који дефинишу ту везу не може користити[2] док је TCP веза у стању 2*MSL чекања. Тек по истеку времена једнаком 2*MSL тај пар се може поново употребити[3].

[1] Сокет (*socket*) чине IP адреса клијента, број порта клијента, IP адреса сервера, број порта сервера.

[2] По дефиницији број локалног порта не може се поново употребити докле год је тај број порта број локалног порта прикључнице која је у стању 2MSL ишчекивања.

[3] Неке имплементације имају начин да заобиђу ово ограничење

Било који закаснели сегмент који стигне док је веза у 2*MSL ишчекивању одбацује се пошто се веза дефинисана паром прикључака (која је у 2*MSL ишчекивању) не може поново користити у том периоду. Када се успостави важећа веза са паром прикључака који су претходно били у 2*MSL ишчекивању а пристигну закаснели пакети из претходне везе[1] они неће бити погрешно интерпретирани.

Као што се види на слици 6.16 уобичајено је да клијент иницира активно затварање везе и улази у TIME - WAIT стање. Сервер обично обавља пасивно затварање везе и не пролази кроз TIME - WAIT стање (слике 6.24, 6.25). Последица је да ако клијент прекине са радом и исти клијент убрзо поново започне са радом онда он не сме поново да узме исти број локалног порта. За клијента то није никакв проблем пошто је за њега неважно које бројеве портова користи.

За сервер то није случај. Сервер користи познате бројеве портова. Уколико прекинемо рад сервера који је већ успоставио везу и убрзо покушамо поново да га подигнемо неће моћи да му се доделе претходно додељени бројеви портова. Ти бројеви портова су део везе која је у стању чекања 2*MSL. Може да протекне од 1 до 4 минута пре него што се сервер може поново покренути[2].

Неке имплементације дозвољавају постојање нове везе и за време TIME - WAIT стања уколико је нови иницијални редни број довољно већи од последњег редног броја претходне везе са истим локалним и удаљеним IP адресама и бројевима портова[3].

Концепт времена мировања[4]

Ишчекивање 2*MSL обезбеђује заштиту од интерпретације закаснелих сегмената из претходне везе са истим истим локалним и удаљеним IP адресама и бројевима портова као сегменти текуће везе. Али ово функционише само под условом да не дође до прекида у раду станице која је у ишчекивању да протекне 2* MSL.

---

[1] Веза се дефинише паром прикључака (сокета). Нова веза са истим паром прикључака назива се реинкарнација те везе.

[2] У литератури [3] наводи се да је ово време од 0,5, 1 и 2 минута.

[3] RFC1185 указује на проблеме који и на овај начин могу наступити.

[4] *Quite Time*

Шта се дешава ако је у квару станица са портовима који су у чекању 2* MSL? Поново се подиже после MSL секунди и одмах успоставља везу са истим локалним и удаљеним IP адресама и бројевима портова као и оним који су у чекању 2*MLS. У том случају сегменти који су у кашњењу припадали вези пре квара система могу се погрешно интерпретирати да припадају новој вези креираној после поновног подизања система. Ово се може десити независно од тога који је иницијални редни број употребљен после подизања система.

Да би се заштитили од овакве грешке у документу RFC793 је предложено да TCP процес првих MSL секунди после подизања не треба да формира ниједну везу. Ово време назива се време мировања.

6.5.2 Стање FIN WAIT2

У стању FIN WAIT2 извориште шаље сегмент *Fin_Ack* и одредиште га потврђује. Ако није урађено полузатварање везе чека се на апликацију са другог краја везе да препозна да је добила EOF[1] обавештење и да она затвара свој крај везе (*Close*) што доводи до слања сегмента *Fin_Ack*. Само када одредиште уради затварање везе изворишни крај прелази из стања FIN WAIT2 у стање TIME WAIT.

Ово значи да изворишни крај везе може да остане у овом стању у недоглед. Друга страна је и даље у CLOSE WAIT стању и може у њему да остане заувек, односно све док апликација не затражи затварање везе.

Многе апликације засноване на беркли имплементацији спречавају ово могуће бесконачно чекање у стању FIN WAIT2 на следећи начин: уколико апликација иницира активно затварање везе онда она извршава потпуно затварање, а не полузатварање, указујући да очекује пријем података а затим активира часовник. Уколико је веза неактивна 10min плус 75s, TCP процес помера везу у стање CLOSED. Овакав приступ није у сагласности са спецификацијом TCP протокола.

6.6 Захтев за успоставу TCP везе са непостојећим портом

Споменули смо да је део TCP заглавља и бит RST[2] (слика 6.6). У

[1] *End Of File*
[2] *Reset*

општем случају *Reset*[1] сегмент се шаље увек када стигне сегмент који не одговара тој вези.

Уобичајен случај генерисања сегмента *Reset* је када стигне захтев за успоставу везе а не постоји процес који „ослушкује" на одредишном порту. Када је у питању UDP протокол онда се шаље ICMP[2] порука „одредишни порт је недоступан". TCP протокол уместо тога користи сегмент *Reset*.

6.7 Нагло прекидање TCP везе

Видели смо да је нормалан начин да се веза оконча да једна од страна пошаље *Fin_Ack* сегмент. Овај начин некада се среће под називом нормално ослобађање (прекид) везе. Сегмент *Fin_Ack* се шаље тек када се претходно сви подаци који су били спремни за слање (смештени у реду за чекање) пошаљу. На тај начин не постоји могућност губитка података.

Могуће је окончати везу слањем сегмента *Reset* уместо *Fin_Ack* сегмента. То се назива нагло прекидање везе[3]. Нагли прекид везе доводи до тога да:

- се било који подаци који су у реду чекања за слање одмах одбацују,
- пријемник сегмента *Reset* зна да је удаљена страна нагло прекинула везу,
- API[4] који користи апликација мора да обезбеди могућност наглог прекида уместо нормалног прекида везе.

6.8 Откривање полузатворене TCP везе

TCP се налази у полузатвореном стању уколико је један крај везе затворио везу (нормално или нагло) без обавештавања друге стране о томе. Ово може да се деси када дође до квара на било којој страни[5] везе.

---

[1] То је сегмент код кога је RST=1.

[2] *Internet Control Message Protocol*

[3] *Abortive release*

[4] *Application Interface*

[5] Ако је веза успостављена између сервера или клијента онда било сервер било клијент. Ако је веза успостављена између две станице било која од њих.

Уколико не постоји покушај стране која је у квару да пошаље податке преко полуотворене везе страна која је још увек у отвореној вези неће открити да је друга страна у квару.

Други уобичајен узрок полуотворене везе је када станица на страни клијента искључи напајање уместо да оконча клијентску апликацију а тек затим регуларно искључи (обори) систем.

Ово се на пример дешава када је на персоналном рачунару покренут *Telnet* клијент, а корисник искључи рачунар на крају дана. Уколико у тренутку искључивања рачунара није био у току пренос података сервер никада неће сазнати да клијент није више на другој страни везе. Када се корисник појави следећег јутра, укључи свој рачунар, покрене новог *Telnet* клијента и отвори нову везу са сервером то може да доведе до превише полуотворених веза на серверу[1].

6.9 Истовремено отварање TCP везе

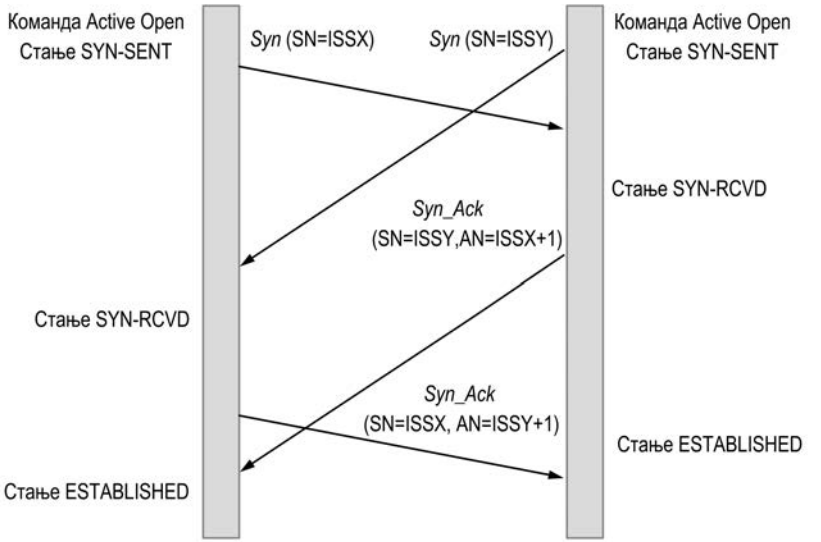
Могуће је, иако мало вероватно, да апликације на оба краја изврше истовремено једна ка другој команду *Active Open*. Сваки крај мора да пошаље свој *Syn* сегмент. Такође се захтева да је сваком од крајева везе познат локални порт другог краја везе. Ово се назива истовремено отварање[2].

На пример једна апликација на страни станице 2 може да има локални порт 6000 и извршава команду *Active Open* са бројем порта 8000 на станици 1. Апликација на станици 1 има као број локалног порта 8000 и извршаваће команду *Active Open* са бројем порта 6000 на станици 2.

Ово није исто као када се истовремено повезују *Telnet* клијент на станици 1 са *Telnet* сервером на станици 2 и *Telnet* клијент на станици 2 са *Telnet* сервером на станици 1. У овом случају оба *Telnet* сервера извршавају команду Passive Open а не команду *Active Open*. *Telnet* клијенти додељују себи пролазне (ефемерне) бројеве портова а не бројеве портова који су познати другим *Telnet* серверима.

TCP протокол је пројектован тако да може да разреши истовремено отварање везе. Правило је да се као резултат добија само једна а не

[1] Постоји начин да један крај TCP везе открије да је други нестао уз помоћ опције „TCP *keepalive*".
[2] *Simultaneous Open*



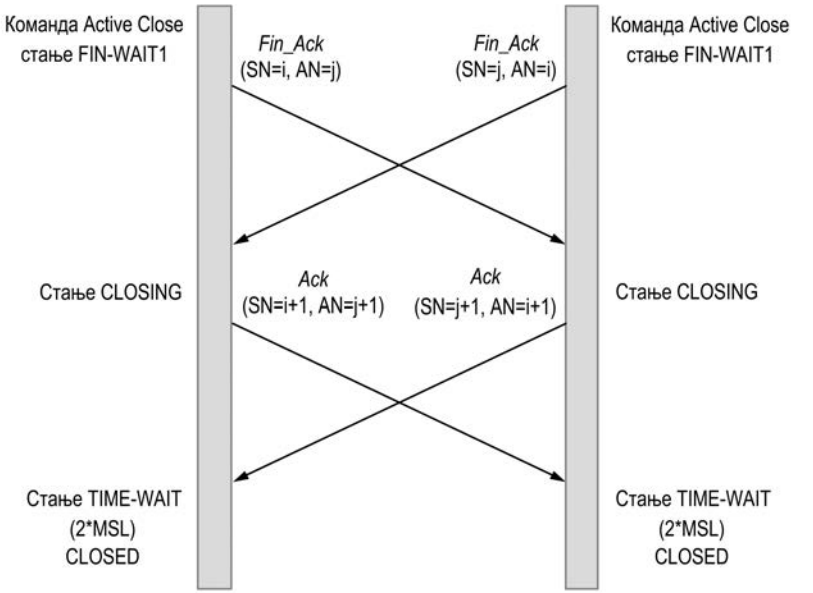
Слика 6.26 Размена сегмената за време истовременог отварање везе

две везе. Код неких других стандарда, као што је на пример транспортни протокол код OSI референтног модела, као резултат истовременог отварања дао би две отворене везе. Када дође до истовременог отварање везе прелаз из једног стања у друго је различит од оног на слици 6.16. Сценарио је следећи (слика 6.26):

- оба краја шаљу *Syn* сегмент у исто време и улазе у SYN SENT стање са својим иницијалним редним бројевима (на слици 6.26 су означени са ISSX и ISSY),
- када сваки од крајева прими *Syn* сегмент прелази у стање SYN RCVD и шаље свој *Syn_Ack* сегмент чиме потврђује примљени *Syn* сегмент из супротног правца,
- по пријему *Syn_Ack* сегмента са потврдом оба краја везе прелазе у стање ESTABLISHED.

6.10 Истовремено затварање TCP везе

Споменули смо у претходном поглављу да једна страна (уобичајено је да је то клијент) извршава активно затварње везе проузрокујући слање *Fin_Ack* сегмента. Могуће је да обе стране извршавају команду активног



Слика 6.27 Размена сегмената за време истовременог затварања везе

затварања и да се TCP протоколу дозвољава истовремено затварање везе (слика 6.27).

Када апликације пошаљу примитиве *Close* (слике 6.19, 6.25) TCP целине на страни сервера и клијента из стања ESTABLISHED прелазе у стање FIN WAIT1и шаљу *Fin_Ack* сегменте. Сваки од крајева везе када прими *Fin_Ack* супротне стране прелази из стања FIN WAIT1 у стање CLOSING и шаље завшни *Ack* сегмент. Кад сваки од TCP процеса на крајевима везе прими одговарајући *Ack* сегмент прелази у стање TIME WAIT. Са истовременим затварањем везе исти број сегмената се размењује као и у нормалном затварању везе.

6.11 Методе у размени података

TCP стандард обезбеђује прецизне спецификације за TCP протокол који се користи за размену података између станица. У овом поглављу анализираћемо различите методе (стратегије). Они се могу категоризовати на следећи начин:

- метод за слање података[1],
- метод за испоруку података[2],
- метод за прихватање података[3],
- метод за поновно слање података[4],
- метод за потврде о пријему података[5].

6.11.1 Метод за слање података

TCP целина је слободна да изабере начин на који ће да шаље податке у границама кредита[6] који поседује у случају да се не ради о подацима које треба одмах испоручити[7].

Када корисник проследи, податке TCP целини она их смешта у предајни меморијски простор. TCP целина може да направи сегмент од сваке појединачне групе података које јој корисник проследи или да сачека да се скупи одређена количина података пре него што направи сегмент (слика 6.4). Метод (начин) који одабира биће прилагођен тренутним захтевима везе.

Ако је пренос података:

- велики и неравномеран постојаће мала количина додатних некорисних података[8] (премашење[9]),
- систем код кога се често шаљу сегменти мале величине биће бржи у обради података и одговору на њих.

6.11.2 Метод испоруке података

Када се не користи функција за тренутно испоручивање података кориснику, TCP целина може да испоручује податке кориснику према свом плану, односно:

- може да испоручује податке кориснику оним редом којим они стижу или,

[1] *Send Policy*
[2] D*eliver Policy*
[3] *Accept Policy*
[4] *Retransmit Policy*
[5] *Acknowledge Policy*
[6] Отвора прозора
[7] Са постављеним PSH битом на вредност 1..
[8] Под појмом корисни подаци подразумевају се само подаци (*data, payload*) који се преносе. Подаци који нису корисни, тзв. некорисни, односе се на заглавље.
[9] *Overhead*

- може прво да прикупи одређену количину података у свој пријемни бафер па да их онда све заједно пошаље кориснику.

Који ће се метод користити зависи од тренутних захтева. Ако је интензитет преноса података неравномеран и ако се преноси велика количина података може се десити да корисник не добија податке довољно често како би било пожељно. С друге стране, ако је интензитет преноса равномеран и на тај начин се преносе мали сегменти, може да дође до непотребне обраде сваког сегмента код предајне и код пријемне стране, па и до непотребних честих прекида које пријемна TCP целина шаље кориснику да би преузео податке.

6.11.3 Метод за прихватање података

TCP целина прима податке оним редом којим су они стигли и смешта их у пријемни меморијски простор да би их испоручила кориснику. Може да се деси да подаци не стигну редоследом којим су послати. У том случају TCP целина може да примени два метода:

- метод „по реду"[1] који омогућава прихватање само оних сегмената који стижу у правилном редоследу,
- метод „у оквиру прозора"[2] који омогућава прихватање сегмената који стижу у складу са тренутним отвором пријемног прозора.

Метод који обезбеђује прихватање само оних сегментата који стигну у правилном редоследу (на пример SN=1, SN=201, SN=401...) лако је применити. Последица примене овог метода је велики број ретрансмисија. Могуће је да се деси да неки сегменти не стигну на одредиште у право време (на пример, сегмент са редним бројем 201 може да стигне после сегмента са редним бројем 401). У том случају пријемна TCP целина одбацује тај сегмент. Када време за добијање потврде о пријему истекне предајна TCP целина поново шаље тај сегмент.

Значи да ако је само један сегмент стигао касније сви сегменти који стигну после њега бивају одбачени и предајник ће морати поново да их пошаље. Ово није добро решење јер циљ TCP протокола јесте да обезбеди што мање ретрансмисија.

---

[1] *In-order*

[2] *In-window*

Метод код ког се прихватају сви сегменти са редним бројевима, који су у оквиру пријемног прозора, много је прихватљивији. Разлог је једноставан: битно смањује потребу за ретрансмисијом. Мана овог метода је што је сложенији за реализацију. Мора се стално проверавати да ли примљени сегмент испуњава услов и мора се водити рачуна који су сегменти стигли изван правилног редоследа.

6.11.4 Метод за поновно слање података

TCP целина води рачуна о реду који чине сегменти који су послати али нису још потврђени. Када потврда за сегмент који је послат не стигне у одређеном времену[1] предајна TCP целина га поново шаље. Постоје три метода за поновно слање:

- Слање само првог сегмента[2] - постоји само један часовник додељен реду за ретрансмисију. Када стигне потврда (*Ack* сегмент) да је одређени сегмент успешно примљен онда се тај сегмент уклања из реда за ретрансмисију и часовник се поново покреће. У случају да време истекне пре него што стигне потврда о успешном пријему било ког сегмента из реда поново се шаље први сегмент из реда и часовник се поново покреће.
- Слање групе сегмената[3] - постоји само један часовник додељен реду за ретрансмисију. Када се прими потврда о пријему одређеног сегмента онда се тај сегмент уклања из реда за ретрансмисију и часовник се поново покреће. У случају да време на које је часовник подешен истекне пре него што стигне потврда о пријему било ког сегмента из реда онда се сви сегменти из реда поново шаљу а часовник се поново покреће.
- Слање појединачних сегмената[4] - сваком сегменту који се налази у реду за ретрансмисију додељен је по један часовник. Када се за одређени сегмент прими потврда о пријему, онда се тај сегмент уклања из реда за ретрансмисију а њему додељен часовник зауставља. У случају да време за пријем потврде

[1] Време за добијање потврде о пријему одређеног сегмента истекне пошто је реч о одговарајућем часовнику који се поставља на одређену вредност.

[2] *First only*

[3] *Batch*

[4] *Individual*

истекне на неком од часовника, онда се сегмент на који тај часовник указује поново шаље а часовник поново покреће.

Првоописани метод (слање само првог сегмента) ефикасан је по питању броја ретрансмисија, јер се поново шаљу само они сегменти за које није стигла потврда о пријему. Недостатак овог метода је тај што часовник за ретрансмисију указује само на први сегмент у реду. То значи да није могуће покренути часовник за сегмент који је други у реду док се не добије потврда о пријему за сегмент који је први у реду за ретрансмисију, чиме се уноси одређено кашњење. Метод са слањем појединачних сегмената разрешава овај проблем, али је доста комплекснији за имплементацију. Слање групе сегмената такође може да реши поменути проблем, али може и да створи нов - велики број сегмената који се непотребно поново шаљу.

Ефективност изабраног метода за ретрансмисију директно зависи од метода (стратегије) за пријем који користи пријемна страна. Ако пријемна страна користи:

- метод редоследа онда ће она одбацити сегменте који су по реду иза изгубљеног сегмента. Тако је у овом случају боље на предајној страни применити метод са групом сегмената јер ће се на тај начин поново послати сви сегменти за које није стигла потврда, а међу њима и сегмент који је изгубљен;
- метод отвора прозора онда је најбоље на предајној страни користити метод слања првог сегмента, или слање појединачних сегмената.

6.11.5 Метод за потврду пријема података

Када пријемна страна прими сегмент који је правилног редоследа[1] има две опције везане за метод (начин) како ће реаговати:

- Тренутно слање потврде - одмах шаље празну потврду (сегмент *Ack*) без података и са одговарајућим потврдним редним бројем (AN).
- Кумулативно слање потврде - када прими податак пријемна TCP целина запамти да је потребно послати потврду. Потврду

[1] Правилан редослед подразумева да пакети стижу истим редоследом којим су и послати, односно са редним бројевима који су узастопни и растући.

не шаље одмах већ ће сачекати први свој сегмент са подацима који се шаље другој страни. Редни број потврде биће уписан у заглавље тог сегмента[1]. Да би се спречила дугачка кашњења треба активирати часовник за отвор прозора (табела 6.1). Уколико часовник сигнализира истицање времена на које је подешен пре него што је *Ack* сегмент послат треба послати празан сегмент који садржи одговарајући број сегмента.

Метод тренутног слања потврде једноставан је и обезбеђује да је удаљена TCP целина потпуно информисана чиме се ограничавају беспотребне ретрансмисије. Овај метод доводи до слања додатних празних, искључиво *Ack* сегмената. Такође може да изазове додатно повећање саобраћаја и самим тим и оптерећење мреже.

Замислимо да TCP целина прими сегмент и одмах пошаље потврду (*Ack*). Подаци се одмах прослеђују апликацији, што обично води повећању пријемног прозора, изазивајући слање још једног празног сегмента који даје додатни кредит предајној TCP целини. Због могућег преоптерећења обично се користи кумулативни метод. Треба уочити да је кумулативни метод сложенији за реализацију и израчунавање укупног времена потребног да се пренесе податак и да стигне потврда за њега (RTT[2]).

6.12 Интерактивни ток података

Анализа TCP саобраћаја показује да се он састоји од две врсте сегмената:

- TCP сегмент који садржи велику количину[3] података који се преносе код апликација као што је пренос датотека (FTP), електронска пошта (SMTP) и сл.
- TCP сегмент који садржи интерактивне податке који се срећу код апликација *Telnet* , *X-Windows*, *Rlogin* и сл.

Ако би се посматрала заступљеност ове две врсте сегмената показује се да 90% саобраћаја чине TCP сегменти који носе велику количину података и обично су максимално могуће величине (бар 512

[1] Као што је у претходним поглављима већ поменуто овај мехнизам се на енг. означава са *piggy back.*
[2] *Round Trip Time*
[3] *Bulk*

бајтова корисничких података). Само 10% TCP сегмената су они који носе интерактивне корисничке податке и обично значајно мање величине[1].

Као пример узећемо *Telnet* везу са интерактивним едитором који реагује на сваки тастер. Размотрићемо метод најгорег случаја када TCP целина пошиљаоца прими један карактер, формира сегмент дужине 21 бајт[2] и прослеђује га IP целини.

IP целина додаје своје заглавље и шаље га као IP пакет дужине 41 бајт[3] (слика 6.28). Пријемник одмах шаље потврду дужине 40 бајтова. Касније, када едитор прочита примљени бајт, TCP целина шаље ажурирану величину отвора прозора померајући отвор прозора један бајт удесно. Овај пакет је такође дужине 40 бајтова. Коначно, након што едитор обради карактер шаље одјек (exo) карактера као пакет дужине 41 бајт. Види се да су за сваки откуцани карактер употребљена 162 бајта и послата четири сегмента. Уколико је пропусни опсег система мали овај метод је неприхватљив.

6.12.1 Неиглов алгоритам

Један од метода за побољшање ефикасности пошиљаоца познат је као Неиглов[4] алгоритам[5]. Неигл је предложио једноставан метод: када подаци стижу до пошиљаоца бајт по бајт, он шаље само први бајт, а све преостале бајтове складишти док не стигне потврда за бајт који је послат. Затим шаље све ускладиштене карактере у једном TCP сегменту и започиње складиштење поново све док сви послати карактери не буду потврђени. Предност овог метода је у томе што је самотактујући. Што потврде брже стижу карактери се брже шаљу. Када се сегменти шаљу преко споре рачунарске мреже ширих подручја[6] корисно је смањио број малих сегмената[7] који су присутни у мрежи. Када се користи Етернет локална рачунарска мрежа онда је 16ms средње време које је потребно

---

[1] Поменуте интерактивне апликације као што су нпр. *Telnet, Rlogin* носе у просеку по 10 бајтова корисничких података (литература [6]).

[2] Податак је дужине 1 бајт (слика 6.28) и заглавље TCP сегмента је 20 бајтова.

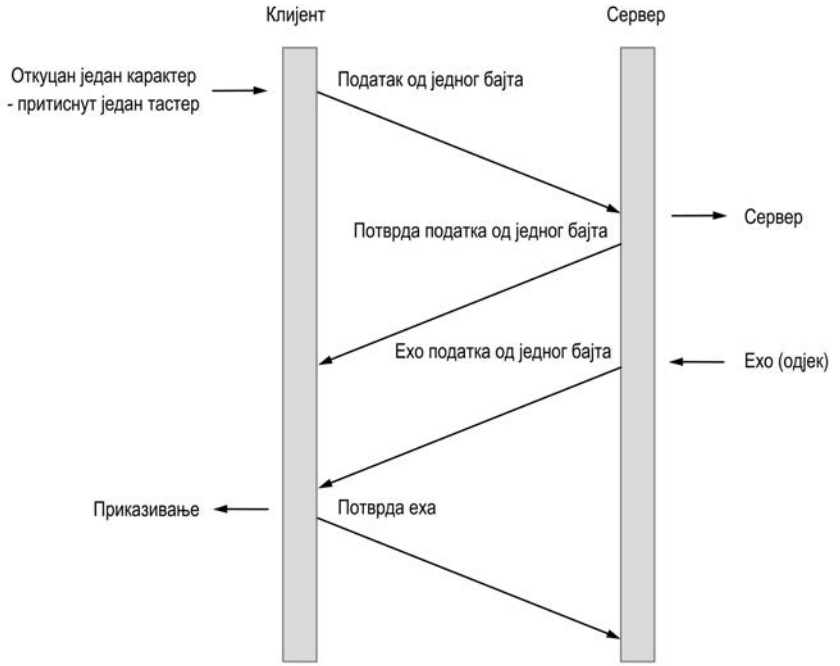
[3] Заглавље IP датаграма је 20 бајтова. Заједно са упакованим (енкапсулираним) TCP сегментом је укупно 20+21=41. Премашење је 4000%.

[4] *Nagle*

[5] Документ RFC896 издат 1984. год.

[6] WAN (*Wide Area Network*)

[7] *Tiny*



Слика 6.28 Један могући начин за размену сегмената
код интерактивне апликације

да се: пошаље један бајт, добије његова потврда и његов ехо[1] [6]. Да би генерисао податке веће брзине од тог времена корисник би морао да откуца више од 60 карактера у секунди што је мало вероватно. Ово значи да се Неиглов алгоритам ређе користи у локалним рачунарским мрежама[2].

Алгоритам додатно омогућава слање новог пакета уколико пристигне довољно података да би се попунила половина отвора прозора или максималан сегмент.

Неиглов алгоритам се доста користи али није најбоље решење за неке апликације. На пример код *X-Windows* апликација (које су значајно заступљене на Интернету) покрети миша треба да се пошаљу ка удаљеном рачунару. Сакупљање ових покрета и слање у групи (одједном) чини кретање миша неправилним.

[1] Детаљније се може наћи у литератури [8].

[2] Постоји могућност да се он по потреби искључи.

6.13 Пренос велике количине података

Код преноса велике количине података[1] TCP протокол обезбеђује контролу тока са применом механизма клизајућег прозора[2] (слика 5.14). Дозвољава се пошиљаоцу да пошаље групу сегмената а и да очекује потврду за послату групу сегмената[3]. Наравно пренос је значајно убрзан у односу на метод заустави се и чекај[4].

Пријемни процес обично може контролисати величину прозора коју дозвољава. Величина прозора[5] клијента и сервера може да буде 2048, 4096[6], 8192 или чак 16384 бајта у зависности од верзија оперативног система. Величина пријемног меморијског простора једнака је максималној величини пријемног прозора за ту везу. За Етернет локалне рачунарске мреже бафер величине 4096 бајтова (и на предајној и на пријемној страни) није оптималан. Пропусна моћ система порасла је за 40% када је меморијски простор (бафер) повећан на 16384 бајтова[7].

6.13.1 Убрзана испорука на пријему

Функција са постављањем PUSH ознаке омогућава TCP корисницима да врше контролу својих меморијских простора.

Употреба меморијских простора повећава ефикасност тако што омогућава предајној TCP целини да може да шаље мање већих сегмената уместо више мањих. На тај начин је мањи број обрада сегмената и на страни пошиљаоца и на страни примаоца. Пријемна TCP целина смањује број захтева који се шаљу апликацији за преузимање података. Негативна страна употребе меморијског простора је што доводи до повећања кашњења у испоруци података.

Ако се користи PUSH функција онда је апликацији омогућено да заобиђе складиштење података а тиме се и кашњење смањује.

---

[1] *Bulk data flow*

[2] *Sliding Window*

[3] Детаљно је објашњено у претходним поглављима.

[4] Метод „заустави се и чекај" (*stop and wait*) објашњен је у 11. поглављу литературе [1]. Користи га једноставан транспортни протокол за пренос датотека TFTP (*Trivial File Transport Protocol*).

[5] Користи се термин објављена величина прозора (*advertised window size*), може се наћи у литератури [6].

[6] Сматра се подразумеваном (*default*) вредношћу.

[7] Детаљније се може наћи у литератури [7].

У оригиналној TCP спецификацији предвиђено је да апликација може да утиче на TCP целину да постави PSH бит на вредност један. У интерактивним апликацијама када клијент шаље команду серверу клијент ће поставити бит PSH=1. Дозвољавање клијентској апликацији да наложи TCP целини постављање PSH бита служи да обавести клијентску TCP целину да процес на страни клијента не жели да се подаци задржавају у меморијском простору пријемника. Када на страни сервера TCP целина прихвати сегмент са постављеном PUSH ознаком то је њој обавештење да одмах пошаље податке апликацији а не да чека да ли ће пристићи још неки подаци.

Многе данашње апликације не подржавају могућност постављања PSH ознаке. Добро конципиран TCP процес сам процењује када треба да постави PSH ознаку. Већи број апликација изведених из беркли имплементације аутоматски постављају PSH ознаку уколико послати сегмент празни предајни меморијски простор. На пријему ту PSH ознаку игноришу пошто никада не касне са испоруком примљених података (одмах их испоручују).

6.13.2 Хитна испорука

TCP целина обезбеђује механизам хитне испоруке[1] који омогућава једном крају да обавести други крај да су „хитни" подаци смештени у нормалан ток података. На пријемнику је да одлучи шта ће и како да уради са тим подацима.

Обавештавање о постојању хитних података обавља се постављањем два поља у TCP заглављу (слика 6.6). Бит означен са URG постављен је на 1 и показивач хитних података је постављен на позитивну вредност (одступање), којој се додаје редни број у TCP заглављу да би се добио редни број последњег бајта хитних података.

TCP целина мора да обавести пријемни процес када стигне сегмент са показивачем хитних података. Апликација на пријему ће очитати податке и она мора да буде у стању да процени како да искористи показивач хитних података.

Сама TCP целина веома мало казује о хитним подацима. Не постоји начин да се покаже где је почетак хитних података у самом току података.

[1] *Urgent mode*

Једина информација која се шаље кроз TCP везу је да се ради са хитним подацима (бит URG у TCP заглављу постављен на 1) и показивач последњег бајта хитних података. Све остало је препуштено самој апликацији.

Поставља се питање која је намена коришћења рада са хитним подацима? Две уобичајене апликације су *Telnet* и *Rlogin* и случај када интерактивни корисник притисне тастер за прекид. Други пример је FTP када интерактивни корисник прекида пренос датотека.

Telnet и *Rlogin* користе рад са хитним подацима од стране сервера ка клијенту пошто је могуће да тај правац података заустави клијентска страна TCP везе (огласи прозор 0). Уколико процес на страни сервера користи врсту рада са хитним подацима, TCP целина на страни сервера одмах шаље показивач и ознаку хитних података иако не може да шаље никакве податке. Када TCP целина на страни клијента прими информацију, она као одговор обавештава процес на страни клијента тако да клијент може да чита са сервера, отвори прозор и дозволи ток података.

Шта се дешава ако пошиљалац неколико пута започиње рад са хитним подацима пре него што пријемник обради све податке закључно са првим показивачем? На пријемној страни губи се информација о претходној позицији показивача. Постоји само један показивач хитних података на пријему и новопристигли показивач се уписује преко претходног. Ово значи да уколико су подаци важни онда изворište мора специјално да их обележи. На пример *Telnet* све бајтове команди у току података обележава бајтом префикса чија је вредност 255.

6.13.3 Успорени почетак

У претходним одељцима претпостављали смо да на самом почетку предајник шаље групу сегмената. Максимална величина групе одговара величини (објављеног) прозора пријемника. Када су станице на истој локалној рачунарској мрежи овакав начин рада је прихватљив. Уколико су предајна и пријемна станица повезане преко рутера а неке од међувеза су мање брзине, може да дође до одређених проблема.

На пример, неки од рутера примљене пакете стављају у редове[1] за чекање. Уколико рутер не стигне да обради пакете из реда, тј. уколико

[1] Редови за чекање су у ствари резервисани меморијски простор (бафер).

се не ослободи простор у прихватном меморијском простору онда су новопристигли пакети изгубљени. На овај начин може се значајно смањити пропусна моћ TCP везе[1].

Алгоритам под називом успорени почетак[2] има за циљ превазилажење поменутих проблема. Принцип на коме ради алгоритам је следећи: брзина којом се пакети убацују у мрежу одређена је брзином којом одредиште враћа потврде.

Спори почетак додаје још један прозор предајној TCP целини: прозор загушења[3]. После успоставе везе пошиљалац одређује величину прозора загушења на основу величине сегмента који је у тој вези у употреби[4] и шаље само један сегмент те величине. Ако потврда (*Ack* сегмент) стигне пре истека часовника загушења онда се прозор загушења увећава за величину још једног сегмента. Шаљу се два сегмента и чека потврда (слика 6.29). Када се сваки од ових сегмената потврди прозор загушења се увећава за четири. На овај начин обезбеђен је експоненцијални раст прозора загушења. У неком тренутку се достигне капацитет Интернета и неки од рутера почне да одбацује пакете. То је знак предајнику да је прозор загушења отишао предалеко и да се треба зауставити. На пример: ако количина од 1024, 2048 и 4096 бајтова пролази кроз мрежу без проблема а количина од 8192 бајта изазива кашњење, прозор загушења биће подешен на 4096 бајтова да би се избегло загушење.

Колика је та максимална вредност изражена у бајтовима која може да се пошаље? Број бајтова који може да се пошаље одговара броју бајтова мањег прозора. То значи да је величина прозора једнака величини мањег од прозора: објављеног прозора (прозора примаоца) или прозора загушења (прозора пошиљаоца). Ако прималац захтева слање 8kB података, а пошиљалац зна да количина података већа од 4kB изазива преоптерећење мреже, шаље се 4kB. У другом случају, ако прималац захтева слање 8kB а пошиљалац зна да количина до 32kB без проблема пролази кроз мрежу, шаље се 8kB. Можемо да закључимо да:

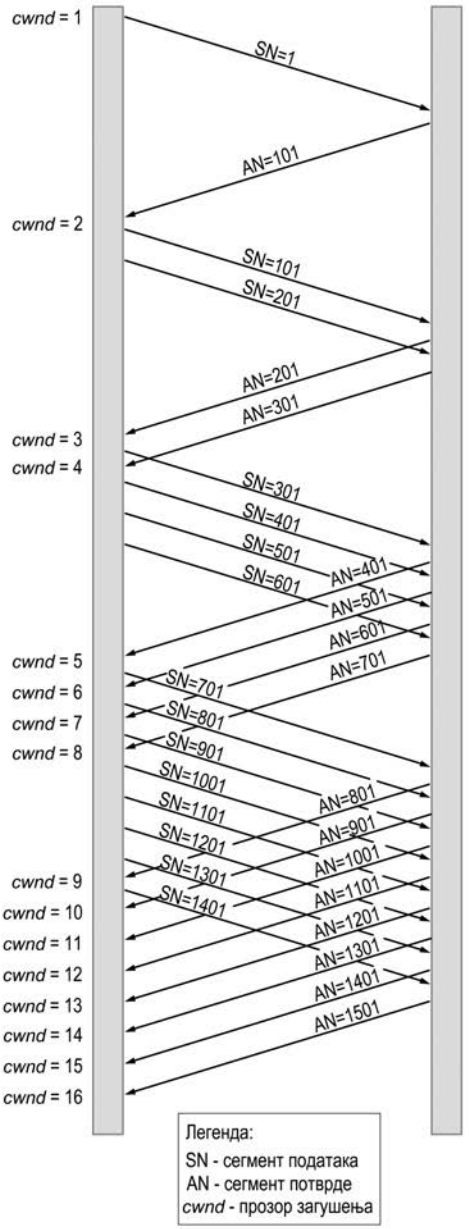
---

[1] Детаљнији опис може се наћи у литератури [12].

[2] *Slow start*

[3] *Congestion window*

[4] Односно максималне величине сегмента који је објављен у тој вези.



Слика 6.29 Пример успореног почетка

- прозором загушења предајник реализује механизам контроле тока предајника,
- објављеним прозором пријемник реализује мехнизам контроле тока пријемника.

У једном од следећих поглавља видећемо како се успорени почетак уобичајено примењује уз метод који се назива заобилажење загушења[1].

6.13.4 Пропусна моћ код преноса велике количине података

У овом делу анализираћемо међусобни утицај величине прозора, контроле тока и успореног почетка.

Слика 6.30 приказује у времену поједине кораке између пошиљаоца и примаоца. Приказано је шеснаест дискретних тренутака у времену. Сегменти који носе податке означени су са C_1, C_2, C_3 итд. Потврде ових сегмената означене су са A_1, A_2, A_3 итд.

У тренутку t=0 предајна TCP целина шаље један сегмент податка. Пошто је у режиму „успорени почетак" (прозор загушења је величине једног сегмента податка) предајна TCP целина мора да сачека да стигне потврда за овај сегмент па тек онда да настави са слањем. У тренуцима t=1, t=2 и t=3 сегменти података се померају за по једну временску јединицу удесно. У тренутку t=4 пријемник очитава сегмент податка и шаље потврду A_1. У тренуцима t=5, t=6 и t=7 потврда A_1 се помера улево за по једну временску јединицу. Добија се да је време потребно да се пошаље сегмент податка и да за њега стигне потврда RTT= 8 временских јединица.

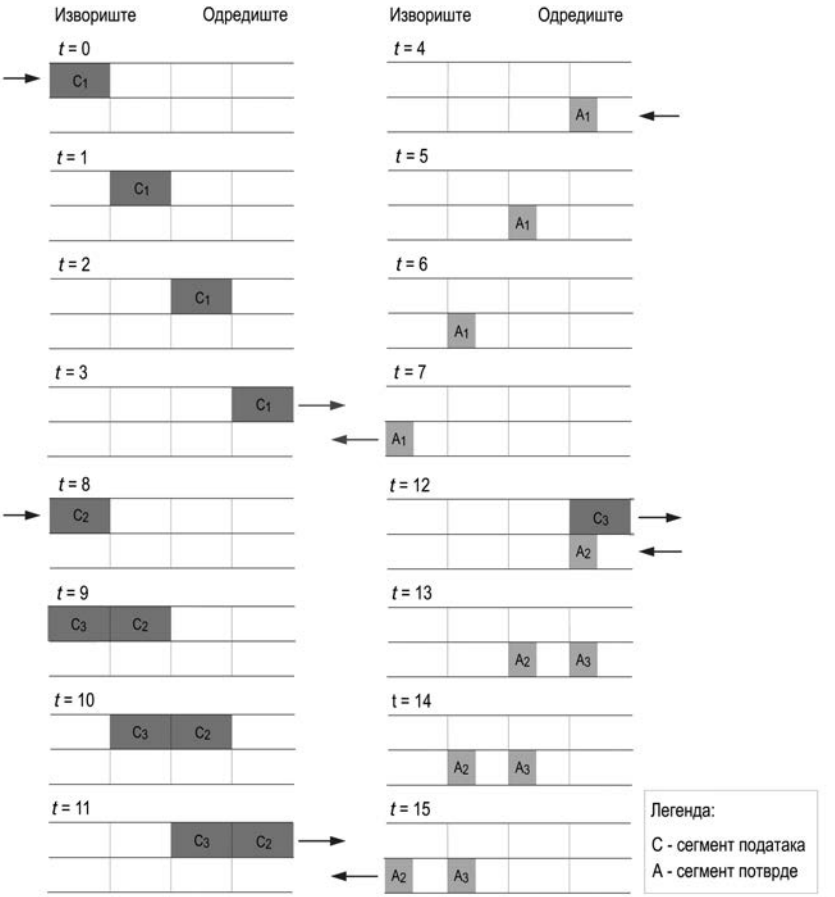
На слици 6.30 су нацртани сегменти потврде (A_x) мање величине од сегмената података (C_x) пошто се они састоје само од IP и TCP заглавља. Приказан је само пренос података у једном правцу. Такође смо претпоставили да је брзина преноса у оба правца иста (што није увек тачно). У општем случају време потребно да пакет стигне на одредиште састоји се од:

- Времена простирања[2] (пропагације) - назива се и кашњење због простирања[3]. Брзина пропагације зависи од физичких

[1] *Congestion avoidance*

[2] У рачунарским мрежама време простирања (кашњење услед простирања) је време потребно да први бит (почетак сигнала) стигне од пошиљаоца до примаоца. Може се израчунати као однос између дужине везе и брзине простирања преко одређеног трансмисионог медијума.

[3] *Propagation delay*



Слика 6.30 Временски тренуци *t*=0 до *t*=15 у примеру преноса велике количине података

карактеристика трансмисионог медијума (да ли је оптичко влакно, бакарни проводник, итд). Креће се у опсегу од $2x10^8$m/s за бакарне проводнике до $3x10^8$m/s за бежичне комуникације, што је једнако брзини светлости. Може се израчунати као:

време простирања = раздаљина (дужина трансмисионог медијума)/ брзина простирања;

- Времена потребног да се пошаље пакет[1] (време слања) - зависи од брзине преноса (колико битова у секунди медијум може да пренесе) и величине пакета (изражене бројем битова). Назива се и кашњење због преноса[2] и на њега нема никаквог утицаја растојање између предајника и пријемника. Може се изразити као:

  време слања = величина пакета/ брзина преноса.

За одређену везу (путању) кашњење због времена простирања је константно, али кашњење због преноса зависи од величине пакета. При малим брзинама преноса доминантно је кашњење због преноса, док је при великим брзинама преноса (нпр. брзине реда Gb/s) доминантан је утицај времена простирања.

Када пошиљалац прими сегмент потврде (A_1) он може да пошаље два додатна сегмента података (на слици 6.30 означена са C_2 и C_3). У тренуцима *t*=8 и *t*=9 прозор загушења је два сегмента података. Ова два сегмента података крећу се удесно ка примаоцу који у тренуцима *t*=12 и *t*=13 шаље A_x сегменте (сегменте потврде). Временски распоред сегмената потврде идентичан је са временским распоредом сегмената података и назива се самотактовање TCP процеса. Пошто пријемник може да генерише потврду (A_x сегмент) тек кад пристигну подаци распоред A_x сегмената указује на брзину пристизања података на пријемној страни[3].

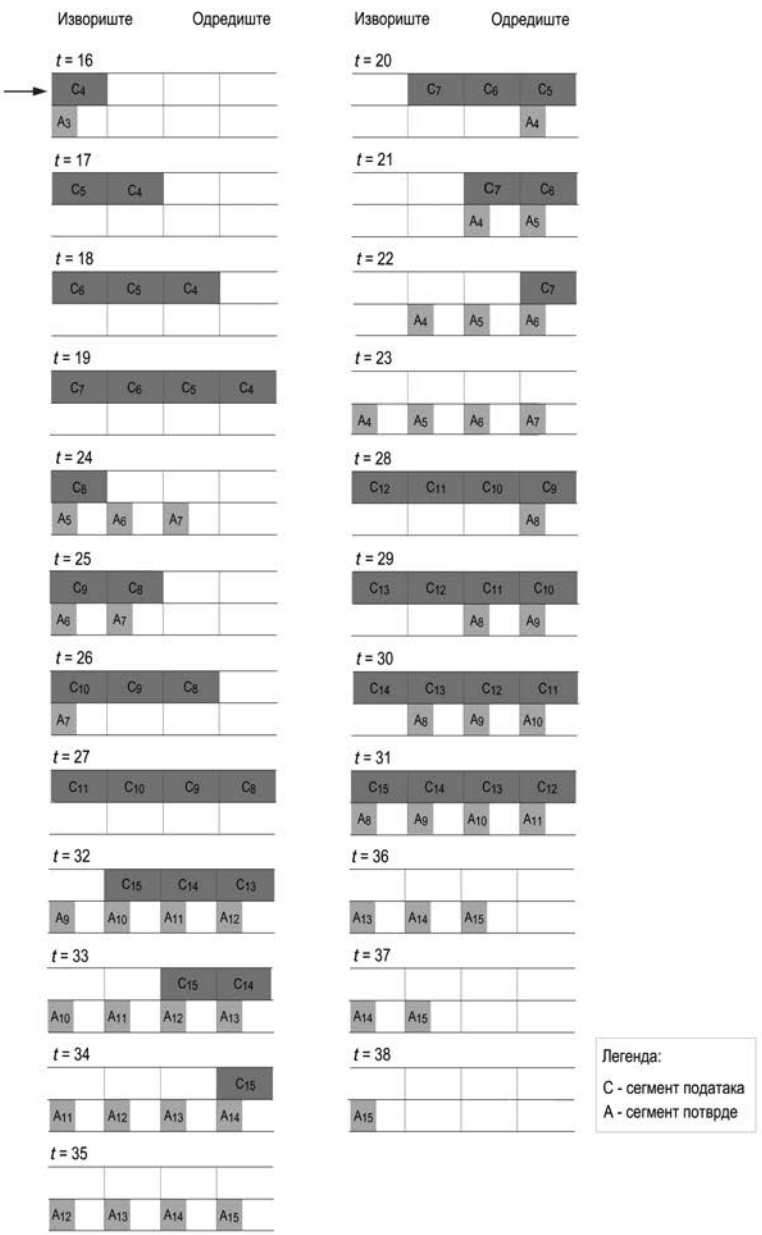
На слици 6.31 приказано је следећих 16 временских јединица. Када стигну две потврде повећава се прозор загушења са два на четири сегмента и четири сегмента података (означена као C_4, C_5, C_6 и C_7) шаљу се у тренуцима *t*=16 до *t*=19. Прва потврда пристиже у тренутку означеном са *t*=25. Четири сегмента потврде увећавају прозор загушења са четири на осам и сегменти података означени као C_8, C_9, C_{10}, C_{11}, C_{12}, C_{13}, C_{14}, и C_{15} шаљу се у тренуцима *t*=24 до *t*= 31.

У тренутку *t*=31 и свим следећим тренуцима, пут између пошиљаоца и примаоца је попуњен подацима. Не може да прихвати више података без

[1] У комуникационим системима време потребно да се пошаље порука је време које протекне од тренутка када се започне слање поруке па док се не заврши слање поруке. У случају дигиталних порука то је време које протекне од тренутка када се први бит поруке (пакета) пошаље на линију до тренутка када се последњи бит поруке пошаље на линију.

[2] *Transmission delay*

[3] У стварности редови чекања у повратку могу променити брзину пристизања потврде, тј. Ах сегмената.



Слика 6.31 Тренуци *t*=16 до *t*=38 у преносу веће количине података

обзира на прозор загушења или објављени прозор (примаочев). У свакој јединици времена прималац уклања један сегмент са мреже а пошиљалац убацује други сегмент у мрежу. Између пошиљаоца и примаоца у једном правцу се преносе више сегмената података а у супротном правцу једнак број сегмената потврде. Ово је идеално стабилно стање везе.

Производ пропусног опсега и кашњења

Сада бисмо могли да одговоримо на питање колико велики отвор прозора може да буде? У нашем примеру за максималну пропусну моћ пошиљалац треба да у сваком тренутку има осам непотврђених сегмената чији је пренос у току. Објављени прозор мора бити исто толико велики пошто толико сегмената пошиљалац може да пошаље. Можемо израчунати капацитет везе (пута) на следећи начин:

$$\text{капацитет [у битима]} = \text{пропусни опсег [b/s]} * \text{RTT[s]} \quad (6.1)$$

Ова вредност може да се мења у широком опсегу у зависности од брзине трансмисионих медијума у рачунарској мрежи и од укупне вредности кашњења RTT између две посматране тачке мреже. На пример за T1 линије[1] са једног на други крај САД и време RTT које се рачуна да је око 60ms добија се капацитет[2] од 11580 бајт. ова. Ово се сматра прихватљивим[3] са становишта величине меморијског простора. Међутим уколико се за пренос са једног на други крај САД користе T3 линије[4] капацитет је 337500 бајтова а добијена вредност је већа од максимално могућег прозора примаоца5 (објављеног прозора). У одељку 6.3 описана је могућност за проширење величине објављеног прозора користећи опционо поље у заглављу TCP сегмента.

И опсег и кашњење могу да утичу на капацитет преносног пута[6] између пошиљаоца и примаоца. На слици 6.32 приказано је графички

---

[1] У овој анализи користимо вредност од 1,544 Mb/s. Брзина података је тачније 1,536 Mb/s пошто се први бит (од укупно 193 бита) користи да означи почетак рама.

[2] Среће се и под називом „производ опсега и кашњења" (литература [12]).

[3] Објашњено у поглављу 5.6.14, део „Величина прозора".

[4] У нашој анализи користимо вредност од 45 Mb/s тачније 44,736 Mb/s. Брзина података је 44 ,21 Mb/s. 65 535 бајтова

[5] Односно максималне величине сегмента који је објављен у тој вези.

[6] У литератури се пут између пошиљаоца и примаоца назива и цев (*pipe*). Аналогија је погодна за сликовито представљање „паковање" сегмената.

како се удвостручавањем времена RTT удвостручава капацитет.

На сличан начин на слици 6.33 приказано је како се двоструким повећањем пропусног опсега дуплира капацитет преносног пута (тзв. цеви).

У доњем делу слике 6.33 претпоставили смо да је брзина рачунарске мреже удвостручена чиме нам је омогућено да шаљемо четири сегмента за половину времена са горњег дела слике. Поново је капацитет преносног пута удвостручен. Претпоставили смо да сегменти у горњој половини слике 6.33 имају исти број битова као и сегменти у доњој половини слике.

6.13.5 Контрола загушености на TCP слоју

Уколико је оптерећење мреже веће него што она може да поднесе долази до загушености. Ни Интернет не представља изузетак. У овом одељку анализираћемо алгоритме који се користе за разрешавање

Слика 6.32 Удвостручавањем времена RTT удвостручава се капацитет преносног пута

овог проблема. Мрежни слој такође покушава да реши проблем загушености, али већину тежих задатака разрешава TCP слој.

Теоретски загушење се може избећи уколико се не шаљу (у рачунарску мрежу тј. Интернет) нови пакети док се одредишту већ претходно послати не испоруче. TCP покушава да реализује овај принцип мењањем величине отвора прозора.

Слика 6.33 Удвостручавањем опсега удвостручава се капацитет преносног пута

Први корак у решавању проблема загушења је његово проналажење. Кашњење може да буде последица загушења на линији или преоптерећености рутера. Одређивање разлике било би тешко.

Данас је губитак пакета услед грешака у преносу релативно редак пошто су на већини дужих путања постављени оптички каблови. Бежичне рачунарске мреже су посебна тема. Тако је највећи део кашњења на Интернету последица загушења. Сви TCP алгоритми који се користе на Интернету претпостављају да су кашњења изазвана загушењем.

На слици 6.34 приказана је хидраулична илустрација овог проблема[1]. Танка цев води до посуде (примаоца) малог капацитета (слика 6.34а). Док пошиљалац не шаље више воде него што посуда може да прими, вода се неће изливати. На слици 6.34б ограничавајући чинилац није капацитет посуде (примаоца) већ унутрашњи, преносни, капацитет мреже. Ако много воде дође пребрзо излиће се (у овом случају преплављујући левак).

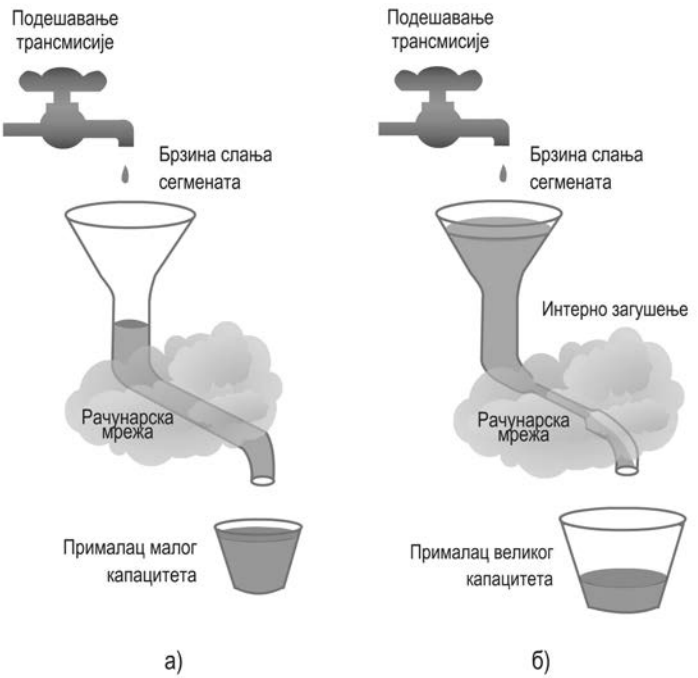
Задатак који Интернет треба да реши јесте како да превазиђе сваки од проблема: капацитет мреже и капацитет примаоца.

Брзина којом TCP целина шаље податке одређена је брзином пристизања *Ack* сегмената. На брзину испоруке података и *Ack* сегмената могу да утичу тачке загушења (уска грла). На слици[2] 6.35

[1] Пример преузет из литературе [13].

[2] Детаљније се може наћи у литератури [13].

претпоставља се да је уско грло негде на Интернету. Веза између предајника и пријемника представљена је као цев. Пречник цеви („дебљина“) пропорционална је брзини којом се преносе подаци кроз њу. Изворište и одредиште налазе се на мрежи великог капацитета и свако од њих може да ради са великим капацитетом. Централни део цеви мањег пречника означава или спору линију (линк) негде на Интернету или рутер који је уско грло. Сваки сегмент представљен је правоугаоником чија је површина пропорционална броју битова које садржи. Када сегменти пролазе кроз уску цев они (правоугаоници) се у времену „продужавају”[1]. Време P_b је минимално време (размак) између сегмената на најспоријој вези. Када сегменти стижу на одредиште размак између њих се задржава без обзира што се брзина повећала, пошто се време између пристизања сегмената није променило. По



Слика 6.34 Хидраулична илустрација
а) загушења на одредишту б) интерног загушења у мрежи

[1] Време потребно да се пошаље иста количина битова повећава се са смањењем брзине преноса.

томе размак између сегмената на пријему је P_r=P_b. Уколико одредиште шаље потврду како сегменти пристижу онда је размак *Ack* сегмената када напуштају пријемник $A_r = P_r$. На крају пошто је размак P_b довољно велики за сегменте података биће довољно велики и за *Ack* сегменте тако да размак остаје исти на најспоријој вези $A_b = A_r$ и на предајној страни $A_s = A_r$.

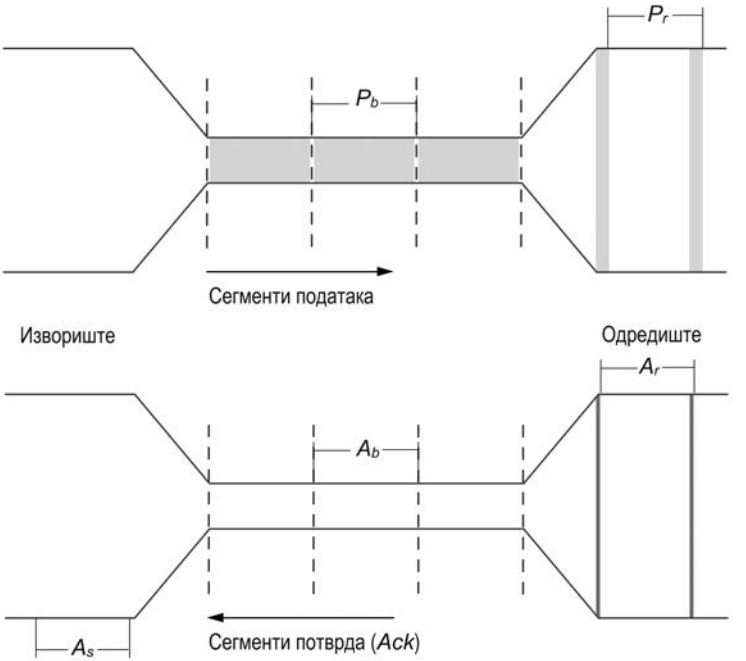
У успостављеном стабилном режиму брзина сегмената података које шаље предајник одговараће брзини пристизања *Ack* сегмената. То значи да је брзина слања сегмената једнака брзини најспорије везе. На овај начин TCP целина аутоматски испитује уско грло рачунарске мреже и регулише свој проток према томе. Овај процес назива се самотактујуће понашање TCP протокола.

Самотактујуће понашање TCP протокола ради једнако добро и ако је уско грло одредиште (пријемник). Претпоставимо да одредиште споро прихвата сегменте било због тога што је заиста споро било због тога што је оптерећено пристизањем сегмената са других веза. На слици 6.36 представљен је овај случај. Претпостављамо да је на најспоријој вези брзина око половине брзине извориште, али је цев на одредишту уска. У овом случају *Ack* сегменти шаљу се брзином којом одредиште прихвата (апсорбује) сегменте података.

Слике 6.35 и 6.36 указује на важан моменат: извориште нема начина да сазна да ли је брзина пристизања сегмената резултат стања на Интернету или резултат стања на одредишту. Прво се превазилази контролом загушења а друго контролом тока.

Уско грло дуж укупног пута између извориште и одредишта (пут сегмента података и пут *Ack* сегмента) јавља се на различитим местима. Дефинише се као логичко или физичко уско грло (слика 6.37). У примеру на слици 6.37 додељује се комплетан капацитет локалне мреже једној TCP вези која има потенцијални пропусни опсег од 100Mb/s до 10Gb/s. У овом случају веза 2Mb/s (E1) између рутера и Интернета постаје физичко уско грло. Када се успостави стабилно стање расположиви капацитет TCP веза може ефикасно да користи.

Много чешћа су логичка уска грла и настају као последица

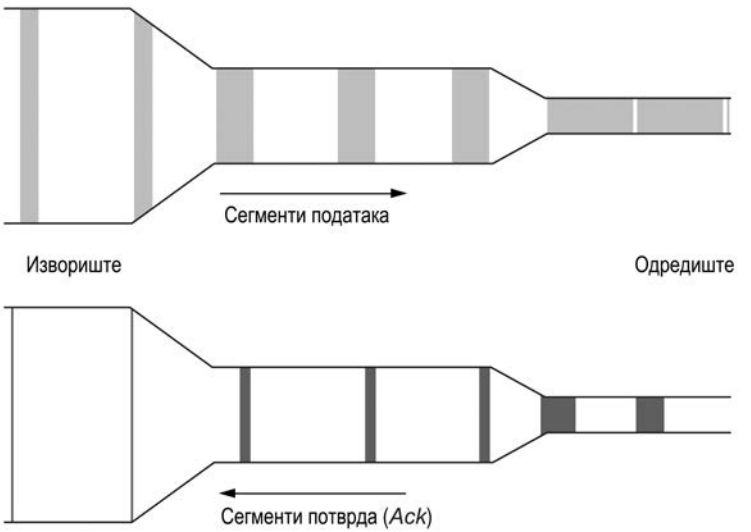


Слика 6.35 Размена TCP сегмената када је загушење у саобраћају последица загушења у рачунарској мрежи (негде на Интернету)

редова за чекање[1] у рутерима, рачунарској мрежи, комутаторима или у самом одредишту. Кашњења изазвана чекањем у редовима мењају се у зависности од саобраћаја, тј. оптерећења мреже, што доводи до потешкоћа у постизању стабилног стања.

Промене у величини кашњења незаобилазни су део Интернета са IP протоколом. Уколико је TCP проток података мали онда је капацитет Интернета неискоришћен. Уколико једно или више TCP извориште користи превелик капацитет остали TCP токови биће презагушени. Уколико већи број TCP извориште користи превелики капацитет сегменти ће у преносу бити изгубљени што доводи до ретрансмисија. Кашњење потврда због преоптерећења рачунарске мреже такође доводи до беспотребних ретрансмисија. Поновљено слање сегмената повећава саобраћај у мрежи, загушење расте, повећава се кашњење и повећава

[1] *Queueing delay*



Слика 6.36 Размена TCP сегмената када је загушење у саобраћају на одредишту

број сегмената који се одбацују. Резултат је још више ретрансмисија које стање у мрежи даље погоршавају.

Види се да без обзира што је TCP клизајући прозор пројектован за контролу тока података са једног на други крај рачунарске мреже мора се узети у обзир потреба за контролом (управљањем) загушења. Од када је објављен документ RFC793 бројне технике, које су имале за циљ да побољшају управљање загушењем, биле су имплементиране на TCP слоју. У табели 6.4 дате су најраспрострањеније технике.

Ниједна од техника из табеле 6.4 не нарушава оригинални TCP стандард. Може се чак констатовати да оне реализују концепте који

Слика 6.37 TCP проток сегмената и контрола загушења

Мерење	RFC 1122	TCP Tahoe	TCP Reno
Процена RTT варијансе[1]	√	√	√
Експоненцијално кашњење RTO[2]	√	√	√
Карнов алгоритам[3]	√	√	√
Успорени почетак	√	√	√
Динамичко подешавање прозора приликом загушења[4]	√	√	√
Брза ретрансмисија[5]			√
Брзи опоравак[6]			√

Табела 6.4 Примена TCP мерења загушења

су у оквиру спецификације за TCP слој. У табели је назначено које од техника су обавезне у RFC1122 а које су примењене у популарним беркли верзијама UNIX оперативног система.

Технике побројане у табели 6.4 спадају у две категорије: управљање часовницима за ретрансмисију[7] и управљање отвором прозора[8]

6.14 Управљање часовницима

TCP протокол обезбеђује поуздану услугу транспортног слоја. Један од начина којим се обезбеђује поузданост је коришћење потврде да су подаци стигли на одредиште. Може се десити да се сегменти података и потврда изгубе негде на Интернету. TCP протокол овај проблем разрешава употребом часовника[9]. Када се сегмент са подацима пошаље часовник се поставља на одређену вредност. Уколико потврда не стигне у том времену (време на које се поставља часовник истече) подаци се поново шаљу (а часовник се поново покреће). Ако је потврда за послати сегмент пристигла пре него што је часовник сигнализирао да је време

[1] *RTT variance estimation*
[2] *Exponential RTO backoff*
[3] *Karn's algorithm*
[4] D*inamic window sizing*
[5] *Fast retransmit*
[6] *Fast recovery*
[7] *Retransmit timer management*
[8] *Window management*
[9] *Timer* - тајмер, временска контрола

истекло, часовник се зауставља. Критична тачка примене овог механизма је метод за поновно слање (ретрансмисиона стратегија). TCP протокол може да управља са четири различита часовника по свакој вези:

- часовником за ретрансмисију - користи се када се очекује потврда од супротне стране;
- часовником истрајности[1] - пројектован је тако да спречи застој који може да наступи на следећи начин. Пријемник шаље потврду са прозором величине нула. Касније, пријемник ажурира прозор, али пакет са ажурирањем је изгубљен. Сада и пошиљалац и прималац чекају да један или други нешто учине. Када време на које је подешен часовник истрајности протекне, предајник шаље пакет који проверава шта је са пријемником. Одговор на пакет провере је величина прозора. Ако је још увек нула, часовник истрајности се поново поставља на одређену вредност и циклус се понавља. Ако је различит од нуле подаци могу сада бити послати;
- часовником који води рачуна да ли су стране (предајна или пријемна) у раду (часовник активности[2]) - када је веза неактивна дуже време (истиче време на које је постављен овај часовник) једна од страна везе може да провери да ли је друга страна везе још увек активна (у раду). Уколико не добије договор веза се прекида;
- часовником 2MS[3] који одређује време колико је веза у стању TIME-WAIT - користи се када се веза затвара да би се осигурало да сви пакети формирани у тој вези или стигну на одредиште или буду изгубљени.

Поставља се питање на коју вредност треба часовник поставити? Овај проблем много је сложенији код транспортног слоја на Интернету него код протокола слоја везе. Код слоја везе кашњење је предвидиво (мале су варијације) тако да време на које је подешен часовник може имати вредност која је мало већа од оне за коју се очекује потврда, као што је то приказано на слици 6.38а. Ако се потврда не појави у одређеном времену значи да су рам или потврда изгубљени. Потврде на слоју везе ретко касне.

[1] *Persistence timer*

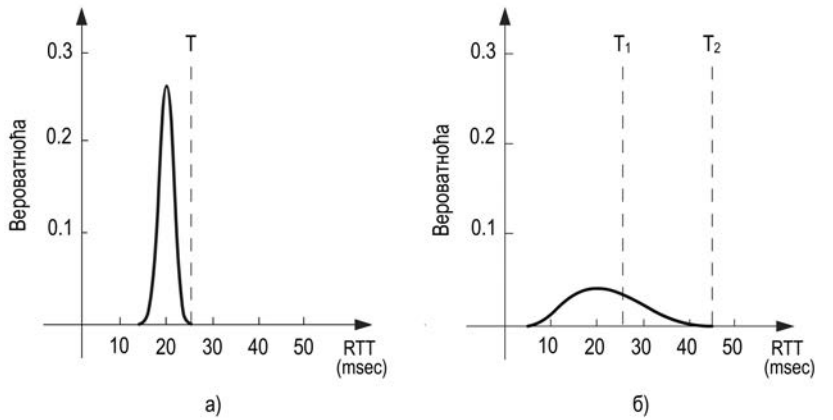
[2] *Keepalive timer*

[3] Двоструко време живота пакета

Расподела (густина) вероватноће[1] пристизања TCP потврда приказана је на слици 6.37б. Одређивање ове функције веома је тешко. Ако је време на које је подешен часовник превише кратко (T1) доћи ће до беспотребног поновљеног слања сегмената чиме се саобраћај на Интернету повећава а самим тим и загушења у саобраћају. Ако је време на које је подешен часовник велико (T2) онда, ако је сегмент изгубљен, протекне превише времена пре него што се сегмент поново пошаље. Како се израчунава време на које се часовник подешава и колико често[2] се сегменти поново шаљу?

Решење је примена динамичког алгоритма који стално подешава временски интервал на који се поставља часовник. Подешавање временског интервала засновано је на константним мерењима перформанси мреже. Алгоритам који се користи предложио је Јакобсон 1986. године.

У овом поглављу поћи ћемо од једноставног примера времена (RTO[3]) на које се поставља часовник за поновно слање сегмената



Слика 6.38 Расподела вероватноће пристизања TCP потврда
а) код слоја везе б) код TCP слоја

[1] *Probability density function*

[2] У примени „ICMP *port unreachable*" опције и TFTP клијенти који користе UDP протокол користи се једноставан метод поновног слања и подешавања часовника - претпоставља се да је време од 5s довољно.

[3] *Retransmission TimeOut value*

(ретрансмисиони часовник), а затим ћемо анализирати типичну примену времена потребног да се пошаље пакет и да стигне потврда за њега (RTT). Анализираћемо и како TCP целина користи измерене вредности да би проценила време на које треба поставити ретрансмисиони часовник, а затим и метод који омогућава да се загушење у мрежи избегне. Ове технике можемо систематизовати као:

- Мерење времена RTT;
- Карнов алгоритам;
- Експоненцијално RTO кашњење;
- Успорени почетак;
- Алгоритми за избегавање загушења;
- Брза ретрансмисија;
- Брзи опоравак.

6.14.1 Мерење времена RTT

Основ за постављање времена TCP часовника је мерење укупног времена RTT које протекне од тренутка започињања слања сегмента до пријема потврде за тај сегмент. Очекује се да се вредност RTT мења у времену у зависности од:

- путање коју сегмент пређе и
- промене интензитета саобраћаја у рачунарској мрежи.

TCP целина треба да прати ове промене и према томе мења вредност времена на које се одговарајући часовници постављају. Измерену вредност RTT означићемо са *M*, а њену процену у тренутку посматрања са *R*. Једноставна техника за предвиђање следеће вредности на основу претходних[1] може се представити једначином:

$$R \leftarrow \alpha * R + (1 - \alpha) * M \qquad (6.2)$$

где је α фактор за који је препоручена вредност од 0,8 до 0,9. Усредњена вредност RTT израчунава се код сваког новог мерења. Значајан део[2] сваке нове процене *R* потиче од претходне процене, а мањи део[3] од мерења *M* које је урађено у том тренутку (слика 6.39).

[1] Специфицирана у документу RFC793.

[2] Од 80% до 90% у зависности од тога колика вредност је узета за α.

[3] Од 20% до 10% у зависности од тога колика вредност је узета за α

Када се добије усредњена процена R, која се мења како се мења вредност RTT, препоручује се[1] да вредност на коју се поставља ретрансмисиони часовник (RTO[2]) буде израчуната на следећи начин:

$$RTO = \beta * R \tag{6.3}$$

Типична вредност за β=2. У стабилним условима са малом варијансом вредности RTT ова формула доводи до велике вредности RTO. У нестабилним окружењима вредност β=2 није добра заштита од беспотребних ретрансмисија. По Јакобсону беспотребне ретрансмисије су додатно повећање саобраћаја мреже која је већ преоптерећена.

Израчунавање вредности RTO на основу одступања (варијансе) и средње вредности обезбеђује много бољи одзив на велике промене времена RTT од израчунавања вредности RTO као средње вредности помножене неком константом.Један од начина је израчунавање стандардне девијације (подразумева израчунавање квадрата и квадратног корена). Једноставнија за израчунавање је средња девијација. Долазимо до следећих једначина које се примењују код сваког мерења времена RTT:

$$E_{rr} = M - A \tag{6.4}$$

$$A \leftarrow A + g * E_{rr} \tag{6.5}$$

$$D \leftarrow D + h * (|E_{rr}| - D) \tag{6.6}$$

$$RTO = A + 4D \tag{6.7}$$

где је *A* поравњана вредност RTT[3] (процењена или усредњена) а D поравњана вредност средње девијације. E_{rr} је разлика између измерене вредности управо добијене и текуће процењене вредност RTT. Вредности *A* и D користе се за израчунавање следећег RTO времена. Параметар *g* користи се за усредњавање и поставља се на вредност 0,125 (1/8). Параметар *h* за израчунавање девијације поставља се на вредност 0,25 (1/4). Што је параметар *h* већи то ће са порастом укупног времена RTT доћи до бржег пораста времена на које се подешава ретрансмисиони часовник[4] RTO.

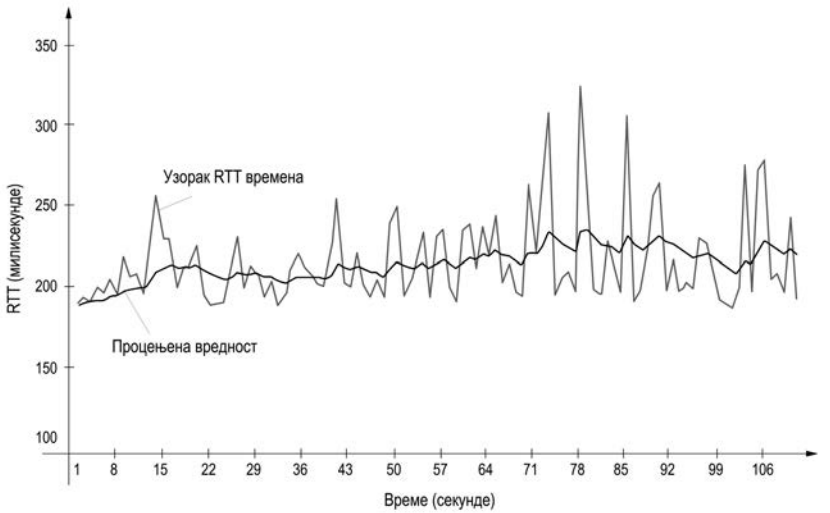
---

[1] Специфицирана у документу RFC795.

[2] *Retransmission TimeOut value.*

[3] *Smothed rount trip time estimate*

[4] Јакобсон је у својим првим истраживањима 1988. год. предложио за вредност 2 константе у једначини



Слика 6.39 Узорци RTT времена и процена вредности

Упоређујући оригинални метод са Јакобсоновим види се да је израчунавање поравњаних вредности слично[1] али је употребљена друга вредност (4 уместо 2). У Јакобсоновом израчунавању вредност RTO зависи од поравњаних вредности RTT и поравњаних вредности средње девијације, док оригинални метод користи умножак поравњаних вредности RTT.

Израчунавање се може обавити једноставним операцијама - сабирањем целих бројева, одузимањем и померањем. Константа 4 је произвољно изабрана и има две предности. Множење са 4 (код израчунавања вредности RTO) рачунар може обавити једноставном операцијом померања. Друго, избегава се сувишно понављање слања пакета пошто мање од 1% свих послатих пакета стиже у времену дужем од четвороструке стандардне девијације.

6.14.2 Карнов алгоритам

Проблем који настаје приликом динамичке процене RTT је шта урадити када сегмент након истицања времена на часовнику буде

(6.7) а касније (1990. год.) вредност 4. У већини примена користи се вредност 4.

[1] $\alpha = 1 - g$

поново послат. Када се добије потврда није јасно да ли се она односи на слање првог или поновљеног сегмента. Погрешан закључак могао би да озбиљно угрози процену вредности RTT.

Фил Карн, радио - аматер, ентузијаста, био је заинтересован за слање TCP/IP пакета радио - везом. Код преноса непоузданим медијумом, радио - везом, на одредиште у најбољем случају стигне 50% пакета. Карн је предложио решење: алгоритам који је по њему добио име и који се користи у већини TCP/IP реализација. Његов предлог био је једноставан:

- не ажурирати вредност RTT по поновљеном слању сегмента,
- време на које се поставља часовник за поновно слање (RTO) удвостручити при сваком неуспелом слању (као у једначини 6.3), све док се не добије потврда за сегменте који стигну на одредиште из првог покушаја.

6.14.3 Експоненцијално RTO кашњење

Када се сегмент пошаље а потврда за њега не стигне у времену RTO, на које је постављен часовник за ретрансмисију, TCP целина мора поново да пошаље сегмент. У документу RFC793 предложено је да се при поновном слању сегмента часовник постави на исто време RTO. Дугачко време у коме потврда није стигла последица је загушења у мрежи па се постављање часовника на исту вредност не препоручује.

Замислимо следећи сценарио: постоји велики број активних TCP веза из различитих извора и све заједно доприносе саобраћају на Интернету. Ефекат загушења има утицаја на сваку од ових веза. То значи да ће готово истовремено много сегмената бити поново послато у мрежу што ће продужити или чак погоршати загушење у саобраћају.

Метод који се предлаже (различит од метода у документу RFC793) је да се вредност RTO повећава при свакој ретрансмисији. На тај начин даје се времена да се загушење на Интернету регулише. Једноставан метод који је примењен је:

$$RTO \leftarrow k * RTO \tag{6.8}$$

Ова релација доводи до експоненцијалног раста времена RTO са сваком следећом ретрансмисијом. Уобичајена вредност за параметар

k је 2. Са овом вредношћу параметра *k* метод се назива бинарно експоненцијални. Исти метод је примењен и код CSMA/CD протокола у локалним рачунарским мрежама.

6.14.4 Алгоритам за избегавање загушења

Успорени почетак, који је описан у једном од претходних поглавља, начин је за иницирање протока података кроз TCP везу. У некој тачки доћи ће се до граничног капацитета неког од рутера и он ће једноставно одбацити сегмент. Избегавање загушења је начин који се примењује када дође до губитка сегмената. Описао га је Јакобсон.

Претпоставка Јакобсоновог алгоритма је да је губитак сегмената због грешке (оштећења) на њима веома мали (много мањи од 1%). Значи да губитак пакета указује на загушење негде у мрежи између изворишта и одредишта. Постоје два показатеља да је дошло до губитка сегмента: истицање часовника и пријем дупликата *Ack* сегмента.

Избегавање загушења и успорени почетак независни су алгоритми са различитим циљевима. Али када дође до загушења у саобраћају треба смањити брзину слања података а затим применити алгоритам успорени почетак. У пракси ова два алгоритма користе се заједно.

Избегавање загушења и успорени почетак захтевају да се прате две променљиве за сваку везу: прозор загушења (*cwnd*[1]) и преломна вредност успореног почетка (*sstresh*[2]). Комбинација ова два алгоритма ради на следећи начин:

- У току иницијализације везе поставити *cwnd* на величину једног сегмента а вредност *sstresh* на 65535 бајтова (62k);
- Предајна TCP целина никада не шаље више сегмената од минимума величине прозора загушења и величине објављеног прозора (тј. прозора пријемне TCP целине). Избегавањем загушења реализује се контрола тока на страни предајника а објављеним прозором реализује се контрола тока на страни пријемника. Прва је заснована на процени предајника о загушењу саобраћаја у мрежи; друга је везана за расположив меморијски простор на страни пријемника (погледати слику 6.5);

[1] *Congestion window*

[2] *Slow start treshold size*

- Када дође до загушења (на које указује истицање часовника или дупликат *Ack* сегмента) половина текуће величине прозора (минимум од *cwnd* и пријемниковог објављеног прозора, али најмање два сегмента) памти се као *sstresh*. Уколико на загушење указује истицање часовника *cwnd* се поставља на један сегмент (тј. успорени почетак);
- Када су нови подаци потврђени од друге стране увећава се *cwnd*, али начин на који се увећава зависи од тога да ли је примењен алгоритам успореног почетка или избегавање загушења.

Уколико је *cwnd* мањи или једнак као *sstresh* ради се са успореним почетком; у супротном ради се са избегавањем загушења. Успорени почетак примењује се све док се не стигне до тачке која је на половини оне вредности која је била када је дошло до загушења (записана је вредност половине отвора прозора која је довела до проблема у кораку 2). Даље се наставља са алгоритмом за избегавање загушења.

Код успореног почетка *cwnd* започиње са једним сегментом и увећава се за по један сегмент сваки пут када стигне *Ack* сегмент. На тај начин се отвор прозора увећава експоненцијално: шаље се један сегмент, па два, па четири итд.

Избегавање загушења диктира да се вредност *cwnd* увећава за 1 када стигне *Ack* сегмент али тек по истеку времена једнаког једном RTT. Овај начин представља адитивно увећавање *cwnd* док је код успореног почетка експоненцијално увећавање. Ми желимо да увећамо *cwnd* највише за један сегмент после укупног истека времена RTT независно од тога колико је потврда примљено. Избегавање загушења се наставља све док се не детектује загушење. Формула која се користи за израчунавање[1] прозора загушења може се представити на следећи начин:

$$cwnd \leftarrow \mathrm{MSS} * \mathrm{MSS} / cwnd \qquad (6.9)$$

Ово прилагођавање се извршава сваки пут када пристигне *Ack* сегмент (који није дупликат). Једначина обезбеђује прихватљиву апроксимацију принципа о увећавању прозора загушења (*cwnd*) за величину једног сегмента података по времену RTT. На пример уколико је MSS величине

[1] Детаљније се може наћи у документу RFC2581.

1460 бајтова и прозор загушења 14600 бајтова онда се 10 сегмената података може послати за време једног RTT. Сваки примљени *Ack* сегмент[1] повећава прозор загушења за 1/10 MSS. После 10 примљених сегмената вредност прозора загушења се повећава за 1 као што је и тражено.

Успорени почетак ће увећати *cwnd* за онолико колико је потврда (*Ack* сегмената) примљено у RTT времену. Слика 6.40 приказује коришћење успореног почетка и заобилажење загушења. Вредност *sstresh* је постављена на вредност 8. Док се ова граница не достигне TCP користи алгоритам успореног (експоненцијалног) почетка да би проширио прозор загушења. Када се достигне *sstresh* вредност *cwnd* се увећава линеарно.

Као илустрацију рада описаних алгоритама загушења погледајмо и слику 6.41. Максимална величина сегмента износи 1024 бајта. Иницијално се отвор прозора загушења поставља на 64k. У тренутку посматрања *sstresh* је постављен на 8k, отвор прозора загушења на 1k за слање редног броја 0. Затим величина прозора загушења расте експоненцијално до вредности *sstresh* 8k. Од те тачке наставља са линеаран раст.

Код 8. слања време на које је часовник постављен истиче. Вредност *sstresh* се поставља на половину тренутне величине прозора загушења (која је сада 12k, па је *sstresh* 6k). И поново се прелази на успорени почетак. Када стигне потврда за 12. слање (нови *sstresh*), прва четири слања су већ удвостручила прозор, после чега његов раст постаје опет линеаран. Овај метод се среће у литератури под називом TCP *Taho*.

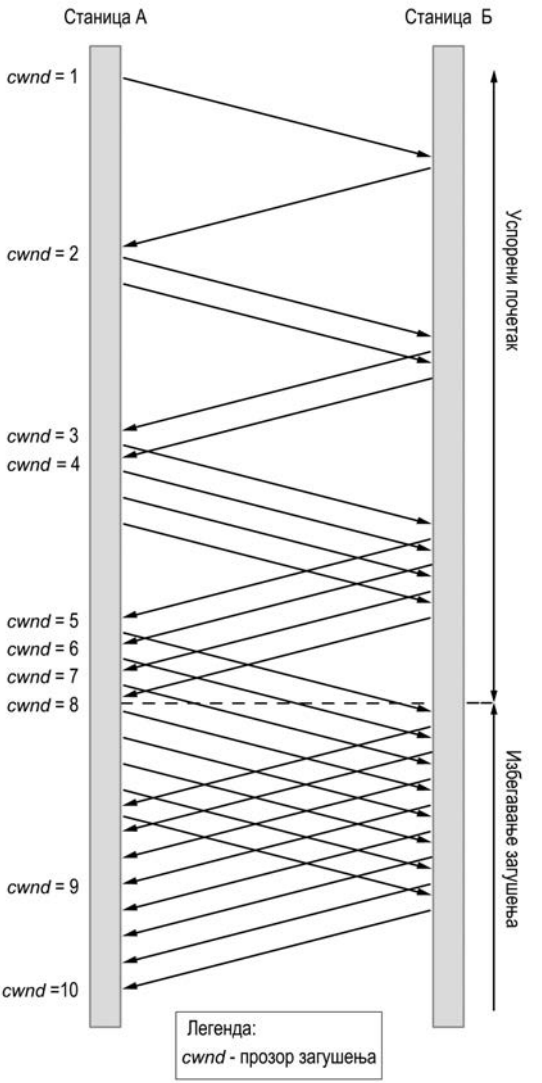
Величина прозора загушења ће расти до величине прозора примаоца уколико у међувремену не истекне време на које је часовник подешен. Када достигне ту величину престаје са растом и остаје константна све док се не промени величина прозора примаоца[2] или часовник не сигнализира истицање времена на које је подешен.

6.14.5 Брзо поновно слање

Са брзим поновним слањем сегмената са подацима избегава се да TCP целина чека да истекне време на које је подешен часовник за

[1] Претпостављајући да за сваки примљени сегмент података пријемник шаље један *Ack* сегмент.

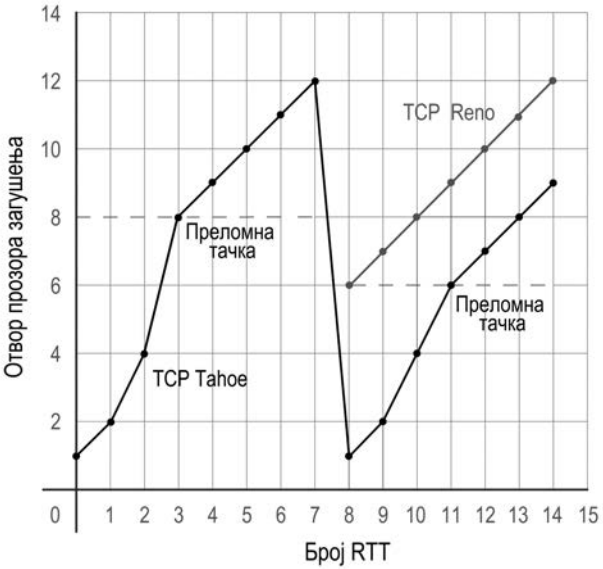
[2] Ако TCP целина добије ICMP SOURCE QUENCH пакет третира се као сигнализирање часовника да је време истекло.



Слика 6.40 Примена алгоритама успореног почетка и избегавање загушења

ретрансмисију па да понови слање сегмента. Измену алгоритма за избегавање загушења предложио је Јакобсон 1990. год.

Пре него што анализирамо у чему се састоје измене подсетимо



Слика 6.41 Илустрација рада алгоритма загушења за избегавање загушења

се да TCP целина може да пошаље потврде (дупликат *Ack* сегмента) када добије сегмент са подацима са редним бројем који не очекује (ван редоследа). Пријемна TCP целина шаље *Ack* дупликат чим добије први сегмент података који је ван редоследа. Намена *Ack* дупликата је да укаже пошиљаоцу да је примио сегмент ван редоследа и коки редни број (AN) податка пријемник као следећи очекује.

Предајна TCP целина не зна да ли је дупликат *Ack* сегмента последица изгубљеног сегмента података или последица само пријема сегмената података у неправилном редоследу. Ако су сегменти података примљени у неправилном редоследу пријемник шаље дупликате *Ack* сегмента све док не обради примљене сегменте података. Тада шаље *Ack* сегмент којим потврђује све примљене сегменте података.

Уколико предајна TCP целина добије три дупликата *Ack* сегмента један за дугим онда она претпоставља да је сегмент податак са редним бројем као у *Ack* сегменту изгубљен. Поново шаље тај сегмент и не чека да часовник за ретрансмисију сигнализира да је време истекло. Овај метод се назива брзи алгоритам за поновно слање (ретрансмисију).

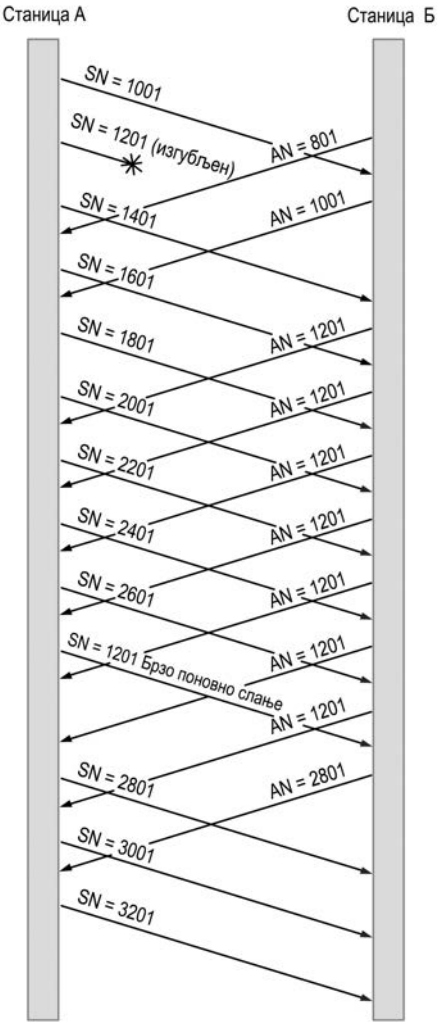
Слика 6.42 приказује процес брзе ретрансмисије. Станица А шаље сегменте са подацима од по 200 бајтова. Сегмент података са редним бројем 1201 је изгубљен, али станица А неће да реагује док време на које је подешен часовник RTO не истекне, или док станица Б не затвори свој прозор. Станица Б прима сегменте 1001 (редни бројеви од SN=1001 до SN=1200) и потврђује их са *Ack* сегментом (редни број AN=1201). Затим прима сегмент 1401 (редни бројеви од SN=1401 до SN=1600). Пошто је овај сегмент изван редоследа станица Б понавља *Ack* сегмент (AN=1201) и понављаће га све док се не појави сегмент са редним бројем 1201. За то време станица А је примила четири дупликата *Ack* сегмента (AN=1201) и послала седам сегмената података са редним бројевима иза сегмента 1201. Станица А поново шаље сегмент података 1201 (редни бројеви од SN=1201 до SN=1400) и наставља са слањем.

6.14.6 Брзи опоравак

Када TCP целина поново пошаље сегмент података користећи брзу ретрансмисију, она зна (тј. претпоставља) да је сегмент изгубљен без обзира што није протекло време на које је подешен часовник за ретрансмисију за тај сегмент. Један од начина би био да се од тог тренутка надаље примене алгоритми успореног почетка и заобилажења загушења (слике 6.40 и 6.41). Овај приступ превише је рестриктиван пошто *Ack* дупликати указују TCP целини да постоји проток података између два краја везе. Пријемник генерише *Ack* дупликат када сегмент (по редоследу следећи) стигне и буде смештен у меморијски простор пријемника. TCP целина не жели да употребом успореног почетка беспотребно нагло смањи проток. Због тога је предложена модификација[1]: поново послати изгубљени сегмент, прозор загушења (*cwnd*) поставити на половину текуће вредности, и наставити са линеарним увећањем прозора загушења. На тај начин се алгоритми брзе ретрансмисије и брзог опоравка примењују заједно (слика 6.42).

Највећи број оперативних система данас користи овај принцип (или неку од модификација RFC3782, RFC2018) и среће се под називом *Reno*. На слици 6.43 представљени су могући начини ограничавања прозора загушења. Треба обратити пажњу на то да је и величина објављеног

[1] Јакобсон, описана у RFC2581



Слика 6.42 Брзо поновно слање

прозора пријемника (на дијаграму представљен испрекиданом линијом) један од ограничавајућих фактора.

Резултати неких истраживања указују да је 54% ретрансмисија последица истицања времена на које је подешен ретрансмисиони часовник, а 46% последица брзе поновне ретрансмисије.

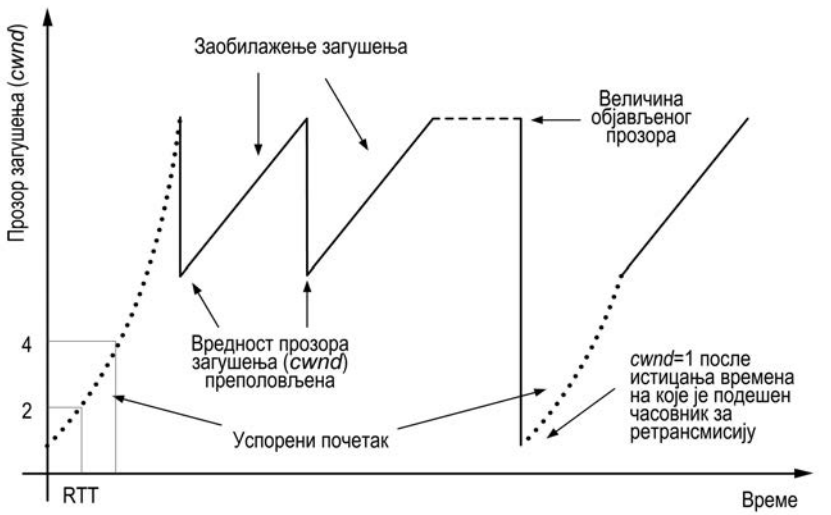
6.14.7 Правци даљег развоја алгоритама за избегавање загушења

Важно је нагласити да постоји потреба да се алгоритми за избегавање (заобилажење) загушења мењају, усавршавају и прилагоде TCP везама великих брзина. Алгоритми који су погодни за пренос велике количине података код апликација SMTP, FTP и *Telnet* не одговарају HTTP и мултимедијалним апликацијама на Интернету.

Потреба за развојем TCP протокола може се илустровати анализом веза великих брзина које су потребне на пример за TCP везе код грид мрежа[1]. Пратећи анализу описану у документу RFC3649 видимо да се веза између средње вредности величине прозора загушења код TCP веза у стационарном стању и вероватноће изгубљених пакета p може изразити релацијом:

$$cwnd \approx 1{,}2 / p^{1/2} \qquad (6.10)$$

Постоје озбиљна ограничења у величини прозора загушења (*cwnd*) која се могу достићи у реалним условима. На пример посматрајмо TCP везу:



Слика 6.43 Различити начини за промену прозора загушења

[1] *Grid* computing - грид израчунавање. Апликација користи више рачунара за истовремено решавање једног проблема. Уобичајено је решавање научних или техничких проблема који захтевају велику обраду или приступ великој количини података.

- у којој се размењују сегменти величине 1500 бајтова,
- у којој је кашњење RTT 100ms, и
- проток од 10Gb/s.

Да би се достигло стационарно стање у оваквој TCP вези потребно је да је средња вредност прозора загушења 83333 сегмента. Такође се захтева и да не дође до одбацивања више од једног пакета на сваких $5*10^9$ пакета што је одговара вероватноћи губитка пакета од $2*10^{-10}$ или вероватноћи битске грешке (BER[1]) од $2*10^{-14}$.

Ово су нереални захтеви за рачунарске мреже које су данас у употреби. Због тога је покретнут већи број истраживања која су довела до нових верзија TCP протокола које су пројектована за рачунарске мреже великих брзина[2].

6.15 Коректност протокола транспортног слоја

Посматрајмо *k* TCP веза таквих да свака од њих има различито кашњење с краја на крај и да свака од њих пролази кроз исти линк – максимално искоришћеног капацитета[3] који је брзине R[b/s]. Претпоставимо да се сваком од веза преносе велике датотеке и да се истим линком преносе подаци који користе UDP протокол. Механизам за контролу загушења сматра се коректним[4] уколико је средња брзина преноса сваке од веза *R/k*. То значи свака од веза добија исти део пропусног опсега линка.

Поставља се питање да ли су TCP алгоритми са заобилажењем загушења[5] коректни? Посебно ако се има на уму да различите TCP везе могу да започну у различитим тренуцима и могу да у датом тренутку имају различите величине отвора прозора. Посматрајмо једноставан случај када две TCP везе деле један линк брзине R (слика 6.44). Претпоставимо (једноставан случај) да две TCP везе[6]:

---

[1] *Bit Error Rate*

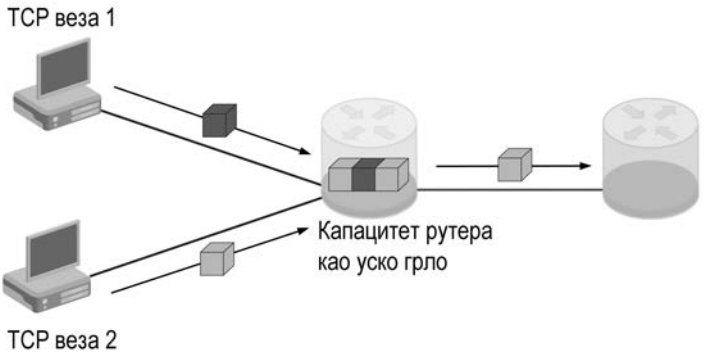
[2] Детаљан опис се може наћи у RFC3649.

[3] *Bottleneck link*

[4] *Fairness*

[5] Означавају се са AIMD (*Additive Increase/Multiplicative* D*ecrease*) - адитивно увећавање/вишеструко умањење.

[6] Детаљније се може наћи у литератури [16].



Слика 6.44 Две TCP везе деле један линк брзине R [*b/s*]

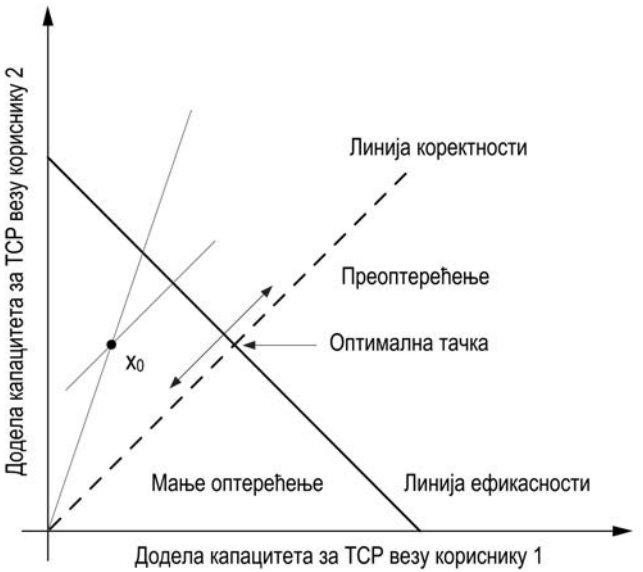
- имају исту максималну величину сегмента (MSS) и време RTT. То значи да имају исту величину прозора загушења и проток[1];
- имају велику количину података за слање;
- не користе ниједну другу TCP или UDP везу и
- раде у режиму заобилажења загушења.

Да бисмо анализирали овај систем са два корисника посматраћемо га као трајекторије у дводимензионалном простору[2]. Као што се на слици 6.45 види додела ресурса (капацитета линка) може се представити тачком (x_1, x_2) у дводимензионалном простору. На сликама 6.45 и 6.46 додела ресурса за TCP везу 2 представљена је хоризонталном осом а додела ресурса за корисника за TCP везу 1 верикалном осом. Свака додела за коју је $x_1+x_2= R$ је ефикасна тј. означава потпуно искоришћење капацитета линка и на слици 6.45 означена је као линија ефикасности. Додела капацитета за коју је $x_1=x_2$ се означава као линија коректности и одговара јој испрекидана линија на слици 6.45. Две линије се пресецају у тачки са координатама (R/2, R/2) и то представља оптималну тачку. Циљ оптималног начина управљања био би да се систем доведе у оптималну тачку независно од почетне позиције. Све тачке испод линије ефикасности (искоришћеност пуног капацитета) мањег су оптерећења[3] и у идеалном случају ће систем захтевати од корисника да повећа

[1] *Throughput*, пропусна моћ

[2] Када би требало спровести анализу за *n* корисника био би потребан *n*-димензионални простор.

[3] *Underloaded*



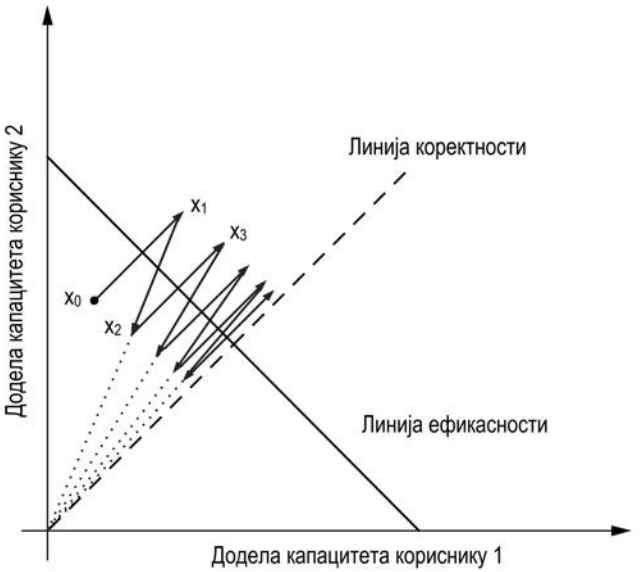
Слика 6.45 Векторско представљање случаја две TCP везе

своје оптерећење. Уочимо тачку $x_0 = (x_{10}, x_{20})$. Адитивно повећање[1] оба корисника за исти капацитет одговара померању у правцу линије нагиба 45º. Вишеструко померање (множење додељених ресурса корисницима за фактор *b*) одговара померању по линији која повезује ту тачку и координатни почетак. Све тачке изнад линије ефикасности указују на прекорачење капацитета[2].

Претпоставимо да је величина TCP прозора таква да корисници 1 и 2 имају на располагању капацитете означене тачком x_0. Пошто је искоришћење линка које заједно користе корисници 1 и 2 мање од R, неће бити губитка података и оба корисника ће увећати свој прозор загушења за један MSS по истеку времена RTT, као резултат алгоритма за заобилажење загушења. Здружени проток обе TCP везе мераће се дуж линије нагиба 45º (једнако увећање за обе везе). Резултујући капацитет је означен тачком x_1. Пошто је проток већи од максималног капацитета линка R доћи ће до губитка пакета. Претпоставићемо да су корисници 1 и 2 то уочили и

[1] *Aditive increase* (AIMD - *Aditive Increase Multiplicative* Decreas) – део механизма за заобилажење загушења у коме се линеарно увећава прозор загушења.

[2] *Overloaded*



Слика 6.46 Адитивно повећање и мултипликативно смањење воде до оптималне тачке

смањују свој отвор прозора два пута. Резултујући проток је означен тачком x_2 и налази се на половини растојања тачке x_1 од координатног почетка. Пошто је здружени проток сада мањи од капацитета R везе корисници поново увећавају свој проток линијом која је под нагибом почевши од тачке x_2. Пошто је здружени проток у тачки x_2 сада мањи од капацитета R везе корисници поново увећавају свој проток дуж линије која је под нагибом од 45° почевши од тачке x_2. До губитка ће поново доћи у тачки x_3 и две везе поново смањују свој прозор за два итд. Види се да искоришћење капацитета које су остварили ове две везе осцилује око линије која дели капацитет линка на једнаке делове.

У овом идеализованом примеру претпоставили смо да TCP везе користе линк пуног капацитета, да корисници TCP веза имају исто време RTT, и да је само једна веза додељена пару извориште-одредиште. У пракси ови услови обично нису испуњени, и клијент/сервер апликације не могу да обезбеде једнако искоришћење капацитета. Показало се да када више веза користи исти линк оне везе са мањим временом RTT су у стању да заузму већи део капацитета од веза са већим временом RTT.

6.15.1 Коректност UDP протокола

У претходном тексту анализирано је како управљање загушењем TCP везе регулише брзину преноса користећи механизам прозора загушења. Многе мултимедијалне апликације не користе TCP већ UDP протокол који не користи механизме за управљање загушењем. Када апликација користи UDP протокол она може да шаље у мрежу аудио и видео податке константном брзином. Повремени губитак података неће, у тренутцима загушења, довести до смањења бризине слања на коректан ниво. Са тачке TCP протокола UDP протокол није коректан: не сарађује са другим везама и не прилагођава брзину слања условима у мрежи. Пошто ће механизам за управљање загушењем умањити проток TCP везе са циљем да се смањи губитак података, а везе са UDP протоколом неће смањити свој проток може доћи до тога да извориште са UDP протоколом заузму много више капацитета линкова од оних са TCP протоколом. Већи број актуелних пројеката истражује овај проблем.

6.15.2 Коректност и паралелне TCP везе

Чак и да се успе у томе да везе са UDP протоколом постану коректне, проблем и даље није комплетно решен. Разлог је тај што не постоји начин да се спрече апликације које користе TCP протокол да користе више паралелних веза. На пример Web претраживачи често користе више паралених TCP веза за пренос више објеката у оквиру Web странице. Када апликација користи више паралелних веза она добија већи део капацитета линка. На пример: линк капацитета R који подржава девет истовремених клијент/сервер апликација. Свака од тих апликација користи TCP протокол. Уколико се појави нова апликација која такође користи TCP протокол онда свака апликација добија капацитет приближно R/10. Међутим уколико нова апликација користи 11 паралелних TCP веза онда нова апликација заузима некоректан капацитет од R/2. Више паралелних веза на Интернету нису реткост.

6.16 Анализа рада часовника

У претходним одељцима видели смо како се користи ретрансмисиони часовник а у овом одељку анализираћемо остале часовнике који су од значаја за рад TCP протокола.

6.16.1 Часовник истрајности

Видели смо да предајна TCP целина шаље податке у зависности од количине коју је пријемна целина у стању да прими, односно од величине прозора. Шта се дешава када се пријемник огласи да не може да прима податке, тј. огласи прозор величине 0? Заустављa се слање предајника док пријемник не огласи прозор различит од 0. Овакав сценарио смо видели на слици 6.5. После пријема сегмента са отвором прозора различитим од нуле (W≠0) наставља се са слањем података. TCP целина мора да предвиди активности за случај да се може десити да се сегмент W≠0 изгуби. Шта се може десити:

- да је предајник заустављен и да чека сегмент са отвором прозора W≠0;
- да је пријемник послао сегмент са отвором прозора W≠0 и да чека да пристигну нови подаци од предајника.

Да би се онемогућило да дође до овакве затворене петље користи се часовник истрајности[1]. Предајна TCP целина периодично шаље упит пријемнику да би сазнала да ли је пријемник повећао свој отвор прозора (W≠0). Сегменти који се користе као упит називају се „испитивање прозора[2]". Карактеристика часовника истрајности је да он никада не одустаје од слања упита. Он наставља са слањем упита сваких 60s све док предајник не отвори прозор или апликација која ту везу користи не затражи затварање везе.

6.16.2 Часовник активности

Може да се деси да ниједан од процеса на крајевима TCP везе не шаље податке и две TCP целине не размењују ништа. То значи да може да се деси да неко покрене процес на страни клијента који успоставља TCP везу са сервером и не настави са радом сатима, данима и месецима а TCP веза ће остати успостављена. Рутери и линкови (нпр. телефонске везе) на путу између два краја везе могу у међувремену да прекину и наставе са радом али ова веза се сматра успостављеном све док је станице на крајевима везе не прекину. Значи, претпоставка је да ниједна од апликација (ни клијент ни сервер) нема часовник на

[1] *Persist timer*
[2] *Window probes*

апликационом слоју који може да уочи да активност на вези не постоји и да везу треба прекинути. Спољашњи протоколи за рутирање (BGP[1]) апликацији шаљу тест сваких 30s. Часовник који иницира слање овог теста везан је само за BGP протокол и независан је од рада часовника активности TCP слоја.

Постоје ситуације када сервер жели да зна да ли је станица - клијент у квару и завршила са радом или је у квару али ће наставити са радом. Часовник активности, који се користи у многим применама, то омогућава али он није део TCP спецификације. Разлог је што се сматра да:

- могу да изазову прекид везе која је неактивна због тренутних али решивих проблема међувеза,
- користе без потребе капацитет везе и додатна су инвестиција.

Постоје разлози за и против коришћења часовника активности. Пример који указује да је потребан часовник активности се када је клијент повезан са сервером *Telnet* протоколом. Ако на крају дана клијент (корисник) искључи рачунар а да се претходно не одјави (откуца *Telnet* команду qui*t*), он оставља полуотворену везу са сервером. Сервер може у недоглед да чека податке од клијента. Употреба часовника активности на страни сервера омогућиће откривање овакве полуотворене везе.

6.17 Ефекат малог прозора

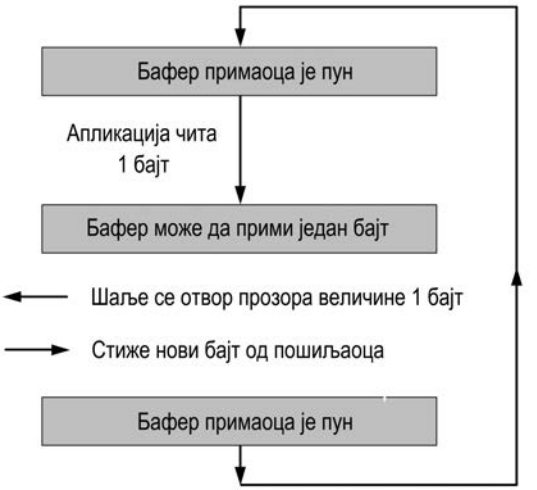
Још један проблем који може настати на TCP слоју као последица употребе отвора прозора за контролу тока је ефекат малог прозора[2] (SWS[3]). Овај проблем настаје када подаци стижу до TCP целине пошиљаоца у великим блоковима, али интерактивна апликација на пријемној страни чита податке бајт по бајт.

Да би анализирали проблем погледајмо слику 6.47. У почетку, меморијски простор TCP целине на страни примаоца је пун и пошиљалац је упознат са тим (нпр. величина отвора прозора му је једнака нули). У том случају, интерактивна апликација чита један карактер из TCP низа. Ова акција проузрокује да TCP примаоца мења отвор прозора

[1] *Border Gateway Protocol*

[2] *Silly window syndrome*

[3] Описао је Кларк (*Clark*) 1982.



Слика 6.47 Ефекат малог прозора

(један бајт) и обавештава пошиљаоца да може да пошаље један бајт. Пошиљалац прима поруку и шаље један бајт. Меморијски простор је сада поново пун, пријемна страна шаље потврду, али поставља величину отвора прозора на нулу. Ово може да траје у недоглед.

Кларк је је предложио решење које се састоји у томе да се спречи пошиљалац да шаље промену отвора прозора за само један бајт. Уместо тога он чека да се значајни меморијски простор ослободи и тек тада обавештава пошиљаоца.

Алгоритам Нигла и решење Кларка су комплементарни. Нигл је покушао да реши проблем који је изазвала апликација пошиљаоца када шаље TCP целини бајт по бајт. Кларк је покушао да реши проблем апликације примаоца која од TCP целине очитава податке бајт по бајт. Оба решења могу се користити истовремено. Суштина је да пошиљалац не треба да шаље мале сегменте а прималац не треба да преузима мале сегменте.

6.18 Карактеристике и правци развоја TCP протокола

TCP протокол се користио у везама малих брзина нпр. 1200b/s на комутираним телефонским линијама. Данас се користи у мрежама већих

брзина: локалним рачунарским мрежама (1Gb/s), мрежама које користе T3, E3 и 10Gb/s везе па се морају узети у обзир и одређена ограничења. У овом одељку анализираћемо неке од предложених измена у TCP протоколу које омогућавају максималну пропусну моћ у мрежама великих брзина.

6.18.1 Откривање максималне јединице за пренос

Када су две станице у истој мрежи дефинисана је максимална јединица података која се може том мрежом преносити (MTU[1]). Када станице комуницирају преко различитих мрежа сваки линк може да има различиту максимална јединицу преноса. Важна величина је максимална јединица података која се може пренети целокупном путањом између те две станице (не максимална јединица преноса у мрежи у којој се станице налазе). Ова величина назива се максимална јединица преноса путање[2] и није константна. Зависи од путање којом се пренос података одвија. Рутирање не мора да буде симетрично па се максимална јединица пута може разликовати у различитим правцима исте везе.

У документу RFC1191 је дефинисан механизам за откривање путање[3] јединице као начин за одређивање јединице MTU у било ком тренутку.

Откривање MTU пута на TCP слоју ради на следећи начин: када се веза успостави TCP целина користи MTU вредност излазног интерфејса или MSS који је други крај објавио као почетну величину сегмента. Одређивање MTU пута не дозвољава TCP целини да прекорачи вредност коју је други крај објавио. Уколико други крај није специфицирао MSS користи се подразумевана вредност 536.

Једном када је величина сегмента одабрана сви IP пакети (датаграми) који се шаљу том TCP везом имају постављен DF бит на вредност 1. Уколико неки од рутера на путу мора да дели IP пакета који има DF=1 он га одбацује и шаље ICMP поруку[4] изворишту „не могу да делим пакет".

---

[1] *Maximum Transfer Unit* - максимална јединица преноса

[2] *Path* MTU

[3] *Path* MTU *discovery mechanism*

[4] Детаљније о ICMP протоколу може се наћи у одељку.

Пошто се путање (руте) могу динамички мењати, када протекне одређено време може се покушати са повећањем величине MTU (само до вредности MTU коју је одредиште објавило). У документу RFC1191 препоручено је да овај временски интервал буде 10 минута.

6.18.2 Дугачке везе великог капацитета

Капацитет везе (линка) назива се „производ опсега и кашњења" и дефинише се као:

$$\text{Капацитетбит} = \text{пропусни опсег}_{b/s} * \text{RTT}_{sec}$$

Користи се и термин „величина цеви" између две крајње тачке. За различите мреже у табели 6.5. приказан је производ пропусног опсега и кашњења у бајтовима пошто је то начин како се уобичајено мери величина меморијског простора (бафера) и величина отвора прозора на сваком од крајева.

Мреже са великим производом пропусног опсега и кашњења називају се мреже велике ширине (LFN[1]). TCP веза која користи мреже велике ширине назива се велика широка цев[2]. Ако погледамо слике 6.24 и 6.25 цев се може проширити у хоризонталном правцу (веће RTT) или се може проширити у вертикалном правцу (шири пропусни опсег) или у оба правца. Бројни проблеми се јављају код дугачких широких цеви. Поменућемо неке од њих:

Величина TCP отвора прозора одређена је 16-битним пољем у TCP заглављу што значи да је максимална величина TCP отвора прозора 216 = 65535 бајтова. Као што можемо видети из последње колоне табеле 6.5 постојеће мреже већ захтевају већи прозор да би обезбедиле максималну пропусну моћ[3].

Губици пакета у мрежама велике ширине могу значајно да умање пропусну моћ мреже. Уколико је само један сегмент изгубљен алгоритми „брза ретрансмисија" и „брзи опоравак", који су описани у претходном поглављу, потребни су да би се у вези (цеви) задржао проток пакета. Чак и са овим алгоритмима губитак више од једног пакета у оквиру

[1] *Long Fat Networks* - велике дебеле мреже а користи се и енг. термин *elefan(t)s*.

[2] *Long fat pipe* - велика дебела цев

[3] Видели смо да је у пољу опције заглавља TCP сегмента предвиђена могућност за повећање максималне величине прозора

Мрежа	Опсег [b/s]	RTT [ms]	Производ опсега и кашњења [бајт]
Локална рачунарска мрежа IEEE802.3	10000000	3	3750
T1 телефонске линије, међуконтиненталне	1544000	60	11580
T1 телефонске линије, сателитске	1544000	500	96500
T3 телефонске линије, међуконтиненталне	45000000	60	337500
Гигабит, међуконтинентален	1 00000 0000		

Табела 6.5 Различите мреже и одговарајући производ опсега и кашњења

једног прозора обично доводи до заустављања протока у вези (цеви). Селективна потврда (SACK) предложена је у документима RFC1072 да би разрешила проблем губитка више пакета у оквиру једног прозора. Ова опција је изостављена у RFC1323 због техничких проблема.

Видели смо да многе примене TCP протокола мере једно RTT време по прозору а не за сваки сегмент у оквиру прозора. За мреже велике ширине потребна су чешћа мерења RTT времена.

Као што је већ објашњено TCP целина означава сваки бајт података са 32 битним редним бројем. Шта спречава сегменте који су у кашњењу, и чија је веза окончана, да се појаве у новоуспостављеној вези између две станице и два порта као из претходно окончане везе? Прво подсетимо се да TTL поље у заглављу IP пакета поставља горњу границу времена његовог постојања у мрежи - 255 прелазака преко рутера или 255s . У одељку 6.6.1 дефинисали смо максимално време трајања сегмента (MSL) као параметар који спречава да до одбацивања IP пакета дође.

У мрежама велике ширине јавља се и проблем са редним бројевима TCP сегмената. Пошто је простор редних бројева коначан исти редни број поново се користи пошто се пошаље 4294967296 бајтова. Шта ће се десити ако сегмент једне TCP везе који садржи бајт са редним бројем N има велико кашњење и стигне на одредиште у току трајање те везе?

До проблема може да дође само уколико се редни број N поново користи (нови круг) пре истека периода од MLS. У Етернет локалним рачунарским мрежама потребно је скоро 60 минута да би се послало толико података да не може да дође до поновног коришћења истих редних бројева. Време које протекне до поновног коришћења истих редних бројева се смањује како се повећава пропусни опсег преносних система.

На пример за T3 преносне системе, брзине 45Mb/s до новог круга коришћења редних бројева долази после 12 минута, а код локалних рачунарских мрежа брзине 1Gb/s после 34 секунде. Проблем код оваквих мрежа није производ пропусног опсега и кашњења већ сам пропусни опсег. Један од начина да се заштити од овог проблема јесу алгоритми који користе опције са временским ознакама сегмената.

6.19 Експлицитно обавештење о загушењу

Један од новијих механизама којим се управља загушењем у мрежи је механизам који се означава као експлицитно обавештење о загушењу (ECN[1]) описан у документу RFC3168.

TCP алгоритми за управљање и заобилажење загушења (описани у претходним поглављима) полазе од претпоставке да је мрежа „црна кутија“. Стање загушености мреже одређу крајње тачке које постепено повећавају оптерећење мреже (увећавајући отвор прозора) док не дође до загушења саобраћаја у мрежи и до губитака пакета. Тада се предузимају кораци за смањење оптерећења у мрежи. Овај приступ означава се као имплицитно обавештавање о загушењу[2].

Овакав механизам одговара апликацијама које нису осетљиве на губитак пакета или на њихово кашњење. Алгоритми који управљају загушењем имају механизме као што су брза ретрансмисија и брзи опоравак који умањује утицај губитка пакета посматрано из угла пропусности система. Међутим ови мехнизми нису од помоћи за апликације које су осетљиве на кашњења или на губитак једног или више пакета. Интерактивне апликације као што су *Telnet*, веб претраживачи

[1] *Explicit Congestion Notification*
[2] *Implicit Congestion Notification*

и пренос аудио и видео података могу да буду осетљиве на губитак пакета нарочито када се користи непоуздан UDP протокол, али и на кашњења до којих долази при поновном слању када се користи TCP протокол. Пошто TCP одређује одговарајући прозор загушења постепено увећавајући величину прозора док не дође до одбацивања пакета, то доводи до повећања редова чекања пакета у критичним[1] рутерима. Када редови чекања премаше одређену критичну границу[2]примењује се метод за одбацивање пакета. Један од њих је и метод којим се последњи пристигли пакети одбацују[3]. Може да се деси да су одбачени пакети из тока података који је осетљив на кашњења.

Механизми са активним надзором редова чекања (AQM[4]) у рутерима откривају загушење пре него што дође до критичне величине редова чекања пакета и обавештавају крајње станице о томе. Тако да активни надзор редова чекања може да смањи беспотребна кашњења за све пакете који се налазе у тим редовима. Предност AQM механизма је описана у документу RFC2309. Важно је нагласити да применом механизма активног надзора редова чекања транспортни протокол (тј. TCP протокол) који користи механизме за управљање загушењем не мора да се ослања на прекорачење меморијског простора рутера[5] као једини показатељ загушења.

Активни надзор редова чекања (AQM) може да користи неколико начина којима крајњим станицама (чворовима) указује на појаву загушења у саобраћају. Један од њих (поред одбацивања пакета) је и коришћење ознаке у заглављу пакета која указује да долази до загушења (CE[6]).

Метод благовременог откривање загушења (RED[7]) је један од AQM механизама који се препоручује за откривање почетног загушења. Користи се на Интернету а описан је у документу RFC2309. Уобичајено је да AQM механизам одбацује пакете када просечна величина реда

[1] *Bottleneck router*
[2] *Queue overflow*
[3] *Tail-drop*
[4] *Active Queue Management*
[5] *Buffer overflow*
[6] *Congestion Experienced*
[7] *Random Early* Detection

чекања пакета премаши граничну вредност[1], пре него што заиста дође до прекорачења величине реда за чекање. Тако да, пошто AQM механизам може да одбаци пакете и пре стварног прекорачења величине реда за чекање то значи да AQM мехнизам није увек покренут недостатком меморијског простора за смештај пакета.

6.19.1 Предности и недостаци ECN механизма

Механизам експлицитног обавештења о загушењу (ECN механизам) има предности у односу над имплицитним механизмима:

1. Употреба ECN механизма спречава беспотребне губитке сегмената. На Интернету је могуће обавестити крајње системе о повећању саобраћаја и могућем загушењу пре него што дође до одбацивања пакета у рутерима. Поновно слање (ретрансмисија), које повећава саобраћај у мрежи, на тај начин је избегнуто;
2. Са ECN механизмом изворište може бити брзо и недвосмислено обавештено о загушењу у мрежи. Не мора да чека истицање времена на које је постављен ретрансмисиони часовник или три дупликата ACK сегмента.

Да би ECN механизам могао да се користи потребно је:

1. увођење два нова бита у TCP заглавље. Крајњи системи треба да буду у могућности да препознају и да постављају вредности ових битова;
2. да TCP целина може да размењује информације са суседним IP слојем (на истом систему);
3. да TCP целина омогући коришћење ECN механизма у фази успоставе везе;
4. да TCP целина може да одговори на пријем ECN информација,
5. увођење два нова бита у IP заглавље. IP целине у крајњим станицама и рутерима треба да буду у могућности да препознају и да постављају вредности ових битова;
6. да IP целина може да размењује информације са суседним TCP слојем (на истом систему);
7. да су IP целине у рутерима у могућности да постављају вредности ECN битова када дође до загушења.

[1] *Threshold*

6.19.2 Механизам ECN и IP заглавље

ECN механизам користи два бита најмање тежине поља диференциране[1] услуге у заглављу IPv4 или IPv6 пакета. Вредности битова, ознаке за пренос које омогућава ECN механизам (ECT[2]) и која обавештава о загушењу (CE[3]) и одговарајуће значење дато је у табели 6.6

AQM мехнизам може да, уместо да одбаци пакет, у заглавље, у поље које указује да је дошло до загушења (CE), постави одговарајућу вредност када такво поље постоји у IP заглављу и када траснпортни слој може да га разуме. Употреба CE ознака са ECN мехнизмом дозвољава пријемнику да се заштити од превеликих кашњења која настају као последица поновних слања због губитака пакета.

Вредност 01 или 10 поставља IP целина пошиљаоца да би указала да TCP веза за овај IP пакет подржава ECN механизам[4]. Рутери посматрају на исти начин пакете са вредностима 10 и 01 и пошиљалац може да користи једну или другу вредност. Основни разлог коришћења различитих вредности је механизам заштите. Пошиљалац користи било ECN(0) било ECN(1). Пријемник шаље исту ECT вредност (exo) у IP пакету који носи ACK сегмент.

Вредност 00 указује да пакет не користи ECN механизам. Вредност 11 поставља рутер да би обавестио одредишта да долази до загушења.

6.19.3 Експлицитно обавештавање о загушењу на TCP слоју

Као подршка ECN механизму користе се следеће ознаке:

- ECN exo (ECE[5]) ознака - користи је пријемна TCP целина да би обавестила предајну TCP целину да је добијен IP пакет са вредношћу ECN поља 11 (CE пакет).
- CWR ознака - користи предајна TCP целина да укаже пријемној TCP целини да је предајник смањио прозор загушења.

Ознаке ECE и CWR користе се у фази успоставе везе. Претпоставимо да станица 1 иницира успоставу TCP везе са станицом 2 и жели да користи

[1] D*ifferentiated Services* , D*iffServ*
[2] ECN *Capable Transport*
[3] *Congestion experienced*
[4] ECN *capable*
[5] ECN-*Echo*

Вредност	Ознака	Значење
00	Non ECT	Пакет не користи ECN механизам.
01	ECT (1)	Пренос који омогућава ECN (ECT). Шаље пошиљалац да укаже да TCP крајње тачке везе могу да користе ECN механизам.
10	ECT (0)	Пренос који омогућава ECN (ECT). Шаље пошиљалац да укаже да TCP крајње тачке везе могу да користе ECN механизам.
11	CE	Загушење откривено. Поставља рутер да би одредишном чвору (кориснику) указао да је дошло до загушења.

Табела 6.6. Вредности, ознаке и значење битова ECN механизма код IPv4 или IPv6 пакета

ECN. Процес иницијализације одвија се на следећи начин:

- Станица 1 шаље *Syn* сегмент станици 2 са вредностим ознака: ECE=1 и CWR=1 . На тај начин станица 1 указује да подржава ECN механизам и да је спремна да га користи и као предајни и као пријемник.
- Уколико станица 2:
  - јесте спремна да користи ECN механизам она као одговор станици 1 шаље *Syn_Ack* сегмент са вредностима ознака: ECE=1 и CWR=0.
  - није спремна да користи ECN механизам она као одоговор станици 1 шаље *Syn_Ack* сегмент са вредностим ознака: ECE=1 и CWR=0.

6.20 Транспортни протокол без успоставе везе

UDP протокол је дефинисан да би се омогућио пренос без успоставе везе (датаграм врста рада) у комуникационим системима са комутацијом пакета. Претпоставља се да је протокол мрежног слоја IP протокола. UDP протокол обезбеђује апликационим програмима процедуре да пошаљу податак другим апликационим програмима уз коришћење минимума механизама протокола. Он не гарантује ни поуздану испоруку, ни правилан редослед, ни заштиту од дуплих података. Апликације које захтевају правилан редослед и поуздану испоруку не треба да користе

Слика 6.48 Формат UDP сегмента

UDP протокол. Документ UDP протокол описан је у документу RFC 768.

Предност UDP протокола у односу на TCP протокол је:

- боља контрола на нивоу апликације над садржајем и временом слања,
- нема успостављања везе,
- не постоје стања везе,
- мање додатно оптерећење заглављем пакета. TCP сегмент има заглавље од 20 бајтова, док UDP има само 8.

Протокол UDP користе многе апликације као што је систем имена домена (DNS), пренос телевизијског сигнала (IPTV[1]) и гласа (VoIP[2]) преко мрежа са IP протоколом, надзор и управљање мрежом (SNMP), игрице преко интернет мреже.

6.20.1 Формат UDP сегмента

Изворишни порт је опционо поље. Уколико постоји указује на порт предајног процеса. Може се сматрати као порт на који се шаље одговор у недостатку других информација. Уколико се не користи поставља се на нулту вредност.

Одредишни порт има исто значење као TCP одредишни порт.

Дужина[3] представља укупну дужину датаграма укључујући заглавље и податке. Изражава се у броју октета (бајтова).

Контролни збир није обавезан за IPv4 али јесте за IPv6 протокол. Израчунава се за псеудозаглавље (заглавље сегмента проширеног IP адресом).

---

[1] *Internet Protocol TeleVision*

[2] *Voice over* IP

[3] *Length*

6.21 Питања и задаци

1. Модификовати и објаснити пример са слике 6.5 уколико пошиљалац шаље сегменте величине 2kB и апликација очита податак величине 8 kB.
2. Каква је функција SYN и ACK битова у заглављу TCP сегмента? Објаснити на примеру успоставе везе.
3. Како се користи пријемни редни број? Који бит треба да буде постављен да би се он могао користити?
4. Описати механизме за слање и пријем хитних података код TCP протокола.
5. Које механизме TCP протокол користи да би разрешио загушење у мрежи?
6. Који фактори отежавају управљање загушењем?
7. Укратко описати Јакобсонов алгоритам.
8. Укратко описати Карнов алгоритам.
9. Укратко описати брзу ретрансмисију и брзи опоравак.
10. У чему се разликују Тахо и Рено имплементације?
11. Каква је разлика између имплицитног и експлицитног обавештавања о загушењу?
12. Које измене треба урадити да би TCP протокол подржавао експлицитно обавештавање о загушењу?
13. Које измене треба урадити да би TCP протокол подржавао имплицитно обавештавање о загушењу?
14. Зашто максималан животни век пакета (TTL) мора да буде довољно дугачак да у међувремену нестану не само сви пакети, већ и потврде за њих?
15. На слици 6 видимо да, осим 32-битног поља Редни број потврђеног пакета постоји и бит ACK. Да ли он стварно служи нечему и зашто је он ту?
16. Опишите два начина за стизање у стање SYN RCVD са слике 6.16.
17. Наведите потенцијалан недостатак Наглеовог алгоритма у веома загушеној мрежи.
18. Размотрите ефекат примене спорог алгоритма на линији на којој је RTT време 10ms и без загушења. Прималац има прозор величине 24kB, а највећи сегмент је 2kB. Колико ће времена протећи док се не пошаље први пун прозор података?

19. Претпоставите да је TCP прозор загушења подешен на 18kB и да се часовник аутоматски искључио. На коју вредност ће се повећати прозор ако наредна четири послата скупа буду успешно примљена? Претпоставите да је 1kB максимална величина сегмента.
20. Ако RTT време тренутно износи 30ms, а потврде стигну после 26, 32 и 24ms, како ће Јакобсонов алгоритам проценити нову вредност RTT? За α употребите вредност 0,9.

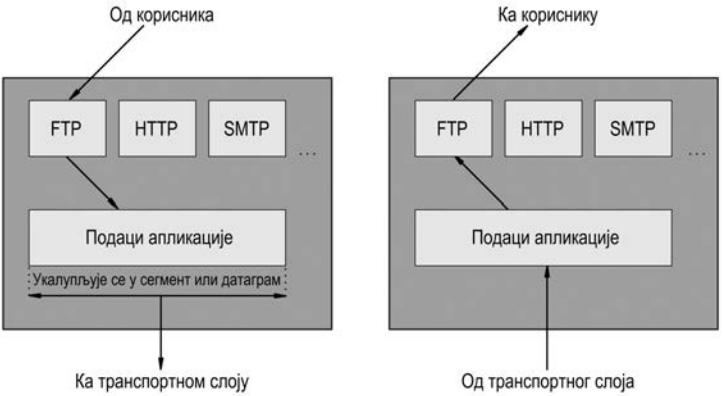
7. Апликациони слој и систем имена домена

Апликације су крајњи циљ рачунарске мреже. Ако не би постојала апликација која користи рачунарску мрежу не би постојала ни потреба за рачунарском мрежом ни за протоколима у рачунарским мрежама. У последњих 40 година направљен је велики број апликација за рачунарске мреже (мрежних апликација) као што су: систем имена домена (DNS[1]), текстуална електронска пошта, удаљени приступ рачунарима, пренос података и текстуални разговор[2] - веб[3] (обухвата кретање по мрежним серверима[4], претраживање и електронску размену), тренутна размена порука (IM[5]) са листом пријатеља, размена података између равноправних станица (P2P[6]), бројне аудио и видео апликације као што су интернет телефонија, размена видео података[7], интернет радио, интернет телевизија... Брзи приступ Интернету и све распрострањеније бежичне мреже отварају могућности за велики број нових апликација.

7.1 Апликација у TCP/IP референтном моделу

У TCP/IP референтном моделу не постоји услуга коју апликацији пружа слој сесије и презентације. Програмери су препуштени сами себи, али то не значи да су они дужни да направе све од самог почетка. На пример апликације могу да користе карактер оријентисану презентациону услугу која се назива мрежни виртуелни терминал (NVT[8]) и која је део

[1] *Domain Name System*
[2] *Chat*
[3] WWW (*World Wide Web)*
[4] *Web surfing*
[5] *Instant Messenger*
[6] *Peer to Peer*
[7] *Video streaming*
[8] *Network Virtual Terminal*



Слика 7.1 Начин функционисања TCP/IP апликација

Telnet спецификације или NetBIOS програмску библиотеку коју користе компаније IBM или Microsoft. Постоје многе услуге презентационог слоја које TCP/IP скуп протокола може да користи.

У примени TCP/IP протокола крајње станице обично обезбеђују различите апликације које омогућавају корисницима приступ подацима о којима воде рачуна протоколи транспортног слоја. Ове апликације користе бројне протоколе који нису део TCP/IP референтног модела као што су: HTTP[1] протокол који користе веб претраживачи, SMTP[2] протокол који користи електронска пошта и бројни други..

У TCP/IP окружењу апликациони протокол[3], апликациона услуга[4] и корисничка апликација[5] обично користи исто име.

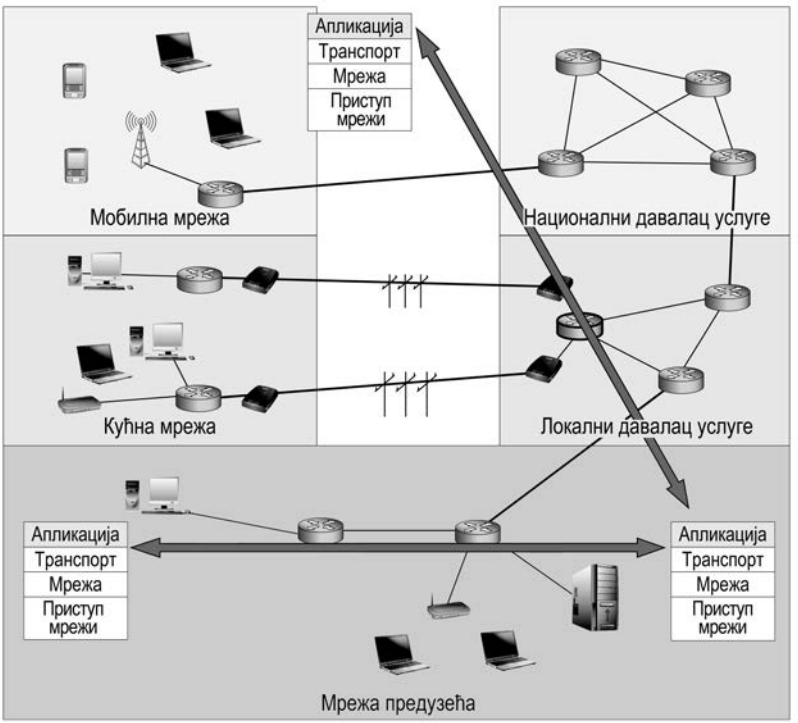
Протокол за пренос датотека (FTP[6] протокол) је и апликациони протокол, апликациона услуга и апликација коју корисник покреће. Некада је збуњујуће који од аспеката се анализира. Начин на који TCP/IP апликација функционише приказана је на слици 7.1 Треба уочити да се овај апликациони слој налази на врху TCP/IP референтног модела (TCP/IP скупа протокола) и интерфејс са програмима или корисницима је директан.

[1] *Hyper-Text Transfer Protocol*
[2] *Simple Message Transfer Protocol*
[3] *Application protocol*
[4] *Application service*
[5] *User application*
[6] *File Transfer Protocol*

7.2 Принципи пројектовања апликација

Када се пројектује и развија апликација за рачунарске мреже (мрежна апликација) основ су програми који раде на различитим крајњим системима и међусобно комуницирају преко мреже (слика 7.2). На пример, свака веб апликација има два програма која међусобно комуницирају: претраживач Интернета[1] (интернет претраживач) на рачунару клијента (стони, преносиви рачунар, преносиви дигитални асистент (PDA[2]), мобилни телефон) и програм веб сервер који се налази на серверу.

Значи развој нових мрежних апликација захтева писање програма на језицима C, C++, и Java који могу да раде на различитим крајњим



Слика 7.2 Комуникација мрежних апликација обавља се између крајњих система на апликационом слоју

[1] *Internet browser*

[2] *Personal Digital Assistant*

системима. Ове апликације не користе се у раду мрежних уређаја (језгра мреже[1], рутерима и комутаторима). Подсетимо се да ови уређаји и не раде на апликационом слоју већ на нижим слојевима (мрежном слоју, слоју везе и физичком слоју).

7.3 Архитектура мрежних апликација

Две најчешће архитектуре које се користе у модерним мрежним апликацијама су: клијент/сервер и архитектура равноправних рачунара (P2P[2]).

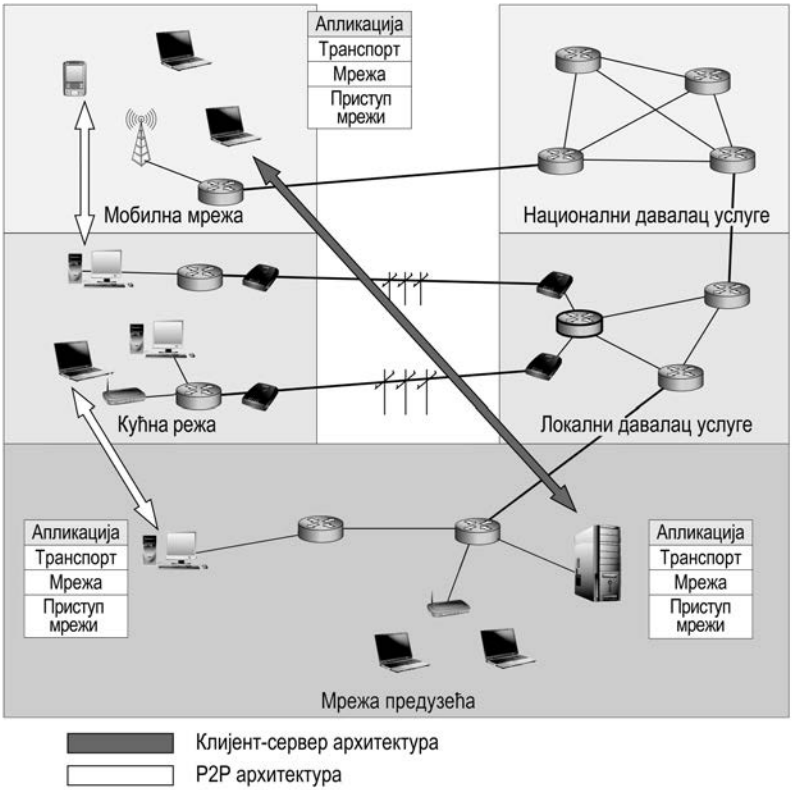
Код клијент/сервер архитектуре постоји рачунар који је увек укључен - сервер. Он обрађује захтеве од других рачунара - клијената. Пример је веб апликација на серверу, који је увек укључен, и који обрађује захтеве интернет претраживача који се налазе на рачунарима клијената. Када веб сервер добије захтев за неким ресурсом (документ, слика...) он захтевани ресурс прослеђује ка рачунару клијента. Клијенти код клијент/сервер архитектуре међусобно не комуницирају директно већ преко сервера.

Сервер у клијент/сервер архитектури користи статичку, општепознату IP адресу[3]. Неке од апликација које користе клијент/сервер архитектуру су веб, FTP, и е-пошта. Клијент/сервер архитектура приказана је на слици 7.2.

Код клијент/сервер апликације сервери често нису у могућности да опслуже све клијенте. Зато се користи група сервера (фарма сервера) која образује веома моћан виртуелни сервер. Услуге које су засноване на клијент/сервер архитектури захтевају веома сложену рачунарско - комуникациону инфраструктуру и подстичу добављаче интернет услуга (ISP[4]) да користе фарме сервера и брзе комуникационе линкове. Услуге које пружају као претраживачи (*Google, Yahoo*), интернет продаја (*Amazon* и *e-Bay*), социјалне мреже (*MySpace, Facebook*...) и размена видео садржаја (*Youtube*) захтевају велика финансијска улагања.

---

[1] *Core*
[2] *Peer to Peer*
[3] *Well-known address*
[4] *Service providers*



Слика 7.3 Клијент/сервер архитектура и архитектура равноправних рачунара

У архитектури равноправних рачунара (P2P архитектури) постоји минималан број сервера или их уопште нема. Уместо комуникације преко сервера корисници директно комуницирају користећи везу својих рачунара. Они су међусобно равноправни па се због тога називају равноправни рачунари или парњаци[1]. Равноправни рачунари нису сервери у власништву добављача интернет услуга, него су то стони или преносиви рачунари у власништу корисника. Оваква архитектура назива се архитектура равноправних рачунара (парњак/парњак[2]). Апликације са великим протоком засноване на архитектури равноправних рачунара

[1] *Peer*

[2] *Peer-to-peer*

су: размена података (*BitTorrent*), интернет телефонија (нпр. *Skype*) и интернет телевизија (*PPLive*). Архитектура равноправних рачунара приказана је на слици 7.3. Неке апликације комбинују клијент/сервер и парњак/парњак архитектуру. Код многих апликација за размену порука сервери се користе да прате IP адресе корисника, али се саме поруке шаљу директно између клијената (без проласка кроз сервер).

Једна од највећих предности парњак/парњак архитектуре јесте што је она самоорганизујућа. Код парњак/парњак апликација за размену података сваки равноправни рачунар генерише саобраћај захтевајући податке. Са друге стране он и повећава капацитет мреже пошто обезбеђује податке другим парњак рачунарима. Архитектура равноправних рачунара не захтева сложену инфраструктуру: ни сервере ни комуникационе линкове. Велики добављачи интернет услуга као што су MSN, *Yahoo* и сами су заинтересовани за парњак/парњак архитектуру због мањих трошкова потребних за одржавање и проширивање система.

7.4 TCP/IP прикључнице

У претходним поглављима анализирана су адресирања на различитим слојевима. Да се подсетимо: адресирање на трећем слоју (IP адресирање) веома је битно за исправан пренос података између уређаја. Са друге стране апликациони протоколи морају се бавити бројевима портова који се додељују свакој апликацији тако да оне могу правилно користити TCP и UDP протоколе.

То значи да апликациони процеси у ствари користе комбинацију IP адресе крајње станице на којој су покренути (прецизније мрежног интерфејса који користе) и број порта који су им додељени. Ова комбинација IP адресе и броја порта назива се прикључница (сокет[1]). Прикључница је значи одређена са:

<IP адреса >:<број порта>

Ако постоји веб сајт који се налази на IP адреси 41.199.222.3 прикључница која одговара HTTP[2] серверу за тај сајт биће: 41.199.222.3:80.

[1] *Socket*

[2] *Hypertext Transfer Protocol*

Понекад се могу прикључнице означити преко имена крајње станице (хоста) уместо IP адресе на следећи начин:

<име хоста>: <број порта>

Да би се користио овакав начин обележавања име мора бити разрешено коришћењем система имена домена (DNS). На пример може се наћи URL[1] адреса веб сајта *http://www.dobarsajt.com:8080*. Ово указује веб претраживачу да прво разреши име *www.dobarsajt.com* у IP адресу користећи DNS а затим пошаље захтев ка тој адреси користећи нестандардан број порта сервера 8080 који се повремено користи уместо порта 80.

Прикључница (сокет) врло је значајна за TCP/IP апликације. У ствари она је основа за апликациони програмски интерфејс (API[2]) са истим именом: прикључница (сокет). Верзија API интерферјса за оперативни систем *Windows* означава се као *Windows Sockets* или *WinSock*. Ови апликациони програмски интерфејси омогућавају апликационим програмима да једноставно користе TCP/IP скуп протокола за комуникацију.

Значи, размена података између два уређаја састоји се од серије порука које се шаљу од прикључнице једног уређаја до прикључнице другог уређаја. Сваки уређај ће нормално имати више таквих истовремених размена (конверзација). У случају TCP протокола веза се успоставља за сваки пар уређаја за време трајања комуникационе везе. Овом везом мора се управљати и то захтева да је она јединствено идентификована. То се реализује употребом пара идентификатора прикључнице за сваки од уређаја који је прикључен.

Без обзира што се станице[3] на мрежи (односно њихови мрежни интерфејси) препознају по IP адресама корисницима је лакше да користе име рачунара (хоста) уместо IP адресе (нпр. *www.viser.edu.rs* = 195.252.117.130). Корисник може да откуцај *www.viser.edu.rs* било где у свету и да добије главну веб страницу сајта Високе школе електротехнике и рачунарства у Београду.

---

[1] *Universal Resource Locator*

[2] *Application Program Interface*

[3] Користе се термини станица, рачунар или хост.

7.5 Основни концепти система имена домена

У TCP/IP окружењу систем имена домена (DNS[1]) представља дистрибуирану базу података која обезбеђује везу (мапирање) између IP адреса (IPv4 или IPv6) и имена рачунара. Свака од апликација TCP/IP архитектуре може да приступи дистрибуираној бази података. Поред повезивања имена рачунара са њиховим IP адресама обезбеђује и додатне информације потребне појединим услугама (нпр. информације потребне за усмеравање електронске поште).

Са стране апликације приступ DNS бази је преко софтверског окружења које разрешава проблем повезивања (мапирања) име - IP адреса (разрешивач[2]). Апликација мора да добије преведено име рачунара у IP адресу пре него што постави захтев транспортној целини да успостави TCP везу или пошаље датаграм користећи UDP протокол. Скуп TCP/IP протокола у оквиру оперативног система не зна ништа о систему имена домена (DNS).

Свака страна (факултет, смер, институција, организациона јединица...) одржава сопствену базу података на серверу коме други системи (клијенти) преко Интернета могу да приступе. Систем имена домена (DNS) обезбеђује протоколе који омогућавају клијентима и серверима да међусобно комуницирају.

На почетку за разрешавање имена рачунара у IP адресу користила се *hosts* текстуална датотека у којој су се налазили подаци о вези (мапирању) између имена рачунара и његове IP адресе. Датотека се налазила на централном рачунару и могла је да се учита коришћењем FTP протокола. Систем је био веома једноставан али неодговарајући за велику рачунарску мрежу као што је Интернет. Текстуалне датотеке[3] *hosts* могу се наћи на следећим местима:

- /etc/*hosts*, на Linux, UNIX и сличним оперативним системима,
- c:\windows\system32\drivers\etc*hosts* на оперативном систему *Windows*.

---

[1] RFC1034 специфицира концепт, а RFC1035 специфицира детаље примене.

[2] *Resolver*

[3] Формат датотеке је обичан текст, на *Windows* оперативном систему може се отворити програмом *Notepad*.

Идеја о хијерархијском систему имена, заснованом на доменима, први пут је објављена 1981. год. у документу RFC799[1]. До 1983. год. реализован је план да се са система „равне“ табеле пређе на хијерархијски систем имена домена (DNS).

Треба обратити пажњу на терминологију. Без обзира што је DNS скраћеница за систем имена домена (*Domain Name System*), слово „S“ означава и сервер (*Server*), и услугу (*Service*)[2].

Основне компоненте DNS система су:

1. DNS простор имена[3] - хијерархијски организован и користи комплексну вишенивоску структуру са тачно одређеним правилима за доделу имена;
2. DNS систем регистрације[4] имена - заснован на идеји хијерархије домена и организација које су за њих ауторитативне;
3. DNS разрешавање[5] - хијерархијски организовано и пројектовано као интеракција између компонената софтвера које разрешавају име[6] и сервера имена[7]. Ове компоненте софтвера проверавају запис[8] у бази података и користе специјални протокол да би се одговорило на захтев (упит) клијента.

Шта ради сервер када нема информације које се од њега траже? Он мора да потражи информације од другог сервера имена. Ово је пример дистрибуираности система имена домена (DNS). Сваки сервер имена не мора да зна како да ступи у везу са свим осталим серверима имена. Уместо тога сваки сервер имена мора да зна како да ступи у везу са сервером имена корена стабла[9].

---

[1] *Internet Domain Names*

[2] У овом уџбенику ће се користити (због боље разумљивости) DNS систем, DNS сервер и DNS услуга без обзира што су систем, сервер и услуга већ део DNS скраћенице.

[3] *DNS name space*

[4] *DNS registration system*

[5] *DNS resolution*

[6] *Name resolver*

[7] *Name server*

[8] *DNS resource records*

[9] *Root name servers*

7.6 DNS домени и хијерархијска архитектура

Најбитнији елемент система имена домена је организација (архитектура) имена која описује како се имена праве и интерпретирају. Архитектура система имена домена заснована је на концепту апстракције која се назива домен. У речницима се реч „домен“ повезује са појмом „сфера утицаја“ или „област управљања или меродавност“. Овакви системи су природно организовани хијерархијски. У DNS системима домен се дефинише или као један објекат или као група објеката који су скупљени на основу неког заједничког својства. У DNS системима то заједничко својство је да их администрира иста организација па је хијерархија имена уско повезана са ознаком о структури DNS хијерархијске ауторитативности (меродавности).

Простор имена домена може се представити као хијерархија домена са обликом окренутог дрвета (слика 7.4). По структури хијерархија домена слична је структури система датотека[1] са кореном[2] који садржи домене, сваки од њих садржи поддомене итд. Теоретски било који број поддомена може бити креиран.

Битно је подсетити се да било која самостална мрежа, или међусобно повезане мреже, могу да имају свој DNS простор имена и јединствену хијерархијску структуру. Тако постоји DNS простор имена глобалног Интернета али може да постоји и локални DNS простор имена.

Поређење са стаблом и његовим деловима са структурама (као што је DNS структура имена) веома је често у рачунарским мрежама. Разлика између технологије и природе је у томе што DNS стабло расте од врха ка дну уместо да расте као у природи увис. Аналогија са стаблом у описивању DNS структуре подразумева коришћење неколико појмова везаних за стабло од којих су неки представљени на слици 7.4. То су:

- Корен - концепцијски врх DNS структуре имена. Домен корена код DNS система садржи читаву унутрашњу структуру. По дефиницији нема име;
- Грана - целовити део DNS хијерархије. Састоји се од домена и

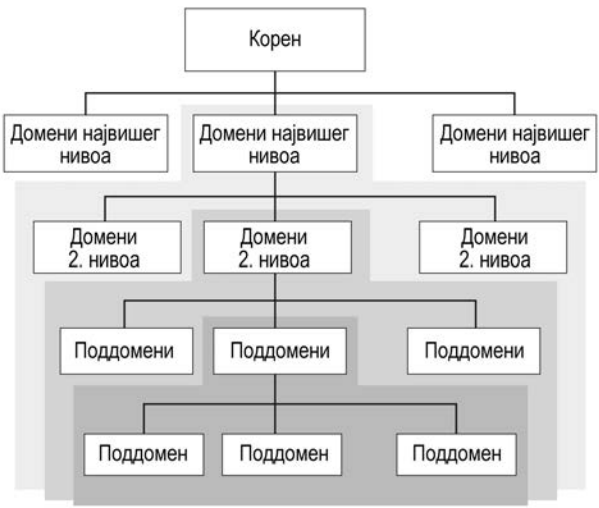
[1] *File system*
[2] *Root*

поддомена и свих објеката у оквиру њега. Све гране се спајају у корену, као и код правог стабла;

- Лист - крајњи објекат у структури, односно домен који нема ништа испод себе (аналогија са гранама на чијем крају се налази лишће).

Не постоји посебан термин који означава домен који није лист. Некада се означава као унутрашњи чвор указујући да се налази у средини структуре. Чвор је генерички рачунарски појам за објекат у топологији или структури и користи се у овом уџбенику. Тако је у DNS структури сваки чвор домен. Може да буде или унутрашњи чвор и садржи додатне домене и/или објекте, или лист који је уређај са одређеним именом. Тако је појам (термин) домен на неки начин нејасан пошто се односи или на скуп објеката који представљају грану дрвета или на лист.

Појам „поддомен“ може се користити генерички као и сама реч „домен“. У том случају односи се једноставно на везу између домена са поддоменима који су у структури испод другог домена. То значи да се за домене на нивоу врха може рећи да су поддомени корена; сваки домен



Слика 7.4 На врху DNS простора имена је корен, испод њега домени највишег нивоа, испод њих домени 2. нивоа и поддомени. Осенчене области представљају гране, гранчице и лишће (најтамније)

2. нивоа је поддомен домена највишег нивоа итд. Али, ипак, понекад поддомен значи одређени домен 3. или нижег нивоа.

У зависности од међусобне позиције домена у стаблу користе се следећи појмови:

- домен изнад одређеног домена у простору имена домена назива се родитељ домен[1];
- домени који су на истом нивоу и имају истог родитеља домена називају се сестре домени[2];
- поддомени одређеног домена називају се децом тог домена.

Важно је напоменути да постоје одређена ограничења: простор имена домена (DNS простор имена) мора бити организован као право тополошко стабло. То значи да сваки домен има само једног родитељ домена и да у топологији стабла петље низу дозвољене.

У стаблу чвор може бити јединствено идентификован на основу свог потпуно квалификованог имена домена (FQDN[3]). Користи се назив потпуно квалификовано име домена пошто оно носи све податке (скуп ознака) о тачној позицији тог домена у оквиру целог DNS простора имена. Потпуно квалификовано име домена дефинише апсолутну локацију домена.

Сваки чвор има ознаку (лабелу). Ознака може да има највише 63 карактера. Два „детета“ у оквиру истог чвора не могу имати исту ознаку.

Постоји и делимично квалификовано име домена (PQDN[4]). То је релативно име и има значење само у одређеном контексту. Делимично име мора се интерпретирати у оквиру тог контекста да би се потпуно одредио чвор. На пример потпуно квалификована имена домена су: *rt.viser.edu.rs*, *nrt.viser.edu.rs*, *avt.viser.edu.rs* а делимично квалификована имена била би: *rt*, *nrt* и *avt*.

Корен стабла је специјалан чвор без ознаке. Име домена сваког чвора стабла је скуп ознака које почињу од тог чвора идући ка корену, са тачком која раздваја ознаке.

---

[1] *Parent domain*

[2] *Siblings domain*

[3] *Fully Qualified Domain Name*

[4] *Partialy Qualified Domain Name*

7.7 Регистрација и администрација

Као што смо видели DNS простор имена састоји се од хијерархије домена и поддомена. Почевши од корена постоји више домена највишег нивоа, испод њих домени 2. нивоа и домени нижих нивоа испод њих. Набројаћемо нека од питања која се могу поставити?

1. Како се прави облик и структура простора имена и ко тим управља?
2. Ко управља кореном стабла и одлучује о томе како ће се домени највишег нивоа звати?
3. Како ће се поделити управаљање поддоменима остатка простора имена?
4. Како ће се осигурати да не дође до сукоба у одабиру имена?

DNS систем може се користити у приватним мрежама којима управља једна организација. Значи додела имена могла би бити локална (приватна). Међутим од највећег значаја је додела имена на јавној мрежи Интернету. То значи да је потребно направити глобални простор имена који обухвата милионе рачунара којима управљају различите организације. Због тога је потребан процес регистрације и администрације који тако сложен задатак може да подржи.

Хијерархијска организација простора имена домена осликава хијерархију организација које управљају доменима и чворовима у оквиру њих. То значи да постоји хијерархија ауторитета[1] која одговара хијерархијској структури DNS имена.

Да би се направио простор имена домена потребно је прво поставити домене највишег нивоа (TLD[2] домени). Сваки од њих треба да буде јединствен и решење је да једна организација треба да руководи постављањем TLD домена. То значи да организација која је меродавна (ауторитативна) за управљање доменом корена (коме припадају TLD домени) практично управља целокупним простором имена домена.

У самом почетку организација која је меродавна (ауторитативна) за DNS систем Интернета и која управља TLD доменима била је Мрежни

[1] *Authority*
[2] *Top Level Domains*

информациони центар (NIC[1]). Касније је организација NIC прерасла у организације које додељују нумерације на Интернету: IANA[2] и ICANN[3].

Са ауторитативношћу јавља се и одговорност за одређени домен да води рачуна, региструје имена у оквиру тог домена. Када се име региструје креира се одговарајући скуп података за то име. Те податке могу користити међумрежни уређаји за разрешавање имена у IP адресу или се могу користити за неку другу намену. Скуп свих података који описује DNS домен чини базу података DNS имена. Као што је организациона надлежност дистрибуирана и хијерархијска тако је и база података дистрибуирана и хијерархијска.

Не постоји једно место где су информације о DNS именима смештене. Уместо тога на DNS серверима налазе се записи о ресурсима[4] који описују домене за које су меродавни (ауторитативни). Чињеница да је база података дистрибуирана има основни утицај на начин на који се разрешавање имена реализује.

Организација IANA је до сада дефинисала следеће групе домена највишег нивоа[5]:

- Домен *arpa* је специјалан домен који је дефинисан као привремени домен за миграцију са старих *host* датотека. Скраћеница потиче од *Advanced Research Project Agency*, оснивача и претече Интернета. Данас се *.arpa* домен користи за интерно управљање Интернетом и за инверзно DNS претраживање.
- Домени означени са три карактера називају се генерички или организацијски домени (gTLD[6]). Термин генерички указује да би свака организација могла да пронађе себи одговарајуће место. Овај број је у односу на почетни проширен 2001/2002. године и дат је у табели (7.1). Свака грана стабла представља један DNS домен.

---

[1] *Network Information Center*
[2] *Internet Numbers Authority*
[3] *Internet Corporation for Assigned Names and Numbers*
[4] *Resource Records*
[5] *Top Level Domains*
[6] *generic Top-Level Domains*

gTLD	Целина	Опис
.aero	*air-transport industry*	Организације које се баве авио превозом
.asia	*Asia-Pacific region*	Азија, Аустралија и Пацифичка регија
.biz	*business*	Посвећене искључиво приватном послу
.cat	*catalan*	Посвећено веб сајтовима на каталонском језику
.com	*commercial*	Комерцијалне организације
.coop	*cooperatives*	Организације као на пример кредитна удружења
.edu	*educational*	Образовне институције
.gov	*governmental*	Владине организације у САД
.info	*information*	Може се користити без икаквих ограничења
.int	*international organizations*	Користе организације или програми које воде најмање две нације
.jobs	*companies*	Користе организације које оглашавају послове
.mil	*U.S. military*	TLD домен ограничен на војску САД
.mobi	*mobile devices*	Могу користити мобилни сајтови у складу са стандардима
.museum	*museums*	Мора бити верификован као музеј
.name	*individuals, by name*	Отворени TLD домен. Било која особа може се регистровати
.net	*network*	Отворени TLD домен. Било која особа или целина може се регистровати
.org	*organization*	Отворени TLD домен. Било која особа или целина може се регистровати
.pro	*professions*	За сада је *.pro* резервисан за адвокате, рачуновође, инжењере који поседују лиценцу у Француској, Канади, Великој Британији и САД.
.tel	*Internet communication services*	Услуга која укључује везу између телефонске мреже и Интернета
.travel	*travel and tourism industry related sites*	Мора бити верификован као целина која се бави путовањима

Табела 7.1 Домени највишег нивоа (TLD домени)

- Домени означени са два карактера називају се домени држава или географски домени (ссTLD[1]). Ознаке одговарају коду државе и могу се наћи у стандарду ISO3166.

7.7.1 Сервери имена корена стабла

Сервери имена корена стабла[2] (сервери корена стабла) постављени су на врху пирамиде DNS система. Данас су ти сервери улазне тачке DNS услуге и задатак им је да чувају податке[3]. Сервери имена корена стабла имају податке о томе ком серверу имена локални DNS сервер треба да се обрати као следећем у низу разрешавања упита. Када сервер имена корена стабла добије захтев за разрешавање нпр. IP адресе за gTLD домен *.edu* као резултат даје листу сервера имена који можда знају одговор. Континуирано памћење одговора у меморији (кеш) значи да нема потребе слати упит кореном серверу имена после првог упита.

Оператори сервера корена стабла одржавају своју сопствену инфраструктуру за случај кварова, шифроване е-поште и сигурности. Сервери имена корена стабла користе дистрибуирани систем „један ка једној од станица“ (еникаст[4]) где је направљен већи број одвојених система широм света, али се сви они појављују и ради као један систем са једном IP адресом. Сматра се да употреба еникаст система минимизира ефекат одбијања услуге (DoS[5] напада).

Да се подсетимо: начин рада еникаст система сличан је начину рада система „један ка групи станица“ (мултикаст[6]): постоји један ка више асоцијација између адреса и одредишта на мрежи (код мултикаста постоје групе). Код система „један ка једној од станица“ (еникаст) одредишна адреса представља скуп пријемних крајњих тачака али само једна од њих, за коју се установи да је „најближа“ или „најбоља“, бира се да у одређеном тренутку добије информацију од одређеног пошиљаоца. За разлику од система емитовања „једног ка свима“ (бродкаст) у коме

---

[1] *country-code Top Level Domains*

[2] *Root name server.* Може се користити и термин коренски сервери имена или сервери имена корена.

[3] Њихове базе податак су минималне и углавном дају одговоре на основу већ ускладиштених информација (кешираних).

[4] *Anycast*

[5] *Denial-of-Service*

[6] *Multicast*

се порука упућује свим одредиштима или мултикаста у коме се порука упућује свим заинтересованим одредиштима, порука са еникаст адресом упућује се само једној од крајњих станица. Постоји много више од 13 (означених са A-M) сервера корена стабла као што се може видети у табели 7.2.

Најновије информације о серверима корена стабла (адресе IPv4 и IPv6, где се користи,...) могу се наћи на *www.root-servers.org*. На пример IPv4 адреса за *B.rootservers.net* промењена је 2004. године.

Сервер	Оператор	Локација, број (глобални/локални)	Софтвер
A	*Verisign*	*Dulles*,VA (6/0)	BIND[1]
B	*Information Sciences Institute*	*Marina Del Rey*, CA	BIND
C	*Cogent Communications*	*Herndon VA; Los Angeles; New York; Chicago* (6/0)	BIND
D	*University of Maryland*	*College Park*, MD, US, (1/0)	BIND
E	NASA[2]	*Mountain View,* CA (1/0)	BIND
F	I*nternet Systems Consortium Inc.*	Дистрибуирана, користе еникаст (2/43)	BIND 9
G	DISA[3]	Дистрибуирана, користе еникаст (6/0)	BIND
H	*U.S. Army Research Lab*	*Aberdeen*, MD, US (2/0)	NSD[4]
I	*Netnod*	Дистрибуирана, користе еникаст (38)	BIND
J	*Verisign*	Дистрибуирана, користе еникаст (63/7)	BIND
K	RIPE NCC	Дистрибуирана, користе еникаст (5/13)	NSD
L	ICANN	Дистрибуирана, користе еникаст (107)	NSD
M	WIDE *Project*	Дистрибуирана, користе еникаст (5/1)	BIND

Табела 7.2 Списак сервера имена корена стабла, њихове локације и софтвер који користе

[1] Највише коришћен софтвер за DNS системе на Интернету. На *Unix-like* оперативним системима *de facto* стандард.

[2] *National Aeronautics and Space Administration*

[3] *Defense Information Systems Agency*

[4] На Интернету NSD (*Name Server Daemon*) је програм отвореног кода за DNS системе. Развиле су га *NLnet Labs* (Амстердам) и RIPE NCC (*Réseaux IP Européens Network Coordination Centre*).

Треба уочити да већина сервера корена стабла представља дистрибуирани систем понекад размештен по целом свету иако је груписан под истим именом. Ту долази до изражаја предност еникаст система адресирања. У прошлости потреба DNS сервера да прихвати иновирање података од било ког извора била је сигурносна слабост. Савремени DNS сервери прихватају нове податке само од ауторизованих извора или дигитално потписаних података. Тренутно се континуирано ради на развоју сигурносног проширења система имена домена (DNSSec[1]).

Од 1993. године примарни сервери имена морају да знају IP адресе сваког сервера корена стабла (не њихова DNS имена). Сервер имена корена стабла зна име и локацију (тј. IP адресу) ауторитативних сервера имена за домене другог нивоа. Ово доводи до итеративног процеса, тако да сервер коме је упућен захтев мора да ступи у везу са сервером у корену стабла.

7.8 Зоне и ауторитативност

Подсетимо се да се две ствари из TCP/IP скупа протокола глобално администрирају: мрежни део IPv4 или IPv6 адресе и име домена које иде уз ту адресу. Део IP адреса крајње станице (хоста) и остали подаци система имена домена администрирају се локално.

Треба анализирати додељивање (делегирање) одговорности у оквиру система имена домена (DNS). Ниједна од целина није задужена за све ознаке у стаблу. Уместо тога једна целина води рачуна о делу стабла (доменима на нивоу врха) и делегира одговорност другима за одређену зону (слика 7.5). Зона је део DNS стабла који се администрира независно. Многи домени другог нивоа деле своје зоне у мање. На пример високошколске установе се деле у зоне које се односе на одређене делове високошколске установе. Једном када се додели[2] меродавност (ауторитативност[3]) ствар је администратора зоне да обезбеди сервер имена за ту зону. Када се нови систем инсталира у

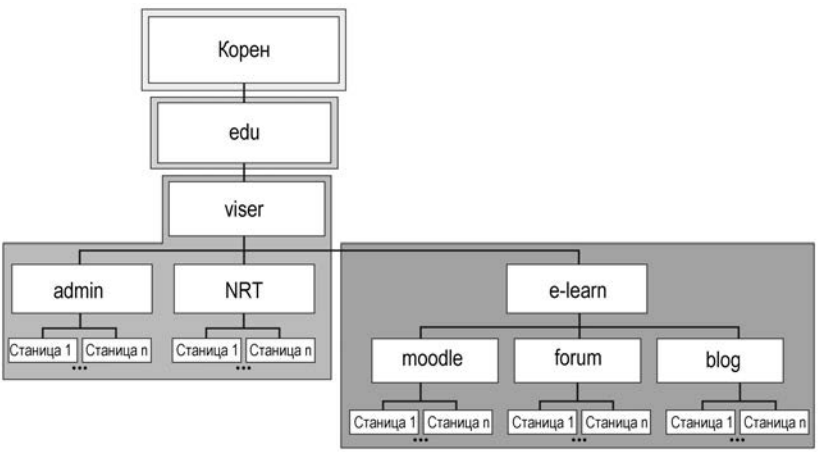
[1] *DNS Security Extensions*
[2] *Delegated*
[3] *Authority*

одређеној зони DNS администратор те зоне додељује му име и IP адресу и доставља ове информације родитељском домену, који са своје стране креира запис којим се неки поддомен делегира. Овде се види потреба за делегирањем.

За малу организацију као што је на пример Висока школа електротехнике и рачунарства једна особа може да обавља овај посао али за Београдски универзитет то би било немогуће. Сервер имена може да буде ауторитативан (меродаван) за једну или више зона.

Особа која је задужена за зону мора да обезбеди примарни сервер имена за ту зону и један или више секундарних сервера имена. Примарни и секундарни сервери морају да буду међусобно независни и редундантни сервери тако да услуга коју пружа та зона није угрожена ако је један од система (примарни или секундарни) у квару. Разлика примарног и секундарног сервера имена је у томе што се информације уносе и складиште у примарни сервер имена а секундарни сервер те информације добија од примарног сервера и складишти их.

Све промене - додавање домена или рачунара обављају се на нивоу примарног сервера. Пребацивање информација од примарног сервера ка секундарном серверу назива се пренос зоне[1].



Слика 7.5 DNS зона ауторитета

[1] *Zone transfer*

7.9 Интернационални домени држава (IDN ccTLD)

Једанаест интернационалних имена домена (IDN[1]) на нивоу врха (TLD) инсталирано је за тестирање октобра 2007. год. и моћи ће да се користе одређени период времена. Сваки од ових домена представља реч „тест“ на одговарајућем језику. На пример на руском *http://пример.испытание/*. Ова група домена садржи најмање једну ознаку у софтверској апликацији, или цео део, у алфабету као што су арапски, кинески, руски, грчки... Систем имена домена који разрешава имена у IP адресе користи ASCII карактере[2].

Октобра 2009. године организација IANA дозволила је прављење кода држава које користе интернационална имена домена (IDNA[3]). На пример IDNccTLD[4] за Руску Федерацију је *.рф* а DNS име је *.xn--p1ai*. Постављена су IDNccTLD домени са арапским писмом (за Египат, Саудијску Арабију и Уједињене Арапске Емирате), са ћириличним писмом (за Руску Федерацију, Србију) и кинеским писмом (за Републику Кину, Хонг-Конг и Тајван).

Од 27.1.2012. год. власници интернет сајтова у Србији и дијаспори могу да региструју и ћирилични .срб домен.

7.10 DNS база података

DNS база података ослања се на хијерархијску базу података која садржи запис о ресурсима (RR[5] запис) који садржи име, IP адресу и друге информације о крајњој станици. Кључне карактеристике базе података су следеће:

- променљиве дужине хијерархија за имена: DNS дозвољава суштински неограничен број нивоа и користи тачку „ .“ да би раздвојио имена;
- дистрибуирана база података: база података налази се на

[1] *Internationalized Domain Name*

[2] Суштински могу се користити не-ASCII карактери, тј. регистровати домени националним алфабетом а он се локално кодира у компатибилан ASCII кôд (*ASCII Compatibile Coding*).

[3] *Internationalizing Domain Names in Applications* механизам дефинисан 2003. године за вођење рачуна о интернационалним именима домена која садрже не-ASCII карактере.

[4] *Internationalized country code top-level domain*

[5] *Resource Records*

Слика 7.6 Формат DNS записа ресурса (RR записа)

DNS серверима размештеним по Интернету и приватним интранетима;

- дистрибуција се контролише базом података: DNS база података подељена је у хиљаде посебно управљаних зона којим управљају различити администратори. Дистрибуцијом и иновирањем записа управља софтвер база података.

На пример сваки пут када се шаље е-писмо или када се приступа некој веб страници мора да постоји DNS претраживање које даје адресу сервера е-поште или веб сервера. На слици 7.6 приказана је структура RR записа.

Запис (величине 16. битова) се састоји из следећих елемената:

- име[1] - име власника записа, односно име чвора на који се запис о ресурсу односи;
- врста (тип)[2] - одређује врсту ресурса у запису. Различите врсте записа дате су у табели 7.3;
- класа[3] - одређује фамилију протокола. Једина обично коришћена вредност је IN за Интернет;
- време трајања (TTL[4]) - обично када се RR запис очитава са сервера имена он се и памти у кеш меморији тако да не мора често да се шаље упит серверу имена. Вредност указује на то

[1] *Name*
[2] *Type*
[3] *Class*
[4] *Time to Live*. Разликовати од TTL за спречавање бесконачог времена живота IP пакета.

колико дуго запис о ресурсу може да буде сачуван у кеш меморији пре него што се поново очита из извора информације. Нулта вредност значи да се запис о ресурсу може користити уколико је трансакција у току и не треба да се чува у кеш меморији;

Врста записа	Опис
A	Адреса крајње станице. Ова врста записа повезује име система са његовом IP адресом. Неки системи (нпр. рутери) имају више адреса и постоји посебан запис за сваку од њих
CNAME	Каноничко име[1]. Специфицира надимак (псеудоним[2]) за крајњу станицу и повезује га са каноничким (правим) именом
HINFO	Информација о крајњој станици. Одређује процесор и оперативни систем који крајња станица користи
MINFO	Поштанско сандуче[3] или информација о списку е-адреса[4]. Повезује поштанско сандуче или списак е-адреса са именом крајње станиц
MX	Размена поште[5]. Идентификује систем који преусмерава пошту у организационој јединици
NS	Меродаван[6] (ауторитативан) сервер имена за овај домен
PTR	Показивач имена домена. Користи се у зонама за реверзно претраживање
SOA	Почетак зоне ауторитета[7] (представља глобалну информацију о зони). Укључује параметре везане за ову зону
SRV	За задату услугу обезбеђује сервер или сервере имена у домену у коме пружа услугу
TXT	Произвољан текст. Обезбеђује начин да се додају текстуални коментари у бази података
WKS	Добро позната услуга[8]. Може да се добије списак апликационих услуга доступних крајњој станице

Табела 7.3 Врсте записа ресурса (RR записа)

[1] Користи се и термин канонски (правилан), може се наћи у литератури [6].
[2] *Alias*
[3] *Mailbox*
[4] *Mail list*
[5] *Mail exchange*
[6] *Authoritative*
[7] *Start of zone authority*
[8] *Well-known service*

- дужина података о ресурсу[1] - изражена бројем бајтова (октета);
- подаци о ресурсу[2] - низ променљиве дужине који описује ресурс. Формат ове информације мења се у зависности од вредности поља „врста“ или „класа“.

Примери записа[3]:

- уколико је A врста записа „име“ је име станице а „податак о ресурсу“ је IP адреса станице - значи да A врста записа обезбеђује стандардно повезивање IP адресе - име станице;
- уколико је NS врста записа значи да „име“ означава име домена (нпр. „*viser.edu.rs*“) а „податак о ресурсу“ је име станице ауторитативног DNS сервера који зна како да обезбеди IP адресу станицама у домену. Овај запис се користи да усмерава DNS упите даље у ланцу упита. На пример („*dns1.viser.edu.rs*“);
- уколико је „врста“ записа MX „Податак о ресурсу“ је име крајње станице (хоста) или канонично име сервера електронске поште (нпр. „*mailserver1.viser.edu.rs*“).

7.11 DNS поруке

Све комуникације у оквиру DNS протокола реализују се разменом порука. Користи се UDP порт 53 (углавном) и TCP порт 53 за пренос зоне (трансфер зоне) и теоретски велике одговоре. Порука је подељена у 5 делова (од којих су неке поруке празне у одређеним случајевима) као што је представљено у табели 7.4

Заглавље	Опис
Упит (*Question*)	Упит серверу имена
Одговор (*Answer*)	Запис о ресурсу као одговор на упит
Ауторитет (*Authority*)	Запис о ресурсу указује на ауторитет (меродавност)
Додатак (*Additional*)	Запис о ресурсу који садржи додатне информације

Табела 7.4 Општи формат DNS поруке

[1] *RDLength*

[2] *Rdata*

[3] Неки од типова, WKS, HINFO, MINFO постоје само у теорији и тешко се може срести њихова имплементација (услуга која би то користила). Проблеми које су они имали да реше или су решени на други начин или нису ни интересантни за решавање.

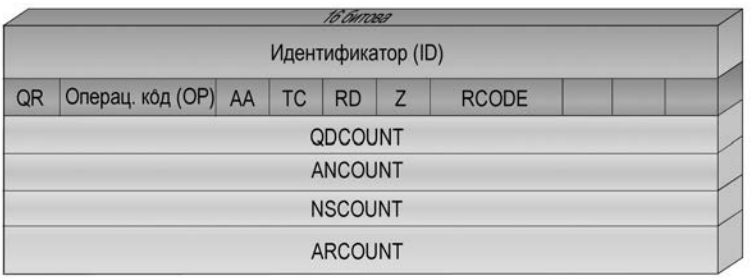
Заглавље DNS поруке је увек присутно. Оно укључује поља која указују који од осталих делова је присутан и такође указују на то да ли је порука упит или одговор, стандардни упит или неки операциони кôд итд...

Имена делова иза заглавља изведена су на основу њихове употребе у стандардном упиту.

Заглавље DNS поруке садржи следећа поља (слика 7.7)

Где је:

- ID - 16-битни идентификатор упита, који додељује програм који прави било коју врсту упита. Овај идентификатор се копира у одговарајући одговор и користи га онај који је поставио упит да би придружио одговор постављеном упиту.
- QR - једнобитно поље које указује да ли је реч о упиту (0) или одговору (1);
- операциони кôд[1] - четворобитно поље које специфицира врсту упита у овој поруци. Ово поље поставља извориште упита и копира се у одговор. Вредности су:
  - 0 - за стандардни упит[2],
  - 1 - за инверзни упит[3],
  - 2 - захтев за статусом сервера[4] и
  - од 3 до 15 - резервисане за будућу употребу;
- AA[5] - овај бит је важећи у одговору и указује да је сервер који



Слика 7.7 Заглавље DNS поруке

[1] *Opcode*
[2] *Query*
[3] *Iquery*
[4] *Status*
[5] *Authoritative Answer*

шаље одговор ауторитативан (меродаван) за име домена које се налази у делу упита. Садржај у делу за одговор може да садржи више имена због псеудонима (CNAME). Бит AA одговара имену које је било наведено у упиту или имену првог власника у делу одговора;

- TC[1] - указује да је ова порука скраћена због тога што је дужа од дозвољене за тај преносни канал[2];
- RD[3] - овај бит може бити постављен у упиту и ископиран у одговору. Уколико је постављен упућује сервер имена да реализује рекурзивни упит. Рекурзивни упит је опцион.
- RA[4] - овај бит је постављен на 1 или обрисан (0) у одговору и указује да ли сервер имена може да подржи рекурзивни упит;
- Z - резервисан за будућу употребу. Мора да буде постављен на нулту вредност у свим упитима и договорима;
- RCODE[5] - ово четворобитно поље је постављено као део одговора. Одговори могу да буду кодирани од 0 до 5 и означавају да: није дошло до грешке, грешка је на серверу, грешка је у имену (име домена не постоји), није имплементиран (та врста упита не постоји) и сервер имена је одбио да одговори;
- QDCOUNT - колики је (број) упита;
- ANCOUNT - колики је (број) одговора са записима о ресурсима;
- NSCOUNT - колики је (број) ауторитативних записа о ресурсима и
- ARCOUNT - колики је (број) додатних записа о ресурсима.

Упит садржи поља која описују упит послат серверу имена. Ова поља су врста упита, класа упита, име домена. На пример адреса станице која одговара имену (*Type* A) или сервер електронске поште (*Type* MX). Ова три поља имају исти формат: листа (можда и празна) међусобно повезаних записа о ресурсима.

---

[1] *TrunCation*

[2] Ово је зато што постоји максимална дужина UDP сегмента. Теоретски би прималац требало да покуша да упостави TCP сесију са DNS сервером како би примио комплетан одговор. Успостављање TCP везе може се користити за DoS напад, па је обично TCP порт блокиран на серверима без обзира што је по стандарду дозвољен.

[3] *Recursion Desired*

[4] *Recursion Available*

[5] *Response code*

Одговор садржи запис о ресурсима који је одговор на постављени упит. Подсетимо се да у сваком запису о ресурсу постоји врста (на пример A, NS, CNAME и MX), вредност и TTL. У делу „Одговор“ може да се нађе више записа о ресурсима пошто станица може да има више IP адреса.

Одговор садржи секцију надлежност[1] која садржи запис о ресурсу који указује на ауторитативан (меродаван) сервер имена.

Секција додатак[2] садржи запис о ресурсима који је везан за упит, али није тачан одговор на постављени упит. На пример одговор на MX упит садржи запис о ресурсу који даје каноничко име за сервер електронске поште. Додатно поље садржи нпр. A запис са IP адресом сервера електронске поште.

7.12 Програмска подршка DNS система

Постоји више алата који се користе код DNS система. Најчешће коришћени су: *nslookup*[3], *dig*[4], и *host*. Алат *nslookup* је доступан на многим оперативним системима. Може се послати најједноставнији DNS упит серверу који је конфигурисан на радној станици. На пример, у командној линији (или терминалском прозору) уколико се откуца:

> nslookup www.google.com

добија се као одговор:

Server: www.l.google.com

Addresses: 74.125.79.99, 74.125.79.104, 74.125.79.147

Aliases: www.google.com

Алат *dig* омогућава да се произвољним DNS серверима шаљу упити за жељени тип записа. Програм може приказати и статусне битове који постоје у одговору на DNS упит што некад може да олакша решавање проблема. Алат *dig* подржава и упите за интернационална имена домена

[1] *Authority section*
[2] *Additional section*
[3] *Name server lookup*
[4] *Domain Information Groper*

(IDN упите), део је BIND[1] софтверског окружења и све више замењује старије алате као што су *nslookup* и *host*.

Без обзира на то што *hosts* датотека више није основни механизам који се користи у TCP/IP системима за разрешавање имена, она још увек има важну функцију у разрешавању имена на свакој појединачној станици. Први разлог је тај што се може користити у мањим TCP/IP мрежама. У њима је могуће проблем разрешавања имена решити тако што се на све крајње станице (хостове) ископира идентична *hosts* датотека са подацима о вези IP адреса-име за све рачунаре у мрежи. Други разлог је тај што и када се у мрежи користи DNS систем корисно је да се битни сервери наведу у *hosts* датотеци како би дати рачунар био отпоран на тренутне проблеме у раду DNS система. На оперативним системима *Linux* и *Windows* прво се проверавају подаци из host датотеке пре обраћања DNS серверу.

У конзоли оперативног система Linux (нпр. Ubuntu) ако се откуца:

less /etc/hosts

свака линија у датотеци описује једну станицу. Формат линије је:

ip_adresa ime_hosta или ip_adresa ime_hosta pseudonim_hosta.

Пример садржаја *host* датотеке дат је у табели 7.5.

Можемо видети да се на основу записа у *host* датотеци *localhost* мапира у адресу локалне петље 127.0.0.1. На *Linux* оперативном систему,

ip_adresa	ime_hosta	pseudonim_hosta
127.0.0.1	localhost	
172.17.50.4	newu.viser.edu.rs	newu
# The following lines are desirable for IPv6 capable hosts		
::1	localhost ip6-localhost ip6-loopback	
fe00::0	ip6-localnet	
ff00::0	ip6-mcastprefix	
ff02::1	ip6-allnodes	
ff02::2	ip6-allrouters	

Табела 7.5 Пример TCP/IP *host* датотеке

[1] *Berkeley Internet Name Domain, Berkeley Internet Name Daemon*

уколико рачунар има статичку адресу, у *host* датотеци би требало да се налази мапирање имена рачунара и IP адресе. Пошто није могуће пописати све рачунаре на Интернету у сопствену *hosts* датотеку за нормално функционисање станица на Интернету важно је да станица буде конфигурисана најмање са једном адресом DNS сервера.

Софтверско окружење BIND је најчешће коришћена имплементација DNS сервера. На *Linux*[1] системима сматра се де факто стандардом. Прву верзију написала су четири студента дипломских студија са Калифорнијског универзитета[2] под покровитељством организације DARPA. Верзија 4.3BSD направљена је 1988. године а од верзије 4.9.3 њено одржавање прешло је у надлежност организације *Internet Systems Consortium*. У тренутку писања овог уџбеника препоручује се употреба BIND9 сервера. Ова верзија укључује подршку за DNSSEC[3] и IPv6 мултипроцесорски рад. Верзија BIND9 у кључена је у готово све *Linux* дистрибуције.

7.13 Процес разрешавања имена

Подаци о DNS именима смештени су у дистрибуираној бази података која се налази на великом броју сервера на различитим местима у свету. Због тога се разрешавање имена не може одрадити кроз једноставан процес слања једног упита на који се добије један одговор. Прво је потребно наћи сервер који садржи одговарајуће податке. То обично захтева размену више порука почевши од поруке упућене кореном серверу до поруке упућене серверу који садржи запис о ресурсу који корисник (клијент) потражује.

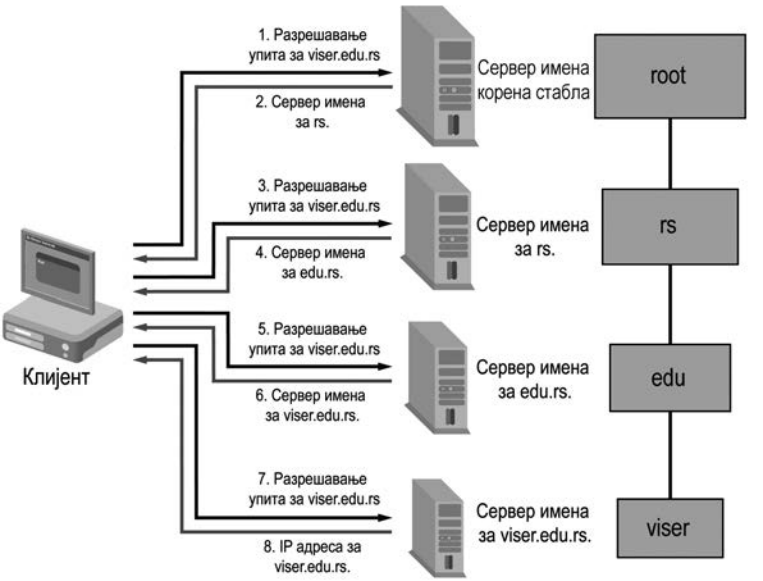
Користе се два начина разрешавања имена: итеративно и рекурзивно. У итеративном начину разрешавања имена уколико клијент пошаље захтев ка DNS серверу имена који не садржи информацију која је клијенту потребна он га упућује на други DNS сервер коме клијент упућује нови захтев (слика 7.8).

Код рекурзивног начина разрешавања имена уколико клијент пошаље захтев DNS серверу који не садржи захтевану информацију (запис о

[1] Свим „Unix-like“ системима.

[2] *University of California, Berkeley*

[3] *DNS Security Extensions*



Слика 7.8 Поједностављен приказ итеративног упита

траженом ресурсу) тај DNS сервер преузима одговорност за слање захтева другим серверима и изналажење потребних информација у прослеђивању клијенту. У том случају он се појављује као клијент у односу на друге сервере којима се обраћа.

7.13.1 Рекурзивно разрешавање имена

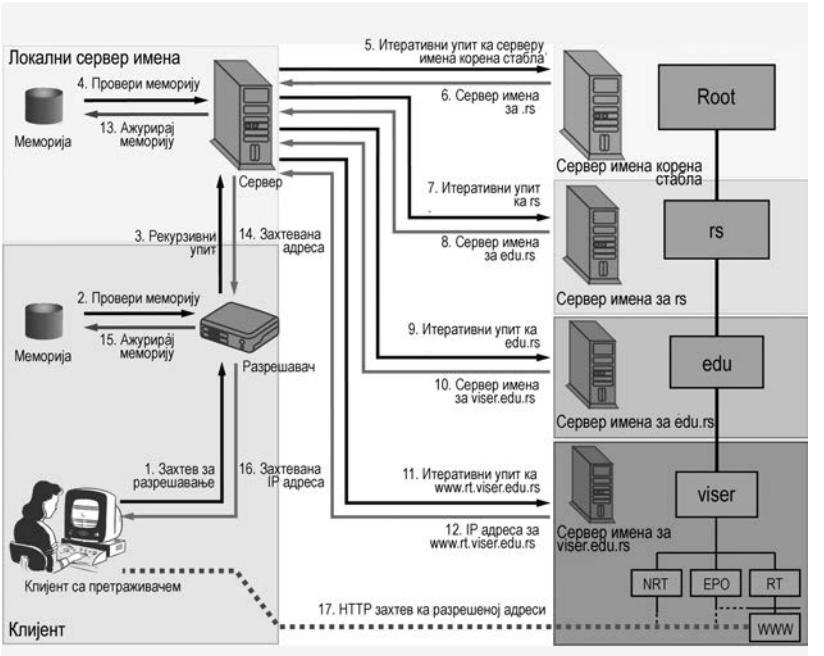
Пођимо од претпоставке да је корисник откуцао у свом претраживачу адресу „*www.rt.viser.edu.rs*“. Поједностављену процедуру за разрешавање имена чине следећи кораци, графички представљени на слици 7.9:

1. Веб претраживач препознаје DNS упит и активира софтвер за локално разрешавање имена (разрешивач) и прослеђује му име *www.rt.viser.edu.rs*;
2. Разрешивач приступа својој меморији и проверава да ли у њој постоји IP адреса која одговара том имену и одмах је прослеђује (ако постоји) веб претраживачу. Уколико је не нађе (што је претпостављено у овом примеру) проверава садржај

host датотеке. Пронађе ли у њој име и одговарајућу статичку адресу одмах ће је искористити. У противном извршава се 3. корак;

3. Разрешивач прави рекурзивни упит и шаље га ка „*dns.example.org*“ користећи адресу локалног сервера имена коју, наравно, разрешивач зна;
4. Локални DNS сервер прихвата захтев и проверава садржај своје кеш меморије. Поново претпостављамо да одговарајућу информацију није пронашао. Уколико би је пронашао, послао би поруку разрешивачу коју би означио „не-ауторитативном“[1]. Сервер такође проверава да би видео да ли он има у својој зони запис о ресурсу којим може да разреши „*www.rt.viser.edu.rs*“. Наравно да нема пошто је реч о различитим доменима;
5. „*dns.example.org*“ прави итеративни захтев за именом и шаље га ка кореном серверу имена корена стабла;
6. Сервер имена корена стабла не разрешава имена. Он шаље име и адресу сервера имена за „*rs*“ домен;
7. „*dns.example.org*“ прави итеративни захтев за именом и шаље га ка „*rs*“ серверу имена;
8. „*rs*“ сервер имена шаље име и адресу сервера имена за „*edu.rs*“ домен;
9. „*dns.example.org*“ прави итеративни захтев за именом и шаље га ка „*edu.rs*“ серверу имена;
10. „*edu.rs*“ серверу имена шаље име и адресу сервера имена за „*viser.edu.rs*“ домен;
11. „*dns.example.org*“ прави итеративни захтев за именом и шаље га ка „*viser.edu.rs*“ серверу имена;
12. „viser.edu.rs“ сервер имена је ауторитативан за *www.rt.viser.edu.rs* и шаље IP адресу те станице ка „dns.example.org“.
13. „*dns.example.org*“ смешта у кеш меморију разрешено име. Уочимо да ће он највероватније да смести у кеш меморију и друге податке, на пример оне добијене у 6, 8. и 10. кораку иако то нисмо експлицитно нагласили;

[1] *Non authoritative*



Слика 7.9 Разрешавање имена – рекурзивни упит

14. Локални сервер имена шаље разрешено име ка разрешивачу на локалном рачунару;
15. Локални разрешивач памти у својој меморији информацију;
16. Локални разрешивач прослеђује адресу претраживачу;
17. Претраживач започиње HTTP захтев IP адреси *rt* рачунара.

7.13.2 Инверзни упити

Претраживач шаље инверзни упит, захтев серверу имена са задатком да нађе (разреши) име рачунара за познату IP адресу. Ово је обрнуто од онога што је до сада анализирано. У DNS простору имена (који смо до сада анализирали) не постоји веза између имена рачунара и IP адреса. Само претраживање свих домена гарантовало би коректан одговор. Да би се ово избегло направљен је специјалан домен '*in-addr.arpa*'.

7.13.3 Памћење времена трајања

Када сервер имена обрађује рекурзивни упит то може захтевати

слање неколико упита док се не пронађе одговор. Сервер имена памти[1] у својој меморији све информације (IP адресе) добијене у том процесу за одређено време. Ово време назива се време трајања (TTL[2]). Информација о времену трајања (TTL) садржана је у повратним подацима. Када DNS сервер запамти податке, он запамти и њихово време трајања тако да мора и да започне смањивање тог времена да би знао када да податке избрише из своје меморије.

7.14 Сигурносно проширење система имена домена

Систем имена домена DNS није пројектован да буде сигуран, што је неопходна особеност за коришћење на Интернету. Напади на сигурност у раду система имена домена могу се поделити у неколико група али највећи број њих одговара добро познатим механизмима напада. Неки од напада посебно су везани за начин рада DNS протокола. Документ RFC3833 даје преглед напада и могућност да се сигурносно проширење система имена домена (DNSSec) успешно од њих одбрани.

DNSSec је пројектован да заштити DNS разрешивача (клијента) од фалсификованих DNS података који би могли да усмере људе са одређене веб локације (као што је на пример банка) на погрешну IP адресу. Фалсификоване информације могу се ставити на одговарајуће место процесом који се означава као тровање DNS меморије. Када се користи DNSSec систем сви одговори на упите су дигитално потписани. Дигитални потпис може да проверава DNS разрешивач да би видео да ли су информације идентичне са информацијама које су добијене од ауторитативног DNS сервера.

Документ RFC4367 описује како се DNS систем користи за дистрибуцију сертификата укључујући и оне који се користе за е-пошту, тако да је могуће користити DNSSec као глобалну инфраструктуру за сигурну е-пошту. DNSSec не обезбеђује поверљивост података и његови одговори су аутентификовани али не и шифровани. Он у ствари не пружа директну заштиту од одбијања услуге (DoS) али постоје неке предности које су последица аутентификације и дигиталног потписа. Друге методе

[1] *Cache*

[2] *Time To Live* - време трајања

се морају користити за пренос велике количине података као што је то са преносом података са примарног на секундарни сервер (трансфер зоне). Наравно (по RFC4367) не може да се спречи да корисници доносе погрешне претпоставке о именима домена. На пример да је за неку организацију *imeorganizacije.com* увек тражена веб локација компаније (или банке). Најмање што DNSSec може да потврди је да су подаци добијени од система имена домена заиста подаци власника домена. Најважније спецификације које у време писања овог уџбеника описују DNSSec систем су: RFC4033, RFC4034 и RFC4035.

7.15 Питања и задаци

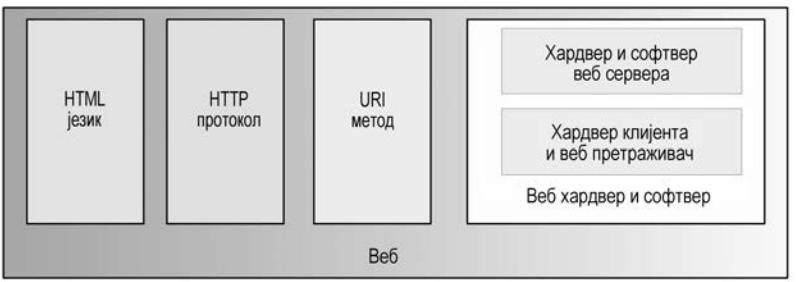
1. Које су две најчешће архитектуре које се користе у модерним мрежним апликацијама?
2. Објаснити принципе пројектовања апликација.
3. Објаснити начин рада клијент/сервер архитектуре.
4. Објаснити начин рада архитектуре равноправних рачунара (P2P архитектуре).
5. Чиме је одређена прикључница у мрежама са скупом TCP/IP протокола?
6. Како је у TCP/IP окружењу представљен систем имена домена (DNS)?
7. Шта систем имена домена обезбеђује?
8. Како апликација приступа DNS бази?
9. Шта је апликацији потребно да би успоставила транспортну везу?
10. Како се може представити простор имена домена?
11. У DNS структури постоје појмови: корен, грана и лист. Описати их користећи слику 7.4.
12. Како се јединствено идентификује чвор у оквиру стабла?
13. Колико карактера ознака чвора може највише да има?
14. Шта је делимично квалификовано име домена (PQDN)?
15. Описати сервере имена корена стабла и како се они користе.
16. Која је функција примарног сервера имена а која секундарног?
17. Какав је међусобни однос примарног и секундарног сервера имена?
18. Навести формат и описати DNS запис ресурса.
19. Која све поља садржи заглавље DNS поруке?

20. Описати концепт интернационалног имена домена?
21. Каква је ситуација у Србији са интернационалним именом домена?
22. Који се алати користе код DNS система? Описати их.
23. Анализирати садржај TCP/IP host датотеке на вашем рачунару.
24. Описати један пример итеративног упита.
25. Описати један пример рекурзивног разрешавања имена.
26. Ко је код рекурзивног начина разрешавања одговоран за слање захтева другим серверима имена уколико DNS сервер нема запис о ресурсу који клијент од њега тражи.
27. Код рекурзивног упита дефинише се време трајања (TTL). Описати његову намену.
28. Како се извршава инверзни упит?
29. Како функционише сигурносно проширење система имена домена.
30. Да ли DNSSec обезбеђује поверљивост података?

8. Веб системи

Веб (WWW[1]) систем састоји се од међусобно повезаних хипертекст[2] докумената којима се може приступити преко Интернета. Користећи веб претраживач (претраживач веб система) могу се видети веб странице које садрже текст, слике, видео записе и друге мултимедијалне садржаје и кретати се по њима користећи хиперлинкове[3]. Идеја о веб систему потекла је 1989. год. од физичара Тима Бернеса-Лиа[4] док је радио у Европском центру за нуклеарна истраживања (CERN[5]). Центар за нуклеарна истраживања (CERN) и Масачусетс институт технологије (MIT[6]) су 1994. год. потписали споразум о оснивању веб конзорцијума (W3C[7]) који је наставио рад на даљем развоју веб система, стандардизацији протокола и подршци у међусобном раду.

Функционалне компоненте веб система приказане су на слици 8.1.



Слика 8.1 Основни функционални делови веб система

[1] *World Wide Web*

[2] *Hypertext*

[3] *Hyperlinks*

[4] *Tim Berners-Lee*

[5] *Conseil Européen pour la Recherche Nucléaire* (*European Council for Nuclear Research*). Конвенцију о формирању CERN-а је 29. септембра 1954. год. потписало 11 земаља Западне Европе. Данашњи назив ове организације био би Европска лабораторија за физику честица (*European Laboratory for Particle Physics*) али је акроним CERN и даље у употреби.

[6] *Massachusetts Institute of Technology*

[7] *World Wide Web Consortium, www.w3.org*

8.1 Главне компоненте веб система

Познато је да изједначавање веб система и Интернета није потпуно тачно али указује на важну чињеницу да је веб систем уско повезан са Интернетом. Када се сам веб систем описује три компоненте се сматрају суштинским:

I HTML[1] језик: текстуални језик који описује документ са хипертекстом. Идеја је да се додају конструктори (називају се тагови[2]) у текстуални документ, којим се омогућава повезивање једног документа са другим. Такође се омогућава специјална врста форматирања и комбиновања различитих врста медија. HTML језик је постао стандардни језик за повезивање информација хипертекстом и покренуо развој бројних сличних језика.

II HTTP[3] протокол: протокол апликационог слоја TCP/IP референтног модела који користе веб системи. Он обезбеђује пренос докумената са хипертекстом (и друге врсте докумената) између клијента и сервера. Подржава пренос више докумената једном везом, памћење (кеширање[4]), преусмеравање[5] и аутентификацију.

III Јединствени идентификатори ресурса (URI[6]): метод којим се дефинишу ознаке ресурса на Интернету тако да се оне могу једноставно пронаћи или указати на њих (референцирати). URI је оригинално развијен као средство којим корисници веб система могу да лоцирају хипертекст документе тако да могу да их прегледају. URI метод није везан само за веб систем, мада се најчешће повезује са веб системом и HTTP протоколом. Јединствени локатор ресурса (URL[7]) је подскуп јединственог идентификатора ресурса (URI). Термин се често користи као да је истог значења у контексту веб система.

---

[1] *Hyper Text Markup Language*
[2] *Tags*
[3] *Hypertext Transfer Protocol*
[4] *Cashing*
[5] *Proxying*
[6] *Uniform Resource Identifier*
[7] *Uniform Resource Locator*

Уз већ поменуте три главне компоненте веб система постоје бројни елементи који су подршка раду веб система као целине. Међу њима најважније улоге играју хардвер и софтвер који обезбеђују клијент/сервер комуникацију. То су веб сервери и веб претраживачи.

Веб сервери су рачунари на којима је покренут специјалан софтвер који им омогућава да обезбеде хипертекст документа и друге датотеке клијентима који их потражују. Данас на Интернету постоје милиони таквих рачунара.

Веб претраживачи су HTTP клијентски софтвери који се покрећу на рачунару-клијенту, са TCP/IP скупом протокола и имају за циљ приступ веб документима и веб серверима.

8.2 Преглед архитектуре веб система

Посматрано са стране корисника веб систем (или само веб) састоји се од огромне збирке докумената или веб страница[1]. Свака страница може да садржи везу (линк) ка другим страницама које се налазе било где у свету. Корисници могу да следе линк који указује на нову страницу. Идеја о томе да једна страница указује на другу (хипертекст[2]) потекла је од професора[3] Масачусетс института технологије (MIT[4]) и датира из 1945. године, много пре појаве Интернета. Странице се прегледају програмом који се назива претраживач[5]. Данас се највише користе *Internet Explorer* и *Firefox*.

Веб страница се састоји од објеката. Објекат је једноставан документ као што су HTML документ, слика у JPEG[6] формату, Java аплет или видео запис - све адресирано са једним јединственим локатором ресурса (URL). Већина веб страница се састоји од основног HTML документа и неколико објеката на које се референцира. На пример веб страница која садржи HTML текст и две JPEG слике састоји се од 3 објекта.

---

[1] *Web page*

[2] *Hypertext*

[3] *Vannevar Bush*

[4] *Massachusetts Institute of Technology*

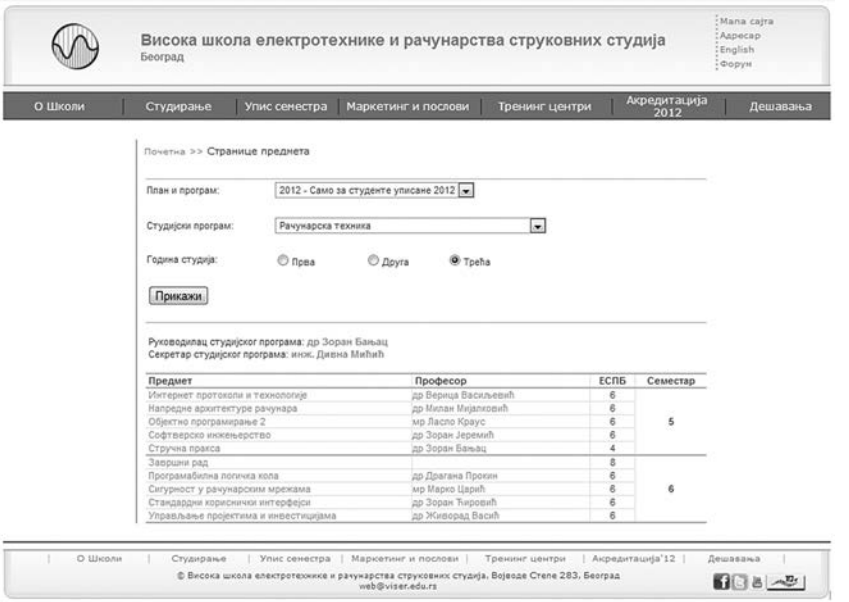
[5] *Browser*

[6] JPEG је уобичајен метод за компресију слика. Име „JPEG" потиче од имена организације (*Joint Photographic Experts Group*) која је направила стандард.

Слика 8.2 Почетна страница *www.viser.edu.rs*

Претраживач преузима захтев за траженом страницом, интерпретира текст и команде за форматирање на њој. Затим приказује страницу, исправно форматирану на екрану. Пример је дат на слици 8.2. Као и многе веб странице и ова почиње насловом, садржи неке информације, а завршава се е-адресом групе (појединца) који одржава страницу. Делови текста који су линкови ка другим страницама (хиперлинкови) често су истакнути.

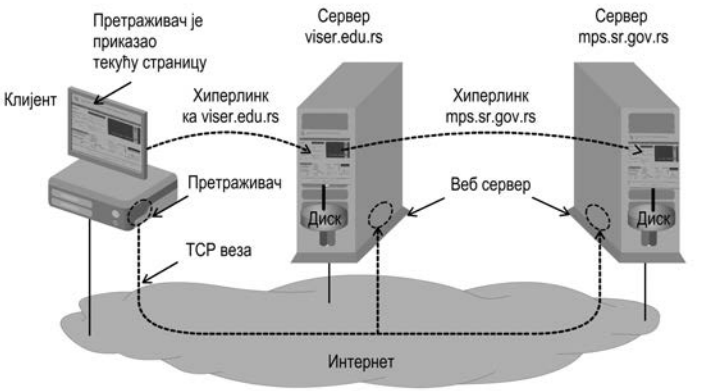
Корисници који желе да сазнају више о предметима школе могу да притисну тастер миша на обележено име. Претраживач потом преузима



Слика 8.3 Страница са предметима: *www.viser.edu.rs/predmeti.html*

страницу за коју је везано име и приказује је, као што је приказано на слици 8.3.

Основни модел како веб систем ради приказан је на слици 8.4: претраживач приказује веб страницу на клијентском рачунару.



Слика 8.4 Клијент приступа веб серверима: *viser.edu.rs* и *mps.sr.gov.rs*

Када корисник притисне тастер миша на линију текста који је повезан са страницом на *viser.edu.rs* серверу, претраживач прати хиперлинк слањем поруке серверу тражећи од њега страницу. Када је добије, страница бива приказана. Ако ова страница садржи хиперлинкове ка другој страници на другом серверу[1] (*mps.sr.gov.rs*) претраживач шаље захтев серверу за ту страницу, итд.

8.3 Активности клијента и сервера

Анализираћемо прво са мало више детаља шта се дешава на страни клијента. Као што смо већ описали претраживач је у суштини програм који може да прикаже веб страницу и прихвати притисак тастера миша на објекте приказане странице. Када је објекат одабран претраживач прати хиперлинк и прихвата одабрану страницу. Убачени хиперлинкови користе URL да би указали на друге веб странице.

Када корисник притисне тастер миша на хиперлинк, претраживач покрене серију корака да би узео захтевану страницу. Претпоставимо да корисник претражује веб и проналази линк за почетну страницу Високе школе електротехнике и рачунарства која је *http://www.viser.edu.rs/* . Када се линк одабере следеће активности се одвијају:

1. Претраживач одреди URL (на основу онога што је одабрано);
2. Претраживач тражи од система имена домена (DNS) IP адресу *www.viser.edu.rs*;
3. DNS одговара са IP адресом 172.7.1.11;
4. Претраживач отвара TCP везу преко порта 80 и IP адресе 172.7.1.11;
5. Шаље захтев тражећи датотеку */predmeti.php*;
6. Сервер *www.viser.edu.rs* шаље датотеку */predmeti.html*;
7. Затвара сеTCP веза;
8. Претраживач приказује све што се налази на страници */predmeti.html*;

Кораци које сервер одрађује су следећи:

1. Прихвата захтев клијента за успоставом TCP везе;

[1] На пример неке информације везане за правила студирања дефинисана у закону о високом образовању а публикована на веб страници Министраства просвете и спорта.

2. Налази име захтеваног документа;
3. Узима документ са диска;
4. Враћа документ клијенту;
5. Затвара (ослобађа) TCP везу.

Савремени веб сервери имају многе додатне улоге али суштински ово су активности које сервер одрађује.

8.4 HTTP протокол

У овом одељку започећемо анализом HTTP концепта и начина рада а затим ћемо се позабавити неким детаљима који су обрађени у интернет стандарду за HTTP1.1, описани у документу RFC2616. Одређени број важних термина, који су дефинисани у спецификацији HTTP стандарда, дат је у табели 8.1.

Термин	Опис
Памћење[1]	Локално смештање порука[2] које се шаљу као одговор и подсистем који управља смештањем порука, проналажењем и брисањем. У кеш меморију смештају се одговори са циљем да се смањи време потребно да се одговори на захтев[3] као и на захтев за капацитетом мреже у случају будућих, сличних захтева[4].
Клијент	Апликациони програм који успоставља везу да би послао захтев.
Оригинални сервер[5]	Сервер на коме је смештен ресурс или на коме ће он бити креиран.
Посредник[6]	Међупрограм који се понаша као сервер и клијент а са циљем да формира захтев у име других клијената. Захтев се обрађује интерно или прослеђивањем са могућим превођењем ка другим серверима. Посредник (прокси) мора да захтев интерпретира и уколико је потребно поново га пре преусмеравања формира. Прокси се обично користи као клијент/сервер портал кроз међумрежне баријере и као помоћна апликација за вођење рачуна о захтевима преко протокола које није имплементирао кориснички агент.

[1] *Cash*
[2] *Response message*
[3] *Response time*
[4] *Request*
[5] *Origin server*
[6] *Proxy*

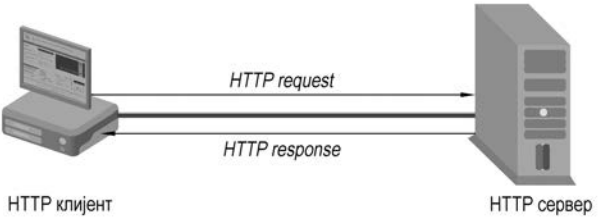
Термин	Опис
Веза[1]	Виртуелна (логичка) веза између два апликациона програма да би се обезбедила комуникација.
Целина[2]	Одређена репрезентација ресурса података или одговор од ресурса који може да се уклопи у поруке захтев или одговор. Целина се састоји од заглавља целине и тела целине.
Међумрежни пролаз[3]	Сервер који се понаша као међусервер неког сервера. За разлику од проксија међумрежни пролаз прихвата захтев као да је он оригинални сервер захтеваног ресурса; клијент који је упутио захтев можда није ни свестан да комуницира са међумрежним пролазом. Међумрежни пролази обично се користе на серверској страни портала преко међумрежних баријера и за превођење протокола када се приступа ресурсима који су смештени на не-HTTP системима.
Ресурс	Објекат[4] или услуга коју URI може да представи.
Сервер	Апликациони програм који прихвата везу са циљем да услужи захтев и пошаље натраг одговор.
Тунел	Међупрограм који ради као „слепи" преклопник[5] између две везе. Једном активиран, тунел се не сматра делом HTTP комуникације без обзира што је иницран HTTP захтевом. Тунел престаје да постоји када су оба дела везе затворена. Тунел се користи када је портал неопходан и када посредник (међупрограм) не може или не треба да интерпретира комуникацију која се преусмерава.
Кориснички агент	Клијент који иницира захтев. Ово су обично претраживачи, едитори, или други алати крајњих корисника.

Табела 8.1 Кључни термини који се односе на HTTP протокол

8.4.1 Општи принципи рада HTTP протокола

HTTP протокол почео је да се примењује као крајње једноставан протокол пројектован да одради само једну ствар: омогући клијенту да пошаље серверу једноставан захтев за хипертекст документом и добије тај документат од сервера. Савремени HTTP протоколи су суштински задржали једноставан принцип захтев/одговор али сада садрже бројне

[1] *Connection*
[2] *Entity*
[3] *Gateway*
[4] *Network data object*
[5] *Relay*



Слика 8.5 Најједноставнији случај: један HTTP клијент и један HTTP сервер

нове особености и могућности које су развијене као подршка великом порасту веб система и непрекидном увећању разноликости начина на који се он користи. Започећемо анализу HTTP протокола анализом комуникације између веб сервера и веб клијента.

У најједноставнијем случају постоји само један HTTP клијент, обично веб претраживач на рачунару клијента и један HTTP сервер (слика 8.5). Он се уобичајено означава као веб сервер. Пошто се успостави TCP веза комуникација се одвија у два корака:

1. Захтев клијента. HTTP клијент шаље захтев који је форматиран на начин како то прописује TCP стандард - HTTP *request*. У овој поруци садржан је податак о ресурсу који клијент жели да добије, или садржи информације које је потребно обезбедити серверу.
2. Одговор сервера. Сервер очитава и тумачи садржај захтева. Спроводи активности које одговарају захтеву и формира поруку HTTP *response* коју шаље натраг клијенту. Одговор указује на то да ли је захтев успешно одрађен. Такође може да садржи и садржај ресурса који је клијент захтевао.

Слика 8.6 Ланац HTTP *request*/HTTP *response*: учествује један HTTP клијент, један HTTP посредник и један HTTP сервер

8.4.2 Посредници у ланцу HTTP *request*/*response*

Најједноставнији пар захтев/одговор између клијента и сервера постаје много сложенији када се посредници[1] нађу на путу виртуелне (логичке) комуникације између клијента и сервера. Посредници као што су прокси, међумрежни пролаз и тунел користе се да би се унапредиле перформансе, обезбедила сигурност или реализовале друге функције од значаја за клијенте или сервере. Прокси се посебно често користи у веб системима пошто могу значајно да унапреде време одзива групе међусобно повезаних клијентских рачунара.

Када су посредници укључени у HTTP комуникацију онда клијенти не комуницирају директно са сервером (и обрнуто) већ сваки од њих комуницира са посредником. Тиме је створен простор у коме посредници могу да остваре функције као што су памћење у меморији, превођење, повезивање (агрегација) и укалупљивање. Посматрајмо комуникацију када је укључен један посредник (слика 8.6). Комуникација која је представљена на слици 8.5 и која је реализована у два корака (без посредника) постаје комуникација која се одвија у четири корака (са једним посредником) и то на следећи начин:

1. Захтев клијента. HTTP клијент шаље захтев (порука HTTP *request*.) ка 1. посреднику.
2. Захтев посредника. Посредник обрађује захтев, и уколико је потребно уноси измене. Затим преусмерава захтев одговарајућем серверу.
3. Одговор сервера. Сервер очитава и тумачи садржај захтева. Спроводи активности које одговарају захтеву и формира одговор (порука HTTP *response*). Пошто је сервер захтев примио од посредника његов одговор се упућује посреднику.
4. Одговор посредника. Посредник обрађује захтев, уколико је потребно уноси измене и преусмерава га клијенту.

На основу претходне анализе види се да посредник има улогу сервера са становишта клијента и клијента са тачке гледишта сервера. Многи посредници су пројектовани тако да могу да убаце различите TCP/IP протоколе представљајући се да су сервери клијентима и

[1] *Intermediares*

представљајући се да су клијенти серверима. Могуће је да се у ланцу између клијента и сервера нађе више посредника (2. посредник, 3. посредник...). Зато HTTP стандард и користи термин ланац захтева/одговора[1].

Важно је напоменути да сервер шаље одговор клијенту а да не задржава (не памти) никакве информације о клијенту. Уколико тај клијент затражи исти објекат више пута у кратком временском периоду сервер не указује клијенту да је поновио исти захтев. Уместо тога он му поново шаље објекат као да је потпуно заборавио да је то већ, веома скоро, одрадио. Пошто HTTP сервер не чува никакве информације о HTTP клијенту HTTP протокол се означава као протокол без памћења стања везе[2]. Веб системи користе клијент/сервер архитектуру. Као што је познато веб сервери су увек укључени, са статичком IP адресом, и примају захтеве од потенцијално великог броја различитих претраживача.

8.5 Врсте HTTP веза

У многим апликацијама на Интернету клијент и сервер комуницирају у дужем периоду времена. Клијенти за то време шаљу бројне захтеве и сервери на њих одговарају. У зависности од апликације пројектанти треба да се определе да ли ће захтев/одговор бити реализован преко исте[3] или различите[4] TCP везе. Када је реч о веб страници са 3 објекта (текста и две JPEG слике) постоји могућност да се та 3 објекта пренесу једном TCP везом или са три TCP везе. Пројектанти могу подешавати HTTP клијенте и сервере да раде на оба начина. Подразумевано[5] подешавање је да се користи једна TCP веза.

Да бисмо боље анализирали потребно време да се HTTP веза реализује, користићемо начин представљања који смо користили у поглављу о TCP протоколу. Треба направити процену о времену које

---

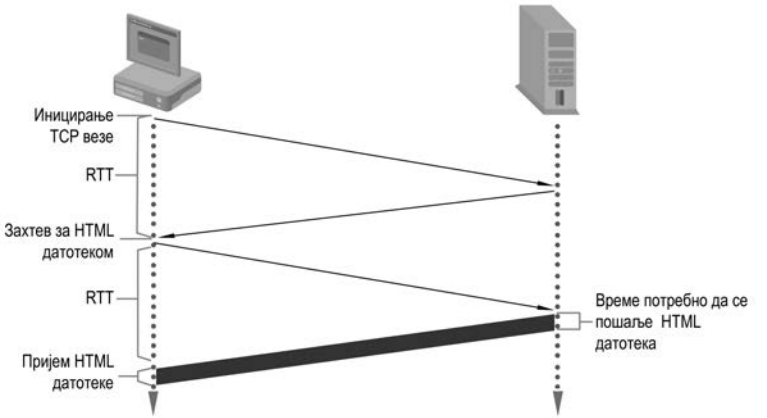
[1] *Request/Response chain*

[2] *Stateles protocol*. У поглављу о TCP протоколу (6. поглавље, одељак 6.6) детаљно су описана стања TCP веза. Основна крактеристика је да постоји запис (TCB) о вези, са свим подацима о тој вези.

[3] *Persistent connection*

[4] *Non persistent connection*

[5] *Default*



Слика 8.7 Израчунавање времена потребног да клијент упути захтев и добије HTML документат

протекне од тренутка када клијент захтева основни HTML документат до тренутка кад га и добије. Време потребно да пакет од клијента стиге до сервера и врати се натраг означава се као RTT[1] време. Шта се дешава када корисник притисне тастером миша на хиперлинк? Као што се види на слици 8.7 то доводи до тога да претраживач иницира успоставу TCP везе између претраживача и веб сервера. Успостава TCP везе захтева размену три сегмента[2]. У току размене прва два сегмента успоставе TCP везе прође RTT времена. По пријему трећег сегмента TCP везе сервер шаље HTML документат користећи успостављену TCP везу. То такође траје RTT времена. То значи да је укупно време које је потребно да клијент добије одговор од сервера једнако збиру два RTT времена и времена потребног да сервер пошаље HTML датотеку.

Коришћење више TCP веза у HTTP комуникацији има бројне недостатке. Прво, нова веза се мора успоставити за пренос сваког од објеката. За сваку од TCP веза треба резервисати меморијски простор за запис о вези (TCB[3]) и код клијента и код сервера. То може да буде

[1] *Round Trip Time*. Ово време укључује: време слања, време простирања (пропагације) од предајника до пријемника и натраг од пријемника до предајника, време чекања у редовима у свим рутерима кроз које пролази и време обраде.

[2] *Three handashake*

[3] *Transmission Control Block*, одељак 6.6 овог уџбеника

веома велики задатак за веб сервере који примају истовремено захтеве од стотине клијената. Друго, на основу слике 8.7 види се да потражња за сваким објектом траје два RTT времена.

Када се користи једна TCP веза за све објекте HTML документа сервер оставља отвореном TCP везу пошто пошаље први објекат. Остале поруке захтев/одговор могу се слати преко исте TCP везе. Могуће је и да више веб страница које се налазе на истом серверу буду послате преко те исте TCP везе. Захтев за објектима може се реализовати један за другим[1] без чекања на одговор на претходно послати захтев. То значи да се шаље 2. захтев пре него што је стигао одговор на 1. захтев, 3. захтев пре него што је стигао одговор на 2. захтев....Сервер са своје стране, такође шаље објекте један за другим. Уобичајено је да HTTP сервер затвара (окончава) везу када се она не користи одређени период времена који се може подешавати. Подразумевани начин је да HTTP протокол користи једну TCP везу са слањем захтева и даје одговоре један за другим.

На самом почетку веб странице биле су статички дефинисане и прослеђиване сваком ко је хтео да их види. Данашње веб странице се некада и тако праве али већина веб система прави своје садржаје динамички када корисник од њих то затражи. Страница која у себи садржи датум и време је пример динамичке веб странице.

Постоји проблем када се користи једна TCP веза а приступа се динамичким веб страницама. Претраживач треба да зна када је HTTP одговор са веб страницом примљен. Када се користе статичке веб странице онда се податак о величини странице налази на почетку заглавља HTTP одговора. Код динамичких веб страница сервер то не може да уради пошто на самом почетку и не зна колико ће велика страница бити.

Да би разрешио овај проблем HTTP протокол користи кодирање делова[2]. Величина дела се шаље претраживачу. Последњи део има величину 0 и може да садржи опционе информације о крају[3] странице.

---

[1] *Pipelining*
[2] *Chunked encoding*
[3] *Trailer*

8.6 Формат HTTP порука

Као што смо видели у претходним одељцима сва комуникација између рачунара који користе HTTP поротокол реализује се разменом HTTP порука. Постоје само две врсте HTTP порука: захтев и одговор. Клијенти обично шаљу захтев а сервери одговарају на тај захтев. Посредници (прокси, међумрежни пролази) могу да шаљу и захтеве и одговоре.

Свака HTTP порука садржи одговарајуће заглавље које преноси важне информације између клијента и сервера. Нека заглавља се појављују само у поруци HTTP *request*, нека само поруци HTT *response* а неке у оба типа порука.

8.6.1 Формат поруке HTTP захтев

Клијент иницира HTTP сесију отварањем TCP везе ка HTTP серверу са којим жели да комуницира. Он затим шаље серверу поруку HTTP захтев (HTTP *request*) која тачно одређује врсту акције коју клијент очекује да сервер одради. Порука HTTP захтев користи следећу структуру:

<*request line*> - линија захтева
<*general headers*> - опште заглавље
<*request headers*> - заглавље захтева
<*entity headers*> - заглавље целине
<*empty line*> - празна линија
[<*message body*>] - тело поруке
[<*message trailers*>] - крај поруке

Опште HTTP заглавље тако се назива зато што за разлику од друге три категорије није предвиђено за одређену врсту поруке или део поруке (захтев, одговор, или само тело поруке). Заглавља се у основи користе да би разменила информације о самој поруци а не о садржају који она носи. Нека од општих заглавља најчешће се користе у једној врсти порука а друга у другој врсти порука. Такође постоје нека општа заглавља која се могу појавити у захтеву или одговору али имају различито значење у сваком од њих.

Погледајмо детаљније једноставну поруку захтев (слика 8.8). Прво што се може уочити је да је написана у ASCII формату. Друго видимо да се порука састоји од десет линија иза којих је обавезно празна линија.

(Request-Line)	GET / HTTP/1.1
Connection	keep-alive
Host	google.com
User-Agent	Mozilla/5.0 (Windows; U; Windows NT 5.0; sr; rv:1.9.1.11) Gecko/20100701 Firefox/3.5.11
Accept	text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language	sr,sr-rs;q=0.8,sr-cs;q=0.6,en-us;q=0.4,en;q=0.2
Accept-Encoding	gzip,deflate
Accept-Charset	UTF-8,*
Keep-Alive	300
Cookie	PREF=ID=0caa5006f17bfeaa:U=769b78077ac25283:....

Слика 8.8 Пример поруке HTTP захтев са неким деловима заглавља

Иако ова порука има десет линија порука може имати много више линија или само једну једину линију.

Прва линија (линија захтева) садржи три поља: поље за метод, поље (опционо) за URL, и поље за верзију HTTP протокола. Највећи број HTTP захтева користи GET метод. У овом примеру претраживач користи верзију HTTP 1.1.

Друга линија (*Connection*) указује серверу да претраживач жели да користи једну TCP везу; хоће да сервер после слања захтеваног објекта задржи (*keep-alive*) TCP везу. Други начин је када претраживач не жели да користи једну TCP везу; хоће да сервер после слања захтеваног објекта оконча (*close*) TCP везу.

Заглавље захтева садржи већи број линија. Њихов распоред није унапред одређен. Анализираћемо линије које се налазе у примеру са слике 8.8:

- Линија *Host* указује на станицу (хост) на којој се тражени објекат налази. У примеру са слике 8.8 то је *google.com*. Треба нагласити да уколико је захтев упућен посреднику онда би се име хоста нашло у линији зхтева као комплетна URL ресурса: GET *http://www.example.com*.
- Линија *User Agent* указује који се кориснички агент користи а то

је претраживач који је упутио захтев серверу. У нашем примеру то је *Mozzila 5.0,* претраживач фирме *Mozilla Corporation*.

- Линија *Accept* дозвољава клијенту да укаже серверу коју врсту медијума би клијент желео да добије у одговору сервера. Заглавље може да садржи списак различитих MIME[1] типова и подтипова са којима клијент зна да ради. Свакој ставци списка може да претходи квалитативни фактор[2] *q* који указује на преференце клијента. Уколико вредност у заглављу није постављена подразумева се да ће сервер претпоставити да може да пошаље било коју врсту медијума.
- Линија *Accept-Charset* указује на то који скуп знакова би клијент волео да добије у одговору. Видимо да се за знакове (*charset*) користи UTF[3]-8 кôд[4]. Списак знакова може да користи вредности q фактора. Подразумева се да уколико је изостављен клијент може да прихвати било који скуп карактера.
- Линија *Accept-Encoding* указује коју врсту кодирања садржаја клијент хоће да прихвати. Ово се често користи да би се контролисало да ли сервер може (или не може) да тражени садржај пошаље у компримованом облику.
- Линија *Accept Language* слично намени претходних *Accept* линија садржи списак ознака које указују на ком језику клијент може да прихвати или на ком језику очекује одговор.

8.6.2 Формат поруке HTTP одговор

Анализираћемо уобичајену поруку HTTP одговор, која би могла да

---

[1] *Multipurpose Internet Mail Extensions* - Интернет стандард који омогућава да се е-поштом преносе: текстуалне поруке скупом карактера који нису ASCII карактери (не ASCII), нетекстуални прилози, тела поруке која се састоје из више делова, заглавља исписана у не ASCII карактерима. Употреба MIME стандарада се проширила и на описе других врста садржаја на Интернету које користе протоколи HTTP, SIP (*Session Initiation Protocol*), RTP (*Real-time Transport Protocol*) као и за складиштења у неким комерцијалним производима. Детаљнији опис може се наћи у 9. поглављу овог уџбеника.

[2] *Quality value*

[3] Стандард у рачунарској индустрији који садржи више од 100000 знакова. Примењен је у многим новим технологијама као што су: XML, у *Java* језику за програмирање, *Microsoft.net* платформи и модерним оперативним системима.

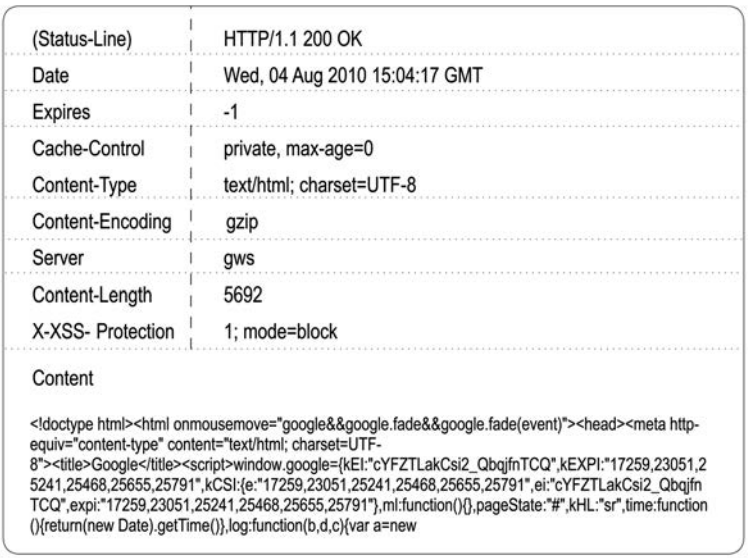
[4] 8-*Unicode Transformation Format*, кодирање карактера као и *Unicode*, усаглашен са ASCII. Доминантно карактер кодирање за датотеке, е-пошту, веб странице и софтвер који се користи за рад са текстуалним документима.

буде одговор на HTTP захтев анализиран у претходном одељку. Формат се незнатно разликује од HTTP захтева:

<*status-line*> - статусна линија
<*general-headers*> - општа заглавља
<*response-headers*> - заглавља одговора
<*entity-headers*> - заглавља целине
<*empty-line*> - празна линија
[<*message-body*>] - тело поруке
[<*message-trailers*>] - крај поруке

Линија *Status* има двоструку намену: указује клијенту која верзија HTTP протокола се користи и сумира резултате обраде захтева клијента. Резултат се поставља као статусни кôд и разлог који се уз њега шаље. Структура типичног HTTP одговора, који се шаље као одговор на HTTP захтев приказна је на слици 8.9. Заглавље одговора обезбеђује детаље статуса сумираног у првој линији одговора. Иза статусне линије следи опште заглавље.

Линија *Date* указује на датум и време када је сервер направио и послао



(Status-Line)	HTTP/1.1 200 OK
Date	Wed, 04 Aug 2010 15:04:17 GMT
Expires	-1
Cache-Control	private, max-age=0
Content-Type	text/html; charset=UTF-8
Content-Encoding	gzip
Server	gws
Content-Length	5692
X-XSS- Protection	1; mode=block

Content

<!doctype html><html onmousemove="google&&google.fade&&google.fade(event)"><head><meta http-equiv="content-type" content="text/html; charset=UTF-8"><title>Google</title><script>window.google={kEI:"cYFZTLakCsi2_QbqjfnTCQ",kEXPI:"17259,23051,25241,25468,25655,25791",kCSI:{e:"17259,23051,25241,25468,25655,25791",ei:"cYFZTLakCsi2_QbqjfnTCQ",expi:"17259,23051,25241,25468,25655,25791"},ml:function(){},pageState:"#",kHL:"sr",time:function(){return(new Date).getTime()},log:function(b,d,c){var a=new

Слика 8.9 Пример HTTP одговора са неким деловима заглавља

одговор. Треба нагласити да то није време када је објекат направљен или последњи пут промењен. То је време када сервер преузима објекат од фајл система, убацује објекат у HTTP одговор и шаље одговор.

Линија *Server* указује да је поруку направио *Appache* веб сервер; аналогна је User-agent линији у HTTP захтеву.

Линија *Content-Length* указује на број бајтова у објекту који је послат. Заглавље је битно пошто указује примаоцу на крај поруке. Треба истаћи да се ово заглавље може поставити само у случају када је дужина поруке потпуно одређена пре него што је послата. Ово није могуће уколико је садржај који се шаље динамички генерисан.

Линија *Content-Type* указује на врсту садржаја[1] тела целине који треба примаоцу да буде послат. У нашем примеру реч је о документу text/html, а као врста знакова (Charset) користи се UTF-8 кôд.

Линија *Expires* специфицира датум и време када ће целина у поруци бити сматрана застарелом[2]. Заглавље ће бити игнорисано уколико је присутно заглавље Cashe-Control које садржи директиву *max-age*.

Линија *Cache-Control* садржи директиве које управљају начином памћења (кеширања) било да је реч о захтеву (HTTP *request*) или одговору (HTTP *response*). Ове директиве утичу на све уређаје (посреднике, међумрежне пролазе) на путу од клијента до сервера, и обрнуто од сервера до клијента. Оне надвладавају сва подразумевана подешавања везана за памћење садржаја у свим уређајима у ланцу од клијента до сервера и обрнуто. Постоји велики број директива које се могу наћи у

[1] *Media type* – идентификатори за формат датотека. Означава се и као *Internet media type* и представља дводелни идентификатор за формат датотека (*file formats*) на Интернету. Идентификатори су првобитно дефинисани у документу RFC2046 за е-пошту са SMTP протоколом. Њихова употреба је проширена и на друге протоколе као што су HTTP, RTP и SIP протоколи. Ови типови су означени као MIME типови (*MIME types*), и понекад се називају садржајем типа (*Content-types*). Оригинални назив *MIME types* односи се на означавање не ASCII делове поруке е-поште састављених помоћу MIME (*Multipurpose Internet Mail Extensions*) спецификација. Без MIME типова, клијенти е-поште не би могли да интерпретирају прикачене датотека ако су оне слике или табеле. *Media type* састоји се од два или више делова: *type, subtype* и нула или више изборних параметара. На пример, подврсте текста (*text*) имају опциони *charset* параметар који може бити укључене да укаже на нпр. кодирања карактера (*character encoding*) нпр. *text/html; charset=UTF-8*. Дозвољене *charset* вредности дефинисани су листом IANA. Дефинисани су идентификатори: за апликације (*type application*) - за вишенаменске датотеке; за аудио (*type audio*); за слике (*type image*); за поруке (*type message*); за 3D моделе (*type model*); за архивирање и друге објекте направљене од више делова (*type multipart*); за читљив текст и изворишни кôд (*type text*); за видео (*type video*).

[2] *Stale* - бајат

Cache-Control директиве	HTTP	Опис
no-cache	Захтев или одговор	Када је присутан, упућује уређај да преусмерава захтеве који следе а који потражују исти садржај на продужење ваљаности. То значи да се мора проверити са сервером да ли су запамћени подаци још увек важећи.
public	Одговор	Указује да одговор може бити запамћен у било којој кеш меморији укључујући и ону дељену (меморија коју користе многи клијенти).
private	Одговор	Указује да је одговор намењен одређеном кориснику и да се не може запамтити у дељеној кеш меморији.
no-store	Захтев или одговор	Указује да целокупан захтев или одговор не треба да буде запамћен. Ово се некада користи да би се спречило памћење поверљивих докумената у кеш меморији где неауторизовани корисници могу њима да приступе. Као што указује HTTP стандард ово је веома рудиментарна мера сигурности и не треба јој превише веровати пошто је недобронамерни оператори могу једноставно игнорисати.
max-age	Захтев или одговор	Када се налази у захтеву указује да је клијент вољан да прихвати одговор који није старији од вредности која је назначена у директиви. Када се налази у одговору, указује на максималну старост одговора пре него што се он прогласи застарелим. Ово је алтернатива коришћењу *Expires* заглавља и надвладава је.

Табела 8.2 Директива *Cache-Control*

документу RFC2616. Иако су уопштеног типа, неке од директива се појављују само у захтевима а неке само у одговорима.

У примерима који су приказани на сликама 8.8 и 8.9 анализиран је само део линија заглавља које се могу користити у HTTP захтеву и HTTP одговору. У спецификацији за HTTP протокол (RFC2616) може се наћи детаљан опис и осталих линија заглавља.

Поставља се питање како претраживач зна коју линију заглавља да убаци у захтев? Како сервер зна које линије заглавља да убаци у свој одговор? Претраживач прави заглавља у складу са верзијом HTTP протокола који се користи и начина на који је корисник подесио претраживач. Слично је и са веб сервером: постоје различите верзије и подешавања која утичу на то која ће од линија бити убачена у поруку HTTP одговор.

Поред типова медијума HTTP протокол је „позајмио“ од MIME спецификације обележавања начина кодирања и начин на који се у заглављу указује на дужину целине. Иако је HTTP протокол сличан MIME спецификацији он није идентичан. Може се рећи да је HTTP 1.1 поделио MIME обележавање кодирања у два дела:

Кодирања садржаја[1] која се примењују на целину која се преноси HTTP поруком сматрају се кодирањем с краја на крај[2] пошто се кодира само једном пре него што се пошаље и декодира једном пошто стигне до пријемника (сервера или клијента). Када је примењен овакав начин кодирања метод се означава у линији *Content Encoding* заглавља целине. Клијент може такође да означи које кодирање садржаја он може да примени користећи заглавље (*Accept Encoding*).

Намена кодирање у преносу[3] је да се обезбеди сигуран пренос између уређаја. Примењује се на целу HTTP поруку. Ова врста кодирања мења [4]се пошто уређаји на путу од изворишта до одредишта користе различите врсте кодирања. Метод (уколико постоји) означава се у општем заглављу *Transfer-Encoding*.

За сада се *Content Encoding* користи једино за компресију. Документ RFC2616 дефинише три различита алгоритма за кодовање:

- *gzip* - компресија коју користи UNIX *gzip* програм, као што је описано у документу RFC1952;
- *compress* - компресија коју користи UNIX *compress* програм и
- *deflate* - метод дефинисан у документима RFC1950 и RFC1951.

---

[1] *Content Eencoding*
[2] *End to end*
[3] *Transfer Encoding*
[4] *Hop by hop*

8.7 Методи HTT протокола

HTTP протокол је пројектован за веб системе и општији је него што је то било потребно имајући у виду будуће објектно-оријентисане апликације. Због тога постоји активност која се назива метод уместо само захтев за веб страницом. Сваки захтев састоји се од једне или више линија ASCII текста, у коме реч у првој линији је име захтеваног метода. Уграђени методи дати су у табели 8.3.

Метод GET захтева од сервера да пошаље страницу (односно објекат у најопштијем случају, али у пракси обично само неку датотеку). Страница је MIME кодирана. Велика већина захтева веб серверима реализује се GET методом. Уобичајени облик GET метода је GET *filename* HTTP/1.1 где *filename* именује (одређује) ресурс (датотеку) коју треба преузети а 1.1 је верзија протокола која се користи.

Метод HEAD тражи само заглавље поруке, без захтева за стварном страницом. Овај метод може се користити да би се добио податак о томе када су вршене последње измене на страници, време за прикупљање података потребних за индексирање или само за проверу исправности URL адресе.

Метод PUT је обрнут од GET метода: уместо учитавања података са сервера, уписују се подаци на сервер. Овај метод омогућава изградњу скупа веб страница на удаљеном серверу. Тело HTTP захтева садржи саму страницу означену URL адресом. Може бити кодирано, а може се

Метод	Опис
GET	Захтев за очитавањем веб странице
HEAD	Захтев за очитавањем заглавља веб странце
PUT	Захтев за складиштењем веб странице
POST	Додавање именованом ресурсу (веб страници)
DELETE	Уклањање веб странице
TRACE	Слање еха долазећем захтеву
CONNECT	Користиће се у будућим применама
OPTIONS	Упит одређене опције

Табела 8.3 Уграђени методи HTTP протокола

захтевати провера идентитета (аутентификацију) да се докаже да онај ко шаље захтев[1] заиста има дозволу за уписивање података на сервер. У том случају се најчешће користи FTP протокол. Метод PUT се не користи често или његова употреба није дозвољена.

Метод POST сличан је PUT методу. Уместо замене постојећих података новим, нови подаци се додају на постојеће податке. На пример: слање поруке ка групним порукама[2] или додавање датотека у систем огласне табле.

Метод DELETE чини оно што би се могло очекивати: уклања ресурс. Као и код PUT метода, провера идентитета и одобрења играју главну улогу. Не постоји гаранција да ће DELETE метод успети, јер чак и ако је даљински HTTP сервер спреман за брисање странице датотеке има начин да забрани HTTP серверу да је мења или уклања. Не користи се често из истог разлога као ни PUT

Метод TRACE се користи за проналажење грешака. Он упућује сервер да пошаље назад захтев. Метод је користан када захтеви нису обрађени правилно и клијент жели да зна које захтеве је сервер заправо добио.

Метод OPTIONS обезбеђује начин на који ће клијент слати упит серверу о његовим својствима или оне о одређеној датотеци.

Сваки захтев на који се добије одговор састоји се од линија статуса, и могуће додатне информације (нпр., веб странице: цела или њен део). Статусна линија садржи троцифрени статусни кôд који указује на то да ли је захтев био задовољен, а ако није, зашто није. Прва цифра се користи да би поделила одговор у пет главних група (као што је приказано у табели 8.4):

- кодови 1xx ретко се користи у пракси,
- кодови 2xx значе да је захтев био успешно одрађен и да је садржај (ако постоји) враћен,
- кодови 3xx указују клијенту да потражи страницу на неком другом месту: помоћу неке друге URL адресе или у својој кеш меморији (о чему ће више речи бити касније),

[1] *Caller*

[2] *Newsgroup*

Кôд	Значење	Примери
1xx	Информација	100 - сервер прихвата да води рачуна о захтевима клијента
2xx	Успешност	200 - захтев је успешно реализован 204 - не постоји тај садржај
3xx	Преусмеравање	301 - страница је пресељена 304 - меморисана (кеширана) страница је и даље важећа
4xx	Грешка клијента	403 - забрањена страница 404 - страница није пронађена
5xx	Грешка сервера	500 - интерна грешка на серверу 503 - пробај поново касније

Табела 8.4 Кодови статусних линија

- кодови 4xx значе да захтев није успео због грешке на страни клијента (нпр. захтев за непостојећом страницом),
- кôд грешке 5xx значи да сам сервер има проблем, било због грешке у својим кодовима било због привременог преоптерећења.

8.8 Механизми за управљање стањима[1] HTTP протокола

Напоменули смо да HTTP сервер не памти стања[2] везе. Овакав поједностављен рад сервера омогућио је да данашњи сервери могу да воде рачуна о стотинама истовремених TCP веза. Међутим, понекад се јавља потреба да веб сервер зна идентитет корисника: било да би ограничио његов приступ било да користи садржаје у функцији идентитета корисника.

Документ RFC6265 описује начин како да се направи веза у којој се памте стања везе код размене HTTP захтева и одговора. Уведена су три нова заглавља *Cookie*, *Cookie2*, *Set-Cookie2*. Она преносе информацију о стањима везе између учесника: сервера и корисничког агента. До сада, HTTP сервери су одговарали на захтеве клијента без повезивања тог захтева са претходним или будућим захтевима клијента.

[1] *State Management Mechanism*
[2] *Stateless*

Cookie[1] је жетон, мали пакет података који се размењује између комуникационих програма где подаци нису од значаја пријемном програму. Садржај је невидљив и обично се и не анализира све док прималац не проследи касније жетон натраг пошиљаоцу. Жетон се користи као карта - да идентификује одређени догађај или трансакцију. Термин је изведен од магичног жетона[2] добро познати концепт из UNIX оперативних система који је био узор и за концепт и за назив. Пошто су само део текста, жетони се не могу извршавати.

Механизам са управљањем стањима дозвољава клијентима и серверима да размењују информације које постављају HTTP захтеве и одговоре у шири контекст који се означава као сесија. Овај контекст може се искористити да се направи на пример „карта куповине[3]" у коме се оно што је купац одабрао може повезати у једну целину пре саме поруџбине. Други пример био би систем за прегледање магазина у коме претходно читање корисника утиче на то шта ће му бити касније понуђено.

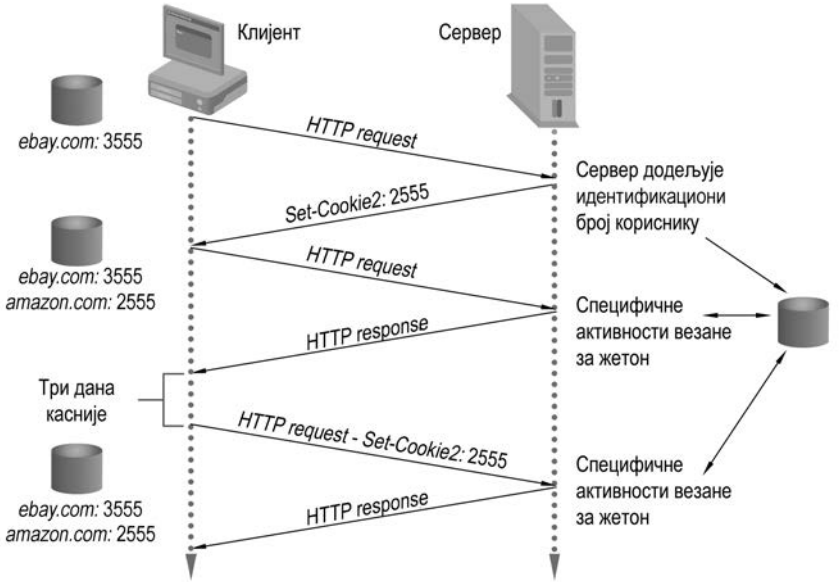
Не захтева се ни од клијента ни од сервера да користе жетоне. Сервер може да не пошаље клијенту садржаје који је тражио ако му клијент не врати жетон који му је он послао.

Узећемо као пример корисника који купује преко веб сервера *ebay.com* и *amazon.com* (слика 8.10). Када захтев (HTTP *request*) стигне до веб сервера *amazon.com* за њега се креира јединствен идентификациони број и формира се запис о захтеву у бази података веб сервера *amazon.com*. Веб сервер *amazon.com* шаље одговор (HTTP *response*) кориснику чије заглавље садржи линију *Set-Cookie2* са идентификационим бројем (на пример 2555). Када претраживач корисника добије одговор сервера он види да је заглављу постављена линија *Set-Cookie2*. Претраживач памти вредност 2555 у посебној жетон датотеци заједно са именом сервера од кога је жетон добио. Датотека у себи већ садржи запис о жетону са веб сервера *ebay.com* (3555) коме је корисник раније приступао. Корисник наставља да претражује *amazon.com* сајт, захтевајући веб страницу. Претраживач корисника ће сваки пут

[1] Име *cookie* потиче од поређења са неотвореним колачем који доноси срећу (*fortune cookie*) због скривене поруке у њему.

[2] *Magic cookie*

[3] *Shopping cart*



Слика 8.10 Механизми за управљање стањима HTTP протокола

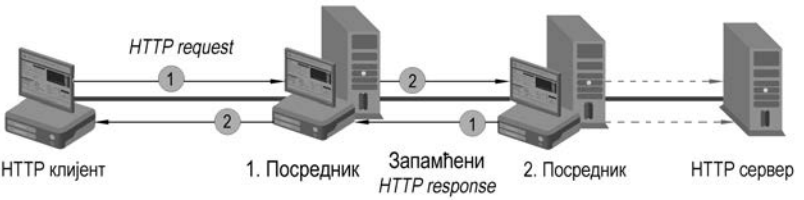
у заглавље HTTP захтева поставити одговарајући идентификациони број (за *amazon.com* је 2555). На тај начин је *amazon.com* сервер у стању да прати активност корисника и формира картон куповине који ће корисник моћи на крају сесије одједном да плати. Ако би корисник приступио сајту *amazon.com* месец дана касније његов претраживач поново ће поставити линију *Cookie2* (2555) у заглављу свог HTTP захтева.

Иако је коришћење жетона корисно у куповинама на Интернету, неки корисници могу да сматрају да је то напад на њихову приватност. Такође, постоји могућност злоупотребе.

8.9 Унапређење перформанси веб сервера

Популарност веб система изузетно је велика. Сервери, рутери, линије често су преоптерећени[1]. Као последица великог кашњења када се приступа веб серверима пројектанти су предложили различите начине за побољшање перформанси.

[1] Среће се и назив за WWW*: World Wide Wait.*



Слика 8.11 Утицај памћења (кеширања) у ланцу HTTP захтева и HTTP одговора

Једноставан начин за побољшање перформанси је памћење објеката у случају да се они поново користе. Овај начин нарочито је ефикасан код веб сервера који се често посећују као што су *www.google.com* или *www.wikipedia.org*. Памћење објеката који се касније користе означава се као кеширање[1].

Предности памћења објеката код HTTP протокола су вишеструке. Прво смањује се заузимање преносног капацитета елиминисањем непотребних захтева и одговора. Друго, једнако важно, краће је време одзива за корисника који учитава објекат. Подсетимо се да многе веб странице данас садрже слике које су значајно веће од самих HTML страница које их референцирају. Памћење ових датотека обезбеђује да се та страница учитава много брже. На слици 8.11 приказано је како се памћењем објеката смањује заузетост преносног капацитета скраћивањем ланца између HTTP захтева и HTTP одговора.

HTTP памћење објеката може се применити на различитим местима у ланцу HTTP захтева и HTTP одговора. Уколико је место где се објекат памти ближе извору HTTP захтева бржи је одзив него када се објекат узима са извориште. Са друге стране што је место на коме се памте објекти даље од оног који објекат захтева (и ближи извориште) већи број уређаја може да га користи. Постоје три класе уређаја у којима се могу памтити објекти тј. у којима се кеш може наћи: памћење објеката код веб клијената, памћење објеката код посредника, памћење објеката на веб серверу.

8.9.1 Памћење објеката код веб клијената

Памћење објеката код веб клијената (локалног клијента) познато је већини корисника Интернета. Обично је уграђено у софтвер веб

[1] *Caching*

претраживача и због тога се назива памћење (кеш) веб претраживача. У локално резервисаној меморији (кешу) памте се документи и датотеке којима је корисник скоро приступао тако да су оне њему брзо доступне уколико их корисник поново захтева. Означава се и као приватни кеш.

8.9.2 Памћење објеката код посредника

Уређаји као што су посредници (прокси сервери[1]) који се налазе између веб клијената и сервера имају такође могућност да памте објекте. Уколико корисник жели неки објекат који није у његовој локалној (кеш) меморији посредник му га може понудити (слика 8.11). Овај начин није ефикасан као када се објекат добавља из локалне меморије али је значајно бољи од приступања удаљеном веб серверу. Када се користи памћење код посредника велики број уређаја може да приступа меморији посредника. Због тога се користи и назив јавна[2] или дељива[3] кеш меморија.

8.9.3 Памћење објеката на веб серверу

Веб сервери такође могу да користе меморију за памћење објеката (кеш меморију). На први поглед изгледа чудно да сервер има меморију за своје сопствене објекте, али постоје ситуације када је то веома корисно. На пример клијенти захтевају објекат који се добија као резултат веома сложених упита из базе података. Уколико такав објекат тражи већи број клијената постоји значајна предност у његовом памћењу.

Пошто је веб сервер највише удаљен од корисника у овом приступу најмање су уштеде: захтев клијента и одговор сервера треба да прођу кроз целу мрежу. Удаљеност клијента значи да сви корисници могу да приступају меморији.

Контрола памћења на страни сервера реализује се на исти начин као друге врсте контроле које су имплементиране код HTTP протокола: употребом специјалних заглавља. Најважнија је *Cache Controle* која се налази у општем заглављу. Она има бројне директиве које омогућавају да се управља меморијом. Важни делови заглавља су и *Expires* и *Vary*[4].

Када корисник добије објекат директно од сервера, сигурно је добио

---

[1] *Proxy server*

[2] *Public*

[3] *Shared*

[4] Додатне информације могу се наћи у 13. одељку документу RFC2616.

текућу (најновију) верзију објекта. Међутим када се користи памћење у кеш меморији то не мора да значи. Ресурси се не мењају често, али ипак после одређеног времена дође до промене на свим ресурсима. Због тога објекти у HTTP кеш меморији не могу да се памте неограничено. Што је дуже објекат запамћен то је већа вероватноћа да је застарео. Зато се велики број функција HTTP протокола, а везаних за рад са кеш меморијом, односи на старење[1] кеш меморије.

8.10 HTTP сервер посредник

Анализирајући општи модел рада HTTP протокола користили смо модел представљен на слици 8.6: у комуникационом ланцу између клијента и сервера могу се наћи један или више посредника. Најважнији тип посредника је HTTP сервер посредник. Сервер посредник ради и као клијент и као сервер. Прихвата захтев од клијента као да је он сам сервер и прослеђује га (можда и мења) ка стварном серверу за кога је HTTP сервер посредник клијент. Сервер враћа одговор HTTP серверу посреднику, који га преусмерава ка клијенту. HTTP сервери посредници могу да буду:

- транспарентни[2] - не мењају HTTP захтев и HTTP одговор, или
- нетранспарентни[3] - мењају са одређеним циљем HTTP захтев и HTTP одговор.

Пошто HTTP сервер - посредник има способност да обради захтев клијента и одговор сервера може се користити да унапреди много важних особености комуникација као што су:

- сигурност - проверу одлазећег захтева и долазећег одговора HTTP сервер посредник може да обави. Пример је спречавање приступа (филтрирање) неподобним садржајима или као рампа за одговоре који могу да оштете рад клијента (програми са скривеним вирусима);
- памћење у меморији - може да се постави дељена меморија у посреднику па да ресурс који је захтевао један корисник може да буде доступан и другим корисницима;

[1] *Aging*
[2] *Transparent*
[3] *Non transparent*

- перформансе - у неким случајевима постојање HTTP сервера посредника може значајно да побољша перформансе система нарочито у смањивању кашњења.

8.11 Јединствено адресирање ресурса

Означавање хиперлинковима има неке важне импликације на то како су веб документи и ресурси адресирани. Веб системи су у суштини протоколи за пренос порука слично FTP протоколу. Потреба да се дефинишу хиперлинкови значила је да је потребно заменити традиционалан FTP модел заснован на употреби скупа команди системом у коме су ресурси јединствено представљени једноставним компактним низом знакова. Резултат је дефиниција једног од три битна елемента веб система: јединствен идентификатор ресурса (URI[1] идентификатор). Развијен је са наменом да веб ресурси буду једноставно и конзистентно идентификовани. Јединствен идентификатор ресурса[2] (URI идентификатор) подељен је у две категорије: јединствену адресу (локацију) ресурса (URL[3] адреса) и јединствено име ресурса (URN[4] име). Данас се они користе и за друге протоколе и апликације.

URL адреса одређује ресурс по месту на коме се налази што може да буде проблем а разлога је више. Први је у томе што ресурс (нпр. бесплатан кориснички програм) може да се налази на више веб сервера. Уколико се ресурс уклони са URL адресе која је постављена у претраживачу добија се одговор који указује на погрешан резултат претраживања: HTTP 404 – NOT FOUND. То значи да када се веб страница премешта и мења име или IP адресу (или и једно и друго) оставља за собом стотине страница са утиснутим застарелим (неважећим) линковима.

Као што је и за очекивати јединствено име ресурса (URN име) означава ресурс по имену а не по месту (локацији) на којој се налази. Јединствено име ресурса (URN име) мора да буде препознато од стране неке званичне организације и мора да буде тачно дефинисано неким формалним језиком. Рад на томе до данас се споро одвијао.

---

[1] *Uniform Resource Identifier*

[2] Детаљније се може наћи у документу RFC3986.

[3] *Uniform Resource Locator*

[4] *Uniform Resource Names*

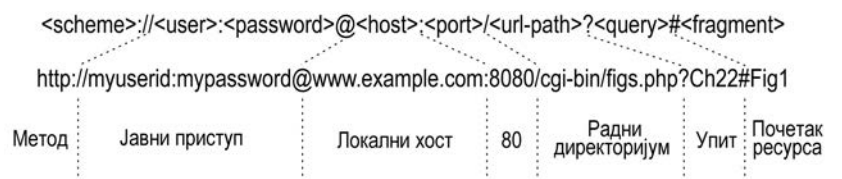
Врста идентификатора која се за сада користи у веб системима је јединствена адреса ресурса (URL адреса). Јединствена имена ресурса (URN имена) још увек се развијају. Јединствену адресу ресурса у оквиру веб система (веб URL адреса) специфицира употреба HTTP протокола за преузимање ресурса и обично се означава као HTTP URL адреса. Она одређује како се HTTP протокол користи да би се ресурс пузео и обезбеђује информације о томе где и како се може пронаћи и преузети (име хоста, путању директоријума име датотеке у којој је смештен).

8.11.1 URL адреса

URL адреса састоји се од два дела: метода који се користи да би се приступило ресурсу и адресе (локације) ресурса. Заједно ова два дела URL адресе обезбеђују начин на који корисник приступа датотекама, објектима, програмима, аудио и видео садржајима веб система.

Метод се означава преко шеме[1] и обично указује на TCP/IP апликациони протокол као што су *http* или *ftp*. Шеме могу да укључе и знакове као што су: плус („+”), тачка („.”) или црта („-”), али обично садрже само слова. Методе не праве разлику између великих и малих слова па је HTTP исто што и *http* (усвојено је да су мала слова). Део URL адресе који одређује место (локацију) налази се иза шеме а одвојен је знаком навођења и две косе црте („://”). Формат локације зависи од врсте шеме. У пракси URL адреса користи TCP/IP и интернет конвенције. Многе шеме користе заједничку синтаксу. На пример *http* и *ftp* шеме користе DNS имена или IP адресе за идентификацију циљне станице и очекују да пронађу ресурс у хијерархијској структури датотека. Генерални израз за http метод дат је на слици 8.12:

Поља имају следећа значења:



Слика 8.12 Генерални израз за *http* метод

[1] *Scheme*

- <*scheme*> - метод који се користи да би се приступило ресурсу. Подразумевани метод за веб претраживач је *http*.
- <*user*> и <*password*> - URL ауторизација састоји се од идентификације (ID) и лозинке (*password*) одвојено са (:). Многи приватни веб сајтови захтевају ауторизацију и уколико се не постави на URL од корисника се тражи ова информација. Када се не постави подразумева се јавни приступ ресурсима.
- <*host*> - код URL адресе специфицира се преко DNS имена или IP адресе.
- <*port*> - представља TCP или UDP порт који заједно са информацијом о станици одређује прикључницу (сокет) на којој се метод који одговара шеми може наћи. За *http* подразумевани порт је 80 али се и 8080 често користи. Обично се 80 (као подразумевани) изостављa.
- <*url-path*> - указује на одређени ресурс који треба допремити користећи HTTP протокол. Обично је то путања директоријума почевши од подразумеваног директоријума где се ресурс може наћи. Уколико је ово поље изостављено веб сајт има подразумевани директоријум на који се корисник поставља. Коса црта која му претходи није технички део путање али се користи као разграничавач и мора да следи порт. Уколико се <*url-path*> завршава са још једном косом цртом указује на директоријум а не на датотеку (већина веб сајтова сами проналазе да ли се путања завршава на датотеци или директоријуму). Двострука тачка (..) помера корисника навише за један ниво од подразумеваног директоријума. Путања зависи од тога да ли је написана великим или малим словима.
- <*params*> - ови параметри контролишу како се метод користи на ресурсима и специфични су за одређену шему. Сваки параметар има облик <*parameter*>=<*value*> и одвојени су са (;). Уколико не постоје параметри подразумевана активност везана за ресурс се реализује.
- <*query*> - ово URL поље садржи информације које користи сервер да би формирао одговор. Обично се користи кôд за реализацију интерактивних функција пошто вредност упита може

да специфицира корисник а затим да се преко веб претраживача проследи веб серверу. Док параметри зависе од шеме упит зависи од ресурса. Исти резултат може се добити HTTP POST методом.

- <*fragment*> - ово URL поље се користи да би указало тачно за који део ресурса је корисник заинтересован. Подразумева се да се кориснику представља почетак стварног ресурса.

8.12 Питања и задаци

1. Објаснити принципе пројектовања апликација.
2. Који су основни функционални делови веб система представљеног на слици 8.1?
3. Укратко описати три главне компоненте веб система: HTML језик, HTTP протокол и јединствени идентификатор ресурса (URI).
4. Описати кораке који се одвијају приликом претраживања веба и приступа почетној страници неког линка (нпр. *http://www.viser.edu.rs/*).
5. Набројати и описати кључне термине који се односе на HTTP протокол.
6. Каква је функција посредника у ланцу HTTP request/response?
7. Израчунати време за које клијент може да добије HTML документ.
8. Које врсте HTTP веза постоје? Објаснити.
9. Описати структуру HTTP захтева.
10. Анализирати пример поруке HTTP захтев приказан на слици 8.8.
11. Анализирати пример поруке HTTP одговор приказан на слици 8.9.
12. Набројати и описати неки од уграђених метода HTTP протокола.
13. Описати функцију и начин обележавања статусних кодова.
14. Описати механизме за управљање стањима HTTP протокола.
15. Описати пример представљен на слици 8.10.
16. Који се механизам користи за побољшање перформанси веб система?
17. Објаснити разлику између транспарентног и нетранспарентног HTTP сервера посредника.
18. Који је допринос HTTP сервер – посредник у комуникацији?
19. Каква је намена јединственог идентификатора ресурса?
20. Каква је разлика између адресе локације ресурса (URL адресе) и јединственог имена ресурса (URN имена)?

9. Електронска пошта

Напредак система за пренос података довео је до повећања комуникационих могућности и стварања модерног поштанског система. Данас су ови системи напредовали до тачке у којој свако у развијеном делу света може послати писмо било коме у другом делу света. Једна од природних примена рачунарских мрежа је да се искористе као замена за физички транспорт порука с једног места на друго.

Први системи електронске поште (е-поште) развијени су у самом почетку на великим рачунарима[1]. Корисник је систему приступао преко терминала, имао је једноставно поштанско сандуче на главном рачунару а испоручивање поште сводило се на пребацивање порука из једног поштанског сандучета у друго. Ови системи били су погодни само за локалну комуникацију али не и за слање порука на даљину.

Електронска пошта у скупу TCP/IP протокола није остварена као јединствен протокол или технологија, она је комплетан систем који обухвата већи број компоненти које заједно раде. У овај систем укључени су стандарди који дефинишу методе адресирања и форматирања поште и више протокола који имају различите улоге неопходне за рад система е-поште.

Систем е-поште који се данас користи датира из времена ARPA-NET мреже. Већи део инфраструктуре Интернета за е-пошту обрађен је у документима: RFC5321 који описује протокол SMTP[2] и RFC5322 који дефинише формат порука на Интернету[3].

Истовремено су се развијали и други системи е-поште, али су једноставност и свестраност система заснованих на SMTP протоколу омогућиле да они постану доминантан облик и *de facto* стандард за комуникацију е-поштом на Интернету.

---

[1] *Mainfraime computers*

[2] *Simple Mail Transfer Protocol*

[3] Документи донети 2008. године, заменили су документе RFC2821 и RFC2822.

Електронска пошта је на самом почетку пројектована тако да обезбеди ефикасан пренос текста. Сада се могу пренети датотеке било ког типа: слике, текстуални документи и многе друге датотеке.

Један од битних концепата модерне електронске поште је прављење разлике између протокола за пренос порука између крајњих станица (хостова) и протокола којима корисник преузима поруке са свог локалног сервера. Раздвајање је намерно урађено јер је на тај начин омогућено слање порука корисницима и када они нису повезани[1] на Интернет. Тако је могуће остварити тзв. одложену комуникацију: пошиљалац може послати поруку кад год пожели а прималац је примити када њему то одговара.

9.1 Архитектура система електронске поште

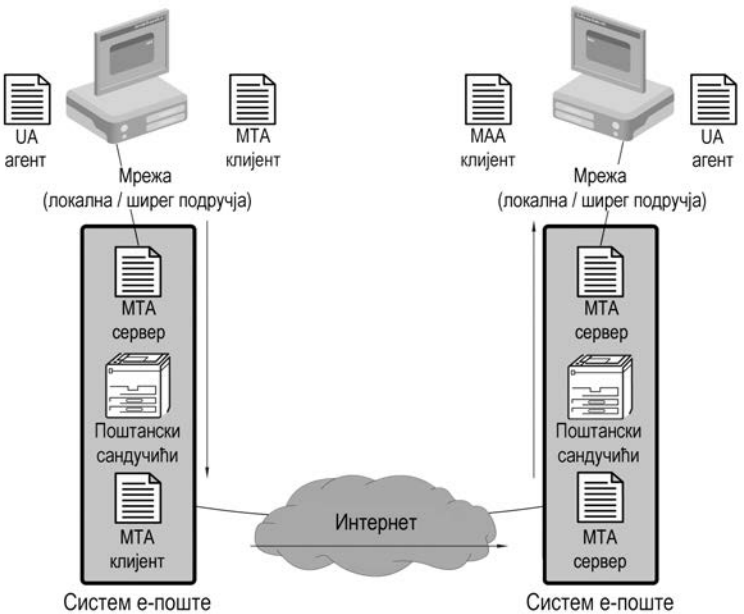
Архитектура система електронске поште развијена је кроз три различите фазе:

- Јединствени дељени систем[2]. Овај систем може да буде велики рачунар или радна станици којима корисници могу да приступе. Администратор прави поштанско сандуче[3] у које се складиште примљене поруке. Посебан програм - кориснички агент (UA[4] агент) формира поруке и смешта их у корисниково поштанско сандуче;
- Дељени системи повезани Интернетом[5]. Полази се од претпоставке да постоји могућност да корисници не деле (не приступају) исти локални систем. Због тога је додат агент за пренос порука (MTA[6] агент). UA агент и даље води рачуна о поштанским сандучићима и порукама у локалу а MTA агент се бави комуникацијом између два система на уобичајени сервер/клијент начин;
- Клијенти и сервери електронске поште повезани Интернетом[7].

[1] *On-line*
[2] *Single shared system*
[3] *Mailbox*
[4] *User Agent*
[5] *Shared systems connected by the Internet*
[6] *Message Transfer Agent*
[7] *E-mail clients and servers connected by the Internet*

Коначан корак у развоју архитектуре система електронске поште учињен је када је узето у обзир да је већина корисника повезана са својим серверима е-поште преко локалне рачунарске мреже или мреже ширег географског подручја. Пошто прималац не мора увек да буде присутан корисницима је потребан агент за приступ порукама (MAA[1] агент) да би преузео поруке са локалног сервера електронске поште. Архитектура последњег сценарија када два корисника 1 и 2 размењују поруке представљена је на слици 9.1. Приказан је ток порука од корисника 1 ка кориснику 2. Када корисник 2 шаље одговор кориснику 1 улога клијента и сервера (као и MTA и MAA) је обрнута.

Архитектура на слици 9.1 приказује два система посвећена управљању поштанским сандучићима електронске поште корисника. Поставља се питање како систем е-поште пошиљаоца зна који уређај



Слика 9.1 Електронска пошта преко Интернета
(представљене су улоге сервера и клијента)

[1] *Message Access Agent*

има улогу примаоца система е-поште? Данас посебни DNS записи[1] обезбеђују пошиљаоцу потребне информације.

9.2 Функције и компоненте система е-поште

Електронска пошта је комплетан систем који садржи одређени број елемената који обрађују различите делове задатка који треба реализовати да би се остварила комуникација е-поштом. Ти елементи треба да обезбеде: стандардни формат поруке, специфичну синтаксу за адресирање примаоца, протоколе за испоруку и преузимање поште.

Уобичајено је да се састоје од два подсистема: корисничких агената који корисницима омогућују читање и слање е-поште и агената за пренос порука који садрже е-пошту од изворишта ка одредишту.

Систем е-поште обезбеђује пет основних функција:

- Састављање поруке[2]. Да би направио поруку корисник приступа UA агенту користећи одговарајући интерфејс (нпр. графички - GUI[3]). Електонска порука се састоји од два дела: заглавља и тела поруке. Заглавље садржи серију поља која описују поруку и која управљају начином како се порука обрађује и испоручује. Тело поруке садржи стварне информације које се шаљу примаоцу. UA агент води рачуна да је порука у правилном, стандардном формату.
- Подношење поруке[4]. Након састављања поруке корисник одлучује када ће порука бити послата и испоручена. Ово омогућује кориснику да буде одвојен од мреже док саставља поруку.
- Испорука поруке[5]. Када је порука спремна за слање њу прихвата локални SMTP систем корисника, лоцира одговарајући систем примаоца и шаље поруку.
- Пријем поруке и обрада[6]. SMTP сервер примаоца прихвата и обрађује поруку. Уколико је све у реду смешта је у одговарајуће

[1] *DNS records*
[2] *Mail composition*
[3] *Graphical User Interface email user agent*
[4] *Mail submission*
[5] *Mail delivery*
[6] *Mail receipt and processing*

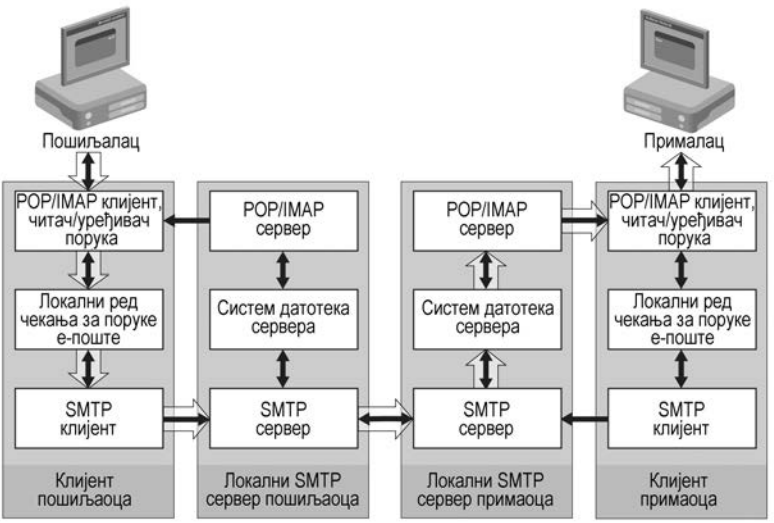
поштанско сандуче одакле ће прималац моћи да је преузме.

- Приступ пошти и преузимање порука[1]. Прималац повремено проверава свој локални SMTP сервер да види да ли има нових порука. Најчешће коришћени протоколи за преузимање поште су POP3[2] протокол и IMAP[3] протокол.

Четири целине су укључене у реализацију гореописаних корака. То су: клијент пошиљаоца, локални SMTP сервер пошиљаоца, локални SMTP сервер примаоца и клијент примаоца. Њихове међусобне релације и протоколи које користи е-пошта приказани су на слици 9.2. Треба имати на уму симетричну природу компонената тако да је могућа двосмерна комуникација.

9.2.1 Поштанско сандуче за е- пошту

Поштанско сандуче за електронску пошту[4] еквивалент је класичном поштанском сандучету осим што је намењено електронској пошти:



Слика 9.2 Пренос поруке у систему е-поште

[1] *Mail access and retrieval*
[2] *Post Office Protocol version 3*
[3] *Internet Message Access Protocol*
[4] *Mailbox*

то је место где се примљене поруке испоручују. Поштанско сандуче је одређено адресом е-поште. За разлику од интернет протокола намењених размени порука, формат у коме би се чувале поруке никада није формално дефинисан RFC документима. Развој механизама и стандарда за поштанске сандучиће препуштен је произвођачима софтвера који развијају клијенте за е-пошту.

9.3 Адресирање у систему електронске поште

Сва комуникација у мрежама захтева специфицирање идентитета корисника коме је комуникација упућена. Адреса е-поште идентификује одређеног корисника, односно његово поштанско сандуче на неком SMTP серверу. Корисник може да преузме своју пошту са више различитих локација, потребно је само да има приступ свом SMTP серверу.

E-пошта се не шаље са једног рачунара на други рачунар на начин како је то реализовано са FTP протоколом већ се шаље од једног до другог корисника. Због тога је и начин адресирања прилагођен корисницима. Не користе се IP адреса и број порта већ систем који специфицира две основне информације: ко је корисник и где се корисник налази. Јасна је аналогија са именом и адресом који се стављају на писма.

Адреса е-поште састављена је од корисничког имена и ознаке домена међусобно повезаних знаком „@“.

<корисничко име>@<ознакадомена>

Где је:

- <ознакадомена> - формирана у складу са синтаксним DNS правилима,
- <корисничко име> - може да садржи бројеве и специјалне знаке у зависности од правила дефинисаних у систему коме припадају.

9.3.1 DNS запис за размену електронске поште

Да би се изашло у сусрет захтевима које су поставили системи е-поште, DNS систем укључује посебна проширења чија је намена да подрже систем адресирања за електронску пошту. Посебан DNS запис за

размену поште, MX запис[1] указује на то који ће SMTP сервер прихватати долазећу пошту за одређено име домена. MX запис је значајан и због његовог DNS дела јер омогућује да се порука пренесе директно од изворишног ка одредишном SMTP серверу, на тај начин се обезбеђује бржа комуникација и већа ефикасност.

9.3.2 Специфицирање идентитета примаоца

Већина клијентских апликација за е-пошту подржава напредне могућности којима је олакшано специфицирање идентитета примаоца. Најзначајније опције су:

- Адресари - сврха је да омогуће апликацији да оствари везу између имена примаоца и његове адресе е-поште;
- Адресирање више прималаца[2] - могуће је постављање адреса већег број прималаца исте поруке тако да ће се порука послати сваком примаоцу;
- Листе за слање[3]- за веће групе ствара се листа адреса која садржи адресе е-поште свих чланова групе. Када један члан групе шаље поруку аутоматски се шаље копија свим члановима групе. Ако члан групе одговори на поруку сви чланови групе добијају одговор.

9.4 Формат поруке електронске поште

Да би се осигурало да сви могу да разумеју поруке е-поште, без обзира ко их шаље, оне морају одговарати уобичајеном формату порука за електронску пошту. Поруке дефинисане у документу RFC5322 састављене су од линија ASCII текста[4]. Свака линија завршава се комбинацијом карактера за крај реда (CR[5]) и прелазак у нови ред (LF[6])[7]. Порука се састоји од заглавља и тела. Тело садржи информацију коју треба пренети примаоцима. Није било могуће пренети слике, звучне, бинарне и остале врсте датотека.

---

[1] *Mail eXchange record*
[2] *Multiple recepient adresing*
[3] *Mailng lists*
[4] US-ASCII
[5] *Carriage Return*
[6] *Line feed*
[7] У литератури се за секвенцу крај реда (CR) и прелазак у нови ред (LF) користи ознака CRLF.

9.4.1 Структура заглавља поруке е-поште

Структура заглавља поруке е-поште је:

<име заглавља>:<вредност заглавља>

Документом RFC5322 дефинисано је више врста заглавља која могу бити укључена у поруке е-поште. Одређени (мањи) број је обавезан и мора бити укључен у сваку поруку. Нека битна заглавља нису обавезна

Група заглавља	Намена	Обавезна заглавља	Опциона заглавља
Заглавља са датумом слања	Садрже датум и време када је порука постала спремна за испоруку	<Date>	
Заглавља о извориште	Садрже информације о пошиљаоцу поруке	<From>	<Sender> <Reply-To>
Заглавља са одредишним адресама	Одређују примаоце поруке	<To>	<Cc> <Bcc
Идентификациона заглавља	Садрже информације потребне да се порука идентификује		<Message-ID> <In-Reply-To> <References>
Описна заглавља	Садрже опционе информације које прималац може да искористи да лакше разуме какву је поруку примио		<Subject> <Comments> <Keywords>
Заглавља о прослеђивању	Намена је да сачувају оригиналне податке о пошиљаоцу и примаоцу код прослеђивања поруке		<Resent-Date> <Resent-From> <Resent-Sender> <Resent-To> <Resent-Cc> <Resent-Bcc> <Resent-ID>
Заглавља за надзор	Заглавља из којих се може сагледати путања коју је порука прешла на путу до примаоца		<Recived> <Return-Path>

Табела 9.1 Групе заглавља електронске поште

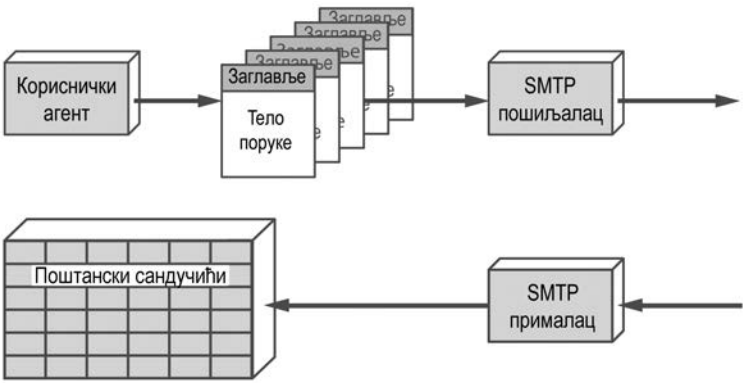
али су скоро увек присутна. Заглавља су подељена у седам група (табела 9.1): заглавља са датумом слања[1], заглавља о извориштy[2], заглавља са одредишним адресама[3], идентификациона заглавља[4], описна заглавља[5], заглавља о прослеђивању[6], заглавља за надзор[7].

9.5 Протокол за пренос електронске поште

Протокол за пренос електронске поште (SMTP[8] протокол) у оквиру скупа TCP/IP протокола дефинисан је документом RFC5321. Поруке које се преносе SMTP протоколом најчешће су у формату дефинсаном у стандарду RFC5322. За сам SMTP протокол формат и садржај поруке нису важни.

9.5.1 Основне операције SMTP протокола

На слици 9.3 илустрован је пут поруке кроз уобичајени систем електронске поште. Протокол SMTP реализује део операција потребних да порука стигне од једног корисника до другог.



Слика 9.3 Пренос порука кроз SMTP систем електронске поште

[1] *Origination Date Field*
[2] *Originator Fields*
[3] *Destination Address Fields*
[4] *Identification Fields*
[5] *Informational Fields*
[6] *Resent Fields*
[7] *Trace Fields*
[8] *Simple Mail Transfer Protocol*

9.5.2 Команде и одговори SMTP протокола

Операције у SMTP систему реализују се преко команди и одговора који се размењују између SMTP клијента и SMTP сервера. Иницијатор је SMTP клијент јер он започиње успоставу TCP везе. Одговори су у облику троцифреног кода који може бити праћен додатном информацијом. Водећа цифра указује на једну од четири категорије одговора: *Positive Completion reply, Positive Intermediate reply, Transient Negative Completion reply* или *Permanent Negative Completion reply*. Ако је екстензија *enhancedstatuscodes* укључена кодови одговора могу бити проширени додатном информацијом.

9.5.3 Комуникација у SMTP системима

Операције у SMTP системима одвијају се у три фазе: успостављање везе, размена једног или више парова команда-одговор и раскидање везе.

Успостављање везе

Када SMTP клијент има поруку(е) за слање он успоставља везу са одредишним SMTP сервером на следећи начин (слика 9.4):

Команде	Опис
HELO	Поздравна порука за почетак SMTP сесије (раније имплементације)
EHLO	Поздравна порука која се шаље SMTP серверу који подржава екстензије. Сервер одговара списком подржаних екстензија. Аргумент уз команду садржи потпуно квалификовано име домена (FQDN[1]) SMTP клијента ако је оно на располагању
MAIL	Указује примаоцу да је пошиљалац спреман да шаље податке. Прималац после потврдом започиње слање
RCPT	Преноси информације о примаоцу поруке
DATA	Уклањање веб странице
RSET	Прекида се процес слања поруке ако се добије порука о грешци
VRFY	Проверава исправност корисниковог поштанског сандучета
EXPN	Тражи потврду да је адреса е-поште на одређеној листи за слање
NOOP	Команда која служи одржавању везе
QUIT	Прекид SMTP сесије

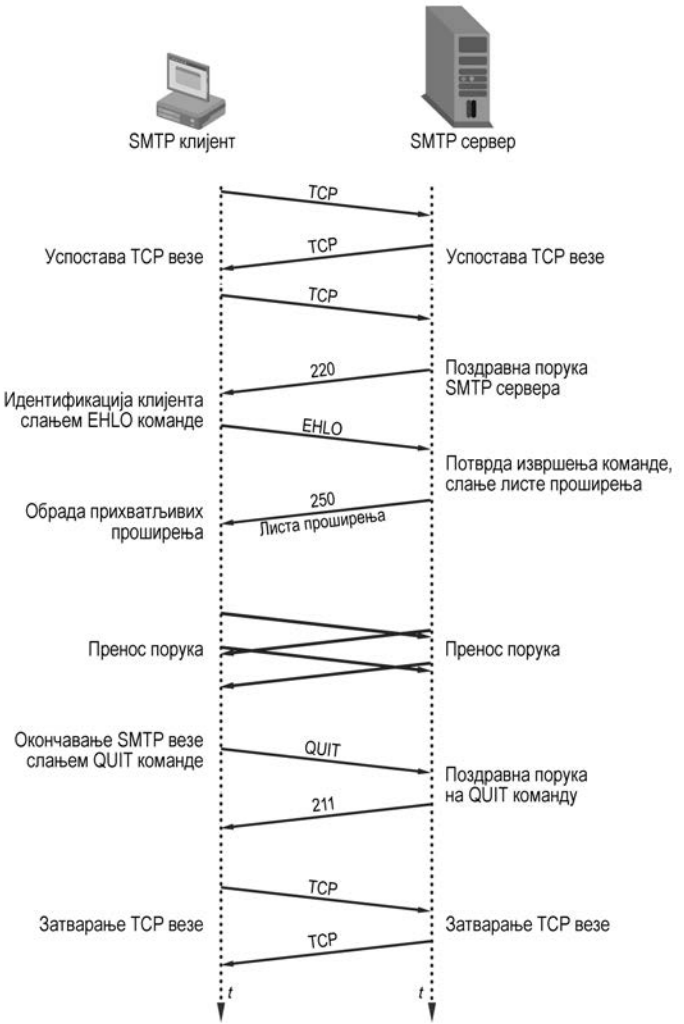
Табела 9.2 Команде SMTP протокола

[1] *Fully Qualified Domain Name*

Одговори	Опис
Positive Completion	
220	Поздравна порука која се шаље приликом успостављања TCP везе
221	Поздравна порука која се шаље као одговор на QUIT команду
250	Позитивна потврда извршења претходно задате команде
251	Потврда да ће SMTP сервер проследити поруку не-локалном примаоцу
252	Најчешће значи да је SMTP прималац покушао да провери адресу примаоца али није добио дефинитиван одговор
Positive Intermediate	
354	Међуодговор за DATA команду (може се слати пошта)
Transient Negative Completion	
421	Одговор на било коју команду примљену док је SMTP сервер ван функције
450	Указује да је сандуче заузето неким другим процесом
451	Указује на локални проблем на серверу
452	Указује на препуњен хард диск на серверу
Permanent Negative Completion	
500	Одговор на лоше задату или предугачку команду
502	Указује да задата команда није подржана на серверу којем је послата
503	Указује да примљене команде нису у исправном редоследу
550	Указује на проблем у вези са поштанским сандучетом
552	Указује да је поштанско сандуче пуно
553	Указује да је адреса поштанског сандучета погрешна
554	Указује на проблем при трансакцији

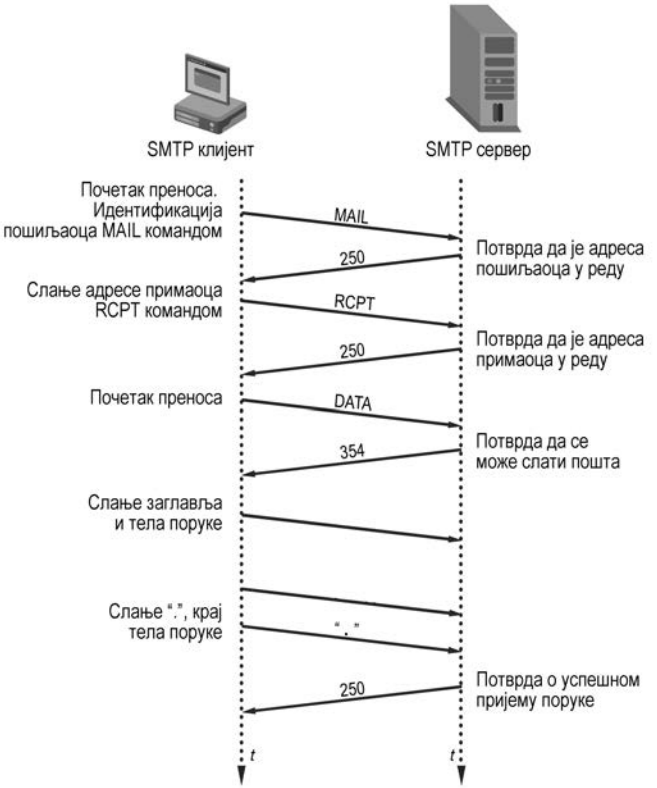
Табела 9.3 Одговори SMTP протокола

- SMTP клијент иницира отварање TCP везе;
- по успостави TCP везе SMTP сервер одговара поздравном поруком 220;
- клијент се идентификује слањем EHLO команде... Старији SMTP системи, који нису у стању да подрже проширења услуга и савремени клијенти који не захтевају проширење услуга могу да користе HELO команду уместо EHLO команде;



Слика 9.4 Процес комуникације у SMTP системима: успостава, пренос и раскидање везе

- сервер прихвата идентификацију и шаље одговор 250;
- ако је сервер недоступан клијент ће у другом кораку одговорити кодом 421 (сервер је ван функције) и прекинути везу;
- SMTP веза се окончава када клијент пошаље QUIT команду;



Слика 9.5 Пренос е-поште у SMTP систему

- SMTP сервер одговара са кодом позит ивног одговора 221 и затвара везу.

Пренос е-поште

Када је веза успостављена SMTP пошиљалац може започети слање једне или више порука. Најчешће се пренос поруке изводи у три корака, а у сваком од њих пошиљалац шаље команде а прималац одговор. Команде које пошиљалац шаље су:

- MAIL команда са аргументом којим се идентификује пошиљалац поруке,
- једна или више RCPT команди са аргументом којим се идентификују примаоци порука,

- DATA команда са аргументима који су тело и текст (линије) поруке.

Аргумент MAIL команда даје повратну путању која се користи за извештавање о грешкама. Ако је прималац спреман да прихвати поруке онда шаље као одговор „250", а ако није може послати неки од одговора који указују на немогућност извршавања команде или грешку при извршавању команде.

Аргументи RCPT команде идентификују појединачне примаоце поруке, ако има више од једног примаоца онда се ова команда користи више пута. На ову поруку могуће је добити један од следећих одговора:

- позитивни одговори - да је одредиште прихваћено или да ће порука бити прослеђена,
- трајно негативан одговор - да порука неће бити прослеђена одредишту, или да се поштанско сандуче не налази на том серверу,
- привремено негативан одговор - прималац може одбити да прими поруку из неког од разлога али пошиљалац треба опет да покуша да испоручи поруку.

DATA команда шаље се када прималац прихвати поруку намењену бар једном одредишту. Ако је спреман да прихвати поруку SMTP прималац треба да пошаље одговор са ознаком 354. По пријему одговора SMTP пошиљалац шаље поруку као низ линија ASCII текста. Крај слања поруке означен је слањем линије која садржи само један знак: „." (тачку). После ње прималац треба да пошаље потврдан одговор „250".

Затварање везе

SMTP пошиљалац затвара везу[1] у два корака:

- пошиљалац шаље QUIT команду и по добијању потврде
- започиње раскидање TCP везе.

9.5.4 Проширени SMTP протокол

Проширени SMTP (ESMTP[2]) дефинисан је документом RF1869. ESMTP клијент започиње сесију EHLO (уместо HELO) командом. Ако

[1] *Connestion closing*

[2] *Extended SMTP, Enhanced SMTP*

сервер на ову команду одговори потврдно, уз одговор ће послати и листу кључних речи које описују подржана проширења. Поред отварања сесије коришћење EHLO указује да је клијент у стању да обради проширења услуга и захтева да сервер пошаље листу екстензија услуга које подржава. Сервер нуди нпр. различите начине аутентификације укључујући и једноставно пријављивање са лозинком. Нека од проширења су:

1. 8-битна MIME подршка (8BITMIME[1]) - RFC1652 дефинише подршку за пренос порука кодираних на овај начин,
2. ауторизација (AUTH[2]) - описује механизам ауторизације да би се одговорило на нове сигурносне захтеве,
3. обавештење о статусу испоруке (DSN[3]) - омогућава SMTP пошиљаоцу да захтева од примаоца обавештење ако дође до проблема при испоруци поруке,
4. унапређени статусни кодови (ENHANCED-STATUSCODES) - дефинише проширење стандардног троцифреног одговора у SMTP протоколу,
5. декларисање величине поруке (SIZE[4]) - омогућава слање информације о величини поруке, на основу чега прималац може да одлучи да ли ће прихватити поруку или не,
6. команда за проточност (PIPELINING[5]) - уместо да се шаље једна по једна команда дозвољава слање команди једна за другом.

9.6 Модели и протоколи за преузимање и приступ електронској пошти

Да би корисник прочитао поруку која му је упућена прво треба да је преузме из поштанског сандучета на SMTP серверу и пребаци на свој рачунар. Документ RFC1733 описује три модела за преузимање и приступ е-пошти. То су:

---

[1] *8-bit MIME support*
[2] *Authorization*
[3] *Delivery Status Notification*
[4] *Message size declaration*
[5] *Command pipelining*

1. Приступ е-пошти у реалном времену[1]. Кориснику би био омогућен сталан и директан приступ његовом поштанском сандучету на серверу.
2. Одложени приступ е-пошти[2]. Овај модел предвиђа да се корисник повеже са сервером на коме се налази његово сандуче, пребаци поруке на свој рачунар и затим их обрише из сандучета сервера. После тога се читање и све друге активности у вези са електронском поштом одвијају без успостављене везе са сервером[3]. Најпознатији пример овог модела је POP[4] протокол.
3. Модел са прекинутим приступом е-пошти[5]. Представља комбинацију два претходно наведена модела. Корисник пребацује поруке на свој рачунар али се поруке не бришу са сервера. Када се корисник поново повеже са сервером извршиће се усаглашавање стања поштанског сандучета на серверу са променама које је направио на свом рачунару док није био повезан са сервером. IMAP протокол је најраспрострањенији представник овог модела.

9.7 Протокол POP3

Протокол POP верзије 3 (POP3[6]) једноставан је и популаран метод „одложеног приступа“, где клијентски уређај приступи серверу, преузме поруке и затим их обрише са сервера. Клијентски програми са POP3 протоколом користе TCP везе на порту 110. Протокол POP3 описује се дијаграмом стања (слика 9.6).

Редослед стања је следећи:

1. Ауторизација[7]. Када сервер изрази спремност да прихвата команде клијент треба да се идентификује да би му се омогућио приступ поштанском сандучету.

[1] *Online access model*
[2] *Offline access model*
[3] *Offline*
[4] *Post Office Protocol*
[5] *Disconected access model*
[6] Дефинисан документат RFC1081 објављен 1988. год.
[7] *Autorization*

Слика 9.6 Дијаграм стања POP3 протокола

2. Трансакција[1]. У овом стању клијенту је дозвољено да изврши операције над сандучетом. Уобичајене операције су: преузимање списка порука, преузимање самих порука, означавање преузетих порука спремних за брисање.
3. Ажурирање[2]. Када је клијент завршио са операцијама и пошаље QUIT команду аутоматски наступа ово стање. Поруке, у претходном кораку, означене за брисање сада се бришу и TCP веза се прекида.

9.8 Протокол IMAP4

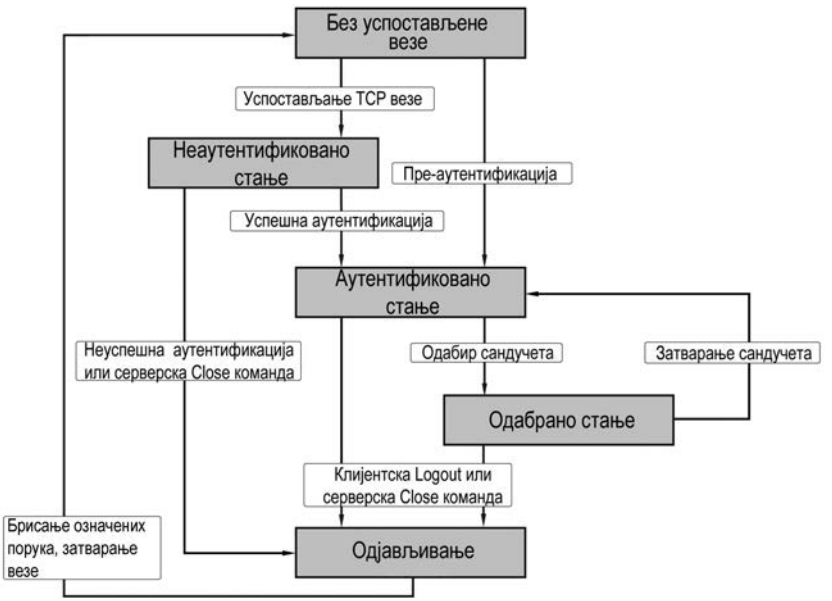
Протокол IMAP4 (IMAP верзије четири) је протокол за пренос електронске поште са клијентским и серверским функцијама (дефинисан документом RFC3501). IMAP протокол задржава оригинале порука на серверу, а клијенту шаље њихове копије. Корисник може имати више поштанских сандучића и у било ком тренутку одабрати који жели да користи. Клијентски програми са IMAP протоколом могу поставити критеријуме према којима ће бирати које поруке да преузму. Пренос измена са стране клијента на сервер врши се периодично и зове се (ре)синхронизација.
IMAP протокол користи на транспортном слоју TCP протокол (порт 143).

[1] *Transaction*
[2] *Update*

9.8.1 Основна стања и операције IMAP протокола

Сесија IMAP4 има четири различита стања. Команде које се могу задати биће прихваћене само ако је сесија у одговарајућем стању али постоје и оне које се увек прихватају. Четири стања IMAP4 сесије су (слика 9.9):

1. Неаутентификовано стање. Сесија уобичајено почиње овим стањем осим ако није извршена аутентификација на неки други начин. Сврха овог стања је да клијент обезбеди информације везане за аутентификацију и пређе у следеће стање.
2. Аутентификовано стање. Када је клијент аутентификован прелази се у аутентификовано стање. За то време дозвољено је вршити само операције над целим сандучићима. Клијент неће моћи да се бави појединачним порукама док је у аутентификованом стању.
3. Одабрано стање[1]. Пошто одабере поштанско сандуче клијент може да приступи и манипулише појединим појединачним порукама из поштанског сандучета. Када заврши оперције над



Слика 9.9 Дијаграм стања IMAP протокола

[1] *Selected state*

порукама у једном сандучету дозвољено му је да се врати у претходно стање и одабере друго сандуче.

4. Стање одјављивања[1]. Да би прекинуо сесију клијент треба да зада команду којом се одјављује. У ово стање могуће је доћи

Команде	Опис
Команде за било које стање	
CAPABILITY	Сервер треба да одговори које функције подржава
NOOP	Сервер треба да одговори које функције подржава
LOGOUT	Команда којом се прекида актуелна сесија
Команде за неаутентификовано стање	
LOGIN	За пренос корисничког имене лозинке
AUTHENTICATE	Захтевање употребе одређеног метода аутентификације
STARTTLS	Захтев да се користи TLS[2] протокол при аутентификацији
Команде за аутентификовано стање	
SELECT	Одабир једног од сандучића. Ако се успешно изврши сесија прелази у одабрано стање
EXAMINE	Исто као SELECT али без дозволе да се врше промене у сандучету
CREATE	Прављење новог сандучета
DELETE	Брисање неког од сандучића
RENAME	Преименовање постојећег сандучета
STATUS	Захтева детаљнију информацију о стању у одређеном сандучету
Команде за одабрано стање	
CLOSE	Затвара актуелно сандуче и враћа сесију у аутентификовано стање
EXPUNGE	Трајно брише поруке означене за брисање и затвара сандуче
SEARCH	Претражује поруке у сандучету на основу задатог критеријума
FETCH	Захтева информације о скупу порука
STORE	Поставља у поштанско сандуче додатне информације о једној или више порука
COPY	Копира скуп порука у друго сандуче назначено у команди

Табела 9.4 IMAP команде

[1] *Logout*

[2] *Transport Layer Security*

из било ког другог. Ако дуже времена сесија није активна биће прекинута и веза ће бити затворена.

9.8.2 IMAP команде и повратне информације

По успостави сесија почиње размена команди и одговора (низови ASCII текста завршени CRLF секвенцом). Команде (табела 9.4) шаље клијент а сервер шаље на њих одговоре. IMAP протокол може да

Резултати	Опис
OK	Позитиван одговор на команду, обично се шаље са ознаком команде на коју се односи
NO	Негативан одговор на команду, може се послати са ознаком команде на коју се односи али не мора
NOOP	Не ради ништа али се може искористити да ресетује часовник који одређује када ће сервер слати обавештење о новопристиглим порукама.
BAD	Порука о грешци

Табела 9.5 IMAP повратне информације - резултат

Одговори	Опис
PREAUTH	Шаље се на почетку сесије клијенту да укаже да није потребна аутентификација
BYE	Поздравна порука приликом затварања сесије
ALERT	Порука сервера кориснику IMAP клијента да га обавести о нечему
CAPABILITY	Листа опција које сервер подржава
PARSE	Обавештава о грешци приликом обраде заглавља
PERMANENT	Саопштава клијенту листу ознака за поруке (флагс) које може да користи
FLAGS	Обавештава клијента да је сандуче отворено само за читање тј. за писање
READ-ONLY и WRITE-ONLY	Шаље децимални број који јединствено означава поруку
UIDNEXT	Потврђивање исправности ознаке поруке
UIDVVALIDITY	Шаље низ ознака порука које још нису прочитане
UNSEEN	Преименовање постојећег сандучета

Табела 9.6 IMAP повратне информације - одговори

користи систем обележавања[1] команди да би се нагласило на коју од команди се одговор односи. Постоје два типа повратних информација: резултат[2] (табела 9.5) и одговор[3] (табела 9.6). Резултат указује на статус извршавања команде а одговор је било која врста информације која се шаље од сервера ка клијенту.

9.8.3 Процес аутентификације IMAP протокола

IMAP сесија почиње када IMAP клијент успостави TCP везу са IMAP сервером. Под нормалним околностима IMAP сервер не зна ко је клијент и зато почиње сесију у неаутентификованом стању. Клијенту неће бити дозвољено ништа да ради док се не аутентификује. Може се догодити да сервер већ зна клијентов идентитет на основу неког другог система аутентификације који није део IMAP протокола а означава се као преаутентификација. IMAP4 дефинише три механизма аутентификације:

1. Једноставно пријављивање[4]. Корисник се идентификује корисничким именом и лозинком (LOGIN команда);
2. Пријављивање са TLS протоколом[5]. Употреба овог протокола активира се STARTTLS командом после чега се обавља аутентификација (командама LOGIN или AUTHENTICATE);
3. Уговорени метод аутентификације[6]. Команда AUTHENTICATE омогућава употребу било ког метода аутентификације. Предуслов је да метод буде истовремено подржан на страни сервера и клијента.

Аутентификовано стање, управљање сандучићима

Када уђе у аутентификовано стање клијент је ауторизован и дозвољено му је да шаље команде серверу. Команде које може да употреби односе се само на целе сандучиће. Ако клијент жели да се бави одређеним порукама мора пре тога да укаже у ком се сандучету

[1] *Tag*
[2] *Result*
[3] *Response*
[4] *Plain login*
[5] *TLS login*
[6] *Negotiated authentication method*

оне налазе. Бирање једног од сандучића врши се командама SELECT или EXAMINE и сесија улази у одабрано стање. Сервер на ове команде одговара слањем корисних информација о одређеном сандучету као што су: број порука у сандучету, број нових порука, које ознаке[1] за поруке могу да се користе итд.

Одабрано стање, управљање порукама

У одабраном стању клијент може да пошаље команде које се односе на е-поруке али и команде које се користе у аутентификованом стању. Из одабраног стања може се изаћи на три начина: може се задати LOGOUT команда и прекинути сесија, може се задати CLOSE команда и затворити актуелно сандуче али сесија неће бити прекинута и могу се користити SELECT или EXAMINE команде при чему се актуелно сандуче затвара и прелази на друго сандуче.

9.9 Веб пошта

За приступ и преузимање е-поште поред POP3 и IMAP4 протокола највише се користи веб пошта[2].

Пораст популарности веб система (WWW[3]) довео је до нове категорије система е-поште засноване на веб технологији. Ови алати засновани на веб технологији не захтевају самостални читач е-поште. Корисник једноставно треба да посети одговарајућу интернет страницу користећи свој претраживач[4] Интернета и приступи пошти кроз понуђени веб интерфејс. Корисник зато може приступити својим порукама са било ког рачунара (или мобилног уређаја) који је повезан на Интернет. Примери овакве услуге су: *Hotmail, Yahoo Mail*, и *GMail*[5]. Веб пошта свестрана је и једноставна за употребу. Није потребно вршити никаква подешавања (што је случај са POP3 и IMAP4 протоколима) тако да се корисници навикли на веб услуге лако сналазе са понуђеним интерфејсом. Пошто поруке нису стварно присутне на корисниковом рачунару било која

[1] *Flag*
[2] *Webmail*
[3] *Word Wide Web*
[4] *Browser*
[5] *Google Mail*

операција која се изводи над њима захтева везу са Интернетом. Скоро сви интернет посредници нуде својим корисницима ову услугу.

9.10 Вишенаменска проширења за електронску пошту на Интернету

Документом RFC5322 дефинисан је формат порука за размену порука е-поште у мрежама са TCP/IP протоколом. Поруке морају да буду написане са ASCII карактерима. Да би се омогућило да се е-поштом преносе произвољне врсте датотека и да би било могуће писати поруке и другим скупом карактера (не ASCII) дефинисан је стандард за вишенаменска проширења за електронску пошту (MIME[1] стандард).

Коришћењем техника MIME стандарда (описаног у документима RFC2045, RFC2046, RFC2047, RFC4288, RFC4289 и RFC2049) могуће је конвертовати различите информације које нису ASCII кодиране у ASCII кодиране. На тај начин RFC5322 порукама могуће је пренети: нетекстуалне врсте података, слике, мултимедијалне садржаје, бинарне датотеке, извршне датотеке, посебне врсте датотека (нпр. *pdf*) и текстуалне поруке написане на не-ASCII скупу карактера.

Све ово се постиже поштовањем посебних правила кодирања којима се не-ASCII датотеке пребацују у ASCII облик. Потребно је додати заглавља која садрже информације потребне при декодовању. Употреба MIME стандарда не утиче на SMTP и друге протоколе. Потребно је само изменити корисничке апликације тако да се оствари подршка за MIME стандард.

9.10.1 Основни типови структура

Тачан метод којим су подаци кодовани у тело поруке, описан у MIME заглављима, зависи од укупне структуре MIME поруке. Могуће су две основне врсте (типа):

- Једноставна структура[2]. MIME порука која има једноставну структуру и преноси једну врсту медија нпр. само текст или само слику. Зато је у телу поруке присутна само једна информација о

[1] *Multipurpose Internet Mail Extensions*
[2] *Discrete Media*

кодирању.

- Сложена структура[1]. MIME стандард омогућава да се различите врсте датотека комбинују у једну поруку. На пример, могуће је послати текст са сликом или убацити комплетну другу поруку у тело поруке као једну целину. Сваки од ових делова може имати сложену структуру па ће тело поруке због тога садржати више делова са одговарајућим MIME заглављима.

9.10.2 MIME заглавља

Документ RFC2045 описује пет главних заглавља која преносе основну информацију о садржају сваке MIME целине (целе поруке или дела тела). То су:

1. MIME-верзија[2]. Свака MIME порука има ово заглавље, оно идентификује поруку е-поште као MIME поруку.
2. Врста садржаја[3]. Описује природу података кодираних у MIME целини. Ово заглавље одређује тип и подтип садржаја примаоца, може да одреди какву врсту медија садржи порука и да ли је једноставне или сложене структуре.
3. Кодирање садржаја[4]. За поруке које користе једноставну структуру описује метод којим су подаци кодирани у тело поруке. Користећи информацију из овог заглавља корисник зна како да декодира поруку у њен оригинални облик. Намена овог и претходно поменутог заглавља можда најбоље описују суштину MIME стандарда.
4. Идентификатор садржаја[5]. Омогућава да се поруци дода идентификациони кôд везан за MIME садржај.
5. Опис садржаја[6]. Ово је опционо заглавље које омогућава да се призвољни описни текст дода уз MIME целину.

Поред горепоменутих пет главних заглавља постоје и додатна заглавља:

---

[1] *Composite Media*
[2] *MIME version*
[3] *Content-Type*
[4] *Contentt-Transfer-Encoding*
[5] *Content-ID*
[6] *Content-Description*

1. Распоређивање садржаја[1]. У вишеделним[2] MIME порукама заглавље може бити додато делу тела да би се контролисало како ће информација бити приказана кориснику.
2. Лоцирање садржаја[3]. Омогућује да на локацију дела тела MIME поруке указује јединствен идентификатор ресурса (URI[4]).
3. Дужина садржаја[5]. Приказује дужину MIME целине у бајтовима.

9.10.3 Заглавље врста садржаја

Пошто MIME подржава пуно различитих врста података неопходно је да свака порука има информацију која описује њен садржај, да би прецизно декодовање било могуће. Ово је функција заглавља „врста садржаја“:

<*type*>/<*subtype*> [; параметар1 ; параметар2 ; ...параметарN]

Сврха ових различитих елемената у заглављу је да опишу садржај MIME целине идући од општег ка специфичнијем. Први елемент „<*type*>“ назива се врховна[6] врста медија[7] и описује општи[8] облик податка. На пример, указује да је MIME целина текст, слика, звучни запис итд.

Други елемент „<*subtype*>“, даје прецизнију информацију о облику и формату податка. Ове елементе може да прати један или више параметара који су најчешће опциони али могу бити захтевани за неке типове података. Параметри дају још детаљнију информацију о природи података.

Пример 9.1:

Content-Type: text/plain; charset=“us-ASCII”
Content-Type: multipart/mixed; boundary=gc0p4Jq0M2Yt08jU534c0p

Врста садржаја: дискретне врсте

Постоји пет врста медија који се преносе: текст, слика, звук, видео

[1] *Content-Disposition*
[2] *Multipart*
[3] *Content-Location*
[4] *Uniform Resource Identifier*
[5] *Content-Length*
[6] *Top-Level*
[7] *Media Type*
[8] *Overall*

и апликација. Њих прате групе подврста тако да постоји комбинација врста/подврста као што су:

- текст[1] могуће су комбинације:
  - *text/plain, text/enriched, text/html, text/css...*
- слике:
  - *image/jpeg, image/gif, image/tiff…*
- звук и видео
  - *audio/basic, audio/mpeg video/mpeg, video/dv, video/quicktime…*

Апликација се издваја као посебна врста медија. Ако податак не припада ниједној од претходно наведених категорија за његов опис се користи ова врста медија. Подврста је искоришћена да укаже на врсту апликације која користи овај податак. Неке често коришћене комбинације су: *application/octet-stream, application/postscript, application/applefile, application/msword, application/pdf, application/vnd.ms-powerpoint, application/zip*…

Композитне врсте садржаја

Ако се преноси композитна, односно сложена врста медија уместо пет дискретних врста у *Content-Type* заглављу наћи ће се информација о једној од две композитне врсте:

1. Вишеделна заглавља[2]. Дозвољава да више скупова података буде послато једном MIME поруком. Сваки од делова биће представљен као једна засебна дискретна врста медија. Постоји шест врста оваквих заглавља и оне су описане у табели 9.7
2. Заглавље - порука[3]. Дозвољава да у поруци буде упакована друга порука. Једна порука може се наћи у другој, нпр. при прослеђивању или одговору на примљену поруку.

Код вишеделних заглавља основни процес је следећи:

- сваки засебан део обрађује се као да ће бити послат у облику тела поруке која припада неком од дискретних типова медија,
- вишеделна порука се саставља а састоји се од преамбуле[4],

[1] *Text*
[2] *Multipart*
[3] *Message*
[4] *Preamble text area*

Вишеделна заглавља	Опис
Вишеделно/мешовито заглавље[1]	Делови тела поруке нису стварно повезни него су само упаковани заједно да би се пренели у једној поруци
Вишеделно/алтернативно заглавље[2]	Различити делови тела поруке су у ствари иста информација представљена на више начина
Вишеделно/паралелно заглавље[3]	Указује кориснику да све делове тела треба да прикаже истовремено
Вишеделно/сажетак заглавље[4]	Овим се дозвољава да порука носи и неку врсту сажетака[5]
Вишеделно/повезано заглавље[6]	Указује да су делови тела поруке међусобно повезани
Вишеделно/шифровано заглавље[7]	Користи се за шифроване податке

Табела 9.7 Вишеделна заглавља

линије разграничења за којим следи први део тела,

- Посебан параметар: линија разграничења[8] је укључена у *Content-Type* заглавље које се односи на целу поруку. Ово је у ствари низ случајних вредности које прималац користи да подели тело поруке на одговарајуће делове.

Пример 9.2. MIME поруке, слање обичног текста и слике (као прилог) приказан је на слици 9.10. Поруку шаље Ана са адресе *ana@gmail.com,* Банету на адресу *bane@gmail.com*. Порука се преноси као вишеделна/мешовита, први део поруке је текст који је Ана написала Банету а други је слика коју је ставила као прилог.

9.10.4 Методе кодовања порука

Да би коришћењем MIME стандарда био послат не-ASCII податак

[1] *Multipart/mixed*
[2] *Multipart/alternative*
[3] *Multipart/parallel*
[4] *Multipart/digest*
[5] *Digest*
[6] *Multipart/related*
[7] *Multipart/encrypted*
[8] *Boundary*

```
Message-ID: <4AAE74B5.7000100@gmail.com>
Date: Mon, 14 Sep 2009 18:52:05 +0200
From: Ana Ivanovic <ana@gmail.com>
User-Agent: Thunderbird 2.0.0.23 (Windows/20090812)
MIME-Version: 1.0
To: Bane Dimitrijevic <bane@gmail.com>
Subject: Slika
Content-Type: multipart/mixed; boundary="------------102020503010501020107 04"
This is a multi-part message in MIME format.
--------------010202050301050102010704
Content-Type: text/plain; charset=ISO-8859-1; format=flowed
Content-Transfer-Encoding: 7bit
Ovo je poruka koji Ana salje Banetu
Kao dodatak tekstu u obliku priloga salje sliku "MIME-poruka.png"
Tekst je napisan US-ASCII karakterima a prilog je kodiran metodom
Osnova64
--------------010202050301050102010704
Content-Type: image/png; name="MIME-poruka.png"
Content-Transfer-Encoding: base64
Content-Disposition: inline; filename="MIME-poruka.png"
iVBORw0KGgoAAAANSUhEUgAAAecAAAJuCAIAAAAFOF2OAAAACXBIWXMAAAsTAAALEwEAmpwY
AAAKT2IDQ1BQaG90b3Nob3AgSUNDIHByb2ZpbGUAAHjanVNnVFPpFj333vRCS4iAlEtvUhUI
IFJCi4AUkSYqIQkQSoghodkVUcERRUUEG8igiAOOjoCMFVEsDIoK2AfkIaKOg6Olisr74Xuj
...
AFBtAAAAUG0AAABQbQAAgGoDAACAagMAAIBqAwBAJ/F/DwCFAdaRkHwuPQAAAABJRU5ErkJg
gg==
--------------010202050301050102010704—
```

Слика 9.10 Пример MIME поруке, слање текста и слике као прилог

потребно је да се кодира. Заглавље *Content-Transfer-Encoding* садржи информацију како су делови тела поруке кодирани. Документ RFC5322 захтева да цела порука мора бити кодована у US-ASCII скупу карактера са седмобитном презентацијом. Ово значи да је у једном бајту од 256 могућих вредност у употреби само 128 тако да није могуће директно преносити датотеке које користе свих 256 вредности.

Да би се овај проблем превазишао дефинисане су следеће врсте кодирања:

1. 7-битно кодирање. Указује да је порука већ у ASCII формату компатибилном са RFC5322. Ово је подразумеван метод кодирања. Ако заглавље *Content-Transfer-Encoding* није присутно порука се декодира као да је написана у овом облику.
2. 8-битно/бинарно кодирање. Ово је опција која је укључена у MIME систем да би се омогућио директан пренос података без

кодирања. Проширење је дефинисано документом RFC1652 - SMTP[1].

3. Наводи који могу бити приказани[2]. Ово је специјална врста кодирања која се користи када је већина података који се преносе ASCII текст али постоје и одређена одступања од RFC5322 стандарда. Ова одступања, тј. делови текста који се не могу приказати обичним ASCII карактерима, посебним методом кодовања конвертују се у ASCII текст, остатак текста остаје непромењен.
4. Кодирање по основи 64[3]: Ово кодирање омогућује представљање произвољног бинарног податка у облику ASCII текста чиме се одговара захтевима RFC5322 стандарда. Скоро сви прилози који се шаљу уз поруке е-поште кодирани су на овај начин.

9.10.5 Примена MIME система са не-ASCII заглављем

Све до сада поменуте методе кодовања односе се на садржај тела поруке. MIME систем[4] омогућава да се не-ASCII карактера користе и за заглавља (описано у документу RFC2047). Због тога се могу у делу за тему[5] поруке користити знакови локалног алфабета (нпр. ћирилице).

Синтакса МИМЕ кодиране речи

Све не-ASCII вредности у заглављима замењују се кодним речима према следећој синтакси[6]:

=?<*charset*>?<*encoding*>?<*encoded-text*>?=

Где је:

- <*charset*>
  - означава скуп карактера који се користи, на пример „ISO-8859-1“;
- <encoding>
  - могућа је једна од вредности „B“ ако се користи кодирање са

[1] *Service extension for 8bit-MIMEtransport*
[2] *Quoted-printable*
[3] *Base64 encoding*
[4] *MIME extension for non-ASCII mail message headers*
[5] *Subject*
[6] *Encoded-word syntax*

основом 64 или „Q“ ако се користи *quoted-printable* метод;

- <*encoded-text*>
  - кôд добијен кодирањем не-ASCII текста.

Пример 9.3. Изглед заглавља са не-ASCII карактерима
Тема како је корисник исписује:

Subject: Kodirana-reč sintaksa

Тема после MIME кодирања:

Subject: =?UTF-8?B?IktvZGlyYW5hLXJlxI0gc2ludGFrc2Ei?=

9.11 Интернационализација е-поште

Организација IETF је иницирала радне групе за стандардизацију и техничко разрешавање питања интернационализације е-адреса (EAI[1]). Резултат радне групе су документи RFC6530, RFC6531, RFC6532 и RFC6533. Документ RFC6530[2] омогућава не-ASCII карактерима да се користе и у локалном делу адресе и у делу адресе за домен.

Основни EAI концепт укључује размену поште по стандарду UTF-8. Локални сервери су задужени за локални део адресе а део адресе домена је дефинисан интернационалним стандардом за домене (IDN[3]), који се такође шаље као UTF-8 кôд. Сервер електронске поште такође је одговоран за повезивање (мапирање) између EAI формата и ASCII еквивалента[4].

EAI концепт омогућава корисницима да се њихове е-адресе могу исписати и са локалним скупом карактера и са ASCII скупом карактера. Апликације које могу да препознају интернационална имена домена и е-адресе морају да омогуће и конверзију ових репрезентација. Значајна потражња за оваквим адресама очекује се у Кини, Јапану, Русији и другим областима у којима су основна писма не-латинични карактери.

EAI адресе нису у складу са препорукама описаним документом RFC5322 и због тога неће радити са серверима електронске поште на

[1] *Email Address Internationalization, среће се и под називом IMA – Internationalized Mail Address*
[2] *Overview and Framework for Internationalized Email*
[3] *Internationalized Domain Name*
[4] *ASCII alias*

којима су препоруке примењене (већина данашњих сервера). Сервери на којима су имплементиране препоруке дефинисане стандардом RFC6530 могу да подрже интернацинална имена е-адреса.

Пример 9.4. EAI адресе:

- Латинички алфабет:
  - pelé@example.com
- Грчки алфабет:
  - rδοκιμή@παράδειγμα.δοκιμή
- Јапански карактери:
  - 甲斐@黒川. 日本
- Ћирилички карактери:
  - чебурашка@ящик-с-апельсинами.р

9.12 Питања и задаци

1. Објаснити архитектуру система е-поште представљену на слици 9.1.
2. Који од транспортних протокола користи систем е-поште?
3. Које основне функције обезбеђује систем е-поште?
4. Објаснити намену сваке од целина приказаних на слици 9.2.
5. Које целине учествују у реализацији система е-поште?
6. Како је у DNS систему реализована подршка за е-пошту?
7. Како се специфицира идентитет примаоца?
8. Описати формат поруке е-поште.
9. Набројати и описати врсте заглавља која могу бити укључена у поруке е-поште.
10. Детаљно описати пренос порука кроз SMTP систем електронске поште.
11. Навести и описати неке од команди SMTP протокола.
12. Навести и описати најчешће коришћене одговоре SMTP протокола.
13. Нацртати и описати фазе у реализацији SMTP протокола.
14. Описати проширење SMTP протокола.
15. Набројати и описати моделе за преузимање и приступ електронској пошти.
16. Који TCP порт користе клијентски програми са POP3 протоколом?
17. Набројати и описати стања кроз која пролази POP3 протокол.

18. Набројати и описати стања кроз која пролази IMAP протокол.
19. Набројати и описати IMAP команде.
20. Набројати и описати IMAP повратне информације.
21. Како се реализује процес аутентификације код IMAP система?
22. Описати веб пошту.
23. Упоредити веб пошту са POP3 и IMAP протоколима.
24. Која је намена MIME стандарда?
25. Описати структуру MIME порука.
26. Описати заглавља која преносе основну информацију о садржају сваке MIME целине.
27. Које све делове садржи заглавље врста садржаја. Описати их.
28. На шта се све односи MIME кодовање?
29. Како се примењује MIME систем за поруке са не-ASCII заглављем?
30. На шта се односи интернационализација е-поште.

10. Протокол за пренос датотека

Једна од најважнијих група апликација која омогућава премештање података између међусобно умрежених уређаја јесу апликације за пренос датотека[1] и порука. Ово је група апликација које се свакодневно користе у комуникацији умрежених система. Могу се категоризовати на следећи начин:

- Апликације за општи пренос датотека - подразумевају премештање било које врсте датотека између умрежених уређаја. Протоколи који се користе код ових апликација јесу протокол за пренос датотека (FTP[2] протокол) и елементарни протокол за пренос датотека (TFTP[3] протокол);
- Апликације за пренос порука - омогућавају различите врсте комуникација али у којима се користе специјалне врсте датотека као што су HTML странице или поруке које се размењују електронском поштом. Протоколи који се користе код ових апликација (а биће обрађени у овом уџбенику) јесу протокол за електронску пошту (SMTP[4]), протокол за пренос хипертекст докумената (HTTP[5]) и протокол за надзор и управљање мрежама (SNMP[6]).

10.1 Основне карактеристике протокола за пренос датотека

Протокол за пренос датотека (FTP протокол) једна је од често

[1] Подаци су организовани у дискретне јединице датотеке (фајлове – *file*). Датотеке које се посебно креирају за комуникацију називају се поруке (*message*).

[2] *File Transfer Protokol*

[3] *Trivial File Transfer Protocol*

[4] *Simple Mail Transfer Protocol*

[5] *Hyper Text Transfer Protocol*

[6] *Simple Network Management Protocol*

коришћених апликација. Овај протокол представља интернет стандард за пренос датотека. Мора се пажљиво правити разлика између преноса датотеке[1], што FTP протокол омогућава, и приступа датотеци[2] што омогућавају апликације као што је NFS[3] систем. Када се користи FTP протокол комплетна датотека се копира из једног система у други систем. Да би се користио FTP протокол потребна је ауторизација (кориснички налог[4]) за приступ серверу. Са друге стране постоји и анонимни FTP[5] (без ауторизације) за оне организације које желе да пруже опште информације корисницима.

Апликације како се мрежа користи концепцијски су подељене у две категорије: директне и индиректне. Директне мрежне апликације дозвољавају кориснику да приступи удаљеној станици као да је она локална и стварају му илузију као да мрежа и не постоји. Индиректна мрежна апликација преузима датотеку са удаљене станице и користи је на локалном систему. Ако је потребно, поново је враћа натраг удаљеној станици.

Ова два метода примењена су у TCP/IP мрежним апликацијама и то: *Telnet* за директан приступ и FTP протокол за индиректан приступ.

FTP протокол је од самог почетка пројектован тако:

- да се може користити између различитих крајњих станица (хостова) и
- да ради под различитим оперативним системима који користе различите структуре датотека а некада и различите врсте карактера.

FTP протокол подржава ограничен број кодова (ASCII, бинарни, итд.) и различите структуре датотеке (проток бајтова или записа[6]). Спецификација за FTP протокол описана је у документу RFC959.

Да би се обезбедио поуздан пренос FTP протокола на транспортном слоју користи се TCP протокол.

---

[1] *File Transfer*

[2] *File access*

[3] *Network File System* развијен од стране *Sun Microsystem* о1984. год.

[4] *Account to login*

[5] *Anonymous FTP*

[6] *Record*

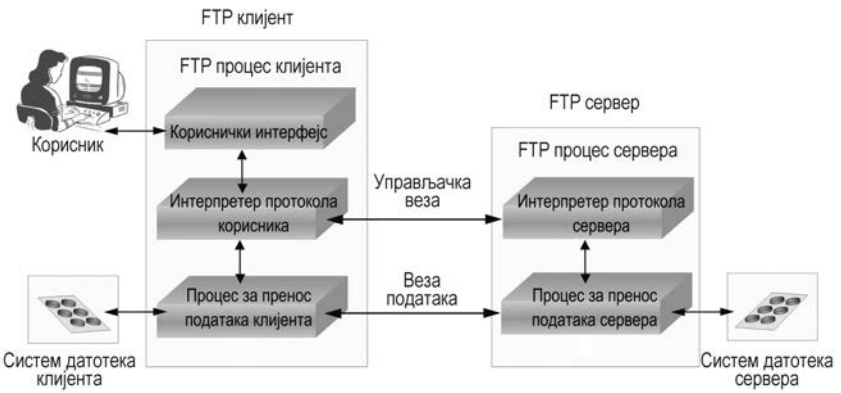
10.2 Модел FTP протокола

Стандард RFC959, у коме је описан FTP протокол, заснован је на FTP моделу приказаном на слици 10.1. Овај модел дефинише улоге целина које учествују у преносу датотека и две комуникационе (логичке) везе које су успостављене између одговарајућих целина. Дефинисане су и компоненте FTP протокола које управљају овим каналима. Протокол FTP заснован је на клијент/сервер архитектури.

Код преноса датотека (за разлику од других апликација) FTP протокол користи две TCP везе: управљачку везу[1] и везу података[2].

Управљачка веза главна је TCP веза која се формира код успостављања FTP сесије. Одржава се све време док клијент комуницира са сервером а користи се (само) за размену наредби[3] које клијент шаље серверу и одговора[4] који сервер враћа клијенту. Не користи се за пренос датотека.

Веза података успоставља се сваки пут када се датотека преноси између клијента и сервера. Она се успоставља и у другим приликама (као што ће се касније видети). Када се пренос заврши веза се окончава.



Слика 10.1 Процеси укључени у пренос датотека

[1] *Control connection*
[2] *Data connection*
[3] *Commands*
[4] *Replies*

Пошто су канали за управљање и пренос података раздвојени FTP модел дели и софтвер који је одговоран за те канале.

Компоненте FTP процеса сервера као што је приказано на слици 10.1 су:

- Интерпретер протокола сервера (SPI[1]) - одговоран је за надзор управљачке везе на страни сервера. Он ослушкује захтев клијента за успоставом везе са сервером, прихвата команде интерпретера протокола корисника (UPI[2]), шаље одговоре и управља преносом података сервера;
- Процес за пренос података сервера (SDTP[3]) - користи се за слање и пријем података ка или од процеса за пријем података клијента (UDTP[4]). Он у интеракцији са локалним системом датотека сервера очитава или уписује датотеке.

FTP процес на страни клијента (слика 10.1) садржи следеће делове:

- Интерпретер протокола корисника (UPI) - одговоран је за надзор управљачке везе на страни клијента. Иницира FTP сесију слањем захтева интерпретеру протокола сервера (SPI). Једном када је веза успостављена интерпретер протокола корисника обрађује команде које добија од корисничког интерфејса (UI[5]), шаље их интерпретеру протокола сервера и прихвата одговоре. Он такође надзире и процес за пренос података клијента (UDTP[6]).
- Процес за пренос података клијента (UDTP) - шаље или прима податке од процеса за пренос података сервера. Он или успоставља везу података или прихвата везу за податке коју иницира сервер. Ради у интеракцији са локалним системом датотека клијента.
- Кориснички интерфејс (UI) - обезбеђује FTP интерфејс боље прилагођен кориснику[7] и омогућава:

[1] *Server Protocol Interpreter*
[2] *User Protocol Interpreter*
[3] *Server Data Transfer Process*
[4] *User Data Transfer Process*
[5] *User Interface*
[6] *User Data Transfer Process*
[7] *User friendly*

- употребу једноставних команди за FTP функције (уместо интерних FTP команди) и
- враћа резултате и информације ка кориснику FTP сесије.

На основу описа FTP система види се да се корисник не бави наредбама и одговорима који се размењују управљачком везом. Ти детаљи препуштени су интерпретерима протокола (PI). Када је потребна размена података између клијента и сервера тај део одрађује процес за пренос података (DTP). Поље означено као кориснички интерфејс представља одабрану врсту корисничког интерфејса и конвертују се у одговарајуће FTP наредбе које се преносе управљачком везом. Одговори које сервер враћа (истим путем) могу се конвертовати у одговарајући формат да би били приказани интерактивном кориснику. Два интерпретера протокола су ти који позивају функције преноса података када је то неопходно.

10.2.1 Успостављање FTP управљачке везе

Управљачки канал успоставља се на стандардан клијент/сервер начин. То значи да сервер извршава пасивно отварање[1] транспортне везе на добро познатом порту за FTP протокол (порт 21) и чека на захтев за успоставом везе клијента. Клијент извршава активно отварање[2] TCP везе користећи ефемералне[3] бројеве портова. Када је веза успостављена, успостављена је и управљачка веза. Наредбе и одговори могу се тада размењивати између интерпретера протокола корисника и сервера.

10.2.2 Наредбе и одговори FTP протокола

Наредбе су представљене великим словима ASCII карактера, дужине 3 или 4 бајта. Наредбе специфицирају параметре за пренос података (порт, врсту преноса, врсту репрезентације и структуру података) као и природу рада са системом датотека (смештање, преузимање, додавање, брисање итд.). Клијент може послати серверу више од 30 различитих FTP наредби. У табели 10.1 дате су неке уобичајене наредбе, од којих ће већина бити представљена у овом поглављу.

---

[1] Користи се примитива *Passive Open*, детаљно је описано у одељку 6.6 овог уџбеника.

[2] Користи се примитива *Active Open*, детаљно је описано у одељку 6.6 овог уџбеника.

[3] *Ephemeral* - привремени, кратког трајања. Потиче од грче речи εφήμερος – *ephemeros*, што значи "у трајању од једносг дана". Додела порта је привремена и важи само за време трајања везе.

Пошто је комуникациони канал успостављен потребно је да се изврши фаза аутентификације, односно пријављивање корисника[1] на FTP сервер.

Одговори су троцифрени бројеви иза којих је (опционо) ASCII кодирана (текстуална) порука. Идеја је да софтвер на основу броја у одговору одређује како да одговор обради. Опционо низ је текстуална порука. Клијент шаље и нумеричку и текстуалну поруку у одговору тако да интерактивни корисник може да одреди значење одговора само читањем текстуалне поруке а не мора да памти нумерички кодирани одговоре. Свака од три цифре у коду има различито значење[3]. У табели 10.2 описана су значења прве и друге цифре кодираног одговора.

Трећа цифра даје додатно значење поруци о грешци[4]. Неки од уобичајених одговора заједно са могућом пратећом текстуалном поруком су:

- 125 - веза за податке је већ отворена; пренос почиње,
- 200 - наредба је у реду,

Наредба	Опис
ABOR	Прекини претходну FTP наредбу и било какав пренос података
LIST *filelist*	Прегледати списак датотека или директоријуме
PASS *password*	Лозинка на серверу
PORT *n1, n2, n3, n4, n5, n6*	Клијентова IP адреса (*n1, n2, n3, n4*) и PORT (*n*5 x 256 + *n*6)
QUIT	Одјављивање[2] са сервера
RETR *filename*	Нађи (узми) датотеку
STOP *filename*	Смести (стави) датотеку
SYST	Сервер враћа тип система
TYPE *type*	Специфицира тип датотеке: слово „A“ за ASCII, слово „I“ за слику
USER *username*	Корисничко име на серверу

Табела 10.1 Уобичајене FTP команде

[1] *Login sequenceand authentication*

[2] *Logoff*

[3] SMTP (*Simple Mail Transfer Protocol*) протокол - користи исту конвенцију за наредбе и одговоре.

[4] *Error message*

Наредба	Опис
1yz	Позитиван прелиминарни одговор. Активност је започета , али се очекује још један одговор пре слања друге наредбе
2yz	Позитивно завршен одговор. Може се послати нова наредба
3yz	Позитиван међуодговор. Наредба је прихваћена, али мора бити послата још једна команда
4yz	Привремени негативни одговор о завршетку. Тражена активност није одрађена, али стање грешке је привремено, па се наредба може поновити касније
5yz	Трајни негативни одговор о завршетку. Наредба није прихваћена и не би је требало понављати
x0z	Грешке у синтакси
x1z	Информације
x2z	Веза. Одговори који се односе на управљачку везу или везу података
x3z	Аутентификација и вођење евиденције о активностима појединачних корисничких налога . Одговори за наредбе приликом пријављивања или вођења евиденције
x4z	Није специфицирано
x5z	Статус система датотека

Табела 10.2 Значења прве и друге цифре троцифрено кодираног одговора

- 214 - помоћна порука - HELP (за људе),
- 220 - сервер је спреман да пружи услугу новом кориснику,
- 230 - корисник је пријављен, може се наставити са даљом процедуром,
- 226 - затварање везе података. Захтевана активност над датотеком је успешна, (на пример пренос датотеке),
- 227 - улазак у пасивни режим рада
- 331 - име корисника (*username*) је у реду, потребна је лозинка (*password*),
- 425 - веза за податке не може се отворити,
- 452 - грешка у исписивању датотеке,
- 500 - грешка у синтакси (наредба није препозната),
- 501 - грешка у синтакси (неважећи аргументи).

Свака FTP наредба даје једноредни одговор. На пример, наредба QUIT даје као одговор: „221 *Goodbye*“.

10.3 Репрезентација података у преносу FTP протоколом

Приликом пројектовања FTP протокола претпоставка је да су датотеке објекти у рачунарској масовној меморији који имају заједничке карактеристике независно од врсте рачунара. Датотеке имају симболичка имена која омогућавају да се могу јединствено идентификовати у оквиру одређеног система датотека или директоријума. Оне имају свог власника и заштиту од неовлашћеног приступа или могућности да неко други (осим власника) датотеку мења. Датотека се може направити, очитати из ње, копирати из ње, у њу уписати или се може обрисати. Полазећи од овог једноставног концепта FTP протокол успева да подржи пренос датотека између различитих рачунара који се налазе на различитим мрежама.

Да би остварио пренос између различитих рачунара и оперативних система FTP протокол обезбеђује механизам за преговарање о опцијама преноса и складиштења у четири димензије:

- врста (тип) датотеке,
- управљање форматом,
- структуром и
- начином (режимом) преноса.

10.3.1 Врсте датотека

Постоје следеће врсте датотека[1] који се преносе FTP везама:

- ASCII датотеке (подразумеване). Текстуална датотека преноси се везом за податке као мрежни виртуелни терминал (NVT[2]). То захтева од пошиљаоца да конвертује локалну текстуалну датотеку у NVT ASCII датотеку, а од примаоца да конвертује NVT ASCII датотеку у локалну врсту текстуалне датотеке. Крај сваког реда преноси се помоћу NVT ASCII репрезентације крај реда[3] и прелазак у нови ред[4]. То значи да прималац мора да скенира сваки бајт тражећи пар: крај реда и прелазак у нови ред;

[1] *File-type*
[2] *Network Virtual Terminal*
[3] *Carriage return*
[4] *Linefeed*

- EBCDIC[1] датотеке. Алтернативни начин за слање текстуалних датотека када су оба краја EBCDIC системи.
- *Image* датотека. Означава се и као бинарна врста датотека. Подаци се шаљу као један повезан ток битова. Нормално се користи за пренос бинарних датотека. Овај начин користан је код размене произвољних датотека између два рачунара истог типа и под истим оперативним системом. Преносом је обезбеђено да је датотека на одредишном рачунару бит по бит копија датотеке са изворишног рачунара.
- *Local* датотеке. Овај начин користи се за пренос датотека код којих су подаци смештени у логичким бајтовима који садрже број битова различит од 8. Ова врста преноса заједно са начином на који се подаци структурирају омогућава да се подаци смештају на одредишни рачунар на начин конзистентан начину смештања на изворишном (локалном) рачунару.

10.3.2 Управљање форматом

Управљање форматом односи се само на ASCII и EBCDIC врсте датотека. Постоје три опције:

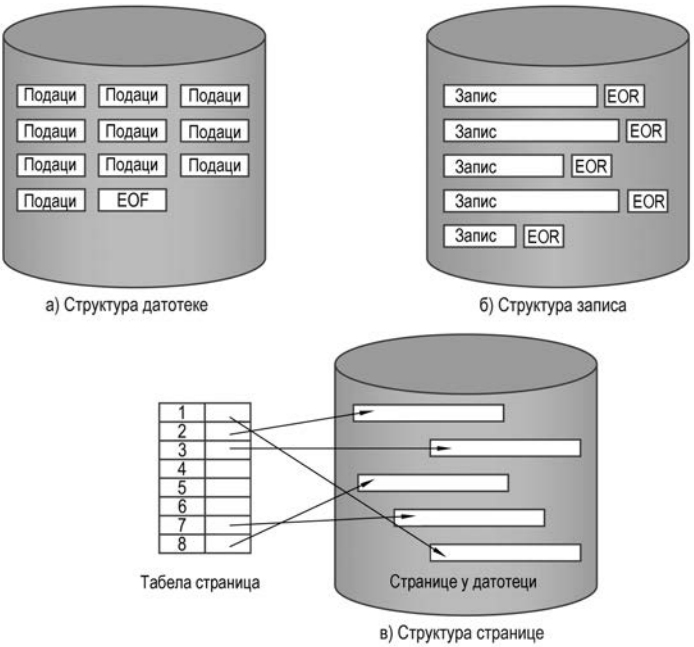
- *Nonprint* (није предвиђено штампање). Датотека не садржи информације о вертикалном форматирању. Одговара датотекама за које није предвиђено штампање на линијском штампачу. Ово је подразумевана опција;
- *Telnet* форматирање. Датотека садржи карактере за вертикално форматирање које интерпретирају штампачи (крај реда, прелазак у нови ред,...). Исте датотеке су и у спецификацији за Telnet протокол.
- *Carriage control/Fortran* форматирање. Користи конвенције у форматирању програмског језика *Fortran*: први карактер сваког реда је управљачки карактер програмског језика *Fortran*.

10.3.3 Структура података

Дефинисане су следеће структуре података:

- *File structure* (подразумеван). Датотека је скуп бајтова (дефинисан

[1] *Extended Binary Coded Decimal Interchange Code*



Слика 10.2 FTP врсте датотека

опцијом врста датотеке) који се завршава ознаком о крају датотеке (слика 10.2а);

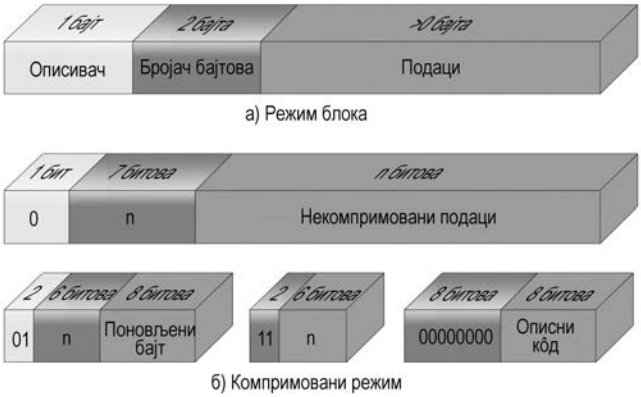
- *Record structure*. Датотека се састоји од скупа узастопних записа, одвојених ознаком о крају записа (EOR[1]) (слика 10.2б).
- *Page structure*. Датотека се састоји од скупа специјално индексираних страница. Документ RFC1123 не препоручује примену ове структуре (слика 10.2в).

10.3.4 Начин преноса

FTP дефинише три различите врсте преноса који одређују како се датотеке преносе везом за податке:

- *Stream mode* (подразумевани) - датотека се шаље као ток бајтова. Када је датотека структуре File structure онда на њен крај указује окончавање веза података. Када је структура датотеке Record

[1] *End of Record*



Слика 10.3 FTP формати врста (режима) преноса

structure онда се двобајтни управљачки кôд користи да укаже на крај записа и крај датотеке;

- *Block mode* - датотека се шаље као серија блокова и сваком блоку претходи једно или више заглавља (слика 10.2а).

Заглавље чини описни кôд[1] и бројач бајтова[2] (табела 10.3):

- *Compressed mode*. Омогућава да се побољша ефикасност преноса дозвољавајући извориштy да секвенцу која се понавља кодира крађом секвенцом[5] (слика 10.3б).На пример у текстуалним датотекама се компримују низови празних места а у бинарним датотекама низови са нула бајтова. Овај начин се

Описни кôд	Значење
128	Крај блока података је крај записа (EOR[1])
64	Крај блока података је крај (EOF[2])
32	Постоји могућа грешка у блоку података
16	Блок података је ознака о поновном слању
0	Не шаље се никакав описни кôд

Табела 10.3 Описни кодови и њихово значење у преносу *Block mode*

[1] *Descriptor Field*
[2] *Byte Count*
[3] *End-Of-Record*
[4] *End Of File*
[5] *Run length coding*

ретко користи пошто данас постоје бројне, ефикасније методе за компресију података.

10.4 Управљање везом

Веза података користи се на три начина:

- слање датотеке од клијента до сервера,
- слање датотеке од сервера до клијента и
- слање списка датотека или директоријума од сервера до клијента.

FTP сервер радије шаље списак датотека користећи везу података него већи број договора користећи управљачку везу. Тиме се избегавају ограничења у величини списка директоријума и омогућава клијенту да смести списак директоријума у датотеку, уместо приказ списка на екрану.

Већ смо нагласили да управљачка веза остаје отворена за све време трајања комуникације између клијента и сервера. За то време веза података може се више пута успостављати и раскидати (затварати) по потреби.

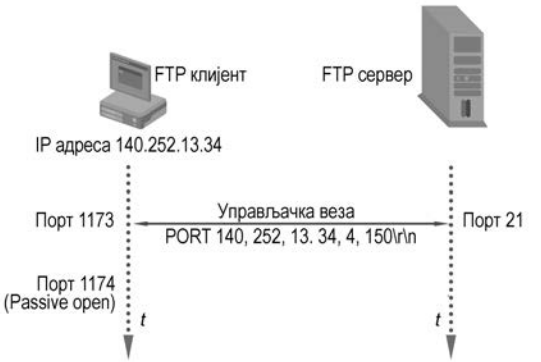
Постављају се следећа питања:

- Како се бројеви портова бирају за везу података?
- Ко извршава активно а ко пасивно отварање везе?

Као што смо напоменули уобичајен начин преноса је ток (*Stream mode*) и крај датотеке (EOF) који је одређен затварањем везе података. То значи да се нова веза података захтева за пренос сваке датотеке или прегледање директоријума.

Нормална процедура преноса (описана у полазном документу RFC959) је следећа:

1. Успостављањем везе података управља клијент пошто је клијент тај који поставља команду која захтева везу података (узми датотеку, постави датотеку, прегледај директоријуме,...).
2. Клијент бира ефемерални број порта за клијентску страну везе података (подразумевана опција, не и једина). Клијент захтева пасивно отварање везе преко овог порта.
3. Клијент шаље овај број порта преко управљачке везе користећи PORT команду.



Слика 10.4 Пренос команде PORT кроз управљачку везу

4. Сервер прима број порта преко управљачке везе и захтева активно отварање везе ка том порту клијента. Страна сервера везе података увек користи порт 20.

На слици 10.4 приказано је стање везе у току извођење прва три корака. Претпоставља се да је ефемерални порт клијента за управљачку везу 1173 а за везу података 1174. Клијент шаље наредбу PORT и и његови аргументи су шест децималних бројева ASCII кодираних одвојених запетом. Прва четири броја специфицирају IP адресу на страни клијента ка којој ће сервер тражити активно отварање (нпр. 140.252.13.34) и следећа два броја специфицирају 16-битни број порта. Пошто је 16-битни број порта сачињен од два броја у нашем примеру његова вредност је 4*256+150=1174.

Слика 10.5 FTP сервер реализује активно отварање везе података

Слика 10.5 показује стање везе када сервер захтева активно отварање (*Active Open*) ка клијенту код везе података. Серверска страна везе користи порт број 20. Сервер реализује активно отварање везе података. Уобичајено сервер одрађује и активно затварање везе података, осим када клијент шаље податке серверу на *Stream* начин. Тада се захтева да клијент затвори (оконча) везу.

10.5 FTP пријављивање и провера идентитета

Нормална процедура пријављивања и провере идентитета (аутентификација) једноставна је: корисник се идентификује тако што интерпретер протокола корисника шаље ка интерпретеру протокола сервера корисничко име користећи команду USER (слика 10.6). После тога шаље се корисничка шифра користећи наредбу PASS. Сервер

Слика10.6 Успостављање везе и провера идентитета корисника

проверава име корисника и шифру у својој бази података да би потврдио да корисник има право да приступи серверу. Уколико је све у реду, сервер то потврђује. Уколико није сервер шаље поруку о грешци а после одређеног броја неуспелих пријава сервер може да оконча везу. Описани метод обезбеђује минималну сигурност, тако да ако се захтева већа сигурност, мора се користити сигурносно проширење FTP протокола или неке друге методе.

10.6 Сигурносно проширење FTP протокола

Као и други старији протоколи, једноставан начин пријављивања који користи FTP протокол је одговарао релативно затвореној природи раног Интернета. По данашњим сигурносним стандардима глобалног Интернета FTP протокол се не сматра сигурним пошто се корисничко име и лозинка шаљу преко управљачке везе као отворен (чист) текст. На тај начин веома је једноставно „пресрести“ везу у међусистемима и преузети информације о налогу.

У документима RFC2228 и RFC4217 дефинисане су опције за безбедniју употребу FTP протокола. Ово се постиже употребом TLS[1] протокола односно његовог претходника SSL[2] протокола. Протоколи TLS и SSL генерално могу да омогуће стандардан начин заштите приватности и интегритета порука које апликације размењују преко мреже. Ослањају се на дигиталне сертификате и инфраструктуру јавних кључева за аутентификацију страна у комуникацији као и на низ алгоритама за симетрично шифровање којима се могу заштитити управљачка веза (која се између осталог користи и за аутентификацију клијента слањем лозинке) као и веза за пренос података. Овакав заштићен FTP се назива и сигурни FTP (FTPS[3]).

У пракси се иницирање договора око параметара заштите постиже додатном FTP командом (AUTH[4]) која се размени преко управљачке везе, која се као и до сада успоставља на TCP порту 21. Ово се

[1] *Transport Layer Security*

[2] *Secure Socket Layer*

[3] *FTP-Secure*

[4] Аутентификација/механизам сигурности. Једна од додатних команди дефинисаних у документу RFC2228 (у односу на команде специфициране у документу RFC959).

зове и експлицитни FTPS јер клијент и сервер експлицитно, путем AUTH команде показују способност и спремност за TLS договор око заштите. У току TLS договора изврши се аутентификација дигиталним сертификатима, договор око кључева и алгоритама за шифровање каснијег саобраћаја.

Одговорајући документи предвиђају и имплицитни FTPS протокол, код кога би након успоставе TCP сесије одмах био покренут TLS договор, без провере да ли обе стране тако нешто подржавају. Како би се спречили проблеми са некомпатибилним клијентима, сервер би у том случају очекивао успоставу управљачке везе на алтернативном порту 990 (уместо 21). Употреба алтернативних портова због заштите се више не препоручује и имплицитни TLS протокол се ређе имплементира.

FTPS протокол омогућава задовољавајући ниво заштите приватности и интегритета, али уз додатни напор потребан за издавање дигиталних сертификата и креирање инфраструктуре јавних кључева. Додатно, шифровање комуникације спречава заштитну баријеру или NAT рутер да разуме PORT команде које се размењују између клијента и сервера што спречава комуникацију или додатно компликује конфигурацију мрежне заштите.

У време писања овог уџбеника често се за безбедно копирање користи SFTP[1] протокол. Протокол SFTP је проширење SSH[2] протокола које омогућава безбедну удаљену администрацију и покретање команди на удаљеним системима. Протокол SSH данас безбедно замењује *Telnet* протокол и користи се за администрацију LINUX/UNIX оперативних система или мрежних уређаја. Протокол SFTP користи механизме заштите SSH протокола и за копирање датотека међу системима.

10.7 Анонимни FTP сервери

Изненађујуће је да многе организације нису виделе потребу за унапређењем сигурности. Напротив: користе се FTP сервери без икакве ауторизације. Разлог је следећи: сервер се користи да би обезбедио информације општој популацији.

[1] *Secure Shell File Transfer Protocol*

[2] *Secure Shell*

Данас многе организације користе веб[1] за дистрибуцију докумената, софтвера и других датотека потребних корисницима. Али осамдесетих година, пре него што је веб постао популаран најчешће се то радило користећи FTP протокол.

Јасно је да је неприхватљиво да се захтева да сваки корисник има корисничко име и лозинку на таквом серверу. Због тога је 1994. године дефинисано коришћење протокола који се назива анонимни FTP протокол.

Користећи овај приступ клијент се повезује са сервером и добија подразумевано корисничко име и лозинку да се пријави као гост[2]. Обично се као корисничко име користи „*anonymous*“ или „*ftp*“. Сервер одговара неком од поздравних порука и као лозинку (обично) тражи е-адресу корисника. Лозинка у овом случају у ствари и није права лозинка већ је податак који сервер користи за статистичку обраду о корисницима који су се пријавили на сервер.

10.8 Активна и пасивна FTP веза података

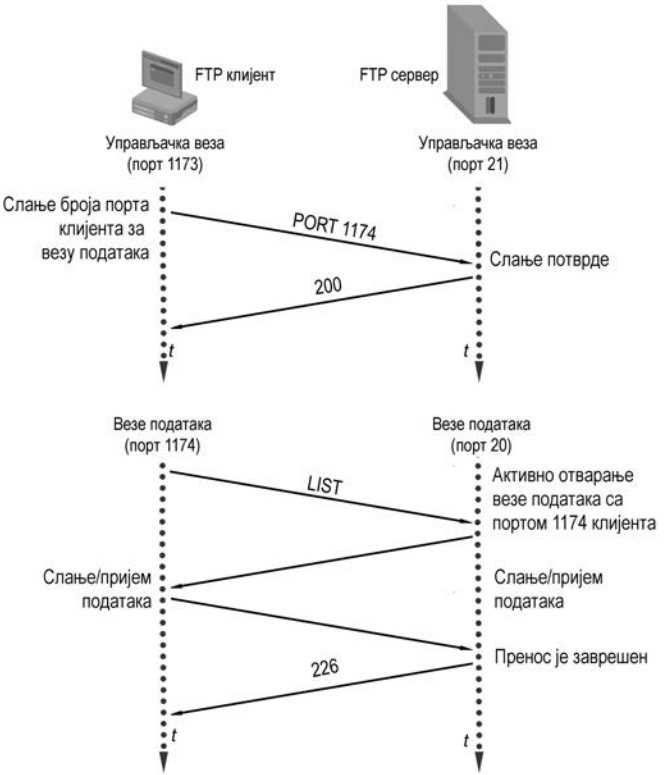
Метод који смо у претходном одељку анализирали и који је представљен сликом 10.6 означава се као нормалан или активан начин успоставе везе података за разлику од пасивног начина који ће бити описан у овом одељку. У нормалном, активном начину, сервер (тј. његов процес за пренос података) иницира успостављање везе података отварањем TCP везе са корисником (тј. његовим процесом за пренос података). Клијент користи ефемерални број порта који може (али се не препоручује) да буде исти као ефемерални број порта управљачке везе. Обично клијент бира различит порт код сваког преноса.

Претпоставимо да је интерпретер протокола корисника успоставио управљачку везу са свог ефемералног порта број 1173 са FTP управљачким портом сервера 21. Наредба PORT претходи преносу података. Процес за пренос података сервера ће успоставити везу са свог порта 20 са портом 1174 клијента. Овај процес приказан је на слици 10.7.

---

[1] *World Wide Web*

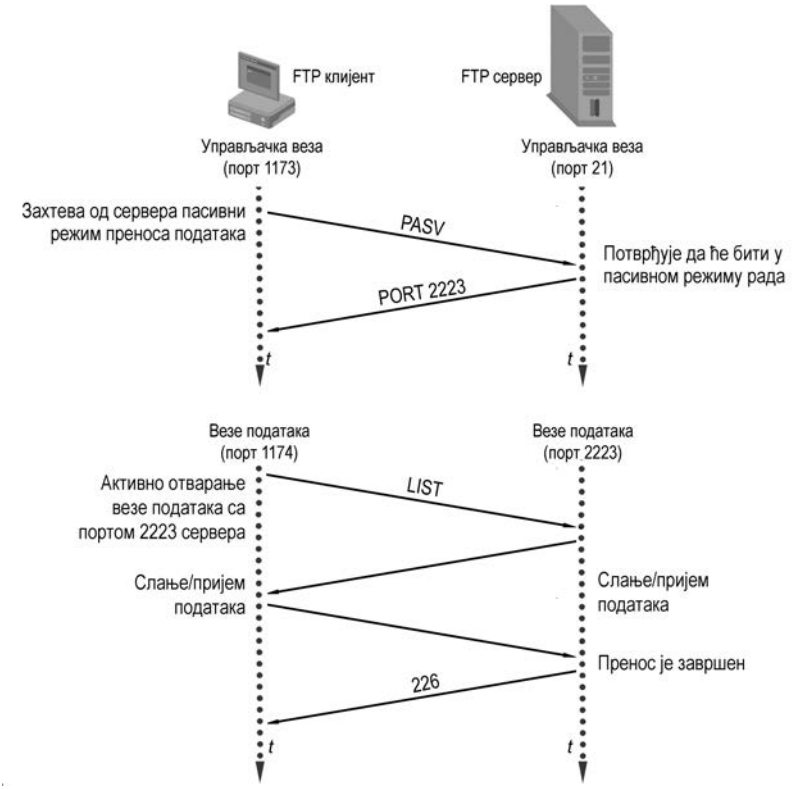
[2] *Guest*



Слика 10.7 Активно отварање FTP везе

Други метод означава се као пасивна веза података. Клијент налаже серверу да буде „пасиван“: да прихвата иницирање везе за успоставом података од клијента. Сервер одговара шаљући клијенту IP адресу сервера и број порта који ће користити. Процес за пренос података сервера ослушкује на том порту на долазећу TCP везу од процеса за пренос података клијента. Подразумева се да се у управљачкој вези користе исти бројеви портова као и код активног начина. Клијент може да изабере различит број порта за везу података уколико је то потребно (обично ефемерални бројеви портова).

Анализирајмо поново овај пример (слика 10.8) али се за пренос података сада користи пасиван начин. Управљачка веза је између порта број 1173 на страни клијента и порта број 21 на страни сервера. Клијент



Слика 10.8 Пасивно отварање FTP везе

шаље PASV команду којом указује серверу да жели да сервер користи пасивно управљање преносом података. Интерпретер протокола сервера ће наложити процесу за пренос података сервера да ослушкује на порту 2223. Интерпретер протокола корисника ће наложити процесу за пренос података корисника да успостави везу између порта 1174 клијента и порта 2223 сервера. Сервер то потврђује и подаци се могу поново у оба смера слати и примати.

10.9 Питања и задаци

1. Како се могу категоризовати апликације за пренос датотека и порука између међусобно умрежених уређаја?

2. Каква је разлика између директних и индиректних мрежних апликација?
3. У TCP/IP мрежним апликацијама навести апликације које одговарају подели на директне и индиректне.
4. Описати захтеве који су постављени пред FTP протокол.
5. Колико TCP веза користи FTP протокол? Како се означавају?
6. Описати начин успостављања и намену управљачке везе.
7. Описати начин успостављања и намену везе података.
8. Набројати и описати компоненте FTP процеса на страни сервера приказане на слици 10.1.
9. Набројати и описати компоненте FTP процеса на страни клијента приказане на слици 10.1.
10. Какву функцију има FTP команда PORT? Какво значење имају параметри које преноси?
11. Које су врсте датотека дефинисане стандардом за FTP системе?
12. Које су структуре података дефинисане стандардом за FTP системе?
13. Описати начине преноса везом података.
14. Како се бројеви портова бирају за везу података?
15. Ко извршава активно а ко пасивно отварање везе?
16. Описати нормалну процедуру преноса (описана у полазном документу RFC959).
17. Шта се подразумева под анонимним FTP сервером?
18. Како се реализује FTP пријављивање и провера идентитета?
19. Навести нека од сигурносних проширења FTP протокола.
20. Описати разлику између активне и пасивне везе података.

11. Протоколи за управљање у рачунарским мрежама

Како број рачунарских мрежа у оквиру једне организације расте заједно са разнородним системима који омогућавају њихово међусобно повезивање и повезивање на Интернет (рутери различитих произвођача, сервери, радне станице...) управљање овим системима постаје веома значајно. Постаје евидентно да:

- сама мрежа, њени ресурси и дистрибуиране апликације бивају од непроцењиве вредности за организацију,
- више ствари може да закаже као што је искључивање мреже или неког њеног дела као и деградирање перформанси до неприхватљивог нивоа.

Велике мреже не могу бити груписане и администриране само од стране људи. Комплексност таквих система диктира коришћење апликација за аутоматизовано управљање рачунарским мрежама. Потражња за таквим апликацијама се повећава, али такође постаје и све теже испоручити такве апликације, поготову ако се мрежа састоји од опреме различитих произвођача. Све већа децентрализација мрежних услуга уз све већи значај радних станица и клијент/сервер система чини кохерентну и координирану администрацију рачунарских мрежа све тежом. У тако комплексним информационим системима многе битне ставке у мрежи су ван контроле њеног администратора.

11.1 Надгледање и управљање мрежом

Интернационална организација за стандардизацију (ISO) дефинисала је области које обухвата надзор и управљање мрежом[1]:

[1] *Network management model*

- Надзор и управљање перформансама[1]. Циљ управљања перформансама је квантификовање, мерење, извештавање, анализа и управљање перформансама различитих елемената мреже (нпр. коришћење пропусног опсега). Елементи мреже су уређаји (нпр. линкови, рутери и крајње станице) као и апстракције као што је путања кроз мрежу. Протокол за управљање мрежом (SNMP[2]) има главну улогу у управљању перформансама на Интернету.
- Надзор и управљање грешкама[3]. Циљ управљања грешкама је да се запише, детектује и одговори на грешке у мрежи. Линија између надзора и управљања грешкама и перформансама је прилично невидљива. Разлика би могла да се дефинише на следећи начин:
  - надзор и управљање грешкама је тренутно вођење рачуна о отказима као што су прекид рада линка, крајње станице, рутера, или комуникационог софтвера,
  - надзор и управљање перформансама захтева дуже посматрање како би се обезбедио прихватљив ниво перформанси у случају променљиве густине саобраћаја и повремених отказа мрежних уређаја.
- Надзор и управљање конфигурацијама[4]. Дозвољава менаџеру мреже да прати који уређаји јесу на мрежи која се надзире и управља софтверску и хардверску конфигурацију тих уређаја[5].
- Надзор и управљање налозима[6]. Пружа могућност мрежном менаџеру да специфицира, запише и управља приступом мрежи корисника и мрежних уређаја. У надзор и управљање налозима спадају корисничке квоте, наплата и додела права приступа ресурсима.
- Надзор и урављање сигурношћу[7]. Циљ надзора и управљања сигурношћу је да управља приступом мрежним ресурсима а према

[1] *Performance management*
[2] *Simple Network Management Protocol*
[3] *Fault management*
[4] *Configuration management*
[5] Детаљнији опис за мреже са IP протоколом може се наћи у документу RFC3139.
[6] *Accounting management*
[7] *Security management*

унапред дефинисаној политици. Употреба заштитних зидова[1] и надгледање и управљање спољашњим приступним тачкама неке мреже пример су надзора и управљања сигурношћу.

У овом поглављу биће анализиране само основе надзора и управљања мрежом а то је инфраструктура за надзор и управљање мрежом. Она подразумева свеобухватну архитектуру, протоколе за надзор и управљање мрежом и базу информација коју користи администратор мреже.

11.2 Систем за управљање мрежом

Систем за управљање мрежом је колекција алата за надгледање и управљање мрежом:

- један интерфејс са моћним, али лаким за коришћење, скупом команди за извршавање већине управљачких задатака,
- минимална количина додатне опреме тј., већина хардвера и софтвера потребних за управљање мрежом укључено је у постојећу корисничку опрему.

Систем за управљање мрежом састоји се од надоградивог додатног хардвера и софтвера имплементираног у постојеће мрежне компоненте. Софтвер који се користи за извршавање задатака управљања мрежом налази се у крајњим станицама и комуникационој опреми (нпр. комутаторима, рутерима…). Систем за управљање мрежом пројектован је тако да види мрежу као јединствену архитектуру са адресама и ознакама које су додељене свакој тачки и специфичним карактеристикама сваког елемента и сваке везе за коју систем зна. Активни елементи у мрежи редовно обезбеђују статусне информације управљачком центру мреже.

11.3 Протокол за управљање мрежом SNMPv1

Протокол за управљање мрежом (SNMP[2]) развијен је као алат за управљање мрежама и међусобно повезаним мрежама које користе TCP/IP протокол. Касније је његова примена проширена и на сва остала мрежна окружења. Назив протокол за управљање мрежом указује на скуп

[1] *Firewall*

[2] *Simple Network Management Protocol*

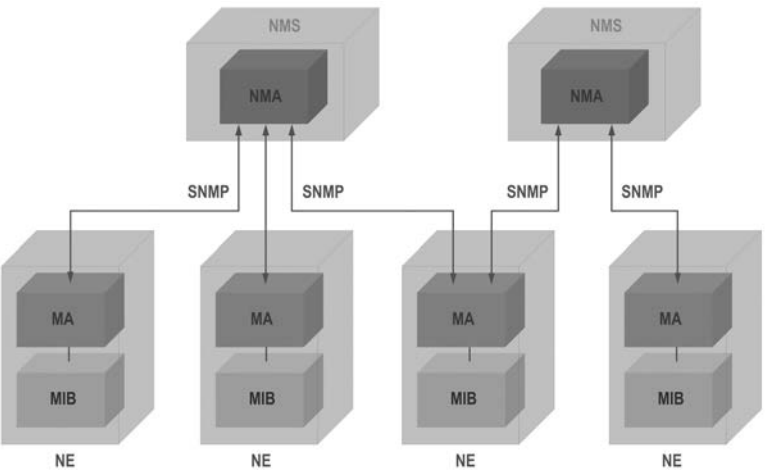
спецификација за управљање мрежом који укључује и сам протокол као и дефиницију и концепт базе података.

Систем управљања мрежом (слика 11.1) обухвата следеће кључне елементе:

- управљачка станица (NMS[1]), менаџер - станица,
- апликација за управљање (NMA[2]),
- елеменат рачунарске мреже (NE[3]),
- управљачки агент (MA[4]), или само агент,
- база података за управљање (MIB[5] база),
- протокол за управљање мрежом (SNMP протокол).

Комуникација може бити двострана:

- Менаџер тражи од агента одговор о специфичној променљивој. На пример, колико је ICMP порука типа „порт је недоступан" послао;
- Агент обавештава менаџера да се нешто важно десило. На пример „мрежни интерфејс је у квару";



Слика 11.1 Компоненте система за управљање мрежом

[1] *Network Management Station*
[2] *Network Management Application*
[3] *Network Element*
[4] *Management Agent*
[5] *Management Information Base*

- Менаџер може да очита или постави вредност неке променљиве код агента. На пример „промени подразумевану вредност поља TTL у IP заглављу на 64”.

Управљачка станица обично је посебан, самосталан уређај али се може реализавати и као део другог система. У оба случаја управљачка станица служи као интерфејс човека-менаџера и управљачког система мреже. Управљачка станица треба да садржи:

- скуп управљачких апликација које се користе за анализу података, опоравак од грешака итд.,
- интерфејс преко кога менаџер надзире и управља мрежом,
- могућност спровођења захтева менаџера мреже у процесу надзора и управљања удаљених елемената мреже,
- базу података са информацијама о управљању мрежом добијеним из база свих управљивих целина у мрежи.

Само последња два елемента су предмет SNMP стандардизације.

Други активни елемент у систему управљања мрежом је управљачки агент[1]. Кључни уређаји као што су станица, мост, комутатор и рутер могу бити опремљени софтвером агената тако да њима може управљати станица - менаџер. Агент шаље управљачкој станици захтеване информације, извршава њене захтеве за управљањем а може и асинхроно снабдевати управљачку станицу информацијама које она није тражила.

Да би се управљало ресурсима у мрежи сваки ресурс или његов део представљен је као мрежни управљачки објекат (MIB објекат). За објекат се може рећи да је то променљива која представља један аспекат управљачког агента. Управљачке информације[2] су представљене као скуп управљивих објеката који заједно сачињавају виртуелно информационо складиште познато као база података за управљање (MIB база).

Управљачка станица извршава функцију надгледања мреже тако што узима податке о MIB објектима. Управљачка станица може да управља агентима или може да промени конфигурационе параметре код агента тако што ће променити вредности одређених променљивих. Управљачка станица и агенти међусобно су везани протоколом за управљање мрежом.

---

[1] *Management agent*

[2] *Management information*

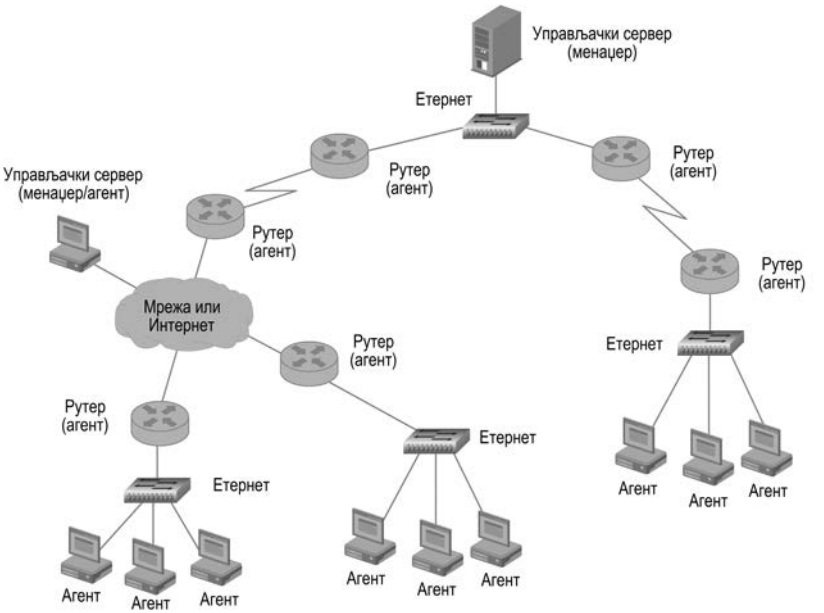
Протокол који се користи за управљање у рачунарским мрежама са TCP/IP протоколом је (једноставан) протокол за управљање мрежом: SNMP протокол. Побољшана верзија овог протокола означена је са SNMPv2 и намењена је мрежама са TCP/IP и OSI архитектурама протокола. Сваки од ових протокола може да користи следеће операције:

- узми (*Get*) - омогућава управљачкој станици да добије вредности објеката код агената,
- постави (*Set*) - омогућава управљачкој станици да подеси вредности објеката код агената,
- обавести (*Notify*) - омогућава агенту да пошаље информацију управљачком систему о значајним догађајима.

У традиционално централизованој шеми управљања мрежом једна крајња станица у конфигурацији има улогу мрежне управљачке станице; може да постоји још једна, или две управљачке станице које имају улогу резервних станица[1]. Други уређај садржи софтвер агента и MIB базу како би омогућио надзор и управљање станици - менаџеру. Како мрежа расте по величини и протоку података таквом централизованом систему све више постаје отежан рад: превише захтева се поставља пред управљачку станицу и превелик је проток података. Извештаји од сваког агента треба да пронађу свој пут кроз мрежу до главне станице. У таквим условима децентрализовани, дистрибуирани приступ, показао се као боље решење (слика 11.2).

У децентрализованој шеми управљања мрежом налази се више управљачких станица највишег нивоа[2] које се могу назвати и управљачки сервери[3]. Сваки такав сервер може директно да управља одређеним бројем агената.Ипак, за многе агенте управљачки сервер даје задужења менаџеру средњег нивоа[4]. Менаџер средњег нивоа надгледа и контролише само оне агенте који су у његовој надлежности. Он такође има улогу и агента који обезбеђује информације и прихвата захтеве за управљањем од сервера - менаџера вишег нивоа. Оваква архитектура распоређује оптерећење и смањује укупан проток у мрежи.

[1] *Backup*
[2] *Top Level*
[3] *Management servers*
[4] *Intermediate manager*



Слика 11.2 Пример дистрибуиране конфигурације за управљање мрежом

11.4 Архитектура протокола за управљање мрежом[1]

Протокол SNMP припада апликационом слоју и део је скупа TCP/IP протокола. Предвиђен је за рад са UDP протоколом. Слика 11.3 приказује типичну конфигурацију SNMPv1 протокола.

За самосталну управљачку станицу управљачки процес контролише приступ централној бази података (MIB бази) на страни управљачке станице и обезбеђује интерфејс према менаџеру мреже. Управљачки процес реализује управљање мрежом користећи SNMP протокол, који је постављен на самом врху изнад UDP, IP протокола и осталих протокола као што су Етернет, ATM[2] или штафетни пренос[3] рамова (протокола за приступ мрежи).

Сваки агент мора да имплементира SNMP, UDP и IP протоколе. Као додатак постоји процес агента који интерпретира SNMP поруке и

[1] *Network Management Protocol Arhitetcture*
[2] *Asynchronous Transfer Mode*
[3] *Frame relay*

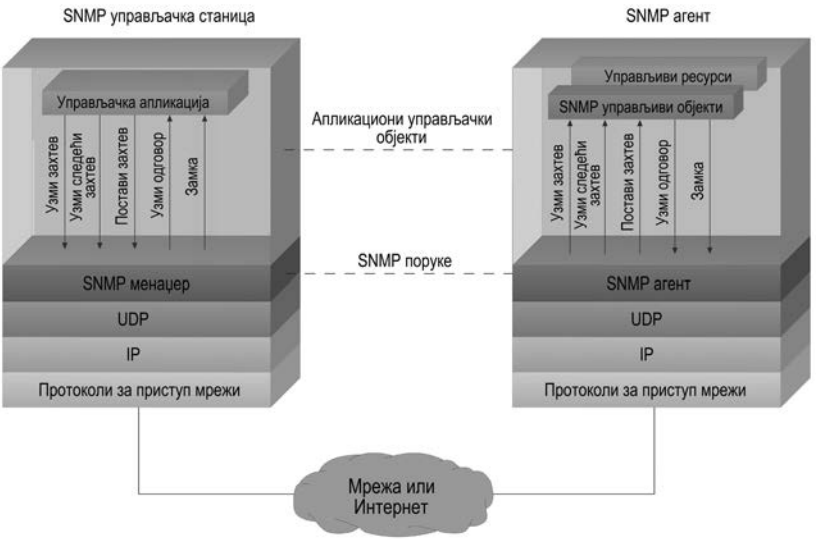
контролише MIB базу података агента. Подршка протоколима као што су FTP, TCP као и UDP протоколи такође је потребна и код агената. На слици 11.3 тамније осенчени делови приказују радно окружење којим ће се управљати. Светлије осенчени делови омогућавају подршку функцијама управљања мрежом.

Слика 11.4 приказује детаљније SNMP протокол. Апликација за управљање може да изда три врсте SNMP порука:

- узми захтев (*Get Request*),
- узми следећи захтев (*GetNextRequest*) и
- постави захтев (*SetRequest*).

Прве две су две варијације функције узми (*Get*). Све три поруке су потврђене на страни агента у облику поруке узми одговор (*GetResponse*), која је прослеђена управљачкој апликацији. Као додатак агент може да изда поруку (*Traps*[1]) као одговор на активност која утиче на MIB базу и остале управљиве ресурсе. Управљачки захтеви се шаљу на UDP порт 161 док агенти шаљу поруку-замку на UDP порт 162.

Протокол SNMP ослања се на UDP порт који је протокол без



Слика 11.4 Улога SNMPv1 протокола

[1] *Замка*

успоставе везе па је и сам SNMP протокол без успоставе везе. Пошто се не успоставља веза између управљачке станице и агената свака размена је посебна трансакција.

11.5 Протокол за управљање мрежом верзија SNMPv2

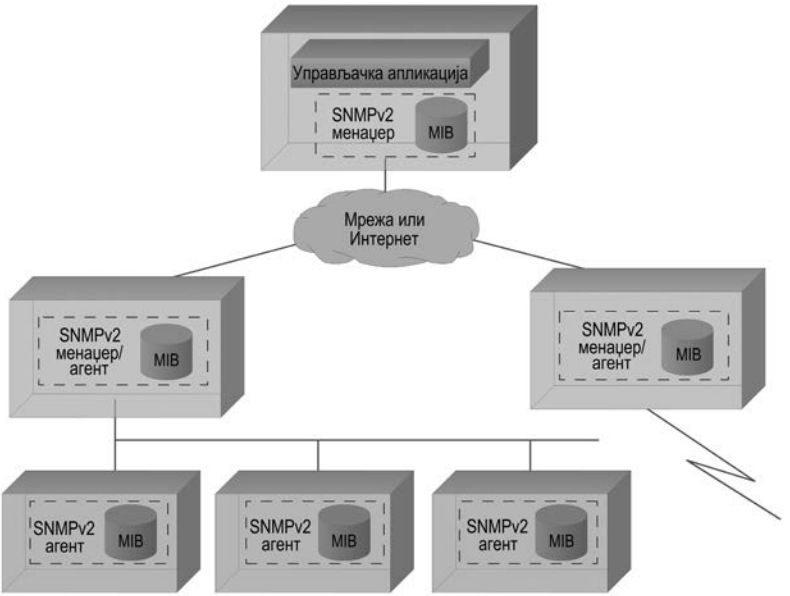
Августа 1988. год. издата је спецификација SNMP протокола и убрзо је постао доминантан стандард за управљање мрежом. Многи произвођачи понудили су самосталне радне станице за управљање мрежом базиране на SNMP протоколу и већина произвођача мостова, рутера, радних станица и персоналних рачунара понудила је софтверске пакете за SNMP агенте који омогућавају њиховим производима да их контролише SNMP управљачка станица.

Као што и само име указује SNMP алат је једноставан начин за управљање мрежом. Стандардом је дефинисана применљива база података за управљању мрежом (MIB база) која је састављена од скаларних променљивих и дводимензионалних табела. Дефинисан је и протокол који омогућава менаџеру да прибави и подеси MIB променљиву и да омогући агенту да пошаље обавештење - замку. Једноставност је један од највећих обележја SNMP протокола. Лако се имплементира, не оптерећује много ни процесор ни мрежне ресурсе.

Структура протокола и MIB база довољно је отворена да није тешко постићи компатибилност између управљачких станица и агент софтвера различитих произвођача.

Распрострањеним коришћењем постали су недостаци SNMP протокола очигледни. Они се односе и на начин рада и на механизме заштите. Резултат је био стандардизација усавршене верзије протокола[1] - SNMPv2. Произвођачи опреме су нову верзију SNMPv2 брзо прихватили. Верзија SNMPv2 не обезбеђује само протокол за управљање мрежом већ оквир у коме се апликације за управљње мрежом могу направити. Апликације, као што су управљање грешкама, надгледање перформанси, наплата итд. су ван опсега овог стандарда. Стандард SNMPv2 дефинише инфраструктуру за управљање мрежом. Слика 11.5 пример је конфигурације која то илуструје.

[1] Описан у документима RFC1901, RFC1905-RFC1909, RFC2578-RFC2580



Слика 11.5 SNMPv2 конфигурација за управљање

Суштина SNMPv2 је протокол који омогућава размену управљачких информација. Сваки учесник[1] у систему за управљање мрежом одржава локалну базу података са информацијама релевантним за управљање мрежом (MIB базу). У оквиру SNMPv2 протокола дефинисана је структура информација MIB базе (SMI[2] структура) и дозвољени типови података. SMI структуру можемо да опишемо као језик за дефинисање управљачких информација. Стандард такође обезбеђује већи број MIB врста[3] које су генерално веома корисне за управљање мрежом. Као допуна нове MIB врсте могу се дефинисати произвођачи или групе корисника.

Потребно је да један систем у конфигурацији мора бити одговоран за управљање мрежом. Може бити више управљачких станица како би

[1] *Player*

[2] *Structure of Management Information*

[3] Термин MIB може се користити на више начина. Када се користи у једнини може да означи базу података о информацијама за управљање на страни менаџера или на страни агента. Такође се може користи у једнини или множини када указује на специфичан скуп управљачких информација које су део свеобухватне базе података MIB. Тако SNMPv2 стандард у себи садржи дефиницију неколико MIB врста и укључује и MIB дефинисан верзијом SNMPv1.

се обезбедила редундантност или просто поделила задужења у великој мрежи. Већина осталих система има улогу агента. Агент прикупља информације локално и складишти их како би им касније приступио менаџер. Информације садрже податке о самом систему и могу садржати информације о мрежном протоку или мрежама на које је агент прикључен. SNMPv2 протокол подржава и високоцентрализовану стратегију управљања мрежом и дистрибуирану стратегију. Код дистрибуираних система неки елементи имају двоструку улогу: управљачке станице и агента. Када је у улози агента елемент ће прихватити команде од вишег управљачког система. Неке од ових команди везане су за локалну MIB базу агента.

Остале команде захтевају од агента да се понаша као посредник за удаљене уређаје. У таковом случају агент - посредник преузима улогу менаџера како би приступио информацијама код удаљеног агента, а преузима улогу агента како би проследио те информације вишем менаџеру.

За све горепоменуте размене користи се SNMPv2 протокола. Он је једноставан протокол типа захтев/одговор. Пошто се протоколом SNMPv2 размењују парови порука захтев/одговор стална, поуздана веза није потребна.

11.6 Структура управљачких информација

Структура управљачких информација (SMI структура) дефинише општи оквир унутар кога се дефинише и конструише MIB база података. Иако је заснована на ASN.1[1] нотацији, SMI структура идентификује типове података који се могу користити у MIB бази података као и на који начин су ресурси у њој представљени и означени. Основна концепција структуре управљачких информација је једноставност и могућност проширивања MIB базе података. Према томе, MIB база може да складишти само просте типове података: скаларе и дводимензионе матрице (табеле).

[1] *Abstract Syntax Notation One* (ISO X.680, 2002.). ASN.1 је стандардна и флексибилна нотација која описује правила и структуре за представљање, кодирање, пренос и декодирање података у комуникационим системима. ASN.1 је здружени стандард организација ISO (*International Organization for Standardization*), IEC (*International Electrotechnical Commission*) и ITU-T (*International Telecommunication Union Telecommunication Standardization Sector*).

Тип податка	Опис
INTEGER	Цели бројеви у распону од -2^{31} до 2^{31} -1, или вредности са листе могућих константних вредности
Integer32	Цели бројеви у распону од 0 до 2^{32} - 1
Counter32	32-битни бројач који се увећава од 0 до 2^{32} - 1 (позитивни цели бројеви који могу бити увећани по модулу 2^{32})
Counter64	32-битни бројач (позитивни цели бројеви који могу бити увећани по модулу 2^{64})
Gauge32	Позитивни цели бројеви који могу бити увећани или умањени, али не смеју прекорачити максималну вредност. Максимална вредност не може бити већа од 2^{32} – 1. MIB варијабла *tcpCurrEstab* је пример: то је број TCP веза које су тренутно у стању ESTABLISHED или CLOSE_WAIT.
TimTicks	Позитивни цели бројеви који репрезентују време по модулу 2^{32}. На пример варијабла *sysUpTime* показује време колико дуго је агент активан изражено у стотинама секунди.
OCTET STRING	ASN.1 формат. Низ бајтова за произвољне бинарне или текстуалне податке. Могу бити ограничени на 65 535 бајтова.
IPAddress	32-бита интернет адреса
PhysAddress	Одређује физичку адресу (нпр. 6 бајтова за етернет адресу).
Opaque	Неинтерпретирани ASN.1 низ
BIT STRING	Списак именованих битова
OBJECT IDENTIFIER	Идентификатор објекта. ASN.1 формат административно додељено име (тип податка) објекту или другом стандардизованом елементу. Вредност је низ величине до 128 позитивних целих бројева.

Табела 11.1 Дозвољени типови података у SNMPv2 протоколу

Структура управљачких информација не подржава стварање и проналажење комплексних структура података. Овакав принцип није у складу са оним који се користи код система за управљање у оквиру OSI окружења код кога се сложена функционалност постиже комплексним структурама података и сложеним претраживањем.

Коришћењем SMI структуре избегавају се комплексни типови података да би се поједноставила имплементација и рад са другим системима. MIB база ће свакако садржати типове података које су

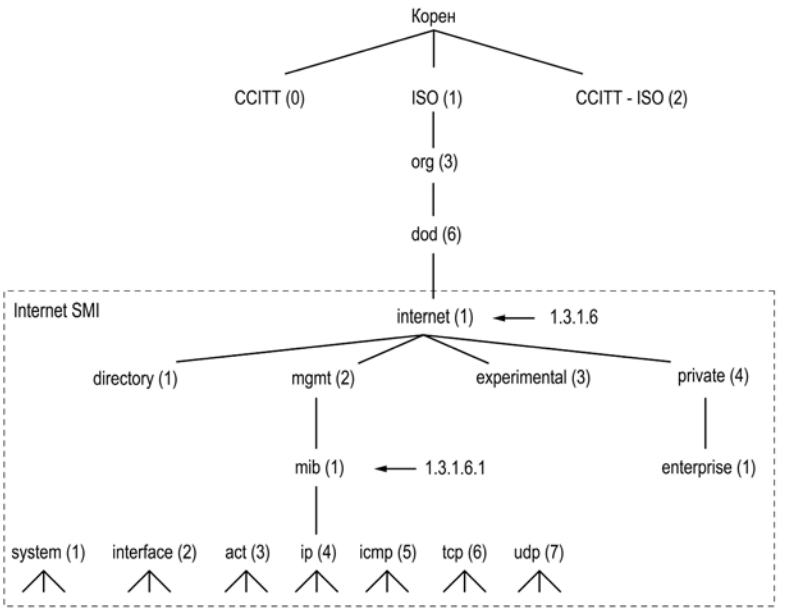
креирали сами произвођачи и док се не поставе строга правила везана за дефинисање типова података међусобно повезивање система различитих произвођача (интероперабилност) неће бити могућа.

Три кључна елемента налазе се у SMI спецификацији. На најнижем нивоу SMI структура одређује типове података који могу да се складиште. Затим SMI структура одређује технику за дефинисање објеката и њихових табела. Коначно, SMI структура обезбеђује шему за повезивање јединствених ознака са сваким стварним објектом у систему тако да податке и агенте могу менаџери да референцирају

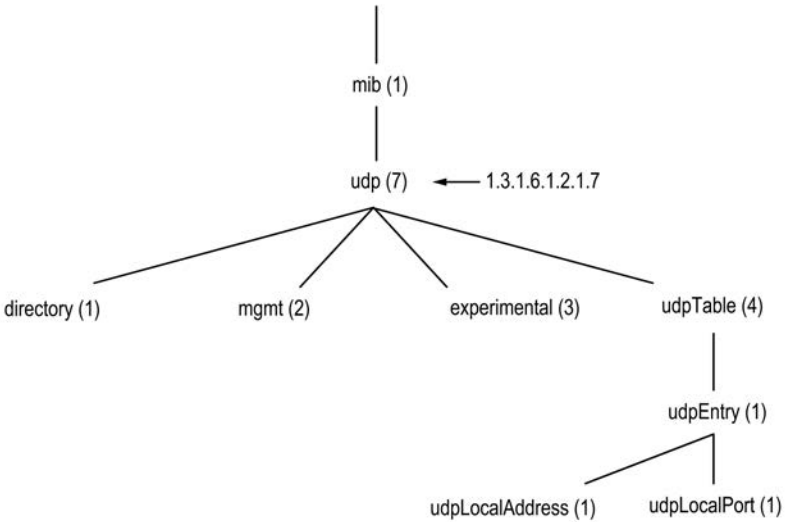
Табела 11.1 приказује типове података које SMI спецификација прописује. То је прилично ограничен скуп типова. На пример, реални бројеви нису подржани. Међутим, довољно је широк да подржи већину захтева у управљању рачунарском мрежом.

11.6.1 Идентификатор објекта

Идентификатор објекта је низ целих бројева одвојених тачком. Бројеве спаја разграната структура (стабло) слична DNS стаблу. На



Слика 11.6 Идентификатори објеката у MIB бази



Слика 11.7 Структура стабла за табелу за IP адресе

самом врху стабла је неименовани корен од кога идентификатор објеката почиње. На слици 11.6 је структура стабла која се користи за системе са SNMP протоколом. Све променљиве у управљачкој MIB бази започињу идентификатором објекта који почиње са 1.3.6.1.2.1.

На слици 11.6 је поред MIB идентификатора објеката представљен и један именовани објекат: *iso.org.dod.inernet.private.enterprise* (1.3.6.1.4.1). То је место где се постављају MIB базе специфичне за одређеног произвођача. Овај чвор садржи листу регистрованих идентификатора који су додељени одговарајућим RFC документом.

11.6.2 База података са информацијама за управљање

Базу података са информацијама за управљање (MIB база) одржава управљачки агент. Из ње менаџер може да добије одговор (упит) или у њу менаџер може да постави одређене вредности. У овом одељку биће дат кратак преглед базе података означене са MIB-II која је првобитно описана у документу RFC1213.

Као што је на слици 11.6 представљено MIB је подељена у групе означене са: *system, interfaces, at*[1], *ip* итд. Описаћемо само променљиве

[1] *Address translation*

Име	Тип податка	R/W	Опис
udpInDatagram	Counter		Број UDP датаграма који су испоручени корисничком процесу
udpNoPorts	Counter		Број примљених UDP датаграма за које није пронађен апликациони процес на одредишном порту
udpInErrors	Counter		Број неиспоручених UDP датаграма за које је неки други разлог а не "непостојање корисничког процеса" као што је нпр. грешка у контролној суми UDP датаграма.
udpOutDatagram	Counter		Број послатих UDP датаграма

Табела 11.2 Променљиве у групи *udp*

из групе UDP. Ово је једноставна група са малим бројем променљивих и једном табелом.

Постоје четири променљиве и табела која садржи две променљиве. У табели 11.2 дат је опис променљивих.

Колона означена са “R/W” је празна уколико се променљива може само очитати[1] или садржи ознаку (*) уколико се променљива може очитати и уписати[2]. Уколико је податак типа INTEGER са ограничењима у табели (нпр. 11.3) биће означена његова горња и доња граница. Две променљиве из *udpTable* су представљене у табели 11.3.

Табела UDP стране која ослушкује, индекс = <*udpLocalAddress*>. <*udpLocalPort*>			
Име	**Тип податка**	**R/W**	**Опис**
udpInDatagram	IpAddress		Локална адреса за страну која ослушкује. вредност 0.0.0.0 указује да је онај који ослушкује спреман да прихвати датаграм на било ком интерфејсу.
udpLocalPort	0...655351		Број локалног порта за оног који ослушкује

Табела 11.3 Променљиве у *udTable*

[1] *Read-only*
[2] *Read - Write*

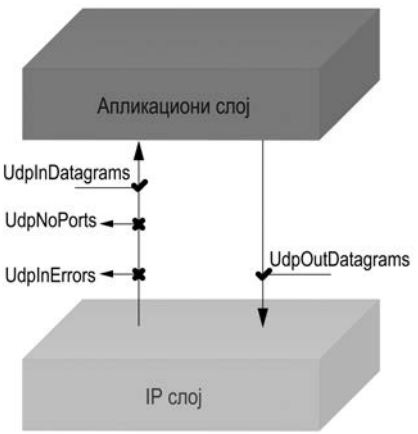
11.7 Дијаграм стања за UDP групу

Постоји међусобна веза између прва три бројача[1] из табеле 11.2. Дијаграм стања за UDP групу визуелно представља везу између различитих променљивих у MIB бази задате групе.

Слика 11.8 представља дијаграм стања за UDP групу. Овај дијаграм показује да је број UDP датаграма који се испоручују апликацији (*udpInDatagram*) једнак броју датаграма које IP слој испоручује транспортном слоју (UDP) умањен за број оних код којих је откривена грешка у контролној суми (*udpInErrors*) и оних за које није пронађена апликација на одредишном порту (*udpNoPorts*). Такође, број UDP датаграма испоручених од IP слоја до транспортног слоја (*udpOutDatagram*) јесте број датаграма које апликација прослеђује транспорном слоју.

11.8 Вредности идентификатора MIB базе

Свака променљива у MIB бази мора бити идентификована када се SNMP протокол референцира на њу, очитава или поставља њену вредност.Треба истаћи да се само чворови „листови“ референцирају. SNMP протокол не ради са одредишним колонама или редовима табела. Посматрајући слику 11.7 чворови „листови“ су они који су описани на слици 11.8. Нису чворови „листови“: *mib, udp, udpTable* и *udpEntry*.



Слика 11.8 Дијаграм стања за UDP групу

[1] *Counter*

11.8.1 Променљиве MIB базе

Променљиве се референцирају додавањем "0" на идентификатор објекта променљиве. На пример бројач *udpInDatagram* из табеле 11.2 чији идентификатор објекта је 1.3.6.2.1.7.1 референцира се са 1.3.6.2.1.7.1.0. Текстуално име је:

iso.org.internet.mgmt.mib.udp.udpInDatagrams.0

Иако се име са којим се референцира променљива обично скраћује на *udpInDatagrams.0* треба нагласити да име променљиве које се јавља у SNMP поруци јесте идентификатор објекта 1.3.6.2.1.7.1.0.

11.8.2 Табеле MIB базе

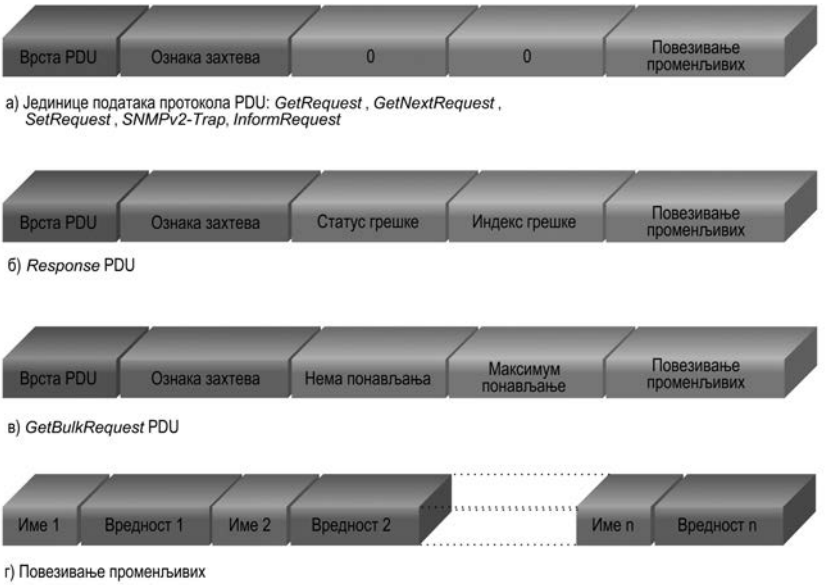
Вредност идентификатора записа табеле је много детаљнија. Погледајмо поново табелу стране која ослушкује (слика 11.7).

Један или више индекса је специфицирано у MIB бази за сваку од табела. За табелу стране која ослушкује MIB дефинише се индекс као комбинација две променљиве *udplocalaAddres* која је у IP address и *udpLocalPort* која је целобројна (индекс је постављен на почетку табеле 11.3). Претпоставимо да имамо три реда на UDP страни која ослушкује: први ред је за IP адресу 0.0.0.0 и порт 67, други за 0.0.0.0 и порт 161 и трећи за 0.0.0.0 и порт 520 (табела 11.4).

11.9 Основна обележја SNMPv2 протокола

Главна компонента SNMPv2 оквира је сам протокол. Протокол обезбеђује директан, базичан механизам за размену управљачких информација између менаџера и агента. Основна јединица која се размењује је порука која је састављена од рама и јединице протокола (PDU[1]). Заглавље рама задужено је за сигурност. Јединица података протокола (PDU) може пренети седам врста SNMP порука. Општи формат приказан је на слици 11.9. Неколико поља је заједничко за више јединица података протокола. Поље ознака захтева[2] је цео број додељен тако да је сваки захтев јединствено означен. То омогућава менаџеру да међусобно

[1] *Protocol data unit*
[2] *Request-id*



Слика 11.9 Формати јединице података протокола (PDU) за SNMPv2

повеже долазеће одговоре са захтевима који чекају на одговор. То такође омогућава агенту да разреши проблем дупликата јединица података протокола који је настао као последица непоуздане мрежне услуге коришћењем UDP протокола. Поље повезивање променљивих[1] садржи листу ознака објеката зависно од јединица података протокола.

Захтев за узимањем вредности (*GetRequest*) који издаје менаџер садржи листу једног или више имена објеката за које се тражи вредност. Ако је операција успешна одговарајући агент ће послати јединицу података протокола одговор (*Response*). Листа *Variable-bindings* ће садржати ознаке и вредности свих тражених објеката. За све променљиве које нису релевантне за MIB ознака и порука о грешци враћају се у листу *Variable-bindings*. Протокол SNMPv2 не дозвољава делимичан одговор на *GetRequest*, што је битно побољшање у односу на SNMPv1. У верзији SNMPv1 ако једна или више променљивих у јединици података протокола *GetRequest-PDU* није подржана агент ће вратити поруку о грешци која гласи: „нема таквог имена (*noSuchName*)". Да би се изборио са том грешком,

[1] *Variable-bindings*

менџер неће вратити никакву вредност или ће укључити такав алгоритам који ће када дође до грешке одстранити недостајућу променљиву поново шаљући захтев и прослеђујући апликацији непотпун резултат.

Јединицу података *GetNextRequest* PDU такође издаје менаџер и она садржи листу једног или више објеката. У овом случају за сваки објекат именован у *variable-bindings* пољу, вредност мора бити враћена за објекат који је следећи по лексикографском реду, тј. следећи по реду у структури стабла ознака објеката. Као што је било и са јединицом података протокола *GetRequest* PDU, агент ће вратити вредности за што је могуће више променљивих. Једна од особености јединице података протокола *GetNextRequest* PDU је да омогући менаџеру да динамички претражује структуру MIB базе. Корисно је ако менаџер не зна унапред скуп објеката који подржава агент или који је у одређеној MIB бази.

Једно од главних унапређења у SNMPv2 протоколу је јединица података протокола захтев за узимањем групе вредности (*GetBulkRequest*.) Сврха ове јединице података протокола је да смањи број размена у протоколу потребних да би се прибавила велика количина управљачких информација. Јединица података *GetBulkRequest* дозвољава SNMPv2 менаџеру да захтева да одговор буде што је могуће обимнији поштујући ограничење у величини поруке.

Захтев за постављањем вредности (*SetRequest*) шаље менаџер да би се вредност једног или више објеката променила. Пријемна SNMPv2 целина шаље одговор (*Response*) који садржи исту ознаку захтева. Извршавање *SetRequest* команде је аутоматско: или су промењене све вредности које су постављене или није ниједна. Уколико је могуће поставити вредности за све променљиве које се налазе у долазећој листи онда се у поруку *Response* уписују те вредности у поље повезивање променљивих. Ако се само једна од вредности променљиве не може променити не шаље се натраг ниједна вредност и ниједна вредност се не ажурира. Статусни кôд о грешци указује на разлог грешке и поље индекс-грешке указује на променљиву у листи која је довела до грешке.

Јединицу података *Trap* прави и шаље SNMPv2 целина који има улогу агента уколико дође до неуобичајеног догађаја. Користи се да би се управљачка станица обавестила (асинхроно) о неким битним догађајима.

Листа се користи да чува информације везане за Trap поруке. Јединице података протокола SNMPv2-*trap* за разлику од јединица података протокола *GetRequest, GetNextRequest, GetBulkRequest, SerRequest* и *InformRequest* PDU не захтевају одговор од пријемне целине.

Јединицу података захтев за информацијом (*InformRequest*) шаље SNMPv2 целина која је у улози менаџера, у име апликације, другој SNMPv2 целини која је у улози менџера да би прибавила управљачке информације за апликацију. Као што је случај и са јединицом података SNMPv2-*trap*, поље листе користи се да пренесе одговарајуће информације. Менаџер који прима *InformRequest* потврђује пријем са јединицом података протокола *Response*.

За SNMPv2-*trap* и *InformRequest* важи да се могу одредити различита стања која указују када је обавештење направљено; а назначено је и које се информације могу послати.

11.10 Протокол за управљање мрежом SNMPv3

Многи недостаци у раду SNMP протокола задржали су се и у верзији SNMPv2. Да би се исправили недостаци SNMPv1 и SNMPv2 протокола везани за сигурност направљена је верзија SNMPv3 протокола која се појавила јануара 1998. год. и која је описана у документима RFC2570-RFC2575. Године 2002. појавили су се нови документи RFC3411-RFC3418 а измењене верзије ових докумената RFC5543, RFC5490 2008. и 2009. године. Скуп ових докумената не приказује комплетне могућности верзије SNMPv3 протокола већ дефинише његову општу архитектуру и особености везане за сигурност.

SNMPv3 протокол обезбеђује три важне услуге: аутентификацију, приватност и контролу приступа. Прва два су део корисничке сигурности (USM[1] модела), а трећи је дефинисан у контроли приступа (VABC[2] моделу). Сигурносним услугама се управља преко идентитета[3] корисника који захтевају услуге. Овај идентитет описује се као принципал[4] а може бити особа или апликација, или група особа или апликација.

[1] *User-Based Security Model*
[2] *View-Based Access Control*
[3] *Identity*
[4] *Principal*

Механизам аутентификације код корисничке сигурности (USM модела) осигурава да је примљена порука послата од принципала чија се идентификација појављује у заглављу поруке. Овај механизам такође осигурава да порука није промењена на свом путу и да није намерно закашњена или поново направљена. Предајни принципал обезбеђује аутентификацију стављајући аутентификациони кôд поруке у SNMP поруку коју шаље. Овај кôд је функција садржаја поруке, идентитета предајног и пријемног пара, времена преноса и тајног кључа који треба да знају само пошиљалац и прималац. Тајни кључ мора бити подешен ван корисничке сигурности (USM модела) као конфигурациона функција. Менаџер конфигурације или мреже задужен је за дистрибуцију тајног кључа и убацивање у базе података различитих SNMP менаџера и агената. То може да се уради физички или користећи неку форму сигурног преноса података изван корисничке сигурности (USM модела). Када пријемни принципал добије поруку он користи исти тајни кључ да би израчунао исти аутентификациони кôд поруке. Ако верзија кôда код пријемника одговара вредности додатој долазној поруци онда пријемник зна да порука потиче од ауторизованог принципала и да порука није промењена на свом путу. Дељени тајни кључ између предајног и пријемног пара мора бити раније подешен. Тренутни аутентификациони кôд који се користи познат је као HMAC[1], који је стандардан механизам аутентификације на Интернету.

Део који се односи на приватност у корисничкој сигурности омогућава менаџерима и агентима да шифрују поруке. Менаџер принципал и агент принципал морају да деле тајни кључ. У овом случају, ако су две стране конфигурисане да користе приватност сав саобраћај између њих биће шифрован помоћу DES[2] алгоритма. Предајни принципал шифрира поруку користећи DES алгоритам и његов тајни кључ и шаље поруку пријемном принципалу који је дешифрује помоћу DES алгоритма и истог тајног кључа.

Део за контролу приступа пружа могућност конфигурисања агената тако да пруже различите нивое приступа MIB бази агента различитим менаџерима. Агент принципал може да ограничи приступ својој MIB бази одређеном менаџер принципалу на два начина:

[1] *Hash-based Message Authentication Code*
[2] *Data Encryption Standard*

- Први начин је да му ограничи приступ на само одређен део MIB базе. На пример, агент може да дозволи већини менаџера приступ само делу за преглед статистичких података о перформансама, а само одређеном менаџеру да омогући преглед и ажурирање конфигурационих параметара;
- Други начин је да агент може да ограничи активности које менаџер може да изврши на одређном делу MIB базе. Рецимо одређени менаџер принципал може само да очитава[1] неки део базе управљачких информација MIB базе агента. Полиса контроле приступа која се користи од стране било ког агента, за сваког принципала мора бити претходно конфигурисана и састоји се од табеле у којој се налазе детаљи права приступа различитих ауторизованих менаџера.

11.11 Питања и задаци

1. Шта обухвата надзор и управљање перформансама?
2. Који је циљ код система за надзор и управљање грешкама?
3. Шта обухвата надзор и управљање сигурношћу?
4. Како се може дефинисати систем за управљање мрежом?
5. Од чега се састоји систем за управљање мрежом?
6. Како је пројектован систем за управљање мрежом?
7. Описати основна обележја протокола за надзор и управљање мрежом SNMPv1.
8. Које су компоненте модела за управљање мрежом?
9. Шта све треба управљачка станица да садржи?
10. Које операције користи SNMP протокол?
11. Описати децентрализовану шему управљања мрежом.
12. Које су новине у верзији SNMPv2 протокола?
13. Како се може дефинисати структура управљачких информација?
14. Описати неке од дозвољених типови података у SNMPv2 протоколу представљених у табели 11.1.
15. Како се користи идентификатор објеката?

[1] *Read-only*

12. Комутација са више протокола на основу ознака

Комутацију са више протокола (вишепротоколарна) на основу ознака (MPLS[1]) развили су произвођачи опреме за рутирање (*Cisco, IBM, Toshiba, Lucent*...) са циљем да се побољша брзина прослеђивања рутера и лакша интеграција IP и ATM мрежа. Своје напоре су усмерили ка томе да испред сваког пакета додају ознаку[2] (кључног концепта из света мрежа са виртуелним колима) на основу које би се пакети усмеравали уместо одредишне адресе. Када се ознаке у облику индекса унесу у табеле рутирања проналажење праве излазне линије своди се на једноставно претраживање те табеле. Усмеравање помоћу ове технике може се обавити веома брзо, а дуж путање могу се резервисати сви неопходни ресурси. Циљ није био да се напусти концепт прослеђивања IP датаграма на основу одредишта ради оне која је заснована на ознакама фиксне дужине и виртуелним колима него да се она прошири. Када су пакети (датаграми) означени рутери могу да прослеђују датаграме на основу ознака фиксне дужине а не на основу IP адреса одредишта.

Означавање токова пакета на овај начин приближава се концепту виртуелних кола. У мрежи X.25, ATM мрежи, мрежи са штафетним преносом рамова и у другим мрежама које користе концепт виртуелних кола у сваки пакет се додаје ознака (идентификатор виртуелног кола). Та ознака се тражи у табели рутирања и пакет усмерава на основу одреднице из ње.

Постоје значајне разлике у погледу тога како се на Интернету формира путања и начина на који то ради мрежа са успостављањем везе, тако да се примењује техника која се разликује од технике комутације

[1] *MultiProtocol Label Switching*
[2] *Label*

канала[1]. Овај нови начин комутације носи различита имена, као што су: комутирање ознака[2] или комутирање етикета[3].

Организација IETF је стандардизовала концепт под именом вишепротоколарно комутирање на основу ознака (MPLS) и специфицирала га у документима RFC3031 и RFC3032. Иако је MPLS протокол намењен за интеграцију са различитим протоколима (IP, IPX, *Apple Talk*,..) у пракси се највише користи у мрежама са IP протоколом.

MPLS протокол је протокол са успоставом везе независан од протокола на слојевима 2 и 3 OSI референтног модела. По функционалности налази се између та два слоја, због чега се често означава као протокол слоја 2,5. Разлог је следећи: IP пакети нису прављени за виртуелна кола, IP заглавље нема поље у које би се могао сместити број виртуелног кола. Због тога се испред IP заглавља морало додати ново MPLS заглавље. Пренос података између мрежних чворова врши се уз помоћ ознаке. Као и ATM технологија, пројектован је за пренос различитих типова саобраћаја. Главна предност огледа се у независности од технологије на другом слоју било да је реч о комутацији ћелија (ATM), штафетном преносу рамова, Етернету или PPP протоколу, тако да нису потребне вишеструке адаптационе технике.

12.1 Основне компоненте MPLS архитектуре

Ради лакшег разумевања како MPLS систем функционише, осврнућемо се укратко на рад рутера базираног на IP протоколу. Сваки IP рутер у основи има две компоненте. Прва компонента заснива се на примени различитих протокола рутирања, као што су RIP, OSPF[4] и BGP[5] протоколи. Ови протоколи креирају руте, и информације о њима размењују се међу рутерима у мрежи. Сваки IP рутер на основу ових информација креира табелу рутирања тј. прослеђивања, познату и као база информација за прослеђивање (FIB[6] база). Друга компонента, тзв.

[1] *Circuit switching*
[2] *Label switching*
[3] *Tag switching*
[4] *Open Shortest Path First*
[5] *Border Gateway Protocol*
[6] *Forwarding Information Base*

прослеђивачка[1], састоји се од процедура које рутер користи како би одредио интерфејс на који ће проследити добијени IP пакет. Тако да код уникаст прослеђивања, рутер користи одредишну IP адресу смештену у заглављу IP пакета, и на основу ње проналази одговарајући ред у FIB табели коришћењем алгоритма најдужег поклапања[2]. У том реду налази се број који репрезентује интерфејс преко ког ће добијени IP пакет бити прослеђен следећем рутеру или самој крајњој станици којој је тај пакет намењен. Рутер прослеђује пакете на основу њиховог префикса. У једном рутеру све адресе које имају исти префикс чине еквивалентну класу за прослеђивање (FEC[3] класу). IP пакети који припадају истој FEC класи биће прослеђени на исти излазни интерфејс.

12.1.1 Еквивалентна класа у прослеђивању

Еквивалентна класа у прослеђивању (FEC класа) је група или ток пакета која се преусмерава истом путањом и која се третира на исти начин у складу са дефинисаним критеријумом за прослеђивање. Сви пакети из исте FEC класе имају исту ознаку. Међутим то не значи да сви пакети који имају исту ознаку припадају истој FEC класи пошто се неки од параметара везаних за квалитет услуге међусобно разликују (нпр. приоритет). То значи да се критеријуми за прослеђивање разликују и ти пакети могу да припадају различитим FEC класама. Рутер који одлучује о томе који пакет припада којој FEC класи је улазни рутер са комутацијом пакета на основу ознака (LSR[4] рутер[5]).

Примери FEC класа су:

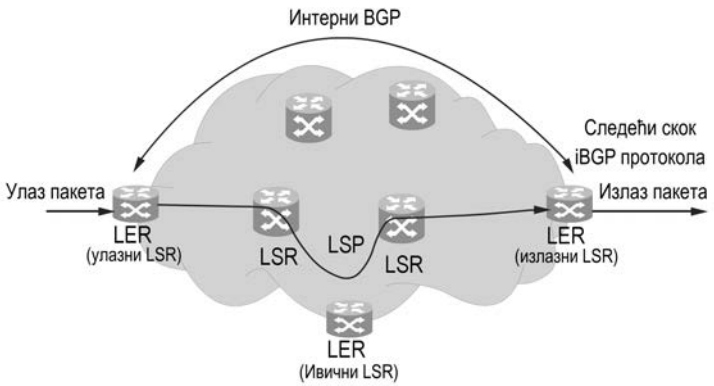
- Пакети са одредишним IP адресама (адресама 3. слоја) којима се поклапа одређени префикс. Често коришћена FEC класа је најдуже поклапање префикса[6] код одредишне IP адресе која се на пример користи код бескласног рутирања између домена (CIDR[7]);

---

[1] *Forwarding*
[2] *Longest match*
[3] *Forwarding Equivalent Class*
[4] *Label Switched Router*
[5] *Ingress LSR*
[6] *Longest prefix match*
[7] *Classless Interdomain Routing*

- Пакет упућен групи станица (мултикаст);
- Пакети са истим третманом у прослеђивању заснованим на приоритетима и диференцираној услузи[1] специфицираној у одговарајућем пољу пакета[2];
- Рамови (2. слоја) који се преносе MPLS мрежом примљени преко једног виртуелног кола или интерфејса улазног LSR рутера (LER[3] рутер) и послати преко једног виртуелног кола или интерфејса излазног LSR рутера[4];
- Пакети са одредишном IP адресом која припада скупу префикса BGP протокола, сви ка истом BGP следећем скоку.

Последњи пример је нарочито интересантан (слика 12.1). Пакети које прихвата улазни LSR рутер (LER рутер) а чија одредишна адреса указује на скуп BGP рутера у табели рутирања (сви са истом адресом следећег скока) припадају једној FEC класи. То значи да сви пакети који уђу у MPLS мрежу добију ознаку на основу тога који је следећи BGP рутер. На слици 12.1 је приказана MPLS мрежа у којој сви ивични LSR рутери користе интерни BGP протокол (iBGP[5]).



Слика 12.1 MPLS мрежа са iBGP протоколом за рутирање

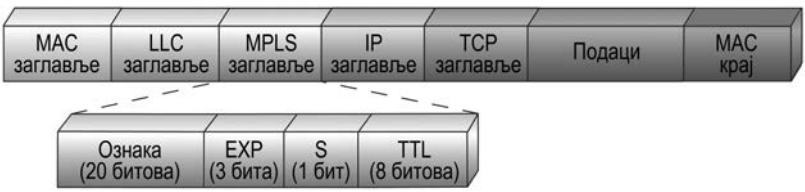
[1] Поље *TOS/DiffServ*
[2] *IP DiffServ Code Point* (DSCP)
[3] *Label Edge Router*
[4] *Egress LSR*
[5] *Internal BGP*

12.1.2 Формат MPLS ознаке

Код MPLS протокола свака FEC класа повезана је са различитом ознаком. Ова ознака се користи код одабира излазног интерфејса преко кога ће бити прослеђен IP пакет без потребе да се претражује његова адреса у FIB табели. Ознака је идентификатор фиксне величине и има локални значај, тј. везана је само за линк између два суседна рутера[1]. MPLS протокол се користи у различитим типовима мрежа. Код IPv6 пакета ознака се може унети у ознаку тока. Пакети IPv4 у свом заглављу немају поље у које се може уписати ознака. Код Етернета и тачка-тачка мрежа које користе протоколе слоја везе (нпр. PPP протокол) ознака се укалупљује (енкапсулира) и смешта између LLC и IP заглавља[2] (слика 12.2).

Као што се са слике 12.2 види, укалупљена (енкапсулирана) ознака има четири поља.

- ознаке (20 битова),
- експериментална улога (3 бита) - најчешће се користи код дефинисања квалитета услуге (QoS[3]),
- S поље (1 бит) - користи се у случају слагања ознака да укаже да се ради о последњој ознаци у низу,
- време живота TTL[4] и има сличну улогу као TTL поље у IP заглављу.



Слика 12.2 Место и формат MPLS ознаке

[1] Њена улога слична је улози VPI/VCI идентификатора код ATM мрежа. Виртуелни идентификатор канала (VCI - *Virtual Channel Identifier*) је јединствени идентификатор који означава одређени виртуелни канал на мрежи. То је 16-битно поље у заглављу ATM ћелије. VCI идентификатор заједно са идентификатором виртуелне путање ((VPI - *Virtual Path Identifier*) користи се да идентификује следеће одредиште ћелије на њеном путу ка крајњем одредишту а при преласку преко ATM комутатора. ATM комутатори користе VPI/VCI поља за идентификацију линка виртуелног канала (VCL - *Virtual Channel Link*) од наредне мреже кроз коју ћелија треба да прође на путу до крајњег одредишта.

[2] У случају мреже са штафетним преносом рамова ознака је смештена у DLCI (*Data Link Connection Identifier*) пољу рама. За пренос преко ATM мреже ознака се смешта у VPI/VCI поље ATM ћелије.

[3] *Quality of Service*

[4] *Time-To-Live*

12.1.3 MPLS чворови и домени

MPLS мрежа састоји се од рутера са комутацијом ознака (LSR рутера). LSR рутер је заправо IP рутер на коме је активиран MPLS протокол. Он препознаје ознаке, прима и шаље означене пакете користећи слој везе.

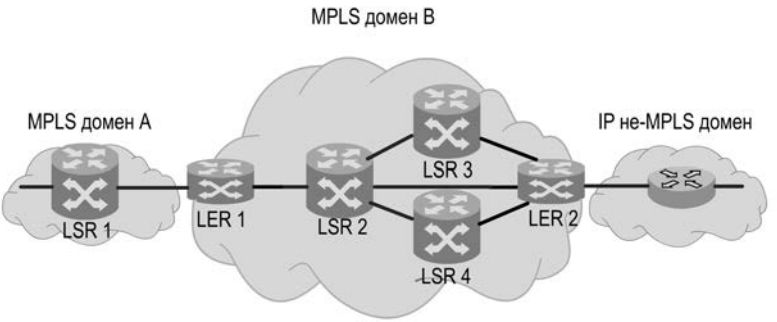
Рутери (у мрежама са IP протоколом) који се налазе на крајевима MPLS домена означавају се као MPLS ивични чворови[1] или гранични рутери са комутацијом ознака -LER рутери. На слици 12.3 то су рутери LER1 и LER2. MPLS ивични чворови су чворови који повезују MPLS домен са чвором који је ван домена или зато што не подржава MPLS ознаке и/или зато што је у другом домену.

Рутер LSR може да обавља три операције:

- постављање ознаке (*push*),
- уклањање ознака (*pop*) или
- замену ознака (*swap*).

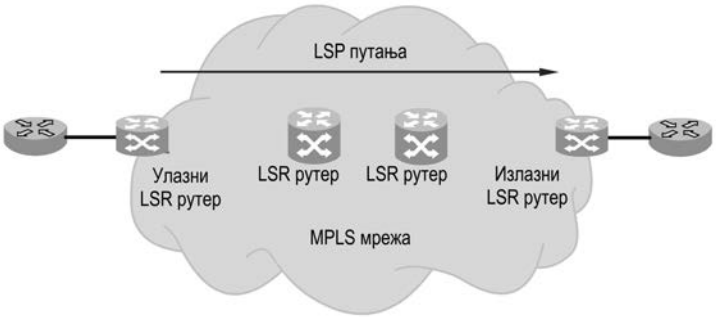
12.1.4 Означена комутирана путања

Означену комутирану путању (LSP[2] путању) чини низ LSR рутера који комутирају означене пакете кроз MPLS мрежу или кроз део MPLS мреже (слика 12.4). У основи LSP путања је путања којом се преноси пакет кроз MPLS мрежу или кроз део MPLS мреже. Први рутер LSP путање је улазни LSR рутер те путање а последњи рутер је излазни LSR



Слика 12.3 MPLS чворови и домени

[1] *MPLS edge node*
[2] *Label Switched Path*



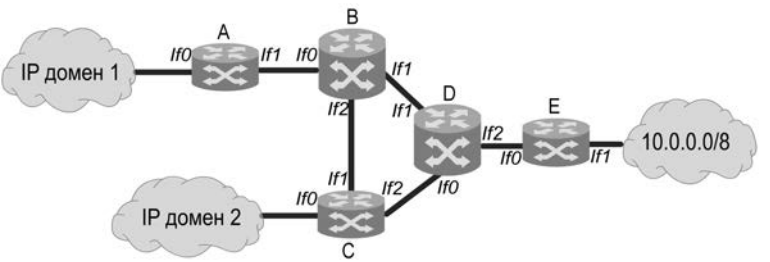
Слика 12.4 LSR путања у MPLS мрежама

рутер те путање. Сви LSR рутери између улазног и излазног LSR рутера су средишњи LSR[1] рутери.

12.2 Пример комутације пакета у оквиру MPLS домена

На слици 12.5 приказан је MPLS домен који се састоји од пет LSR рутера међусобно повезаних тачка-тачка везама. Рутери A и C директно су повезани на IP (не-MPLS) домене, а рутер E на мрежу 10.0.0.0/8. Мрежни префикс у овом случају је 10.0.0.0, па се може рећи да сви пакети чија је одредишна адреса у опсегу од 10.0.0.1 до 10.255.255.254 припадају истој FEC класи.

Пакети намењени мрежи 10.0.0.0/8 од рутера A до рутера E преносиће се преко рутера B и D, а од рутера C до рутера E преко рутера D. На слици 12.5 наведене су и ознаке интерфејса путем којих су остварене



Слика 12.5 Пример MPLS комутације

[1] *Intermediate LSR*

тачка-тачка везе између рутера. Када LSR рутер на свом интерфејсу детектује пакет који припада FEC класи са префиксом 10.0.0.0, додељује му ознаку (из опсега недодељених ознака) и креира ред у својој табели за преусмеравање (LFIB[1] табели). Ова табела садржи информације о улазној[2] и излазној[3] ознаци придруженој одговарајућој FEC класи и излазном интерфејсу рутера. LSR рутер такође чува ознаку и у својој FIB табели у реду који је придружен посматраној FEC класи.

Пре било какве размене информација о ознакама сваки рутер детектованој FEC класи (10.0.0.0) додељује одређену ознаку. Табела 12.1 приказује вредности тих ознака.

Рутери A и C (ивични рутери) не придружују улазним пакетима никакве ознаке јер су повезани на IP домен (не MPLS домене) па самим тим и не очекују да приме пакете са ознакама. Због тога одлуку о интерфејсу на који ће проследити саобраћај намењен FEC класи 10.0.0.0 доносе увидом у своје FIB табеле.

Улазна[4] ознака је у ствари ознака коју LSR рутер очекује да види у улазним IP пакетима који припадају одређеној FEC класи. На пример рутер B очекује да примљени IP пакети намењени мрежи 10.0.0.0/8 имају ознаку 28. Да би тако стварно и било пакети се морају означити на рутеру који је тзв. узводни[5] рутер рутера B. Посматрајући слику 12.3 и ток саобраћаја намењен мрежи 10.0.0.0/8, једини узводни рутер рутера

LSR рутер	Улазна ознака	Излазна ознака	Следећи скок	Излазни интерфејс
A	-	-	*LSR B*	*if1*
B	*28*	-	*LSR D*	*if1*
C	-	-	*LSR D*	*if2*
D	*31*	-	*LSR E*	*if2*
E	62	-	-	*if1*

Табела 12.1 Почетни изглед дела LFIB табела на рутерима за FEC класу 10.0.0.0/8

[1] *Label Forward Information Base*
[2] *Incoming*
[3] *Outgoing*
[4] *Incoming*
[5] *Upstream*

B је рутер A. У том случају рутер B је низводни[1] рутер A рутера. У случају рутера D, узводни рутери су и рутер B и рутер C.

Да би рутер примио пакете са ознаком коју очекује он мора обавестити своје суседе о ознакама које је доделио различитим FEC класама. Тако ће, на пример, рутер B послати информације о својим ознакама рутерима A, C и D. Рутер A себе препознаје као узводни рутер B рутера и користи информације добијене од рутера B да би ажурирао одговарајуће редове у својој LFIB табели (између осталог и ред који се односи на FEC класу 10.0.0.0). С обзиром да за FEC класу 10.0.0.0 рутери C и D нису узводни рутери B рутера, они неће ажурирати своје LFIB табеле информацијама добијеним од рутера B. Међутим, постоји могућност да добијене информације сачувају у својој меморији и да их касније употребе. На пример, ако линк између рутера C и D откаже (падне), тада ће рутер C постати узводни рутер B рутера.

Рутер D шаље информације о својим ознакама рутерима B и C. Пошто се оба рутера препознају као узводни рутери D рутера, они ажурирају своје LFIB табеле. Рутер E шаље своје информације рутеру D који као узводни рутер E рутера ажурира своју LFIB табелу.

Размена информација међу рутерима омогућава допуну редова који се односе на FEC 10.0.0.0 у LFIB табели рутера (табела 12.2).

Рутер E је директно повезан на мрежу 10.0.0.0/8. Прослеђивање пакета намењених тој мрежи обавиће увидом у своју FIB табелу, тј. на основу мрежног префикса.

Шта се након попуњавања LFIB табела на рутерима дешава са

LSR рутер	Улазна ознака	Излазна ознака	Следећи скок	Излазни интерфејс
A	-	28	*LSR B*	*if1*
B	*28*	31	*LSR D*	*if1*
C	-	31	*LSR D*	*if2*
D	*31*	62	*LSR E*	*if2*
E	62	-	-	*If1*

Табела 12.2 Коначни изглед дела LFIB табела на рутерима за FEC класу 10.0.0.0/8

[1] *Downstream*

пакетом који се нађе у MPLS домену какав је приказан на слици 12.3? Рутер A прихвата пакет из IP домена. У случају да је намењен мрежи 10.0.0.0/8, размотриће своје FIB и LFIB табеле и доделиће му ознаку 28. Обележени пакет прослеђује се на излазни интерфејс *if1*. Рутер B, када пакет стигне до њега, анализира своју LFIB табелу, уклања ознаку 28 и убацује нову која има вредност 31. Новодобијени пакет прослеђује даље путем свог интерфејса *if1*. Следећи рутер који прихвата пакет је рутер D. Он ће анализирати своју LFIB табелу, уклониће ознаку 31 и поставиће нову 62. Пакет се прослеђује даље путем интерфејса *if2* рутера D до рутера E. Рутер E уклања ознаку 62 из IP пакета и оригинални пакет прослеђује одредишној мрежи, тј. 10.0.0.0/8. Слично се дешава и са пакетима који су намењени поменутој мрежи а који у MPLS домен улазе преко рутера C.

Низ ознака 28, 31 и 62, представља LSP путању којом се преносе означени пакети од рутера A до рутера E ка одредишној мрежи 10.0.0.0/8.

Метода замене ознака унутар пакета елиминише потребу рутера за анализом FIB табеле. Са друге стране, LFIB табела је мала па њена анализа захтева много мање времена и процесорске снаге рутера у односу на анализу FIB табеле.

Квалитет услуге (QoS) у MPLS мрежама може се имплементирати означавањем приоритета пакета па се у ту сврху користи тробитно експериментално поље ознаке. Доделу приоритета пакета могу обављати ивични MPLS чворови. У зависности од приоритета пакети се прослеђују на излазне интерфејсе рутера према одређеном редоследу.

12.3 Запис следећег означеног скока

Да бисмо поједноставили анализу (пример са слике 12.5) претпоставили смо да LSR рутер одржава за сваку улазну ознаку по један ред у табели. Тај ред садржи везу улазне ознаке са излазном ознаком и информације везане за следећи скок (следећи LSR рутер и излазни интерфејс). MPLS архитектура дозвољава да LSR рутер одржава више редова за једну улазну ознаку. Тај ред се означава као запис следећег означеног скока (NHLFE[1] запис). Он може да садржи информације о

[1] *Next Hop Label Forwarding Entry*

укалупљивању на слоју везе, начин повезивања ознака или неке друге детаље специфичне за слој везе. NHLFE запис мора да садржи излазни интерфејс следећег скока заједно са податком о томе која ће од операција бити извршена на означеном пакету као што је:

- замена (*swap*) највише ознаке по рангу[1] новом ознаком;
- уклањање (*pop*) највише ознаке по рангу;
- замена највише ознаке по рангу новом ознаком и постављање (*push*) нове ознаке у низу ознака[2].

Да би се подржале напредне MPLS функције пакет може да садржи више од једне ознаке и назива се низ ознака. Низ ознака је хијерархијски уређени редослед по моделу „последњи постављен, први преузет[3]“. Наиме, сваки пакет може садржати више ознака организованих у низ. Да би рутер знао где се завршава низ ознака користи се S бит, чија је вредност 1 само ако се ради о последњој ознаци у низу (слика 12.6). Следеће три операције могу се извршити над низом ознака:

- заменити ознаку са врха низа ознака новом ознаком;
- уклонити ознаку са врха низа ознака;
- заменити ознаку са врха низа ознака новом ознаком а затим поставити једну или више ознака у низ.

Мапа долазећих ознака (ILM[4] мапа) повезује (мапира) долазеће ознаке у скуп NHLFE записа. Користи се када су пакети које треба усмеравати означени. ILM мапа део је LFIB[5] базе. Могућност да за улазну ознаку постоји више записа (редова у LFIB бази) може бити корисна код расподеле саобраћаја по више LSP путања[6] са истом ценом. Процедуре

Слика 12.6 Низ ознака

[1] *Topmost label*
[2] *Label stack*
[3] *Last-in, first-out*
[4] *Incoming Label Map*
[5] *Label Forwarding Information Base*
[6] *Multi pathing*

за одабир једног од NHLFE записа изван су оквира MPLS архитектуре и дефинисане су документом RFC3031.

Дефинисана је и FTN[1] мапа која се користи да се повеже FEC класа са скупом NHLFE записа. То значи да се из долазећег IP пакета извлачи податак о FEC класи и придружује једном или више NHLFE записа у FIB бази. Користи се када пакети пристижу неозначени и када их треба означити пре прослеђивања. Потребна је одређена процедура да се одаберe један од NHLFE записа (није дефинисано у документу RFC3031).

Важна оптимизација MPLS система у IP мрежама је избегавање процеса претраживања у излазном LER рутеру у случају када пакет пристиже LSP путањом која захтева претраживање и по IP адреси. На пример (слика 12.5, табела 12.1) долазећем пакету рутера E уклања се ознака 62, затим се он усмерава на претраживање FIB базе на основу IP заглавља. Да би се спречила ова додатна обрада у MPLS мрежи је могуће применити метод који се означава као „уклањање на претпоследњем скоку[2]“ где претпоследњи рутер LSP путање (у примеру са слике 12.5 рутер D) уклања ознаку уместо да захтева да то уради последњи рутер (рутер E). На тај начин се смањује захтевана обрада од последњег рутера.

12.4 Примена низа ознака у MPLS мрежи са тунелом

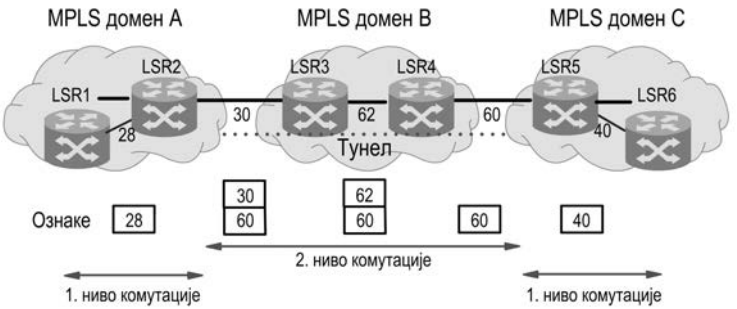
Када MPLS чворови нису узастопни рутери у путањи скок-по-скок[3] а MPLS чвор испоручује пакет ка „партнерском MPLS чвору” путања између њих се означава као тунел. LSP путања и комутација ознака користе се за преусмеравање саобраћаја кроз тунел. Сав саобраћај који се шаље преко тунела сачињава FEC класу. Сваки LSR рутер у тунелу мора да додели ознаку тој FEC класи. Да би пакет био послат кроз тунел улазни чвор тунела поставља у низ ознака ознаку коју разуме излазни чвор и усмерава пакет кроз тунел.

На слици 12.7 приказана је примена низа ознака у MPLS мрежи са тунелом. Мрежу чине три MPLS домена (A, B и C). Између рутера

[1] *FEC-to-NHLFE* - повезивање FEC класе у NHLFE запис

[2] *Penultimate hop popping*

[3] *Hop-by-hop path*



Слика 12.7 Пример постављања ознака у низ код тунела

LSR1 и LSR6 примењено је експлицитно рутирање. Између рутера LSR2 и LSR5, техником прављења низа ознака, креиран је тунел који чине рутери MPLS домена.

LSR1 додељује примљеном пакету ознаку 28 и прослеђује га рутеру LSR2. MPLS домен B у овом примеру представља тунел од LSR2 рутера до LSR5 рутера и може се представити замишљеном линијом између ова два рутера. Као што се са слике 12.7 види LSR5 рутер очекује пакете са ознаком 60. Због тога ће LSR2 рутер примљеном пакету заменити ознаку 28 новом, која има вредност 60. Пошто новодобијени пакет мора проћи кроз MPLS домен B, који физички постоји између рутера LSR2 и LSR5, LSR2 рутер пакету додаје још једну ознаку. Вредност ове друге ознаке одговара вредности коју очекује рутер LSR3 а то је 30. Првобитни пакет сада има две ознаке: 60 и 30. Прослеђује се рутеру LSR3 који уклања горњу ознаку (30) и мења је новом која има вредност 62. Рутер LSR4 прихвата пакет са ознакама 60 и 62. Пошто је LSR4 рутер крајњи рутер у домену B он уклања горњу ознаку (62). Рутеру LSR5 прослеђује се пакет само са једном, доњом ознаком (60) коју је убацио LSR2. На крају, рутер LSR5 ознаку 60 замењује ознаком 40 и прослеђује је рутеру LSR6.

Као што се види пакет који се преноси кроз MPLS домен B има две ознаке. Ознака на врху се користи за комутацију ознака у оквиру MPLS домена B а доњу ознаку користе два гранична чвора која повезују MPLS домене B и C. Коришћењем низа ознака омогућено је успостављање LSP тунела.

12.5 Рутирање на основу ограничења[1]

У мрежама са протоколима без успоставе везе, као што је IP мрежа, сваки рутер доноси независно одлуку о следећем скоку коме ће проследити примљени саобраћај. Ова одлука доноси се на основу садржаја заглавља пакета тј. оног дела који се односи на одредишну адресу, као и на основу информације о топологији мреже коју је рутер научио уз помоћ протокола рутирања. На пример OSPF протокол најкраћу путању рачуна на основу одговарајуће метрике тј. цене линкова. Међутим, у пракси се често јавља потреба да се путања до одредишта рачуна одреди и на основу других критеријума тако да је рутирање последица једног или више ограничења. Резултат приступа који узима у обзир различита ограничења је путања добијена управљањем саобраћајем (инжењерингом саобраћаја[2]). Она се може значајно разликовати од путање са најмањом ценом. У обзир се узимају различити критеријуми као што су:

- захтевани квалитет услуге (QoS),
- капацитет и тренутна загушеност линка,
- критеријум по коме су одређене врсте линкова пожељније за пренос од других,
- линкови које треба избегавати[3]...

На слици 12.8 приказана је мрежа од 5 рутера међусобно повезаних линковима брзине[4] 34Mb/s. Претпоставимо да је у мрежи активиран протокол рутирања најкраћом путањом[5] који је израчунао (нпр. узимајући у обзир број скокова) да је најповољнија путања од рутера A до рутера B преко рутера C (слика 12.8a). Ако између рутера A и B постоји саобраћај брзине 10Mb/s, у мрежи неће постојати загушење. Међутим, у случају да између ових рутера треба да се пренесу четири оваква тока (40Mb/s), доћи ће до загушења јер су међулинкови капацитета 34Mb/s. C друге

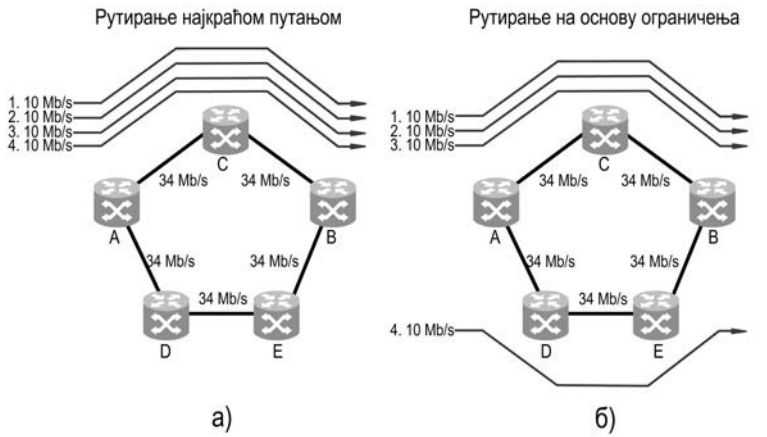
[1] *Constraint-based*

[2] *Traffic engineering*

[3] Детаљан опис може се наћи у документу RFC2702.

[4] Брзина на E3 линијама, временски мултиплекс. Стандардизован од стране ITU-T у свим државама осим САД-у, Канади и Јапану где се користи T/DS (*Digital Signal*) сигнализација линијама. Тачније брзина E3=34,368Mb/s (брзине: E0=64kb/s; E1=2,048Mb/s и E2=8,448Mb/s) а брзина T3/DS3=44,736Mb/s.

[5] *Shortest Path Routing*



Слика 12.8 Разлике између рутирања најкраћом путањом и рутирања на основу ограничења

стране, путања између рутера A и B преко рутера D и E остаје потпуно неискоришћена. Овај проблем јавља се управо из разлога што се приликом рачунања оптималне путање у обзир узима само један параметар (цена путање) а не и загушеност линкова који ту путању сачињавају.

Протоколи са успоставом везе када одређују путање користе рутирање које узима у обзир ограничење капацитета. Обично саобраћај се усмерава на путању са најбољом метриком све док се та путања потпуно не оптерети. Након тога остали саобраћај се преусмерава на друге путање са најбољом метриком. Уколико не постоји ниједна путања која би могла да пренесе жељени саобраћај, мрежа захтев за успоставом везе се одбацује. Слика 12.8 илуструје обе врсте рутирања.

12.6 Карактеристична обележја LDCP протокола

У овом одељку биће описана карактеристична обележја (атрибути) у управљању и дистрибуцији MPLS ознака (LDCP[1] протокола) описаних у документу RFC3031.

У великим мрежама, уместо статичких рута, примењује се неки од протокола рутирања, на пример OSPF протокол. Размењујући

[1] *Label Distribution Control Protocol*

информације са другим рутерима, сваки IP рутер ће сазнати топологију домена у коме се налази, и на основу ње ће за свако одредиште израчунати следећи скок (рутер). Информација о следећем скоку чува се у FIB табели рутера. MPLS протокол користи ове информације приликом креирања LSP путање и такво рутирање зове се скок-по-скок[1] рутирање.

Насупрот овоме, MPLS архитектура дозвољава и креирање LSP путања које се не морају подударати са путањама скок-по-скок. Овакво рутирање зове се експлицитно[2] рутирање и примењује се најчешће у случајевима када је потребно избалансирати искоришћеност мрежних ресурса и на тај начин повећати пропусну моћ мреже[3]. Такође, користи се и код креирања тунела и виртуелних приватних мрежа (VPN[4] мрежа) базираних на MPLS протоколу. Експлицитно рутирање може бити:

- стриктно експлицитно[5] - путања између улазног[6] и излазног[7] LSR рутера прецизно је дефинисана. То значи да су сви LSR рутери преко којих путања прелази експлицитно специфицирани;
- слободно експлицитно[8] - путања између MPLS домена прецизно је дефинисана, а унутар самих домена путање дефинишу ивични LSR рутери[9].

12.6.1 Размена информација између LSR рутера

Генерално гледано, информације о ознакама међу рутерима могу се размењивати на два начина:

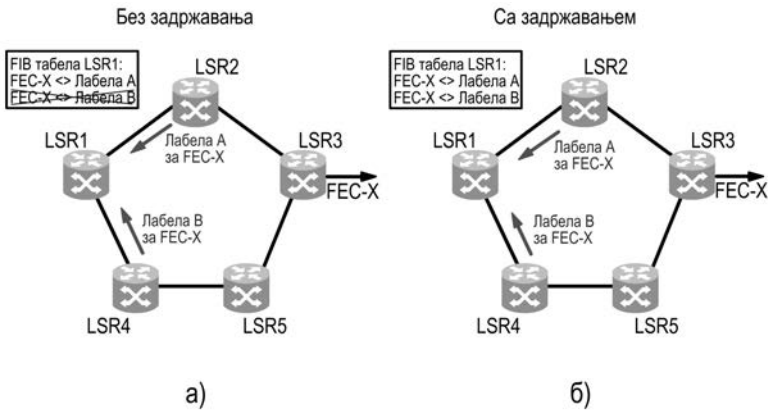
- Без претходног захтева[10] - LSR рутер гради LFIB табелу додељујући одговарајуће ознаке свакој FEC класи (слика 12.9). Након тога обавештава суседне LSR рутере о ознакама

[1] *Hop-by-hop*
[2] *Explicite*
[3] *Throughput*
[4] *Virtual Private Network*
[5] *Strictly explicite routed*
[6] *Ingress LSR*
[7] *Egress LSR*
[8] *Loosely explicited routed*
[9] Стриктно и слободно експлицитно рутирање еквивалентно је рутирању на основу стања линка и хијерархијском PNNI рутирању код ATM протокола.
[10] *Unsolicited downstream*

које је доделио. Ако је суседни рутер узводни[1] у односу на ток саобраћаја за посматрану FEC класу (узевши у обзир информацију добијену IP рутирањем), он ће ажурирати своју LFIB табелу. Уколико суседни рутер није узводни он ће добијену информацију или игнорисати (неће је задржати[2]) или ће је сачувати (задржати[3]) у својој меморији па је касније искористити.

- На захтев[4] - LSR рутер гради LFIB табелу, али је не оглашава суседима. Уместо тога, узводни LSR рутер шаље упит суседу када жели да допуни своју LFIB табелу. У овом случају низводни[5] рутер не мора бити следећи рутер (скок) израчунат IP протоколом рутирања, што се нарочито користи код експлицитног рутирања тј. за креирање експлицитних LSP путања.

Предност првог начина (режима рада) је мање заузеће меморије рутера. Недостатак је што ће рутер у случају промене топологије мреже бити принуђен да шаље захтев суседу да би доделио одговарајућу ознаку жељеној FEC класи. Последица је повећано време прилагођавања на ту



Слика 12.9 Размена ознака без претходног захтева:
а) без задржавања б) са задржавањем

[1] *Upstream*
[2] *Conservative label retention mode*
[3] *Lliberal label retention mode*
[4] *Downstream on demand*
[5] *Downstream*

промену. Предност другог начина је могућност брзог прилагођавања на промене у топологији уколико неки од суседа постане следећи скок за посматрану FEC класу. Слика 12.9 илуструје разлике између ова два начина рада.

И једна и друга метода размене ознака могу истовремено бити имплементиране у мрежи, али се приликом успостављања суседства рутери морају договорити коју ће од ове две методе користити.

12.6.2 Шеме за постављање LSP путања

Код MPLS система могуће је довести до тога да се LSP путања успостави преко LSR рутера одређеним редоследом. Две шеме се могу користити за подешавање LSP путање: независно управљање[1] и наређено (или уређено) управљање[2] LSP путањом.

Код независног управљања LSP путањом LSR рутери могу ознаке придруживати чим препознају нову FEC класу. Информације о додељеним ознаке потом оглашавају својим суседима. Овај начин сличан је класичном IP рутирању, где сваки рутер независно доноси одлуку где ће проследити примљени пакет. Предност независног придруживања ознака је у брзом успостављању LSP путање, пошто се придруживање може остварити паралелно између већег броја рутера па се пренос података може започети пре него што се све ознаке придруже. Недостатак независног придруживања је могуће формирање рутирајућих петљи које захтевају употребу механизама за њихово откривање и отклањање.

Други начин доделе ознака подразумева да LSR рутер везује ознаку за одговарајућу FEC класу само у случају да је он за њу излазни[3] рутер или у случају да је већ примио ознаку од свог низводног рутера за посматрану FEC класу[4]. Овај начин успостављања LSP путање подразумева размену ознака у супротном смеру у односу на ток саобраћаја, тачније од излазног до улазног рутера. Користи се при креирању експлицитних рута.

---

[1] *Independent LSP control*
[2] *Ordered LSP control*
[3] *Egress*
[4] *Ordered control*

12.6.3 Агрегација FEC класа

У пракси се често дешава да се саобраћај који припада различитим FEC класама преноси истом путањом кроз мрежу. Ради упрошћавања и бржег прегледа LFIB табеле, MPLS управљачки протокол омогућава агрегацију FEC класа креирањем јединствене LSP путање.

12.6.4 Спајање више токова саобраћаја у један ток

LSR рутер има могућност да пакете са различитим ознакама, које прима путем различитих улазних интерфејса, прослеђује преко истог излазног интерфејса, додељујући им заједничку ознаку. Као резултат настаје више тачака-тачка[1] LSP путања. Спајање више токова саобраћаја у један врши се када ти токови имају исто одредиште.

12.7 Сигнализациони протоколи за дистрибуцију MPLS ознака

MPLS архитектура описана у документу RFC3031 не прописује као обавезан ниједан протокол за дистрибуцију ознака. Постоји већи број протокола сваки са одређеним предностима и недостацима. У овом одељку биће описани:

- протокол за дистрибуцију MPLS ознака (LDP[2] протокол),
- протокол за рутирање на основу ограничења (CR-LDP[3] протокол),
- протокол за резервацију ресурса и обликовање саобраћаја (RSVP-TE[4] протокол).

12.7.1 LDP протокол

LDP протокол користи се за успостављање и одржавање повезаности ознака одређене LSP путање са FEC класом. Два LSR рутера који користе LDP протокол да би разменили информације о повезаности ознака називају се парњаци (равноправни[5]). LDP протокол проистекао је из протокола компаније CISCO[6] и подразумева размену неколико врста LDP порука између LSR рутера. То су:

---

[1] *Multipoint-to-point*
[2] *Label Distribution Protocol*
[3] *Constraint-Based Routing LDP*
[4] *Resource Reservation Protocol - Traffic Engineering*
[5] *LDP peers*
[6] *Tag Switching*

- *Discovery* порука - периодична размена *Hello* порука ради оглашавања и верификације директно и недиректно повезаних LSR рутера.
- *Session* порука - за успостављање, иницијализацију, одржавање и раскид сесија између равноправних (парњак) LSR рутера.
- *Advertisement* порука - за креирање, промену и брисање повезаност (мапирања) FEC класа-ознака[1].
- *Notification* порука - за размену савета и информисање о грешкама.
- *LDP Discovery* поруке - размењују се коришћењем UDP протокола. Остале врсте порука користе TCP протокол. Ради повећања сигурности може се користити MD5[2] аутентификација.

Сваки LSR рутер мора обавити резервацију простора одређене величине у који ће смештати ознаке. Ово резервисање је неопходно, с обзиром на то да различите врсте уређаја и интерфејса подржавају и различите врсте и различит број ознака. Простор за ознаке може се резервисати:

- на бази самог интерфејса[3]
  - простор ознаке постоји за сваки интерфејс понаособ, или
- на бази целе платформе[4]
  - сви интерфејси деле један исти простор ознака.

LDP идентификатор, величине 6 бајтова указује на величину овог простора. Прва четири бајта чине идентификатор LSR рутера, док преостала два указују на величину простора за ознаке.

LDP сесија се може успоставити између равноправних LSR рутера који јесу директно повезани парњаци и који нису директно повезани. Режим рада на бази интерфејса може се применити када су равноправни LSR рутери директно повезани. LDP сесија између парњака који нису директно повезани користи се код виртуелних приватних мрежа (VPN[5]).

За обавештавање о повезаности FEC класа и ознака LDP протокола користе се две врсте порука:

---

[1] *FEC to-label*
[2] *Message Digest 5*
[3] *Per logical/phisical*
[4] *Per platform*
[5] *Virtual Private Network*

- *Label mapping* (мапирање ознаке) и
- *Label request* (захтев за ознаком).

Када LSR рутери међусобно сазнају једни за друге и успоставе сесију, почиње међусобна размена *Label mapping* порука које садрже FEC класу, ознаку и одређене опционе параметре. Код размене ознака на захтев између рутера се размењују *Label request* поруке које садрже FEC класу и одређене опционе параметре. Дефинисана је као подршка дистрибуцији ознака на захтев.

LDP протокол дефинише и поруке за специфичне ситуације:

- *Label withdraw* (повлачење ознаке) - овом поруком LSR рутер захтева од другог (равноправног) LSR рутера да престане да користи раније додељену ознаку.
- *Label release* (ослобађање ознаке) - овом поруком LSR обавештава да му претходно примљена или захтевана ознака није више потребна.
- *Label abort request* (прекид захтева за ознаком) - овом поруком прекида се текући захтев за ознаком.

12.7.2 Пример употребе LDP протокола

Пример приказан на слици 12.10 илуструје употребу порука *Label request* и *Label mapping* у размени ознака на захтев са независном доделом (управљањем) ознакама[1]. LDP поруке размењују се између равноправних LSR рутера између којих је успостављена LSP путања: од улазног рутера R1, преко R2 рутера до излазног R3 рутера. Путања је успостављена за FEC класу која је специфицирана адресним префиксом a.b/16.

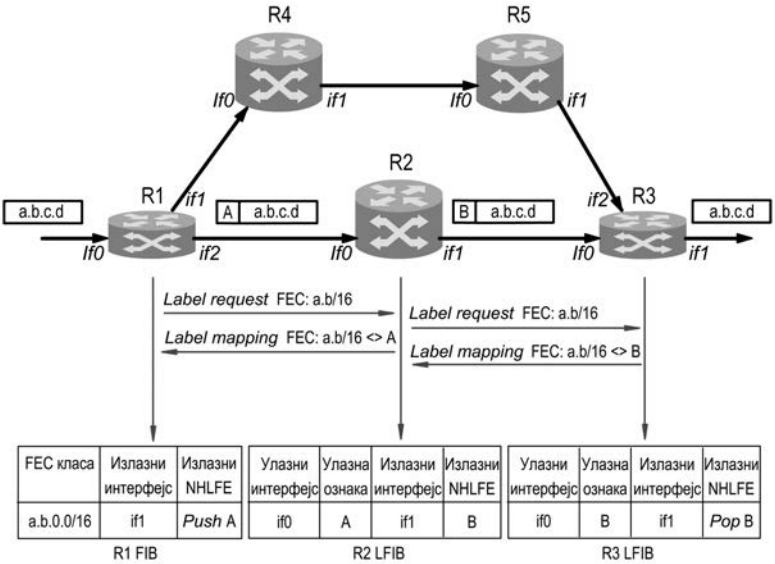
У овом примеру R1 рутер иницира процес захтевајући ознаку за FEC класу a.b./16 од свог следећег скока (R2 рутера).

Пошто се користи независна додела ознака[2] R2 рутер може одговорити R1 рутеру пре него што прими одговарајуће мапирање ознаке од свог низводног рутера - R3 рутера. Исте поруке размењују се и између R2 и R3 рутера, након чега се формира LSP путања. Акцију стављања (*push*) врши улазни ивични рутер који примљеном IP пакету додаје ознаку.

---

[1] LDP *Downstream on Demand Independent Control*

[2] *Independent control*



Слика 12.10 Пример LDP рутирања

Акцију уклањања (*pop*) врши излазни ивични рутер који уклања ознаку и оригинални пакет прослеђује IP мрежи.

LDP подржава и друге начине дистрибуције ознаке. Тако на пример, код режима у коме се ознаке шаљу без претходног захтева, рутери не користе *Label request* поруке, већ само *Label mapping* поруке којима својим суседима оглашавају ознаке које су доделили појединачним FEC класама.

12.7.3 Протокол за рутирање на основу ограничења

Протокол за рутирање на однову ограничења (CR-LDP[1] протокол) у основи функционише као класичан LDP протокол. Могуће је експлицитно рутирање, уговарање параметара саобраћаја, коришћење класа и приоритета саобраћаја. У суштини, CR-LDP протокол сличан је RSVP-TE протоколу али није компатибилан са њим. Ради успостављања LSP путање на којој је могуће вршити расподелу саобраћаја CR-LDP протокол захтева рад у следећим режимима рада:

[1] *Constraint-Based Routing LDP*

- размена ознака између LSR рутера на захтев,
- наређено управљање LSP путањом,
- размена ознака између LSR рутера без претходног захтева, где уколико суседни рутер није узводни добијена информација неће бити задржана.

12.7.4 RSVP-TE протокол

RSVP-TE протокол је проширење основног RSVP протокола који је пројектован тако да подржи архитектуру интегрисаних услуга[1]. Архитектура интегрисаних услуга специфицира квалитет услуге (QoS) тако да различите врсте саобраћаја (посебно аудио и видео) имају несметан пролаз кроз мрежу. RSVP поруке шаљу се или као IP пакет са вредношћу 46 у пољу за протокол или се укалупљују у UDP датаграм. Идеја архитектуре интегрисаних услуга је да сваки рутер који се налази дуж тока саобраћаја може обрађивати RSVP поруке. Због тога употреба RSVP протокола за дистрибуцију ознака код MPLS захтева додатне сигурносне механизме. Један од њих је MD5 аутентификација а други конфигурисање филтера на не-MPLS интерфејсима којима се блокирају RSVP поруке.

Иако у основи RSVP протокол користи мултикаст код MPLS сигнализације користи се уникаст комуникација. Принцип функционисања RSVP подразумева успостављање резервација за једносмерни[2] ток пакета. RSVP поруке преносе се дуж путање скок-по-скок[3] (израчунате IP протоколом рутирања) осим ако није извршено експлицитно рутирање.

Поруке RSVP-TE протокола

Постоје три основне врсте RSVP порука:

- *Reservation setup* - поруке за подешавање резервације
- *Tear down* - поруке за раскид
- *Error* - поруке о грешци

Поред ових, RSVP-TE дефинише и *Hello* поруке.

---

[1] *Integrated services*
[2] *Unidirectional*
[3] *Hop-by-hop*

RSVP поруке за подешавање резервације[1]

Најпре пошиљалац шаље *Path* поруку којом се идентификују ток и карактеристике саобраћаја. Ова порука садржи објекте као што су: идентификатор сесије, ознака пошиљаоца, захтев за ознаком, *Tspec* (група описивача саобраћаја) и опционо објекат експлицитног рутирања (ERO[2]). Идентификатор сесије садржи одредишну IP адресу упарену са 16-битним идентификатором LSP тунела. Код примене LSP тунела, једино улазни рутер мора знати која FEC класа припада ком тунелу. Због тога, за разлику од LDP, мапирање FEC класе на одговарајући LSP тунел није укључено ни у једној RSVP поруци. Ознака пошиљаоца[3] садржи IP адресу пошиљаоца упарену са идентификатором LSP путање. Захтев за ознаком указује да је подржан режим рада „размене на захтев". MPLS обезбеђује услугу контроле оптерећења[4] коришћењем *Tspec*[5] описивача који садрже жетоне за дефинисање брзина и максималну величину пакета.

Када *Path* порука стигне до свог одредишта прималац поруке одговара *Resv* поруком уколико жели да иницира доделу ознаке која је захтевана. *Resv* порука се преноси истом путањом као и *Path* порука, али у супротном смеру. Она такође садржи идентификатор сесије (преузет из одговарајуће *Path* поруке), опциони објекат за снимање руте[6] (којим се памти рута LSP путање почев од излазног ка улазном рутеру) и информацију о стилу резервације. RSVP дефинише два начина резервације:

- са фиксним филтром (FF[7]) - постоји посебна ознака и *Tspec* описивачи за сваки пар пошиљалац-прималац;
- са експлицитним разликовањем (SE[8]) - постоји посебна ознака за сваког пошиљаоца али сви користе исте описиваче тока саобраћаја.

[1] *RSVP Reservation Setup*
[2] *Explicit route object*
[3] *Sender template*
[4] *Controlled load service*
[5] *Traffic Spec*
[6] *Route record object*
[7] *Fixed Filter*
[8] *Shared Explicit*

RSVP-TE поруке за раскид

RSVP-TE дефинише две врсте порука за раскид (*Tear Down* поруке):

- *Path Tear* порука и
- *Resv Tear* порука.

Обе поруке шаљу се у супротном смеру у односу на одговарајућу *Path* и *Resv* поруку. *Tear* поруке могу се користити за прекид било каквог успостављеног стања, у случају да је дошло до грешке па се мора обавити поновно рутирање.

RSVP-TE поруке о грешкама

Постоје и две RSVP-TE врсте порука за објављивање грешке:

- *Path Error* порука и
- *Resv Error* порука.

Ако LSR рутер закључи да не може да подржи захтеване *Tspec* описиваче наведене у *Resv* поруци он је неће проследити свом узводном рутеру. Уместо тога свом низводном рутеру послаће *Resv Error* поруку да би прекинуо покушај успостављања LSP путање.

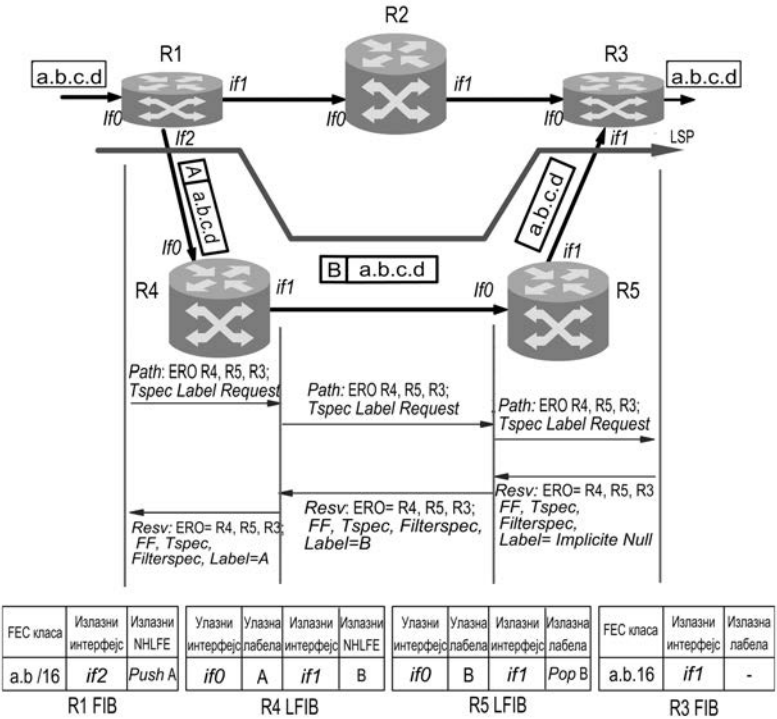
Hello поруке су опционе и омогућавају LSR рутерима да брзо детектују грешку код суседа како би се што пре обавило прерутирање.

12.8 Пример експлицитног рутирања на захтев[1]

У примеру са слике 12.11, поруке за успостављање LSP путање садрже објекат експлицитног рутирања (ERO[2]). Овај објекат указује на путању другачију од оне израчунате неким IP протоколом рутирања. Рутер R1 везује FEC класу a.b/16 за одговарајући LSP тунел и израчунава експлицитну руту до одредишта преко рутера R4, R5 и R3. Рутер R1 иницира успостављање овог LSP тунела слањем *Path* поруке рутеру R4. Ова порука садржи експлицитну руту, *Tspec* описиваче, ознаку пошиљаоца и захтев за ознаком. Примљену *Path* поруку рутер R4 прослеђује следећем рутеру у ERO рути тј. рутеру R5 који је даље шаље излазном рутеру R3.

[1] *Downstream on Demand Ordered Control*

[2] *Explicit Route Object*



FEC класа	Излазни интерфејс	Излазни NHLFE
a.b /16	*if2*	*Push* A

R1 FIB

Улазни интерфејс	Улазна лабела	Излазни интерфејс	Излазни NHLFE
if0	A	*if1*	B

R4 LFIB

Улазни интерфејс	Улазна лабела	Излазни интерфејс	Излазна лабела
if0	B	*if1*	*Pop* B

R5 LFIB

FEC класа	Излазни интерфејс	Излазна лабела
a.b.16	*if1*	-

R3 FIB

Слика 12.11 Пример RSVP-TE рутирања

Примивши *Path* поруку рутер R3 закључује да линк између R3 и R5 може задовољити захтеване параметре и одговара слањем *Resv* поруке која садржи ERO, *Tspec* описивач резервисаног капацитета, спецификацију филтера и тзв. *null* ознаку. Ова ознака указује да је рутер R3 директно повезан на мрежу a.b/16, што је сигнал његовом низводном рутеру R5 да уклони ознаку у пакетима намењеним тој мрежи. Рутер R5 потом шаље *Resv* поруку рутеру R4 додељујући ознаку B. Рутер R4 прихвата захтев, додељује ознаку A и шаље *Resv* поруку назад до рутера R1.

На овај начин обављено је успостављање тунела. Може се приметити да ниједна RSVP-TE порука (за разлику од LDP) не садржи FEC класу пошто само рутер R1 врши мапирање између FEC класе и LSP тунела.

12.9 Виртуелне приватне мреже

Виртуелне приватне мреже (VPN[1]) су мреже које емулирају приватне мреже користећи заједничку инфраструктуру. Приватне мреже захтевају да све локације корисника могу међусобно да комуницирају али да су истовремено потпуно одвојене од других виртуелних приватних мрежа. Виртуелне приватне мреже обично су у власништву једне организације која има своје делове на различитим локацијама, који су међусобно повезани заједничком инфраструктуром добављача интернет услуга (интернет посредника).

Добављачи интернет услуга обезбеђују у највећем броју случајева два модела виртуелних мрежа за своје кориснике:

- модел са прекривањем[2] и
- модел парњак-парњак[3].

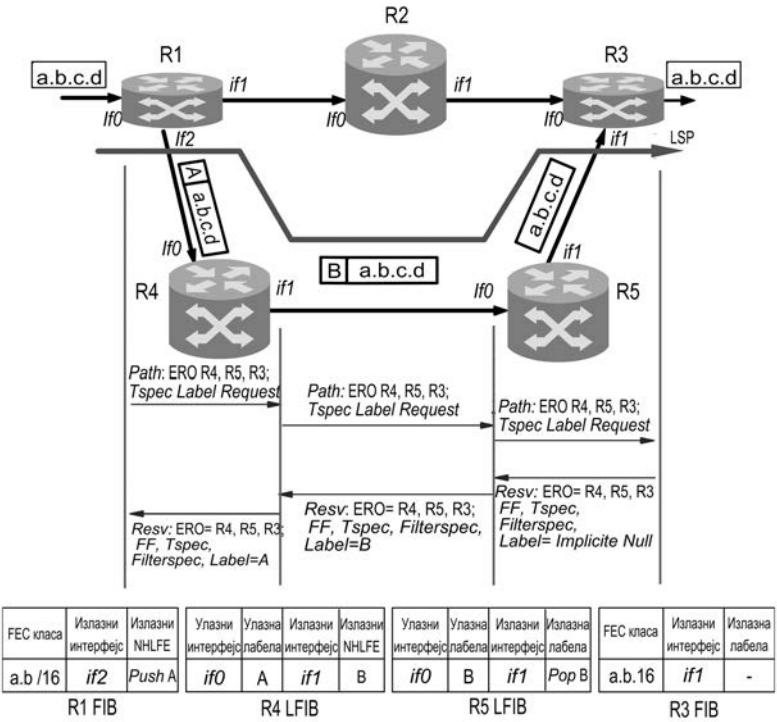
12.9.1 Модел виртуелних приватних мрежа са прекривањем

Када се користи модел виртуелних приватних мрежа са прекривањем интернет посредник између рутера корисника обезбеђује тачка-тачка везе (линкове) или виртуелне канале[4]. Рутери корисника примењују рутирање између равноправних учесника[5] директно преко линкова или виртуелних канала интернет посредника. Рутери или комутатори интернет посредника преносе податке корисника користећи своју мрежу. Не постоји рутирање равноправних између рутера корисника и рутера Интернет посредника.

Ова тачка-тачка услуга може бити реализована на 1, 2. или чак и 3. слоју. Пример услуге 1. слоја је временски мултиплекс (TDM[6]), E1, E3, SONET[7], SDH[8]. Пример услуге 2. слоја су виртуелни канали који се формирају код X.25 система са комутацијом пакета, код система са

---

[1] *Virtual Private Network*
[2] *Overlay VPN model*
[3] *Peer-to-Peer*
[4] *Virtual circuits*
[5] *Routing peering*
[6] *Time Division Multiplex*
[7] *Synchronous Optical Networking*
[8] *Synchronous Digital Hierarchy* (SDH) и SONET су стандардизовани протоколи за мултиплексирање који омогућавају пренос више дигиталних токова података преко оптичких влакана.



FEC класа	Излазни интерфејс	Излазни NHLFE
a.b /16	*if2*	*Push* A

R1 FIB

Улазни интерфејс	Улазна лабела	Излазни интерфејс	Излазни NHLFE
if0	A	*if1*	B

R4 LFIB

Улазни интерфејс	Улазна лабела	Излазни интерфејс	Излазна лабела
if0	B	*if1*	*Pop* B

R5 LFIB

FEC класа	Излазни интерфејс	Излазна лабела
a.b.16	*if1*	-

R3 FIB

Слика 12.12 Модел виртуелне приватне мреже са прекривањем изграђен на мрежи са штафетним преносом рамова

комутацијом ћелија (ATM[1]) и код штафетног преноса рамова (FR[2]). На слици 12.12 приказан је пример модела мреже са прекривањем који је изграђен на мрежи са штафетним преносом рамова.

Узимајући у обзир протокол слоја 3 (IP протокол) и комуникацију парњака са гледишта корисника изгледа као да су рутери директно повезани.

Услуга прекривања такође се може постићи и преко мреже са IP протоколом 3. слоја. Најчешће коришћена врста тунела којом се може изградити мрежа са прекривањем је помоћу тунела са генерички

[1] *Asynchronous Transfer Mode*

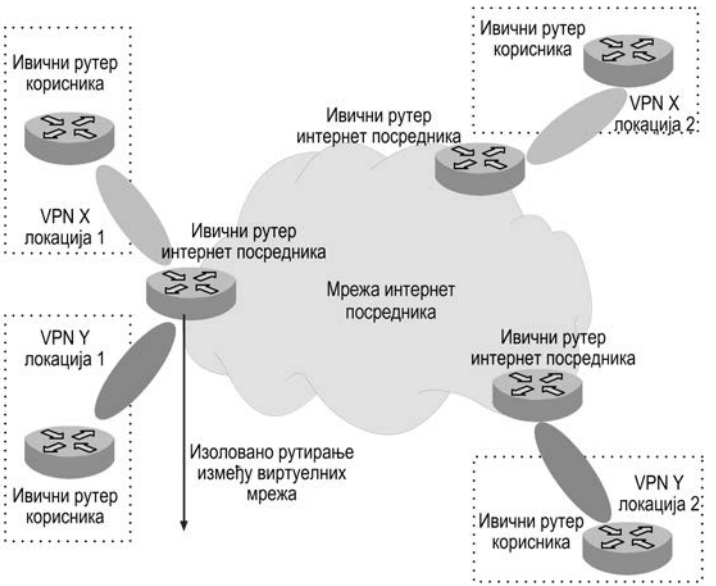
[2] *Frame relay*

укалупљеним рутирањем (GRE[1] тунели). Ови тунели укалупљују саобраћај са GRE заглављем. Указују и који се од транспортних протокола користи. IP заглавље се користи за рутирање кроз мрежу интернет посредника. Предност GRE тунела је да они могу да усмеравају и друге пакете (не само IP пакете).

12.9.2 Парњак-парњак модел виртуелне приватне мреже

Код парњак-парњак модела виртуелне приватне мреже рутери интернет посредника преносе податке корисника кроз мрежу али учествују и у процесу рутирања корисничких мрежа. Другим речима, рутери посредника се на трећем слоју повезују директно са рутерима корисника.

Резултат је да постоји однос непосредног суседства[2] код протокола за рутирање између рутера посредника и рутера корисника. На слици 12.13 приказан је парњак-парњак модел виртуелне приватне мреже.



Слика 12.13 Парњак-парњак модел виртуелне приватне мреже (VPN)

[1] *Generic Routing Encapsulation*

[2] *Neighborship, adjacency*

Пре него што је MPLS мрежа уопште постојала модел парњак-парњак виртуелне приватне мреже могао се постићи упаривањем[1] рутера корисника и рутера интернет посредника. Такође, модел виртуелне приватне мреже захтева приватност или међусобну изолацију различитих корисника. Она се може постићи филтрирањем пакета[2] (листама за приступ[3]) тако да се контролишу подаци који се преносе од рутера корисника и ка рутеру корисника.

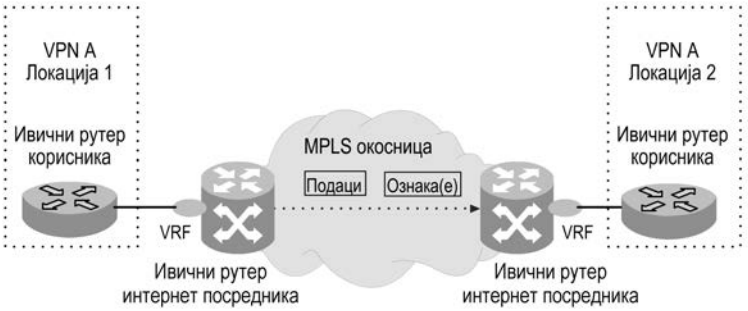
Други начин да се достигне неки од облика приватности је да се конфигуришу филтри рута тако да се оглашавају руте или да се заустави оглашавање рута рутерима корисника. Оба метода могу се и истовремено применити.

Модел виртуелне приватне мреже са прекривањем примењиван је чешће него парњак-парњак модел пре него што су почеле MPLS мреже да се користе. Парњак-парњак модел виртуелне приватне мреже захтевнији је пошто додавање једне корисничке локације захтева много промена у конфигурацијама на великом броју локација. MPLS виртуелна приватна мрежа једна је од примена MPLS система која је омогућила да се парњак-парњак модел може једноставније применити.

Са MPLS виртуелним приватним мрежама један кориснички рутер, који се означава као кориснички ивични рутер (CE[4] рутер) комуницира на IP слоју најмање са једним равноправним рутером интернет посредника (PE[5] рутер).

Приватност у MPLS виртуелним приватним мрежама постиже се концептом виртуелног рутирања/преусмеравања (VRF[6]) и чињеницом да се подаци окосницом[7] преусмеравају као означени пакети. Систем виртуелног преусмеравања/рутирања обезбеђује да се информације о рутирању различитих корисника воде посебно. Значи да примена MPLS протокола у окосници мреже обезбеђује да се пакети преусмеравају на основу ознака а не на основу IP адреса (слика 12.14).

[1] *Peering*
[2] *Packet filters*
[3] *Access Lists*
[4] *Customer Edge*
[5] *Provider Edge*
[6] *Virtual Routing/Forwarding*
[7] *Backbone*



Слика 12.14 MPLS виртуелна приватна мрежа са виртуелним рутирањем/преусмеравањем (VRF)

Када се додаје једна корисничка локација за њу на PE рутеру треба обезбедити само упаривање[1] са CE рутером. Нема потребе за креирањем већег броја виртуелних кола као што је то потребно код модела са прекривањем. Није неопходно ни конфигурисање филтера пакета или филтера рута, што се примењује код парњак-парњак модела над IP мрежом и што представља велику предност MPLS виртуелне приватне мреже за интернет посреднике.

Највећи број корисника има мрежу звездасте топологије са концентратором[2]. Други корисници имају потпуно повезану[3] мрежу око окоснице добављача Интернет услуга. Трећи корисници имају топологију која је нешто између. Предност MPLS виртуелне приватне мреже за кориснике је највећа када имају потпуно повезану мрежу.

Посматрајмо потпуно повезану корисничку мрежу око мреже са штафетним преносом рамова (слика 12.12) и упоредимо је са истом корисничком мрежом повезаном MPLS виртуелном приватном мрежом (слици 12.15). На слици 12.12 сваки ивични рутер корисника упарен је са *n*-1 других граничних рутера корисника (*n* је укупан број граничних рутера корисника). Види се на слици 12.15 да је сваки кориснички рутер упарен само са једним рутером – граничним рутером интернет посредника.

[1] *Peering*
[2] *Hub and Spoke*
[3] *Fully mashed*

Слика 12.15 Парњак-парњак модел виртуелних приватних мрежа примењен у MPLS виртуелној приватној мрежи

Друга предност за интернет посреднике је да он треба да обезбеди само линк између PE и CE рутера. Код модела прекривања интернет посредник треба да обезбеди везе (линкове) или виртуелне канале између свих локација. Много је једноставније предвидети саобраћај а самим тим захтевани пропусни опсег за једну локацију од предвиђања комплетног саобраћаја између свих корисничких локација.

Постоје и одређени недостаци модела виртуелних приватних мрежа парњак-парњак у односу на модел са прекривањем:

- корисник мора да дели одговорност за рутирање са добављачем интернет услуга,
- ивични уређаји добављача интернет услуга имају додатно оптерећење.

Први недостатак је да корисник мора да има рутирање парњак-парњак са интернет посредником. Корисник више не контролише своју мрежу с краја на крај[1] на 3. слоју ни у погледу IP рутирања, као што је то случај са моделом прекривања.

[1] *End to End*

Други недостатак односи се на интернет посредника: додато је оптерећење његовим ивичним рутерима (PE рутерима). Добављач интернет услуга одговоран је за скалабилност и конвергенцију рутирања корисничке мреже пошто PE рутери морају да воде рачуна о свим рутама великог броја корисника и истовремено о временској конвергенцији рутирања.

12.10 Питања и задаци

1. Ко је развио и са којим циљем вишепротоколну комутацију на основу ознака (MPLS)?
2. Набројати и описати компоненте сваког рутера.
3. Како се може описати еквивалентна класа у прослеђивању (FEC класа)?
4. Навести неколико примера FEC класа.
5. Нацртати и описати све компоненте у формату MPLS ознаке.
6. Које операције може да обавља LSR рутер?
7. Објаснити појмове: MPLS чворови, домени и означена комутирана путања (LSP путања).
8. На слици 12.5 представљен је пример MPLS комутације. Објаснити.
9. У табели 12.2 представљен је коначни изглед дела LFIB табела на рутерима за FEC класу 10.0.0.0/8. Објаснити.
10. Како се дефинише и користи низ ознака?
11. Описати како се користи низ ознака у MPLS мрежи са тунелом.
12. Каква је разлика између рутирања најкраћом путањом и рутирања на основу ограничења?
13. Која је намена LDCP протокола?
14. На који се начин информације о ознакама међу рутерима могу размењивати?
15. Описати сигнализационе протоколе за дистрибуцију ознака.
16. У мрежи на слици 12.8а активиран је протокол за рутирање најкраћом путањом. Описати проблем који може да настане због неравномерног оптерећења линкова код саобраћаја већег интензитета од капацитета линкова.
17. Нацртати и описати како изгледа расподела саобраћаја у примеру са слике 12.8 када је примењен принцип рутирања на основу ограничења

18. MPLS архитектура дозвољава креирање LSP путања које се не морају подударати са путањом скок по скок:
    а) како се означава ова врста рутирања,
    б) када се користи оваква врста рутирања,
    в) где се примењује оваква врста рутирања?
19. Које врсте експлицитног рутирања постоје?
20. Каква је разлика између стриктно и слободно експлицитног рутирања?
21. Описати и међусобно упоредити начине на који се могу размењивати информације између LSR рутера.
22. Анализирати пример примене LDP протокола представљеног на слици 12.10.
23. Које поруке користи и на који начин RSVP-TE протокол?
24. Анализирати пример експлицитног рутирања на захтев.
25. Како се дефинишу виртуелне приватне мреже?
26. Која два најчешћа модела виртуелних приватних мрежа користе добављачи интернет услуга?
27. Описати модел виртуелних приватних мрежа са прекривањем.
28. На којим слојевима се може реализовати виртуелна приватна мрежа? Навести примере.
29. Анализирати виртуелну приватну мрежу са прекривањем изграђену у мрежама са штафетним преносом рамова (слика 12.12).
30. Описати парњак-парњак виртуелне мреже са прекривањем.

13. Бежичне мреже

Бежичне мреже су према домету категоризоване у четири групе: бежичне личне мреже (WPAN[1]), бежичне локалне рачунарске мреже (WLAN[2]), бежичне рачунарске мреже градских подручја (WMAN[3]) и бежичне рачунарске мреже ширих географских подручја (WWAN[4]).

13.1 Бежичне локалне рачунарске мреже

Први експеримент бежичног повезивања рачунара реализован је 1970. године у лабораторијама IBM-а у Швајцарској.

Прва бежична рачунарска мрежа ALOHA[5] пуштена је у рад 1971. године. Мрежа се састојала од седам рачунара постављених на четири острва и сервера са којим су рачунари комуницирали бежичном везом. Ова мрежа представља први јавно представљени рад бежичне пакетске мреже за пренос података.

Развој бежичних мрежа наставља се 1985. године пошто је Федерална комисија за комуникације (FCC[6]) одобрила коришћење опсега радиофреквенцијског спектра без посебних дозвола за потребе индустрије, научних истраживања и медицине (ISM[7] опсег). Овај фреквенцијски опсег обухвата три групе фреквенција: 902 - 928MHz; 2,4 - 2,4835GHz и 5,725 - 5,875GHz (слика 13.1).

Модуларност и флексибилност бежичних локалних рачунарских мрежа чини их добрим решењем за поједине објекте као на пример: повезивање рачунара на сајмовима, у болницама, библиотекама,

---

[1] *Wireless Personal Area Network*

[2] *Wireless Local Area Network*

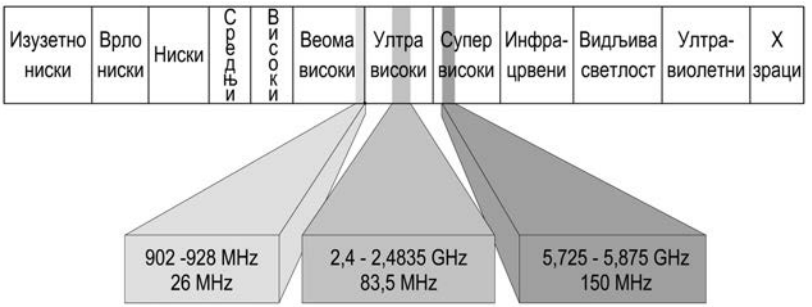
[3] *Wireless Metropolitan Area Network*

[4] *Wireless Wide Area Network*

[5] ALOHA значи здраво на хавајском. Назив се користи и за метод приступа трансмисионом медијуму.

[6] *Federal Communications Commission*, тело у САД које регулише коришћење радио спектра

[7] *Industrial Scientific Medical*



Слика 13.1 ISM опсега који се користи за бежичне локалне рачунарске мреже

повезивање зграда које су раздвојене прометним саобраћајницама или реком, у зградама од историјске вредности у којима није дозвољена промена изгледа зграде, али и као допуна или проширење кабловских мрежа.

13.2 Топологије бежичних локалних рачунарских мрежа

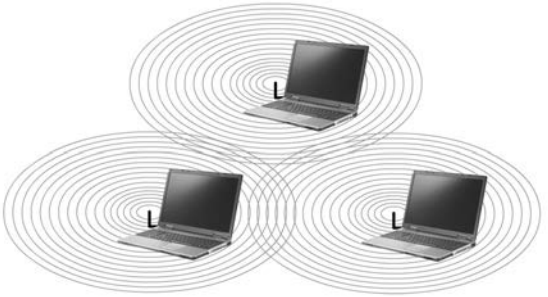
Бежичне мреже пружају велику флексибилност и могућност комбиновања различитих начина повезивања. Искристалисале су се следеће топологије: само за ову прилику (*ad hoc*), инфраструктурна, ћелијска, испреплетана[1] и тачка-тачка.

13.2.1 Топологија само за ову прилику

Топологија само за ову прилику (*ad hoc*) локалне рачунарске мреже (слика 13.2) омогућава корисницима успостављање везе „свако са сваким“ без уређаја за приступ (AP[2]). Као такве погодне су за комуникацију између мањих група корисника на малом растојању и углавном се примењују тамо где је потребно брзо успоставити привремену мрежу[3].

Да би могла да се оствари међусобна комуникација потребно је да сваки рачунар буде у домету свих осталих рачунара, што ограничава покретљивост корисника на релативно мали простор.

[1] *Mesh*
[2] *Access Point*
[3] Нпр. састанци, скупови, сајмови.



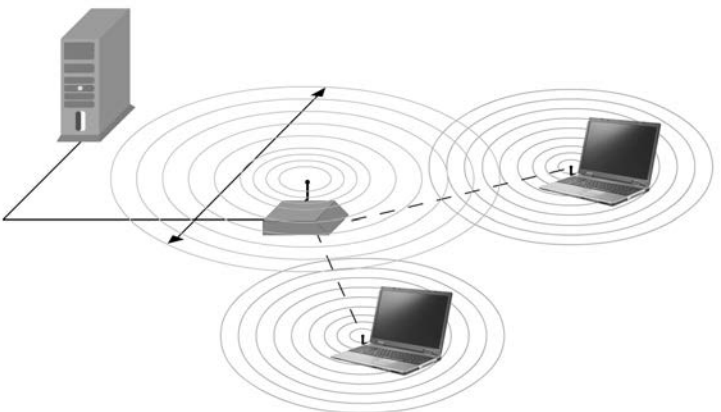
Слика 13.2 *Ad hoc* начин међусобног повезивања рачунара

13.2.2 Инфраструктурна топологија

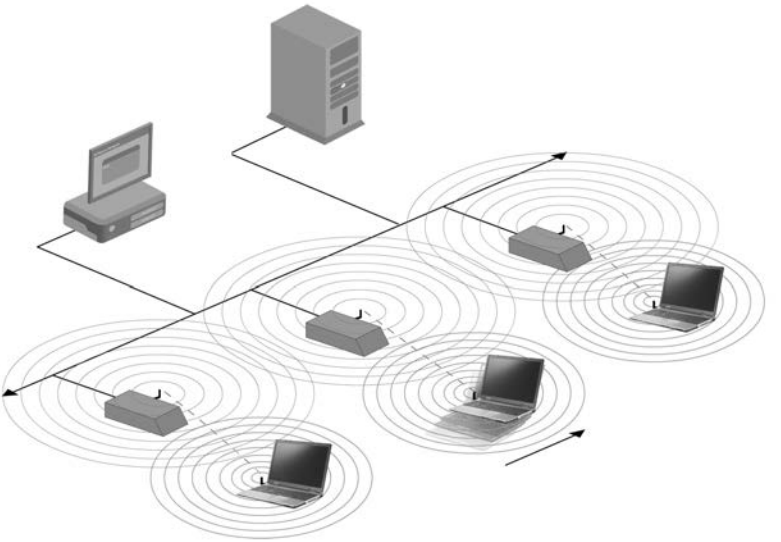
Уколико станице нису у стању да комуницирају без посредства уређаја за приступ (AP) онда се примењује начин рада који захтева одговарајућу инфраструктуру (слика 13.3). Све станице које се налазе у зони покривања уређаја за приступ међусобно комуницирају преко њега.

13.2.3 Ћелијска мрежа

У случају да се жели повећати покривеност може се инсталирати више уређаја за приступ и формирати ћелијска (целуларна) мрежа (слика 13.4). У таквој мрежи станице не могу да комуницирају директно међу собом већ искључиво посредством уређаја за приступ који могу бити међусобно



Слика 13.3 Међусобно повезивање рачунара помоћу уређаја за приступ (инфраструктурна топологија)



Слика 13.4 Ћелијска мрежа са инфраструктурним повезивањем уређаја за приступ

повезани жично или бежично. Одређеним бројем приступних станица може се покрити одређено подручје, при чему се зоне покривања појединих станица, тзв. ћелије, могу међусобно делимично и преклапати. У случају да је у одређеној ћелији велика густина саобраћаја могуће је исту површину покрити са више приступних станица, које међусобно деле мрежни саобраћај и на тај начин омогућавају опслуживање већег броја корисника.

Карактеристика ћелијских мрежа је да корисници могу бити покретљиви у току рада, мењајући приступне станице преко којих комуницирају са остатком мреже. Уређаји за приступ (AP) везују се на ожичену мрежу[1].

13.2.4 Испреплетана мрежа

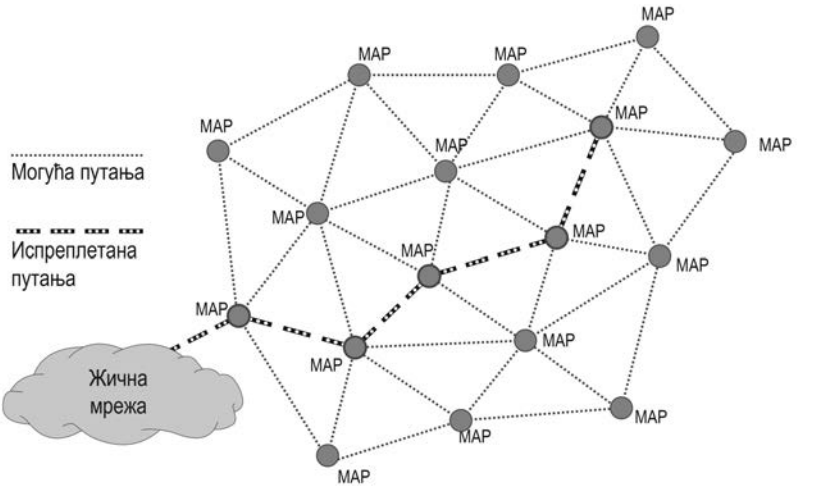
Приступне тачке које се могу бежичним путем међусобно повезивати означавају се као испреплетане бежичне тачке (MAP[2]) и формирају испреплетану бежичну мрежу[3] (слика 13.5).

---

[1] *Кичму - Backbone*

[2] *Mesh Access Point*

[3] Прецизније би било: топологију бежичне испреплетане локалне мреже (*wireless local area network mesh topologies*) а према документу IEEE802.11s за испреплетане локалне мреже.



Слика 13.5 Испреплетана топологија

Са повећаном применом бежичних мрежа појавила се потреба да се обезбеди бежични приступ на местима где није могуће повезивање приступне тачке на комутаторе ожичене мреже. Дужина етернет кабла је ограничена на 100m што отежава постављање приступних тачака унутар пространих објеката. Ситуација је још сложенија код повезивања ван објеката код јавних услуга, општина или хитних служби.

Идеја да се бежичним линковима замене неки жични линкови (нпр. Етренет[1]) једнако је стара као и стандарди за бежичне локалне мреже. Замена каблова бежичним линковима доноси многе предности. Навешћемо неке од њих:

- Повећана флексибилност бежичних линкова над жичним. Неколико линкова између приступних тачака испреплетане мреже формирају преносни систем тако да се подаци могу пренети до жичне мреже.

[1] 1972. године Роберт Меткалф (*Robert Metcalfe*) и његове колеге из компаније Xerox PARC развиле су први експериментални етернет систем за повезивање *Xerox Alto* радну станицу са графичким интерфејсом. Назив ове експерименталне мреже *Alto Aloha Network* је Меткалф променио 1973. год. у Етернет (*Ethemet*) да би показао да систем може да обезбеди повезивање било ког рачунара, а не само *Alto* рачунара. Име потиче од речи етар (*ether*) која описује суштинску карактеристику система: физички медијум (нпр. кабл) преноси бите ка свим станицама практично на исти начин као што се некада сматрало да стари етар (енг. *Luminiferous ether/aether* - пренос светлости) преноси електромагнетске таласе кроз простор.

- Самоорганизовање[1]. Уколико су алгоритми за детекцију најбоље путање ка жичној мрежи имплементирани у испреплетаним приступним тачкама онда је прављење или проширивање бежичне мреже веома једноставно.
- Могућност поновног самоопоравка[2]. У случају отказа или искључења неке од приступних тачака аутоматски се проналази нова, најбоља путања.

13.3 Технологије бежичних локалних рачунарских мрежа

Бежичне локалне рачунарске мреже (WLAN) према технологији (техници) преноса генерално се могу поделити на следеће категорије:

- пренос у инфрацрвеном делу спектра (IR[3]),
- пренос проширивањем спектра сигнала (SS[4]),
- ортогонални фреквенцијски мултиплекс (OFDM[5]),
- пренос са више антена и више праваца пропагације таласа (MIMO[6]) и
- ускопојасне микроталасне технике[7].

Нежељена преслушавања између канала избегавају се фреквенцијским планирањем, а сигурност и избегавање интерференције постижу се употребом посебних радио-фреквенција. Филтар на пријемнику одстрањује све сигнале осим жељеног.

13.4 Преглед стандарда IEEE 802.11

Стандард IEEE802.11 је скуп стандарда (први објављен 1997. год.) за бежичне локалне рачунарске мреже за рад у три фреквенцијска опсега: 2,4GHz; 3,6GHz и 5GHz.

Стандард је поставио основу за производе за бежичне локалне

[1] *Self forming*
[2] *Self-healing*
[3] *Infrared Technology*
[4] *Spread Spectrum*
[5] *Orthogonal Frequency Division Multiplexing*
[6] *Multiple Input - Multiple Otput*
[7] *Narrowband Microwave*

рачунарске мреже који носе ознаку Wi-Fi[1]. Ознака „Wi-Fi“ је заштићени знак[2] Wi-Fi Alliance и ознака производа који раде по стандарду IEEE802.11. Производи Wi-Fi који су успешно прошли све тестове за међусобни рад[3] које је прописала организација Wi-Fi Alliance могу да користе ознаку „*WiFi certified*”.

Бежичне локалне рачунарске мреже могу да раде у једном од два режима: без базне станице (слика 13.2) и са базном станицом (слике 13.3 и 13.4). У овом одељку биће анализирани скуп протокола[4] IEEE802.11, бежичне технике преноса физичког слоја, протоколи MAC подслоја, формат рама и понуђене услуге.

Организација IEEE је од 1997. године до данас издала многе стандарде[5] за бежичне локалне рачунарске мреже. Стандарди који су до сада нашли широку примену су: IEEE802.11, IEEE802.11a, IEEE802.11b, IEEE802.11g и IEEE802.11n. Основна обележја ових стандарда дата су у табели 13.1

Остали стандарди из серије IEEE802.11 који су развијени или на којима се ради а који неће бити анализирани у овом уџбенику биће дати у табели 13.6 на крају овог поглавља.

13.4.1 Фреквенцијски опсези и канали IEEE 802.11 мрежа

Радна група IEEE802.11 предвидела је употребу три различита фреквенцијска опсега: 2,4 GHz; 3,6 GHz и 4,9/5,0 GHz. Сваки од опсега је подељен у мноштво канала. Држава примењује своја правила (регулативе) о томе који се канали користе и колика је максимална снага предајника у оквиру сваког од канала. У неким државама (нпр. САД) оператери аматерског радија могу да користе предајнике већих снага код бежичних система већег домета.

Постоји 14 канала у опсегу 2,4GHz са међусобним растојањем од 5 MHz (слика 13.6). Изузетак је 14. канал који је 12MHz удаљен од 13. канала.

---

[1] *Wireless Fidelity*

[2] *Trademark*

[3] *Interoperability*

[4] *Protocol stack*

[5] У табели 13.6 (на крају поглавља) дат је списак проширења IEEE802.11 стандарда. Један је стандард IEEE802.11 али се користи термин стандард и за проширење стандарда.

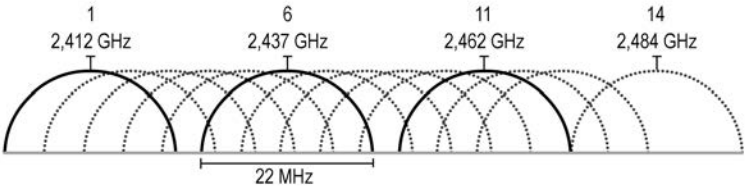
IEEE802.11	Објављен	Учестаност [GHz]	Опсег [MHz]	Брзина података по току [Mb/s]	Дозвољени број MIMO токова	Модулације	Домет у затвореном простору [m]	Домет у отвореном простору [m]
-	1997.	2,4	20	1, 2	1	DSSS, FHSS	20	100
a	1999.	5	20	6, 9, 12, 18, 24, 36, 48, 54	1	OFDM	35	120
		3,7[1]					-	5000
b	1999.	2,4	20	5,5; 11	1	DSSS	38	140
g	2003	2,4	20	6, 9, 12, 18, 24, 36, 48, 54	1	OFDM, DSSS	38	140
n	2009.	2,4; 5	20	7,2; 14,4; 21,7; 28,9; 43,3; 57,8; 65; 72,2	4	OFDM	70	250
			40	15, 30, 45, 60, 90, 120, 135, 150			70	250
ad[2]	2012.	2,4; 5; 60	7000-9000[3]	7000				
ac	~2014.	5	20	До 87,6	8	OFDM		
			40	До 200				
			80	До 433,3				
			160	До 866,7				

Табела 13.1 Основна обележја стандарда серије IEEE802.11

[1] IEEE802.11y-2008 проширио је опсег рада IEEE 802.11 на лиценцирани опсег 3,7GHz. Повећана предајна снага дозвољава опсег од 5km. Дозвољено само у САД.

[2] Означава се и као WiGig, покренут развој од конзорцијума *WiGig Alliance*. Постала је део WiFi групе. У будућности планирају се и веће брзине од 7Gb/s у нелиценцираном опсегу 60GHz.

[3] Различито у различитим земљама



Слика 13.6 Графички приказ канала у опсегу 2,4GHz

У Европи се користи првих 13 канала. У Сједињеним Америчким Државама користе се првих 11 канала док канали 12 и 13 подлежу посебним ограничењима (нпр. мала снага). У Јапану дозвољено је и коришћење 14. канала али само за системе са директном секвенцом (DSSS), и CCK модулације што значи само стандард 802.11b.

Суседни канали међусобно се преклапају и могу да ометају један другог. Поред дефинисања централне учестаности канала IEEE802.11 стандард специфицира и спектралну маску којом је одређена дозвољена расподела снаге сигнала кроз канал.

Маска захтева да је сигнал ослабљен најмање 20dB (за OFDM системе[1]) у односу на своју максималну вредност на одстојању ±11MHz од централне учестаности. Последица је да станице могу да користе сваки четврти или пети канал а да не дође до преклапања.

Да би се спречило међусобно ометање препоручује се коришћење 1, 6. и 11. канала у Северној Америци и 1, 5, 9. и 13. канала у Европи. Коришћење ових канала може се наћи у документу IEEE802.11-2012[2].

13.4.2 Компоненте IEEE802.11 мрежа

У општем случају мрежа IEEE802.11 састоји се од:

- станица (STA[3]) једне или више целина са основном (базном) услугом (BSS[4] целина, инфрастуктурна BSS целина), које су повезане са дистрибуционим системом (DS[5]) (слика 13.7) и

[1] За IEEE802.11b предвиђено је слабљење од 30dB.

[2] Године 2007. радна група је повезала сва дотадашња проширења стандарда IEEE802.11a, b, d, e, g, h, i, j у IEEE802.11-2007 а 2012. године десет проширења IEEE802.11k, r, y, n, w, p, z, v, u, s у стандард IEEE02.11-2012.

[3] *STAtion*

[4] *Basic Service Sets*

[5] *Distribution System*

- испреплетаних станица (MSTA[1]) које припадају целини са испреплетаном базном услугом (MBSS[2] целина) и испреплетаног излазног уређаја (MG[3] излаз) преко кога се MBSS целина повезује са дистрибуционим системом (слика 13.8).

Дистрибуциони систем има функцију окоснице и може да буде ожичена или било која друга комуникациона мрежа. Целина са основном услугом (BSS целина) основна је градивна целина архитектуре IEEE802.11. Група станица (STA[4]) која је под директним управљањем једне координационе функције формира целину са основном услугом (BSS).

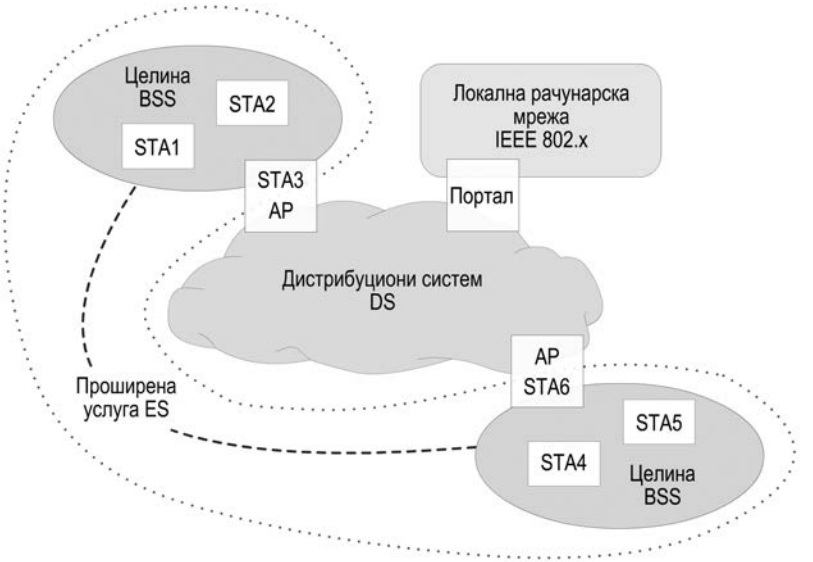
Област коју покрива BSS целина означава се као област са основном услугом (BSA[5] област) и одговара ћелији у целуларним комуникационим мрежама. Све станице у BSS целини могу међусобно да комуницирају. Може се десити да неке станице постану невидљиве због слабљења сигнала, као последице мултипутног феднига или због интерференције суседних станица.

Ad hoc мрежа представља груписање станица у једну целину са основном услугом (BSS). Стандард IEEE802.11 користи назив независна BSS целина (IBSS[6]). Било која станица у целини са основном услугом може да комуницира са било којом другом станицом без уређаја за приступ (AP).

Инфраструктурна мрежа

Када се за комуникацију између станица користи уређај за приступ (AP) онда стандард IEEE802.11 за такву мрежу користи термин инфраструктурна мрежа. Уређај за приступ омогућава проширење опсега и могућност комуникације више целина са основном услугом. Проширена услуга (ESS[7]) састоји се од две или више целина са основном услугом, међусобно повезаних дистрибуционим системом. Проширена услуга

[1] *Mesh STAtion*
[2] *Mesh Basic Service Sets*
[3] *Mesh Gate*
[4] *Stations*
[5] *Basic Service Area*
[6] *Independent Basic Service Set*
[7] *Extended Service Set*



Слика 13.7 Компоненте IEEE802.11 инфраструктурне мреже

појављује се као једна логичка локална рачунарска мрежа на подслоју слоја везе (LLC[1]).

На слици 13.7 уређај за приступ (AP) представљен је као део станице; представља логику у оквиру станице која омогућава и приступ дистрибуционом систему (DS) и саму станицу. Дистрибуциони систем (дефинисан стандардом IEEE802.11) је независан од имплементације. То значи да дистрибуциони систем може да буде жична IEEE802.3 локална рачунарска мрежа, или неки други медијум.

За интеграцију IEEE802.11 архитектуре са традиционалним локалним рачунарским мрежама са жичним трансмисионим медијумима[2], или за повезивање на Интернет, користи се портал. Логика портала уграђује се у уређаје као што су мостови или рутери који су део кабловске рачунарске мреже и који су прикључени на дистрибуциони систем (DS).

Испреплетана мрежа

Већ 2003. год. студијска група IEEE802.11 дефинисала је концепт

[1] *Logical Link Control*

[2] Користи се и термин не-IEEE802.11.

бежичног дистрибуционог система (WDS[1]) као механизам за бежичну комуникацију уз коришћење четири адресе у формату рама (слика 13.16). Студијска група није отишла ништа даље од дефиниције формата рама тј, није дала никакав опис како се тај механизам или формат рама користи. Постојала је велика потреба да се разјасни терминологија, дефинишу карактеристике и начин рада који се може интегрисати са IEEE802.11 стандардом. Радна група[2] је формирана 2004. године а стандард је објављен у јесен 2011. год. као IEEE 802.11s.

Стандард IEEE802.11s је предвидео унапређење MAC функција које обезбеђују бежичну испреплетану локалну топологију[3]. Испреплетана особеност је на располагању MSTA станицама које припадају целини са испреплетаном базном услугом (MBSS целина). Механизам који обезбеђује испреплетаност разликује се међу имплементацијама.

Способност да се реализује испреплетаност[4] је скуп напредних функција, правила за приступање каналу, формат рамова, метода међусобне аутентификације, управљани објекти који се користе за пренос података између аутономних станица које не морају да буду у директној међусобној комуникацији преко једног бежичног линка.

Целина MBSS је IEEE 802.11 локална рачунарска мрежа која се састоји од аутономних станица (MSTA станица). Унутар MBSS целине станице успостављају бежичну везу са суседним станицама међусобно размењујући поруке. Даље, користећи способност за повезивање у више скокова[5] поруке се могу пренети између MSTA станица које нису у међусобно директној вези. Са становишта испоруке порука изгледа као да су све станице у MBSS целини међусобно директно повезане на MAC слоју, иако оне нису све у домету једна другој.

Повезивањем у више скокова проширује се област покривања MSTA станице. У MBSS целини MSTA станица може да буде извориште,

---

[1] *Wireless Distribution System*. Обично се користи као колоквијални термин за механизам за бежичну комуникацију између станица које не подражавају функцију испреплетаности које користе четири адресе. Стандард IEEE802.11-2012 специфицира такав формат само за испреплетану базну целину (MBSS) и искључен је из даљих верзија.

[2] *Task group*

[3] *Wireless LAN mesh topologies*

[4] *Mesh facility*

[5] *Multi-hop capability*

одредиште или да прослеђује саобраћај[1] (неким MSTA станицама то јесте једина функција). MBSS целина може да има интерфејсе ка спољашњим мрежама и да обавља функцију дистрибуционог медијума система (DSM[2] медијума) за инфраструктурну BSS целину.

Пример испреплетане базне целине (MBSS целина) приказан је на слици 13.8. Само MSTA станице могу да формирају MBSS целину, селектују путању и преусмеравају поруке. Станадрд IEEE802.11s каже да MSTA станица[3] није део инфраструктурне базне целине и као последица тога она не може директно да комуницира са не-MSTA станицама[4]. Ово може да буде збуњујуће: како се испреплетана мрежа може повезивати са жичном мрежом другачије него преко IEEE802.11 уређаја. Разлог који стоји иза ове дефиниције је тај што у проширењу стандарда IEEE802.11s испреплетана станица представља функцију. У нашој свакодневној употреби станица је физички уређај али у терминологији IEEE802.11 термин станица је подскуп функција једног уређаја. Приступна тачка може у себи да интегрише функције испреплетане станице и комуницира са другим испреплетаним станицама а такође да интегрише и функције приступне тачке (AP) која обезбеђује асоцирање IEEE802.11 клијентских уређаја.

MBSS целина може се повезати са другим базним целинама (инфраструктурним или испреплетаним) преко дистрибуционог система (слика 13.8). MSTA станица може да комуницира са не-MSTA станицом преко дистрибуираног система.

Подаци се преносе између MBSS целине и дистрибуционог система преко једног или више излаза (MG[5] излаз).

Дистрибуциони систем „дистрибуира“ рамове са једног интерфејса на други. То може да буде дистрибуција рамова од клијента на 2,4GHz на клијента на 5GHz. У сваком случају битно је нагласити да је дистрибуциони

---

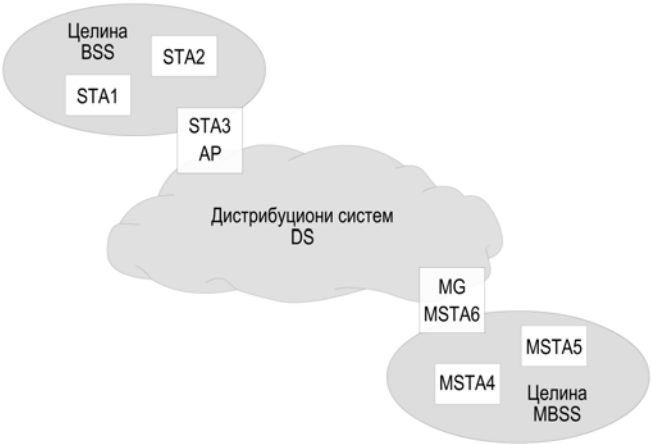
[1] *Propagators of traffic*

[2] *Distribution System Medium*. Трансмисиони медијум или скуп трансмисионих медијума које користи дистрибуциони систем (DS) за комуникацију између приступних тачака (AP), излаза из испреплетане целине MBSS (MG излаз) и портала проширеног скупа услуга (ESS).

[3] Треба имати на уму да термин испреплетана станица у проширењу стандарда IEEE802.11s има значење функције а не уређаја.

[4] *Non-mesh station*

[5] *Mesh gate*



Слика 13.8 Пример повезивања станице (MSTA) у испреплетаној мрежи (MBSS) са станицама у инфраструктурној мрежи (BSS) помоћу MG излаза

систем логичка конструкција а не посебан трансмисиони медијум као што је то жични (нпр. Етернет).

Стандардом 802.11-2007 дефинисана је приступна тачка (AP) као станица која обезбеђује приступ дистрибуционом систему преко бежичног медијума станицама које су на њу повезане. Приступна тачка може да испоручује рамове другим станицама у ћелији или у ожиченој мрежи.

Функција приступне тачке која обавља превођење између бежичне мреже и жичне мреже назива се портал. Портал је функција. То је само логичка тачка где се бежична јединица података MAC слоја (MSDU[1] податак) преводи ка и из не-IEEE802.11 мреже. Приступна тачка уобичајено обавља функцију портала.

Када се превођење одвија између MBSS целине и неиспреплетаног IEEE802.11 дистрибуционог система функција се означава као излаз из испреплетане мреже (MG излаз). То је логичка тачка где се јединица података испреплетаног MAC слоја (MMSDU[2] податак) преводи ка и од неиспреплетаног IEEE802.11 дистрибуционог система. Друге функције се обично захтевају да би се комплетирало превођење. На

[1] *MAC Service Data Unit*

[2] *Mesh MAC Service Data Unit*

пример ако се рам који долази из MBSS целине шаље ка IEEE802.11 ћелији (инфраструктурној BSS ћелији) AP функција води рачуна о преусмеравању рамова ка жичној мрежи. Један уређај може да буде централна тачка ћелије (на тај начин је приступна тачка AP), повезује своје бежичне клијенте на жичну мрежу (на тај начин је портал) док истовремено учествује у испреплетаној базној станици (MBSS целини) и повезује испреплетане станице (MSTA станице) ка дистрибуционом систему (на тај начин је излаз испреплетане мреже).

Недостатак тачног дефинисања појмова и компонената испреплетаних мрежа је довео до различитих реализација произвођача. У време писања овог уџбеника међусобни рад уређаја различитих произвођача није могућ. Произвођачи користе специфичне своје термине, формате рамова и специфичне протоколе. Не може се са сигурношћу знати да ли су или у ком обиму произвођачи реализовали функције које су стандардом предвиђене.

13.4.3 Протоколи MAC подслоја IEEE 802.11

Протокол на MAC подслоју стандарда IEEE802.11 значајно се разликује од протокола на MAC подслоју стандарда IEEE802.3, због веће сложености бежичних у односу на жичне системе. Код IEEE802.3 система станица чека док се трансмисиони медијум не ослободи и тек онда започиње са слањем. Уколико не прими ометајући сигнал (који указује да је дошло до колизије) скоро је сигурно да је рам успешно испоручен одредишту. Код бежичних система ситуација је мало другачија.

Постоји проблем невидљиве станице[1]: све станице нису једна другој у радио-домету па се може десити да се предаја у једном делу ћелије не може детектовати у другом делу исте ћелије. У примеру на слици 2.3 станица C_1 шаље податке ка станици C_2. Станица C_1 испитује канал и пошто је ван домета станице C_3 погрешно ће закључити да је канал слободан и да може да започне са слањем података ка станици C_2.

Постоји и други проблем – изложена станица[2]: станица C_2 жели да пошаље податке станици C_4 и испитује канал. Кад открије да постоји неки сигнал погрешно ће закључити да не може да започне са слањем

[1] Одељак 2.2.1.

[2] *Exposed node problem*

ка станици C_4, пошто станица C_1 шаље податке ка станици C_3. Многи радио-системи су полудуплексни што значи да не могу истовремено и да шаљу и да испитују (ослушкују) медијум на истој радио-учестаности. Последица тога је да IEEE802.11 системи не могу да користе CSMA/CD протокол који се користи код IEEE802.3 система.

13.4.4 Слојевита архитектура протокола по стандарду IEEE 802.11

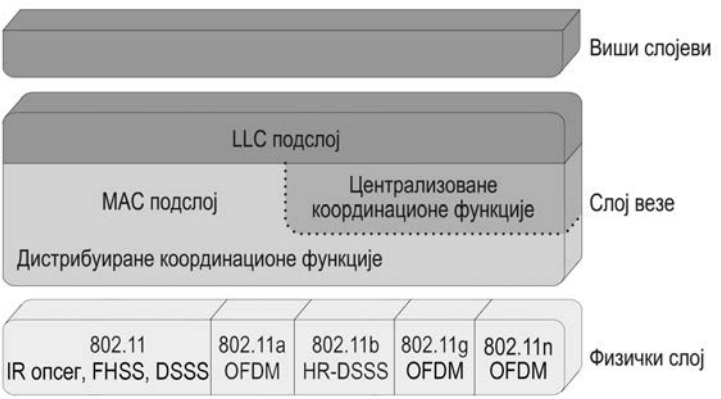
Слојевита архитектура протокола који се користе у првобитном IEEE802.11 стандарду има структуру као што је приказано на слици 13.9. Стандард се односи на прва два слоја OSI референтног модела: физички слој и слој везе. Подслој MAC (слоја везе) одређује начин доделе канала а подслој LLC (слоја везе) има задатак да заштити више слојеве од различитости техника IEEE802 серије.

У стандарду IEEE802.11дефинисане су:

- дистрибуиране координационе функције (DCF[1]) и
- централизоване координационе функције (PCF[2]) (слика 13.9).

Стандардом IEEE802.11-2012 обухваћене су и (слика 13.12):

- хибридне координационе функције (HCF[3]) описане у документу IEEE802.11e и



Слика 13.9 Слојевита архитектура протокола у бежичним локалним рачунарским мрежама у оригиналном стандарду IEEE802.11

[1] *Distributed Coordination Function*

[2] *Point Coordination Function*

[3] *Hybrid Coordination Function*

- испреплетане координационе функције (MCF[1]) описане у документу IEEE802.11s.

Дистрибуирана координациона функција

Основни метод приступа трансмисионом медијуму IEEE802.11 система је дистрибуирана координациона функција (DCF) означена као вишеструки приступ са избегавањем колизије (CSMA/CA[2]). Метод DCF је имплементиран у свим станицама и може се описати на следећи начин (слика 13.10):

1. Станица која жели да шаље податке прво испитује трансмисиони медијум. Ако је слободан чека одређено време T_P да види да ли ће и даље бити слободан. Уколико је и по истеку T_P времена медијум и даље слободан започиње са слањем. CSMA/CA дистрибуирани алгоритам дефинише колики треба да буде минимални размак између суседних рамова;
2. Уколико је трансмисиони медијум заузет, било зато што је већ био заузет, било зато што је постао заузет после T_P времена, станица чека да се заврши слање и да се медијум ослободи;
3. Када се текуће слање заврши станица чека још T_P времена. Уколико је медијум и тада слободан станица чека још неко случајно време кашњења у слању[3]. Ово је период надметања (такмичења[4]) у коме станице могу да се такмиче (боре) за приступ медијуму. Трајање случајног времена одлагања слања одређује се за сваку станицу посебно као мултипл временских целина[5]. Ново, независно случајно време бира се за сваки нови покушај слања. Станица бира број временских целина по случајном избору у опсегу од 0 до времена надметања (CW[6] - прозора надметања). Ово време се иницијално поставља

[1] *Mesh Coordination Function*

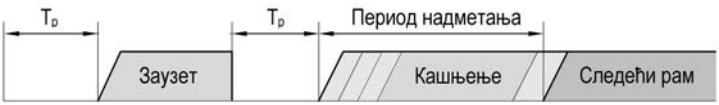
[2] *CSMA with Collision Avoidance* - CSMA са избегавањем колизије

[3] *Random auck off time*. Користи се исти алгоритам са бинарним експоненцијалним кашњењем као код IEEE802.3.

[4] *Contention time*

[5] *Slot time* Нпр. износи 9μsec за мреже по стандарду IEEE802.11a.

[6] *Contention Window*



Слика 13.10 Протокол CSMA/CA

на минималну вредност[1] CW_{min}. Све станице користе исто CW_{min} време али бирају случајно време кашњења у слању. Код система са дистрибуираним координационим функцијама (DCF) све станице раде са истим CW_{min} временом тако да имају исти приоритет у приступу трансмисионом медијуму. Резултат је да не постоји механизам који може да прави разлику између станица и њиховог саобраћаја а самим тим не може да се обезбеди квалитет услуге (QoS).

4. Ако по истеку случајног времена нико није заузео медијум станица може да започне са слањем података.

Одлагање слања (случајно време) које се користи је бинарно експоненцијално и користи се код саобраћаја великог интензитета. Понављање неуспелих покушаја слања података доводи до све дужих одлагања слања, што помаже да се саобраћај развуче у времену. Ако се не би користило случајно кашњење догодило би се следеће: две станице које покушају истовремено да шаљу податке дошле би у колизију. Уколико би без међусобно различитог кашњења покушале поново да шаљу поново би дошло до колизије.

Увек када IEEE802.11 бежична станица (извориште) пошаље рам само ка једном одредишту[2] (директно упућеном) и ако је рам примљен без грешке одредиште ће одговорити са рамом потврде (ACK рам). Уколико изворишна станица добије ACK рам она зна да је пренос успешно реализован. Ако се ACK рам не добије изворишна бежична станица ће морати да поново пошаље рам. Значи: за сваки рам упућен једној станици мора да постоју потврда - ACK рам. Не може се тачно утврдити да ли је дошло до колизије, тј. не постоји детекција колизије.

У сваком случају пренос је успешно окончан по пријему ACK рама

[1] Нпр. износи 15μsec за мреже по стандарду IEEE802.11a.

[2] *Unicast frame*

од станице којој је директно упућен или када је комплетно послат рам ка групи станица.

Да би се додатно минимизирала могућност истовременог слања (колизија) две станице предајна и пријемна станица могу да размењују управљачке рамове[1] RTS и CTS. Рамови се размењују пошто станице установе да је медијум слободан и после одлагања преноса а пре слања података.

Постоје два начина испитивања канала: физички и виртуелни механизми. Начин који се користи код система са дистрибуираном координационом функцијом заснован на протоколу CSMA/CA је испитивање виртуелног канала као што је приказано на слици 13.11.

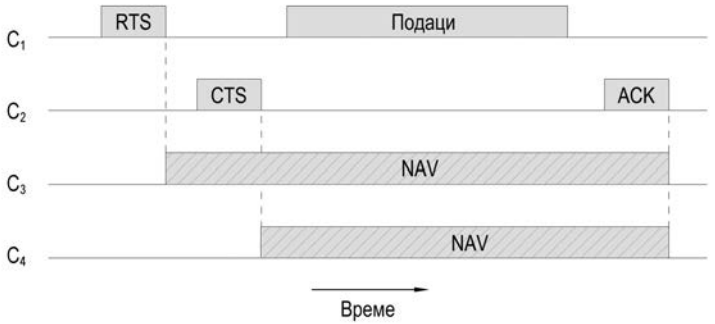
Размена RTS и CTS рамова је један од начина размене информација о резервацији трансмисионог медијума. Рамови RTS и CTS садрже информацију о трајању комуникације[2]. Тако је дефинисан период времена у коме ће трансмисиони медијум бити резервисан за слање рамова и за пријем потврде (ACK рам). Станица која прима RTS рам (који је послала изворишна станица) или CTS рам (који је послала одредишна станица) ће реализовати процес резервације трансмисионог медијума. Може да се деси да станица није у стању да прими податке од изворишне станице али је ипак упозната са тим захтевом.

Анализирајмо пример (слика 2.3) станице C_1 која жели да пошаље податке станици C_2. Станица C_1 прво шаље RTS рам којим тражи дозволу од станице C_2 да јој пошаље податке. Када станица C_2 добије захтев RTS станице C_1, и ако жели да прими податке од ње, она шаље натраг дозволу - рам CTS. По пријему CTS рама станица C_1 шаље рам са подацима и поставља на одређено време часовник за потврду ACK. Када станица C_2 прими рам података она шаље рам потврде ACK и размена је окончана. Уколико време на које је часовник ACK постављен истекне пре него што стигне потврда (ACK) рама цела процедура се понавља.

Посматрајмо како размену података виде станице C_3 и C_4. Станица C_3 је у домету станице C_1, и може да прими RTS рам. Уколико га је примила она може на основу података из рама RTS да закључи да ће станица C_1

[1] Детаљан опис може се наћи у одељку 2.1.2 овог уџбеника.

[2] *Duration field*



Слика 13.11 Употреба испитивања виртуелног канала са протоколом CSMA/CA

бити заузета одређено време. На основу тога станица C_3 поставља на одређену вредност ознаку (индикатор, вектор) заузетости канала (NAV[1]). Рам RTS не може да допре до станице C_4 али може рам CTS, пошто је станица C_4 у домету станице C_2. Станица C, поставља на одређену вредност своју ознаку заузетости канала (NAV). Треба уочити да ознаке о заузетости канала (NAV) нису рамови који се шаљу, они су само интерни показатељи који указују колико дуго станица не треба ништа да шаље.

Други начин дистрибуирања информација о резервацији трансмисионог медијума је поље трајања[2] у рамовима који су адресирани ка појединачним станицама. Ово поље указује на колико је времена трансмисиони медијум резервисан.

За разлику од кабловских локалних рачунарских мрежа бежичне рачунарске мреже су непоуздане и са великим сметњама. Због тога је у IEEE802.11 рачунарским мрежама предвиђено дељење рамова у мање целине - фрагменте. Сваки фрагмент садржи свој контролни збир. Фрагменти су нумерисани и потврђују се помоћу протокола заустави се и чекај[3]. Фрагменти се шаљу један за другим (слика 13.12).

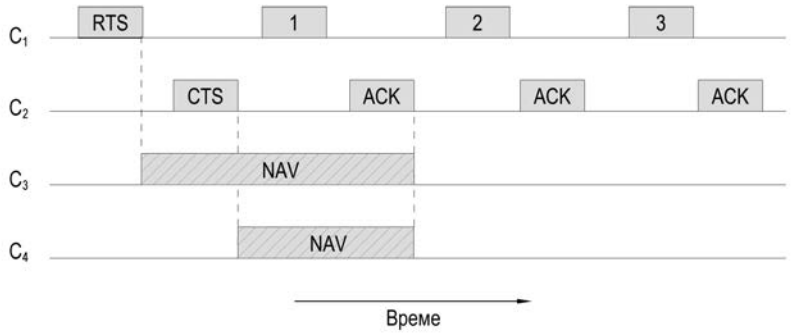
Пошто је трајање батерија значајно за бежичне системе стандард IEEE802.11 је понудио решење које води рачуна о томе. Базна станица може да обавештава мобилну станицу да је у стању мировања[4] све док

[1] *Network Allocation Vector*

[2] *Duration/ID field*

[3] Детаљније објашњење може се наћи у 7. поглављу уџбеника Рачунарске мреже [1].

[4] Стање успаваности



Слика 13.12 Слање фрагмената

је она или корисник експлицитно не активирају. Ово има за последицу да базна станица преузима на себе смештање свих података који се упућују ка станици која је у стању мировања. Станица у мировању податке може да покупи касније.

Централизоване координационе функције

Други начин приступа медијуму код IEEE802.11 система је помоћу централизоване координационе функције[1] (PCF). Базна станица[2] шаље пит (полира) другим станицама у ћелији да би видела да ли имају податке за слање.

Пошто редослед слања потпуно одређује базна станица до колизије не може да дође. Стандард прописује механизам полирања али не и учестаности, редослед или приоритете полирања.

Основни механизам састоји се у томе да базна станица шаље посебан рам[3] периодично (10 до 100 пута у секунди). Посебан рам садржи параметре система као што су учестаност скакања и такт за синхронизацију. Такође се позивају нове станице да се пријаве.

Хибридне координационе функције

Проширењем стандарда IEEE802.11e дефинисане су MAC методе потребне да се задовоље захтеви за квалитетом услуга (QoS[4]) у преносу

[1] У време писања овог уџбеника централизована координациона функција није део WI-Fi спецификације.

[2] Централизовани координатор

[3] *Beacon* - фар

[4] *Quality of Service*

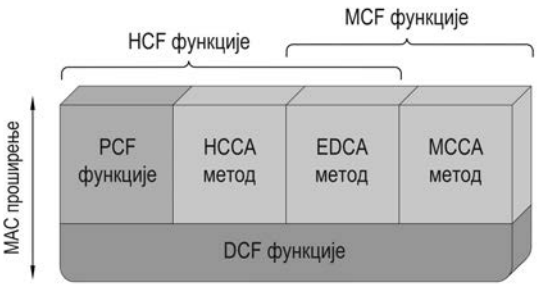
временски осетљивих апликација[1] бежичним локалним мрежама. У том циљу дефинисане су хибридне координационе функције (HCF[2] функције) које обезбеђује два метода приступа (слика 13.13):

- унапређени дистрибуирани приступ каналу (EDCA[3] метод) и
- хибридне координационе функције (HCF функције) за управљање приступом канала (HCCA[4] метод).

Унапређена координациона функција је проширење дистрибуиране координационе функције (DCF функције). EDCA метод обезбеђује приступ трансмисионом медијуму (каналу) саобраћај са приоритетима коришћењем ознака о приоритетима дефинисаним у стандарду IEEE802.1D (табела .13.2).

Хибридна координациона функција (HCFA функција) слична је централизованој координационој функцији (PCF функцији) и обезбеђује механизам за доделу приоритета користећи механизам упита помоћу хибридног координатора (HC[5] координатора). Другим речима одређене станице могу да шаљу податке пре осталих.

Организација Wi-Fi Alliance објавила је WMM[6] сертификат који делимично имплементира IEEE802.11e проширење. WMM сертификат ослоњен је на EDCA метод и обезбеђује четири категорије приоритета (табела 13.2).



Слика 13.13 MAC архитектура по стандарду IEEE02.11-2012 (укључена проширења IEEE802.11e и IEEE802.11s)

[1] *Time-sensitive applications*
[2] *Hybrid Coordination Function*
[3] *Enhanced Distributed Channel Access*
[4] *HCF Controlled Channel Access*
[5] *Hybrid Coordinator*
[6] *Wi-Fi Multimedia*

Приступна категорија[1]	Опис	Ознаке приоритета[2]
Говор[3] (1. категорија)	Највећи приоритет. Дозвољава више VoIP позива уз минимално кашњење и прихватљив квалитет.	7, 6
Видео[4] (2. категорија)	Преносу видео садржаја даје се приоритет над преносом података[5].	5, 4
Најбоља могућа[6] (3. категорија)	Саобраћај од апликација или уређаја који немају захтеве за квалитетом услуга (QoS) Овај саобраћај је мање осетљив на кашњења. Већина традиционалних итернет података подлеже овој врсти саобраћаја.	0, 3
У позадини[7] (4. категорија)	Саобраћај најнижег приоритета (скидање датотека, штампа) који нема стриктне захтеве у погледу кашњења и пропусног опсега.	2, 1

Табела 13.2 Врсте саобраћаја и категорије приоритета у бежичним локалним мрежама

Wi-Fi Alliance је објавила[8] и WMM планирани приступ (SA[9] приступ) који је заснован на HCCA методу. Код овог приступа нема надметања између станица која од њих ће успети да приступи медијуму – унапред је испланирано (заказано). Планирани приступ (SA приступ) даје могућност апликацијама да резервишу мрежне ресурсе на основу карактеристика свог саобраћаја тако што клијент шаље захтев приступној тачки (AP). Овај приступ подржава тзв. параметаризовани, планирани приступ

[1] *Access Categories*

[2] *Priority tags*. Преузето из стандарда IEEE802.1D.

[3] *Voice*

[4] *Video*

[5] Један IEEE802.11g или IEEE802.11a канал може да обезбеди пренос три до четири SDTV тока (*Standard-definition television* – телевизија стандардне резолуције) или један HDTV ток (*High-definition television* – телевизија високе дефиниције).

[6] *Best effort*

[7] *Background*

[8] *The Wi-Fi Alliance* је дефинисала WMM – PS (*Power Save*), што представља IEEE802.11e механизам за продужење века трајања батерија применом напредних метода штедње напајања.

[9] *Scheduled Access*

и одговара HCCA методу. Пошто је утицај случајног кашњења мањи SA приступ може да свеукупно смањи кашњење у мрежи употребом централизованог механизма управљања са планирањем времена[1].

Испреплетана координациона функција

Да би станице које су део MBSS целине биле у стању да детектују једна другу морају да раде на истом каналу. Приступ каналу је кључна компонента испреплетане инфраструктуре па је нова координациона функција направљена за испреплетане мреже: испреплетана координациона функција (MCF[2] функција). Она у приступу медијуму комбинује методе надметања[3] и метод са временском расподелом[4]. MCF функција интегрише елементе EDCA и HCCA метода стандарда IEEE802.11e и формира контролисани приступ испреплетаној мрежи (MCCA[5] приступ).

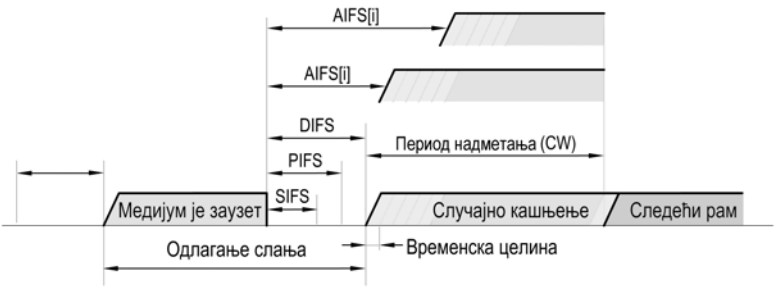
Временски интервали између рамова

Стандард IEEE802.11-2012 је предвидео коегзистенцију начина рада: централизованих и дистрибуираних система у једној ћелији. Један од механизама су дефинисана времена између рамова. Пошто се један рам пошаље чека се одређени временски интервал пре него што било која станица може да пошаље следећи рам.

Дефинисано је шест временских интервала између рамова[6] (слика 13.14):

1. Најкраћи интервал између рамова (SIFS[7] време) користи се код станица које су већ у међусобној вези. То значи да ће по истеку SIFS времена: бити послат рам CTS као одговор на пријем рама RTS, или рам ACK као одговор на фрагмент или цео рам података. Постоји само једна станица која би требало да шаље по истеку SIFS времена.
2. Редуковани интервал између рамова (RIFS[8] време) може се

[1] *Centralized sche duling control mechanism*
[2] *Mesh Coordination Function*
[3] *Contention-based*
[4] *Scheduled access*
[5] *Mesh Controlled Channel Access*
[6] *InterFrame Spacing*
[7] *Short Interframe Spacing*
[8] *Reduced InterFrame Space*



Слика 13.14 Временски интервали између рамова по стандарду IEEE802.11-2012

користити уместо SIFS времена када један предајник шаље више рамова и када се не очекује више одговора међусобно размакнутих SIFS временом. Не може се користити за различите одредишне адресе. RIFS време је време које протекне између последњег послатог симбола претходног рама и почетка преамбуле следећег рама који је присутан на радио интерфејсу.

3. Уколико станица не искористи канал у временском интервалу SIFS базна станица може да пошаље свој посебан рам или упит (PIFS[1] време). Овај механизам дозвољава да станица која шаље податке или секвенцу фрагмената заврши започето слање.
4. Уколико базна станица нема шта да пошаље по истеку временског интервала DIFS[2] било која станица може да покуша да приступи каналу и пошаље нови рам. Примењује се уобичајен начин за приступање каналу и бинарни експоненцијални алгоритми за кашњења уколико дође до колизије.
5. Временски интервал (EIFS[3] време) користи станица која је управо добила рам са грешком или рам који није очекивала[4]. Она шаље обавештење о грешци. Идеја да се додели најнижи приоритет овом обавештењу произлази из чињенице да треба дати времена предајнику који не зна да је дошло до грешке и да се на тај начин спречи интерференција.

[1] *Point Coordination Function Inter Frame Spacing* - PCF размак између рамова

[2] *Distributed Coordination Function InterFrame Spacing* - DCF размак између рамова

[3] *Extended InterFrame Spacing* - проширени размак између рамова

[4] Изостављено са слике 13.13.

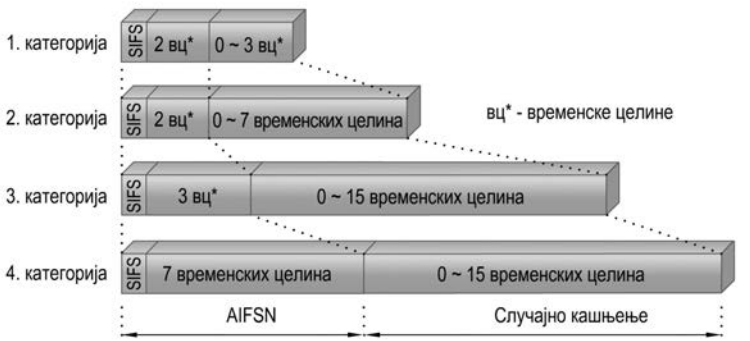
6. Арбитражни временски интервал (AIFS[1] време) указује на број временских целина после којих станица треба да одложи слање пре него што активира алгоритам за поновно слање или започне процедуру слања. Минимална вредност AIFS времена је 2. AIFS време користе станице са имплементираном услугом квалитета (QoS) и које приступају медијуму користећи EDCA приступ за слање свих рамова података, рамова за надзор и услугу, рамова упита и управљачикх рамова (RTS, CTS). Параметар „*i*" у ознаци за арбитражни временски интервал AIFS[*i*] односи се на приступне категорије дефинисане у табели 13.2.

Уколико је медијум слободан пруступа се одмах по истеку времена DIFS и AIFS[*i*].

Алгоритам за разрешавање колизије који обезбеђује приоритете у приступу медијуму је пробабилистички и зависи од два временска параметра који се мењају у зависности од приступне категорије:

- минимално време између рамова (AIFSN[2]),
- прозора надметања (CW) .

Обе вредности су мање за саобраћај већег приоритета. За сваку од приступних категорија вредност одлагања слања израчунава се (слика 13.15) као збир вредности минималног времена између рамова AIFSN и случајне вредности од 0 до вредности прозора надметања (CW).



Слика 13.15 Арбитражни временски интервали у зависности од приступних категорија

[1] *Arbitration InterFrame Space*

[2] *Arbitrary InterFrame Space Number*

Приступна категорија	CW_{min}	CW_{max}	AIFSN
Говор	15	1023	7
Видео	15	1023	3
Најбоља могућа	7	15	2
У позадини	3	7	2
Оригинални DCF	15	1023	2

Табела 13.3 Подразумеване вредности параметара EDCA метода за различите приступне категорије

Вредност CW се мења у времену и иницијално се поставља на вредност која зависи од приступне категорије (табела 13.3).

После колизије прозор надметања (CW) се удвостручава све док не достигне максималну вредност. После завршеног успешног преноса CW се поставља на иницијалну вредност у зависности од приступне категорије. Приступна категорија са најмањим кашњењем стиче право да прва пошаље своје податке.

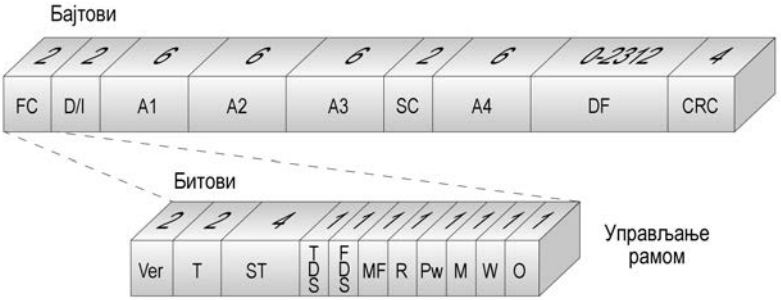
13.4.5 Структура рама стандарда IEEE 802.11

Стандард IEEE802.11 дефинише 3 различите класе рамова: податак, управљање и надзор. За сваки од ових рамова дефинисана су поља унутар MAC подслоја. Формат рама података приказан је на слици 13.16.

Прво поље за управљање рамом (FC[1]) садржи 11 потпоља:

- верзија протокола Ver[2] - дозвољава истовремени рад две верзије протокола у истој ћелији,
- тип (T[3]) - указује да ли је рам података, управљања или надзора,
- подтип (ST[4])- указује да ли је рам RTS или CTS,
- ка DS (TDS[5]) и од DS (FDS[6]) - указује да ли рам одлази ка или долази од дистрибуционог система,
- бит MF[7] - указује да постоји више фрагмената,

[1] *Frame Control field*
[2] *Version*
[3] *Type field*
[4] *Subtype*
[5] *To DS*
[6] *From DS*
[7] *More fragment*



Слика 13.16 Рам података IEEE802.11

- бит поновни покушај (R[1]) - означава ретрансмисију рамова који су већ раније послати,
- бит управљање напајањем (Pw[2]) - користи базна станица за постављање мобилне станице у мирно или активно стање,
- бит M[3] - указује да пошиљалац планира да пошаље још рамова примаоцу,
- бит W - указује да је садржај рама шифрован користећи WEP[4] алгоритме,
- бит 0[5] - указује примаоцу да рамови са постављеним битом на 1 треба да буду обрађени редоследом којим су и примљени.

Поље трајања (D/I[6]) указује на то колико ће дуго (изражено у микросекундама) рам и његова потврда заузимати канал. Ово поље постоји и у управљачким рамовима[7] тако да на основу њега станице могу да поставе своје индикаторе (NAV).

Заглавље рама садржи четири адресе (A_1 - A_4). Све адресе су стандардног IEEE802 формата. Две адресе су изворишна и одредишна адреса мобилних станица. Преостале две адресе су изворишна и

[1] *Retry*
[2] *Power management*
[3] *More*
[4] *Wired Equivalent Privacy*
[5] *Order*
[6] *Duration /Connection ID, Duration/ID*
[7] Поље трајања (*Duration/ID*) је дужине 16 битова. Садржај овог поља се мења у зависности од типа (и подтипа) рама. Зависи и од тога да ли се рам шаље за време CFP периода (*Contention Free Period*) када не може да дође до колизије (постављен NAV вектор). Мења се у зависности од квалитета услуге (QoS) изворишне станице.

одредишна адреса базних станица када се саобраћај одвија између ћелија[1].

Поље за секвенцне бројеве (SC[2]) користи се за обележавање фрагмената. Од 16 битова који су на располагању 12 битова се користи за обележавање рамова, а 4 бита за обележавање делова рама (фрагмената).

Поље података (DF[3]) садржи корисне податке[4] од 0 до 2312 бајтова.

Поље контролног збира (CRC[5]) има уобичајену намену да открије грешке у преносу.

Рамови за надзор имају формат сличан формату рама података осим броја адреса. Пошто се рамови надзора размењују искључиво унутар једне ћелије адреса базне станице није потребна.

Управљачки рамови су краћи и имају само једну или две адресе. Не садрже поље за податке ни поље за секвенцне бројеве. Кључне информације у овим рамовима налазе се у потпољу подтип које указује да ли је рам захтев за успоставом везе (RTS), потврда о успостави везе (CTS) или потврда о успешном пријему рама података (ACK).

13.4.6 Услуге IEEE 802.11

Стандард IEEE802.11 дефинише девет услуга које треба да пружи бежична локална рачунарска мрежа да би обезбедила функционалност еквивалентну оној која постоји у кабловским локалним рачунарским мрежама. Табела 13.4 приказује списак услуга и два начина на које оне могу да се систематизују[6].

На основу табеле може да се уочи следеће:

1. Услугу може да обезбеди или станица или дистрибуциони систем (DS). Услуге станице су примењене у свакој IEEE802.11 станици, укључујући приступне тачке (AP). Дистрибуционе услуге су обезбеђене између целина са основном услугом (BSS); ове услуге могу бити уграђене у приступне тачке (AP) или у друге уређаје са специјалним функцијама који су повезани на дистрибуциони систем.

---

[1] Обично се користе само три адресе

[2] *Sequence Control*

[3] *Data Field*

[4] *Payload*

[5] 32-битни циклични кôд (*Cyclic Redundancy Check*)

[6] *MAC Service Data Unit*

Услуга	Ко обезбеђује	За шта се користи
Приступање (*Association*)	Дистрибуциони систем	MSDU испорука
Одступање (*Dissassociation*)	Дистрибуциони систем	MSDU испорука
Дистрибуција (*Distribution*)	Дистрибуциони систем	MSDU испорука
Поновно приступање (*Reassociation*)	Дистрибуциони систем	MSDU испорука
Интеграција (*Integration*)	Дистрибуциони систем	MSDU испорука
Пријава (*Authentication*)	Станица	Приступ и безбедност локалне рачунарске мреже
Одјава (*Deauthentication*)	Станица	Приступ и безбедност локалне рачунарске мреже
Приватност (*Privacy*)	Станица	Приступ и безбедност локалне рачунарске мреже
Испорука јединица података MAC подслоја (*MSDU dilevery*)	Станица	MSDU испорука

Табела 13. 4 Услуге IEEE802.11

2. Три услуге су употребљене за управљање приступом поверљивошћу IEEE802.11 бежичних локалних мрежа. Од шест могућих услуга три се користе да би се обезбедила испорука MSDU јединица података између станица. MSDU јединица података је блок података који корисник MAC подслоја прослеђује MAC подслоју[1]. Уколико је MSDU јединица података превише велика да би била послата користећи један MAC рам она се може поделити у делове (фрагменте) и послати користећи више MAC рамова.

За информације које се размењују бежичном мрежом користи се RC4[2] алгоритам за шифровање.

[1] Обично је корисник LLC подслој и он прослеђује LLC PDU (*Logical Link Control Protocol Data Unit*) MAC подслоју.

[2] Аутор (1987. год.) RC4 (Rivest Cipher 4) је *Ron Rivest* један од оснивача компаније *RSA Security*. Име

13.4.7 Физички слој стандарда IEEE 802.11

Као што се може видети из табеле 13.4 велика пажња поклања се побољшању свих карактеристика система са бежичним преносом. У овом одељку биће дата кратка анализа физичких слојева оригиналног стандарда IEEE802.11 и каснијих, побољшаних верзија IEEE802.11a, IEEE802.11b, IEEE802.11g и IEEE802.11n.

Оригинални стандард IEEE 802.11

Предвиђена су три начина реализације физичког слоја стандарда IEEE802.11. Први је користио пренос сигнала у инфрацрвеном опсегу спектра а друга два омогућавају радио-пренос података употребом техника проширеног спектра[1], у фреквенцијском опсегу од 2,4GHz до 2,4835GHz. Подржан је пренос на брзинама од 1Mb/s и 2Mb/s.

Први начин реализације стандарда IEEE802.11 користи дифузну инфрацрвену технологију преноса. Може се користити на растојањима до 20m.

Други начин реализације стандарда IEEE802.11 користи пренос података техником проширеног спектра, употребом директне, Баркерове, секвенце. Број канала зависи од фреквенцијског опсега који одређују национална регулаторска тела. Примењују се диференцијалне фазне модулације и то: DBPSK[2] за проток од 1Mb/s и DQPSK[3] за проток од 2Mb/s.

Трећи начин реализације стандарда IEEE802.11 користи пренос података техником проширеног спектра са фреквенцијским скакањем. Сигнал носиоца мења учестаност у зависности од псеудослучајне секвенце. Дефинисано је 78 секвенци за скакање (у САД и Европи). Растојање између учестаности два узастопна скока је 6MHz у САД и већини европских земаља, а 5MHz у Јапану. Коришћени фреквенцијски опсег подељен је на 79 потканала (у САД и Европи) ширине 1MHz чиме је брзина ограничена на 2Mb/s.

RSA је настало од иницијала проналазача *Ron Rivest, Adi Shamir* и *Len Adleman* по којима је RSA криптографски алгоритам јавних кључева (*RSA public key cryptography algorithm*) добио име.

[1] Одељак 7.11 уџбеника Рачунарске мреже [1].

[2] *Differential Binary Phase Shift Keying*

[3] *Differential Quaternary Phase Shift Keying*

Стандард IEEE 802.11a

Први у низу стандарда већих брзина у бежичним системима је IEEE802.11a. Користи се ISM фреквенцијски опсег 5,8GHz и ортогонални фреквенцијски мултиплекс (OFDM[1]). Не користи се техника проширивања спектра већ потканали са носиоцима на различитим међусобно ортогоналним учестаностима. Подаци се деле у више токова а сваки се преноси помоћу свог подносиоца, кроз свој потканал. Могуће брзине су: 6, 9, 12, 18, 24, 36, 48 и 54Mb/s. Користе се до 52 носиоца који су модулисани са BPSK, QPSK, 16QAM[2], 64QAM у зависности од брзине која се захтева.

Стандард IEEE 802.11b

Циљ је био понудити стандард IEEE802.11b који подржава брзине преноса података као и (тадашњи) IEEE802.3 стандард. Понуђене су брзине 5,5Mb/s и 11 Mb/s и рад у ISM фреквенцијском опсегу од 2,400GHz до 2,4835GHz. Само повећање брзине преноса на 11Mb/s омогућено је употребом CCK[3] модулације, или такозване модулације са комплементарним кодовима. Модулације CCK се користе као кодна секвенца при процесу ширења спектра DSSS техником. Да ли ће проток бити 11 Mb/s или 5,5Mb/s зависи од стања у каналу, а уређај се сам томе прилагођава.

Стандард IEEE 802.11g

Стандард IEEE802.11g је побољшана верзија 802.11b. Користи OFDM модулацију али ради у 2,4GHz фреквенцијском опсегу. Теоретски може да достигне брзине од 54Mb/s.

Стандард IEEE 802.11n

Стандард IEEE 802.11n је проширење стандарда IEEE802.11-2007 који је издат 2009. године. Предвиђен је рад у оба опсега 2,4 GHz и мање коришћени 5GHz опсег. Планиране су брзине од 600Mb/s и коришћење ортогоналног фреквенцијског мултиплекса (OFDM) и MIMO[4] преноса.

[1] *Orthogonal Frequency Division Multiplexing*. Среће се и под називом дискретна вишетонска DMT (*Discrete Multitone*) и модулација са више носиоца MCM (*MultiCarrier Modulation*).

[2] *Quadrature Amplitude Modulation* - комбинација PSK и ASK (*Amplitude Shift Keying*).

[3] *Complementary Code Keying*

[4] *Multiple Input - Multiple Otput*

Технологија MIMO користи више антена (просторно распоређених) и мултипутно простирање сигнала у четири одвојена тока података. Користи квадратурну амплитудску модулацију (QAM) са 64 и 256 констелационих тачака. Потканали су ширине 40MHz у односу на садашњих 20MHz (у државама у којима регулаторска тела то дозвољавају).

Пре него што је усвојена завршна верзија стандарда у бежичним мрежама већ су се користили сертификовани производи Wi-Fi Alliance (2007. године) направљени по предлогу стандарда који није био у потпуности усвојен[1].

13.5 Бежичне персоналне мреже

Под појмом персоналне мреже (PAN[2]) подразумева се повезивање рачунара, телефона и дигиталних персоналних уређаја (PDA[3]) које користи једна особа (лично). За повезивање ових уређаја могу се користити и радио или инфрацрвени таласи па је онда реч о бежичној персоналној мрежи (WPAN[4]). Велики број технологија и произвођача присутан је на тржишту бежичних персоналних мрежа. У овом поглављу биће анализирани стандарди организација *Bluetooth SIG* и *Zig Bee Alliance*.

13.6 Стандард Bluetooth

Bluetooth[5] је стандард за бежичне везе, који је 1994. године промовисао Ericsson. Првобитно је био намењен за бежично повезивање мобилних телефона и уређаја, као нпр. дигитални персонални уређај (PDA), а тек касније за међусобно повезивање рачунара. Организација произвођача која развија и унапређује ову технологију позната је под називом *Bluetooth SIG*[6].

---

[1] *Draft*

[2] *Personal Area Network*

[3] *Personal Digital Assistent*

[4] *Wireless Personal Area Network*

[5] Добила је име по викиншком краљу Харалду II Блатану (*Blatann Harald*), који је у X веку ујединио зарађена викиншка племена. Значење имена Блатан су плави зуби (*Bluetooth*).

[6] *Special Interest Group - Ericsson, IBM, Nokia, Toshiba* и *Intel*. Од када је 1998. године ова група издала проглас о званичном почетку развоја ове технологије преко 2000 компанија је потписало споразум који им даје за право да развијају, производе и продају Bluetooth уређаје.

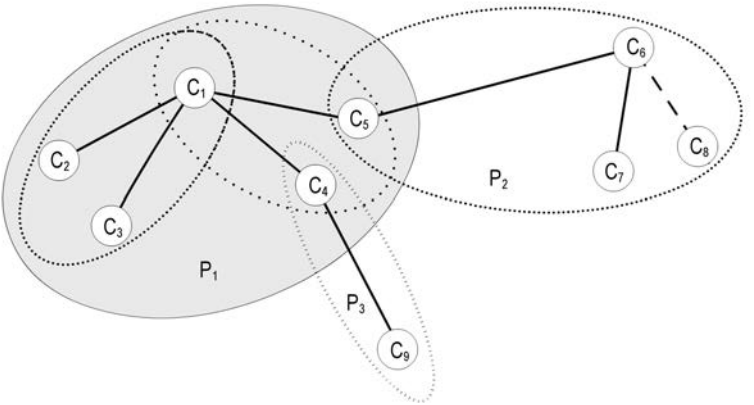
13.6.1 Архитектура Bluetooth мрежа

Основна јединица Bluetooth система је пиконет[1] мрежа. Пиконет мрежа састоји се од две или више станица (уређаја) које користе исти физички канал, што значи да су синхронизоване на заједнички такт сигнал и да користе исту секвенцу скакања. Заједнички такт пиконета је идентичан такт сигналу главне станице (уређаја[2]). Све остале станице (уређаји) које се синхронизују са главном станицом су пратеће станице (уређаји[3]). Само једна станица може да буде главна, а све остале су пратеће. Комуникација се одвија само између главне и пратећих станица. Директна комуникација између пратећих станица није могућа у оквиру канала пиконета.

На једној локацији може да постоји више независних пиконет мрежа. Свака од тих пиконет мрежа ради на различитим физичким каналима (различита главна станица, независни такт пиконет мреже и секвенца скакања). Bluetooth станица може да буде главна само у једној пиконет мрежи.

Разлог је следећи: пошто је пиконет мрежа дефинисана преко синхронизације са главним Bluetooth тактом није могуће бити главна станица у више од једне пиконет мреже. Bluetooth станица може да буде пратећа станица у више независних пиконет мрежа користећи временски мултиплекс. Bluetooth станица која је члан више пиконет мрежа укључена је у разбацану мрежу[4]. Укључење у разбацану мрежу не значи да је омогућено и преусмеравање саобраћаја. Језгро Bluetooth протокола не нуди такву опцију већ се сматра да је то у надлежности протокола виших слојева и ван је Bluetooth спецификације. На слици 13.17 приказан је пример топологије. Станица C_1 је главна станица у пиконету (осенчена област и означена као пиконет мрежа C_1) са станицама C_2, C_3, C_4 и C_5 као пратећим станицама. Пиконет мрежа P_1 за рад користи два канала: један за везу са станицама C_2 и C_3 и други за везу са станицама C_4 и C_5. Приказане су и две друге пиконет мреже: једна са станицом C_6 као главном (означена као пиконет мрежа P_2) и станицама C_5, C_7 и C_8 као

[1] *Piconet*
[2] *Master*
[3] *Slave*
[4] *Scatternet*



Слика 13.17 Пример повезивања станица пиконет мрежа

пратећим и друга са станицом P_3 (означена као пиконет мрежа C_4) као главном и станицом C_9 као пратећом.

Поред активних станица (главне и пратеће) Bluetooth дефинише и неактивну станицу[1]. Када пратећа станица нема потребе да ради (учествује у пиконет каналу), а још увек је синхронизована са тим каналом, она улази у неактивно стање. Повремено, када јој главна станица пошаље одговарајући сигнал (фар[2]) неактивна станица се због синхронизације укључује. Неактивно стање омогућава уштеду напајања (батерије) и могућност да је у пиконет мрежи присутно више од седам пратећих станица.

Начин функционисања пратеће станице веома је једноставан, па су једноставна и ниских цена интегрисана кола које те функције реализују.

13.6.2 Примена Bluetooth мрежа

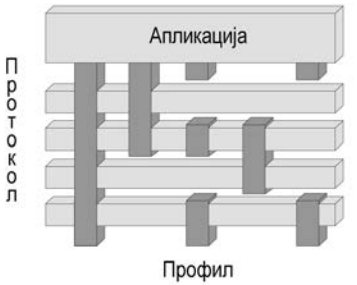
Да би станица користила Bluetooth мрежу мора бити у стању да интерпретира одређене дефинисане профиле. Профили представљају специфичне апликације од којих свака подржава различит скуп протокола (слика 13.18). У табели 13.5 приказано је 13 профила[3].

Профил за генерички приступ (GAP) у ствари није апликација већ

[1] *Parked*

[2] *Beacon*

[3] За сада су дефинисана 33 профила.



Слика 13.18 Веза апликације, протокола и профила код Bluetooth система

основа на којој су све апликације направљене и од које зависе. Њен основни задатак је да обезбеди процедуре за откривање Bluetooth станица, успостављање и одржавање сигурне везе (канал) између станица.

Поред GAP профила и профили SPP, SDAP и GOEP сматрају се генеричким. Изнад профила за генерички приступ (GAP) налазе се серијски профил (SPP), профил за генеричку размену објеката (GOEP) и профил за синхронизацију (SP). Многи често коришћени профили су: приступ фиксној локалној мрежи (LAP), профил за везу слушалице и базне станице (HSP) и профил за факс (Fax Profile) који су изнад серијског профила (SPP). Навешћемо један пример: ако треба синхронизовати[1] дигитални лични уређај (PDA) и рачунар да би преко серијског порта размењивали податке онда оба уређаја морају да подржавају профиле GAP, SPP, GOEP и SP. Остали профили нису потребни.

13.6.3 Скуп протокола код Bluetooth система

Језгро Bluetooth система обухвата четири најнижа слоја и одговарајуће протоколе који су део Bluetooth спецификације, као и заједнички протокол за откривање услуга (SDP[2]) и свеобухватни генерички профил приступа (GAP[3]). Комплетна Bluetooth апликација захтева додатне услуге и протоколе виших слојева који су дефинисани али превазилазе обим овог уџбеника.

---

[1] Користи се и термин *Bluetoth enabled.*

[2] *Service Discovery Protocol*

[3] *Generic Access Profile*

Ознака профила	Опис
GAP (*Generic Access Profile*)	Процедуре за надзор везе
SDAP (*Service Discovery Application Profile*)	Протокол за откривање понуђене услуге
CTP (*Cordeless Telephone Profile*)	Повезује слушалицу са локалном базном станицом
IP (*Intercom Profile*)	Дигитални воки-токи
SPP (*SerialPortProfile*)	Замена за кабл за серијски порт
HSP (*Headset Profil*e)	Омогућава употребу телефона без држања апарата у руци
DUNP (*Dial Up Networking Profile*)	Омогућава да се преносиви рачунар повезује преко мобилног телефона
Fax Profile	Омогућава да се преносиви факс уређај повезује преко мобилног телефона
LAP (*LAN Access Profile*)	Протокол између мобилног рачунара и кабловске локалне рачунарске мреже
GOEP (*Generic Object Exchange Profile*)	Дефинише клијент-сервер релације између објеката у покрету
OPP (*Object Push Profile*)	Обезбеђује начин за размену једноставних објеката
FTP (*File Transfer Profile*)	Обезбеђује пренос датотека
SP (*Synchronization Profile*)	Дозвољава PDA уређају да се синхронизује са другим рачунаром

Табела 13.5 Неки од профила код Bluetooth мрежа

Детаљнија архитектура језгра Bluetooth система представљена је на слици 13.19. Најнижа три слоја груписана су у подсистем који се среће под називом Bluetooth контролер. Остатак система укључујући услугу L2CAP[1] и више слојеве означава се као Bluetooth хост.

Станице комуницирају користећи протоколе дефинисане Bluetooth спецификацијом. Језгро протокола чини: језгро система које нуди услуге преко бројних тачака приступа (SAP[2]). Услуге могу бити разврстане у две групе: управљачке, које се означавају као C[3] раван, и корисничке

[1] *Logical Link Control and Adaptation Layer Protocol*
[2] *Service Access Point*
[3] *Control*

које се означавају као U[1] раван. Прва група води рачуна о успостављању и раскидању канала и веза, а друга о подацима који се том везом размењују.

Интерфејс услуга ка подсистему контролера дефинисана је на такав начин да се Bluetooth контролер може сматрати стандардним делом. У оваквој конфигурацији контролер ради на најнижа три слоја, а L2CAP слој садржи остатак апликације хост система. Стандардни интерфејс назива се интерфејс станица - контролер (HCI[2]). Његове приступне тачке означене су на горњој ивици контролера подсистема. Имплементација ових стандардних интерфејса ка услугама је опциона.

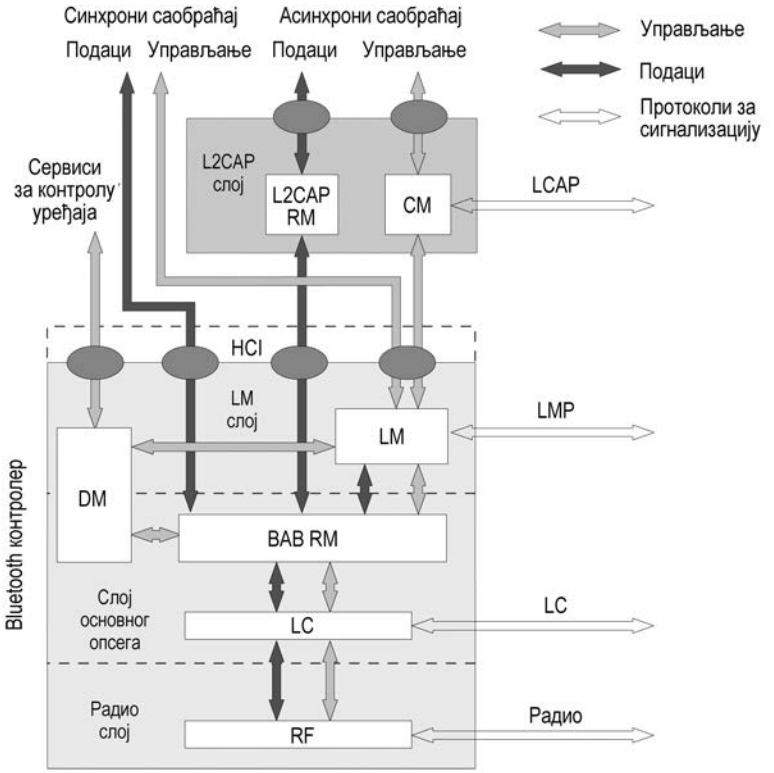
Слој основног опсега (BAB[3]) на неки начин је сличан MAC подслоју, али садржи и елементе физичког слоја. Води рачуна о томе како главна станица управља временским целинама и како су оне груписане у рамове.

Слој L2CAP може опционо да обезбеди додатну детекцију грешке и поновљено слање јединица протокола (L2CAP PDU[4]). Ова опција препоручује се апликацијама које у корисничким подацима захтевају малу вероватноћу неоткривених грешака. Следећа опција слоја L2CAP је контрола тока применом механизма клизајућег прозора[5].

У тексту који следи биће укратко објашњена функција сваког од модула са слике:

- Модул за надзор канала (CM[6]) задужен је за успостављање, управљање и раскидање L2CAP канала укључујући управљање напајањем, проверу идентитета и квалитет услуге. Протокол између парњак целина је L2CAP;
- Модул за управљање ресурсима (L2CAP_RM[7]) води рачуна о редоследу пријема сегмената. Такође води рачуна и о томе да ли апликације примају податке у складу са договореним квалитетом услуга QoS;

[1] *User*
[2] *Host to Controllerer Interface*
[3] *Baseband Layer*
[4] *Protocol Data Unit* - 1. поглавље овог уџбеника.
[5] Детаљније у 5.4 одељку овог уџбеника.
[6] *Channel Manager*
[7] *L2CAP Resource Manager*



Слика 13.19 Архитектура језгра Bluetooth система

- Модул за надзор станице (DM[1]) управља општим радом Bluetooth станице. Задужен је за све операције које нису директно везане за пренос података као што су испитивање присуства суседних станица, повезивање са тим станицама итд.;
- Модул за надзор везе (LM[2]) задужен је за успостављање, модификовање, раскидање логичке везе и иновирање параметара везаних за физичку везу између уређаја. За комуникацију са модулом за управљање везом и удаљеном станицом користи се протокол за управљање везом (LMP[3]);

[1] *Device Manager*
[2] *Link Manager*
[3] *Link Management Protocol*

- Модул за управљање везом (LC[1]) задужен је за кодирање и декодирање Bluetooth корисних података као и параметара везаних за физички канал;
- Модул за управљање ресурсима основног опсега (BAB_RM[2]) одговоран је за приступ медијуму;
- Радио - модул (RF) задужен је за предају и пријем рамова на физичком каналу.

Радио-слој Bluetooth система

Радио-слој преноси битове од главне станице до пратеће и обрнуто. Систем ради у 2,4GHz фреквенцијском опсегу (ISM) као и IEEE802.11 (Wi-Fi) системи тако да се међусобно преклапају. Фреквенцијски опсег је подељен на 79 канала ширине 1MHz, а модулација је GFSK[3] (верзија 1.2). Користи се фреквенцијско скакање са 1600 скокова у секунди. Време између два скока је 625μs.

Домет Bluetooth везе је 1m са предајником снаге OdBm, 10m са предајником снаге 4dBm и 100m са предајником снаге 20dBm. Могу се достићи брзине од 1Mb/s (верзија 1.2), 3Mb/s (верзија 2.1+EDR[4]) и 24Mb/s (верзија 3.0+HS[5]).

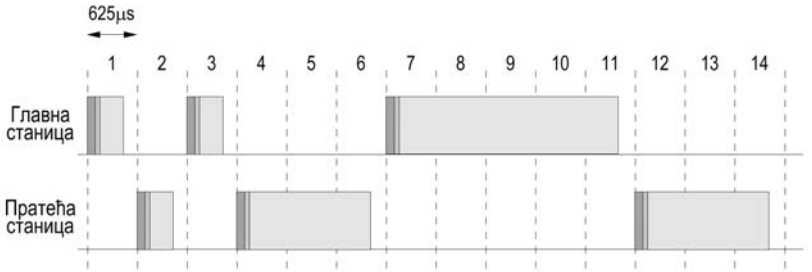
Слој основног опсега Bluetooth система

Овај слој претвара ток битова у рамове и дефинише формате од значаја. У најједноставнијем облику главна станица у свакој пиконет мрежи дефинише серију временских целина величине 625μs. Главна станица шаље у непарним временским целинама, а пратећа станица у парним. Ово је традиционалан временски мултиплекс у коме је главној станици додељена половина временских целина а пратеће станице деле другу половину.

Рамови могу бити дугачки 1, 3 и 5 временских целина (слика 13.20).

Сваки од рамова шаље се преко логичког канала - везе[6]. Постоје две врсте везе:

[1] *Link Contoller*
[2] *Baseband Resource Manager*
[3] *Gaussian Frequency Shift Keying*
[4] *Enhanced Data Rate*
[5] *High Speed*
[6] *Линк*



Слика 13.20 Размена пакета између главне и пратеће станице

- асинхрона без успоставе везе (ACL[1]), и
- синхрона са успоставом везе (SCO[2]).

Код ACL врсте користи се пакетска комутација и примењује се на податке који пристижу у неравномерним временским интервалима. Подаци пристижу са L2CAP слоја на предајној страни и предају се L2CAP слоју на пријемној страни. Саобраћај се испоручује на најбољи могући[3] начин али не постоји гаранција да су подаци успешно стигли на одредиште. Пратећа станица може са главном станицом да успостави само једну ACL везу.

Друга врста - SCO везе користи се за апликације у реалном времену, као што су телефонске везе. За овакав тип канала додељују се тачно одређене временске целине у оба правца. Због природе апликација у реалном времену - критична су временска кашњења - не користе се поновна слања већ механизам за корекцију грешака (FEC[4]). Пратећа станица може да има до три SCO везе са главном станицом. Свака SCO веза може да пошаље 64000b/s кодирани PCM аудио сигнал.

Слој L2CAP

Слој L2CAP има три важне функције:

- Прво, прима пакете до величине од 64kB од виших слојева и дели их на рамове погодне за пренос. На одредишту рамови се поново повезују у пакете;

[1] *Asynchronous ConnectionLess*

[2] *Synchronous Connection Oriented*

[3] *Best effort* - термин који се користи код система без успоставе везе као што су Интернет слој, Етернет и сл.

[4] *Foward Error Correction*

- Друго, води рачуна о мултиплексирању и демултиплексирању пакета из различитих извора. Када се пакети поново повезују L2CAP указује коме од виших слојева су упућени;
- Треће, L2CAP руководи захтевима за квалитет услуге и у фази успоставе везе и за време нормалног тока података. Такође води рачуна и о усаглашавању величине пакета између станица. Ово је важно пошто нису све станице у могућности да раде са пакетима величине 64kB.

13.6.4 Структура рама у Bluetooth системима

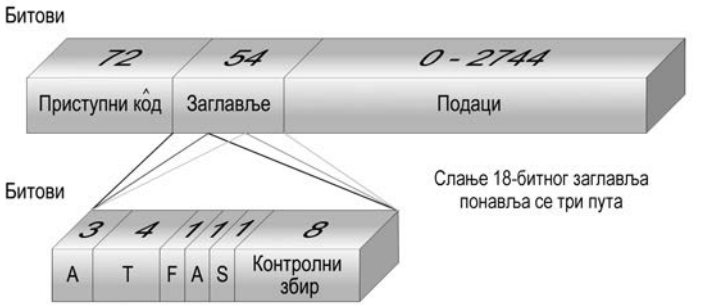
Постоји неколико формата рамова. Један од њих приказан је на слици 13.21 и чине га:

- приступни кôд који идентификује главну станицу тако да пратеће станице које
- се налазе у радио-опсегу две главне станице могу да одреде који од саобраћаја је за њих,
- заглавље дужине 54 бита које садржи поља типична за MAC подслој,
- поље корисничких података величине до 2744 бита. Ова величина одговара трајању пет временских целина.

Заглавља сачињавају следећа поља:

- поље адресе A (дужине 3 бита) - указује којој од осам активних станица је рам упућен;
- поље типа T (дужине 4 бита) - указује на врсту рама (ACL, SCO...), врсту корекције грешке која се користи и на то колико временских целина је рам дугачак;
- поље тока F[1] (дужине 1 бит) - поставља пратећа станица када је њен меморијски простор попуњен и не може да прими више података;
- поље потврде A (дужине 1 бит) - користи се да се пошаље потврда преко рама података[2];
- поље секвенце S[3] (дужине 1 бит) - користи се за обележавање

[1] *Flow control*
[2] *Piggy back*
[3] *Sequence*



Слика 13.21 Карактеристичан рам података у Bluetooth системима

рамова. Пошто се користи протокол заустави се и чекај[1] довољан је један бит;

- поље контролног збира[2] (дужине 8 битова) - користи се за детекцију грешке.

Дужина заглавља од 54 бита постиже се троструким слањем 18-битне целине која се састоји од претходно описаних поља. Вишеструким слањем повећава се поузданост преноса у радио-медијуму.

Код ACL веза за поље података користе се различити формати. Формати рамова у SCO везама су једноставнији: поље података је увек дужине 240 битова. Постоје три верзије: 80,160 и 240 стварне дужине података. Остатак се користи за корекцију грешке. У најпоузданијој верзији (80 битова података) садржај се три пута шаље.

Пратећа станица може да користи само непарне временске целине, тј. 800 временских целина у секунди. Са 80-битним подацима капацитет канала пратеће и главне станице је по 64000b/s што одговара капацитету PCM[3] говорног канала. То и јесте био разлог да се одабере брзина скакања од 1600.

Уколико се користи мање поуздана варијанта (без редундансе) могуће је реализовати истовремено три говорне везе што и представља максимум за SCO пратеће станице.

---

[1] *Stop and wait* – детаљније се може наћи у 11. поглављу уџбеника [1].

[2] CRC

[3] *Pulse Code Modulation* - импулсна кодна модулација, детаљније објашњење може се наћи у 7. и 8. поглављу литературе [1].

13.6.5 Унапређења новијих верзија Bluetooth стандарда

Верзија[1] 2.1+EDR, објављена 2007. године, компатибилна је са ранијим верзијама. Главна карактеристика је сигурно и једноставно упаривање (SSP[2]). Побољшања су везана и за размену додатних информација (EIR[3]) које се размењују у току успостављања веза са другим уређајима што доприноси бољем филтрирању уређаја са којим се повезује. Такође укључене су опције које смањују потрошњу у режиму мале енергетске потрошње. Брзина је 3Mb/s, домет 10m. Користи модулације DQPSK и 8DPSK.

Верзија[4] 3.0+HS објављена је у априлу 2009. године. Подржава брзину преноса до 24Mb/s. У току успоставе везе и договора око параметара везе користи се Bluetooth веза (линк). Пренос података се реализује кроз IEEE802.11 линк. Нова, битна одлика је додатни MAC и физички слој (AMP[5]) реализован по стандарду IEEE802.11 који се користи за брзи пренос података[6]. Користи се Bluetooth радио-пренос у фази откривања уређаја, иницијализовања везе и конфигурисања профила. Али када треба пренети већу количину података биће употребљен други физички слој MAC/PHY по стандарду IEEE802.11. Када је систем неактиван користи се начин рада Bluetooth уређаја са малом потрошњом (потврђено у дугогодишњој примени). Домет је 10m.

Верзија[7] 4.0 објављена је у децембру 2009. године. Основно обележје ове верзије је изузетно мала потрошња уређаја малих димензија који користе батерије величине кованог новца[8]. Сматра се да се на тај начин проширује тржиште на здравство, спорт, рекреацију. Као побољшање основне спецификације јесте могућност да се одаберу два режима: са два скупа протокола и са једним скупом протокола. У режиму рада са два скупа протокола особеност са малом потрошњом је

---

[1] *Bluetooth Core Specification Version 2.1 + EDR (Enhanced Data Rate)*

[2] *Secure Simple Pairing*

[3] *Extended inquiry response*

[4] *Bluetooth Core Specification Version 3.0 + HS (High Speed)*

[5] *Alternate* MAC/PHY

[6] Две технологије су биле предвиђене за AMP: 802.11 и UWB, али је UWB изостављена из спецификације.

[7] *Bluetooth Core Specification Version* 4.0

[8] *Coin-cell batteries*

интегрисана у стандардни Bluetooth контролер[1]. Резултујућа архитектура слична је стандардној архитектури (описаној у претходним одељцима) а резултује малим финансијским улагањима да би се добио нови систем. Произвођачи могу да користе интегрисана кола са постојећим верзијама (Bluetooth v2.1 + EDR или Bluetooth v3.0 + HS) и да додају нови скуп протокола који обезбеђује малу потрошњу.

13.7 Стандарди за персоналне мреже IEEE организације

Организација IEEE формирала је радне групе које се баве бежичним персоналним мрежама. Прва радна група радила је на прилагођавању стандарда Bluetooth SIG и издала 2002. године стандард IEEE802.15.1 (усаглашен са Bluetooth v1.1) који се односи само на физички слој и слој везе (слика 13.22). Иновирана верзија овог стандарда заснована на побољшањима која су укључена у верзију Bluetooth v1.2 издата[2] је као стандард IEEE802.15.1-2005.

Као што смо видели у претходним поглављима Bluetooth[3] спецификација обухвата цео систем, од физичког слоја до апликационог (слика 13.18). Без обзира што ове верзије нису идентичне оне конвергирају ка истом стандарду. Стандард IEEE802.15 односи се на бежичну комуникацију уређаја малих димензија на релативно кратким

Слика 13.22 Стандард IEEE802.15.1

[1] *Classic Bluetooth controller*

[2] По доношењу овог документа радна група задужена за овај стандард донела је одлуку да нове верзије Bluetooth неће бити укључене у IEEE стандарде.

[3] По верзији 1.1.

растојањима као што су PDA, преносиви рачунари, мобилни телефони, MP3 уређаји...

Рад друге групе (IEEE802.15.2) односи се на могућност истовременог рада бежичних личних мрежа по стандардима серије IEEE802.15 и бежичних локалних рачунарских мрежа по стандардима серије IEEE802.11.

Генерално, пошто и бежичне персоналне мреже и бежичне локалне рачунарске мреже користе ISM фреквенцијски опсег 2,4GHz може да дође до интерференције у раду уређаја ових мрежа. Да би се разрешио овај проблем радна група је предложила механизам за истовремени рад[1].

Трећа радна група ради на стандарду IEEE802.15.3 за рад персоналних уређаја на релативно великим брзинама (већим од 20Mb/s) као и на дефинисању нових карактеристика као што су мала потрошња и квалитет услуга који одговара мултимедијалним апликацијама.

Постоји више радних група: IEEE802.15.3a бави се преносом у изузетно широком опсегу спектра (UWB[2]): опсег од 3,1GHz до 10,6GHz, ширине 500MHz, брзине до 480Mb/s. Стандард IEEE802.15.3b-2006 бави се применом и међусобним радом[3] MAC подслоја. Стандард IEEE802.15.3c-2009 односи се на физичку спецификацију преноса са милиметарским таласима (mmWaveWPAN). Користи се опсег од 57GHz до 64GHz а брзине преко 2Gb/s. Предвиђен је за брз приступ Интернету, скидање видео садржаја (стриминг[4]) као што је видео по захтеву[5], телевизија високе резолуције (HDTV[6]), кућни биоскоп, итд.

Четврта радна група издала је стандард IEEE802.15.4-2006 за релативно мале брзине (LR-WPAN[7]), мање од 250kb/s. Стандардом су специфицирани физички слој и MAC подслој. У ову категорију спадају спецификације ZigBee, WirellessHART[8] и MiWi[9]. Минимизирањем

[1] *Coexistence Mechanism*

[2] *Multi-Band OFDM (MB-OFDM)* подржава *WiMedia Alliance*, *Direct Seguence - Ultra Wide Band* подржава *UWB Forum*

[3] *Interoperability*

[4] *Streaming*

[5] *Video on Demand*

[6] *High Definition Television*

[7] *Low Rate WPAN*

[8] *WirellessHART* је отворени стандард за технологију коју је развила *HART Communication Foundation.*

[9] Произвођачка (*proprietary*) спецификација за бежични протокол коју је развила компанија *Microchip Technology*

потрошње батерије у овим уређајима могу дуго да трају. Овај стандард развијен је за мреже са сензорима, за даљинско управљање и аутоматизацију у домаћинствима итд.

ZigBee Alliance има исту улогу у односу на стандард IEEE802.15.4 као Wi-Fi Alliance у односу на стандард IEEE802.11. У одељку који следи детаљније ће бити описана ZigBee технологија.

13.8 Сензорске мреже

Бежичне сензорске мреже (WSN[1]) састоје се од просторно дистрибуираних аутономних сензора који имају заједнички задатак да надгледају физичке услове или услове животне средине као што су температура, звук, вибрације, притисак, кретања или загађиваче. Мотив за развој сензорских мрежа везан је за војне пројекте у надзору бојних поља. Данас се користе у различитим индустријским и цивилним апликацијама као што су надгледање и управљање индустријских процеса, надгледање здравственог стања пацијената, надгледање животне средине, аутоматизација у домаћинствима и управљање саобраћајем.

Поред једног или више сензора, сваки чвор у бежичним сензорским мрежама обично има и радио-примопредајник, микроконтролер и извор енергије. Сензор може да варира по величини: од величине кутије за ципеле до величине зрна прашине. Сензорска мрежа уобичајено формира бежичну *ad hoc* мрежу. Неколико стандарда везаних за бежичне сензорске мреже је за сада или усвојено или је у фази развоја. Поред ових постоје и бројне нестандардне произвођачке спецификације. Навешћемо неке од њих: 6LoWPAN[2], ISA100[3], WirellessHART и MiWi.

За сензорске мрежне апликације кључни захтеви у пројектовању крећу се око ограниченог трајања батерија, ниских цена, малих растојања између уређаја и концепта међусобно увезаних мрежа[4] и комуникациона подршка за велики број међусобно повезаних уређаја

---

[1] *Wireless Sensor Network*

[2] Акроним за *IPv6 over Low power Wireless Personal Area Networks* на коме ради радна група у оквиру IETF (*Internet Engineering Task Force*).

[3] Стандарде за бежично умрежавање намењено аутоматизацији у индустрији које развија организација ISA (*International Society of Automation*).

[4] *Mesh*

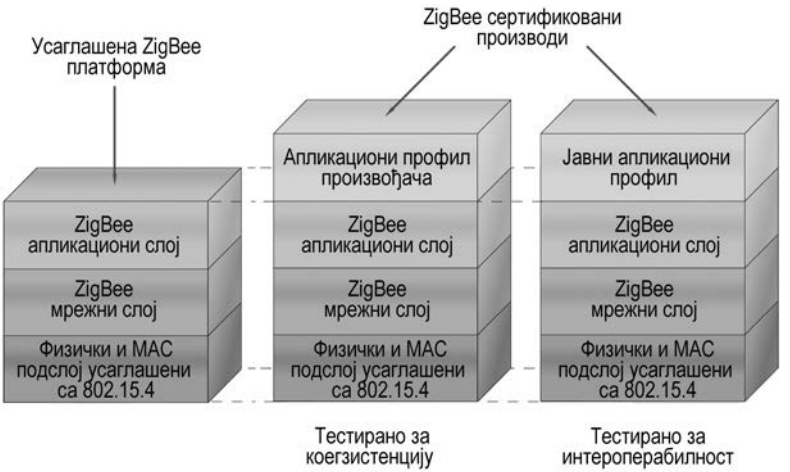
у мултиапликационом окружењу. Ово су биле основе за IEEE 802.15.4 и ZigBee спецификације да обезбеде основ за релативно једноставну, прилагодљиву, великог обима бежичну мрежу са више скокова[1] уз могућност да подржи велики број различитих апликација.

13.8.1 Концепција сертификовања Zig Bee производа

ZigBee Alliance је удружење састављено од великог броја компанија (чији се број стално повећава) које раде заједно на развоју јефтиних, поузданих, са малом потрошњом бежичних мрежа и врше контролу производа насталих на глобалном отвореном стандарду. Да би производ носио логотип ZigBee Alliance, прво мора успешно да прође ZigBee програм сертификације.

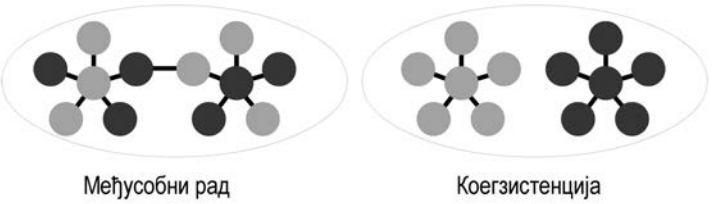
Постоје два програма за сертификацију производа (слика 13.23). Први је ZigBee усаглашена платформа (ZCP[2]) и односи се на модуле, или платформе које су намењене, као градивни блокови, за крајњи производ. ZigBee сертификовани производи односе се на крајње производе који су изграђени на ZigBee усаглашеној платформи.

Производи који користе јавне апликационе профиле[3] тестирани су да



Слика 13.23 Концепт сертификовања ZigBee производа

[1] *Multi-hop*
[2] *ZigBee Compliant Platform*
[3] *Public aplication profiles*



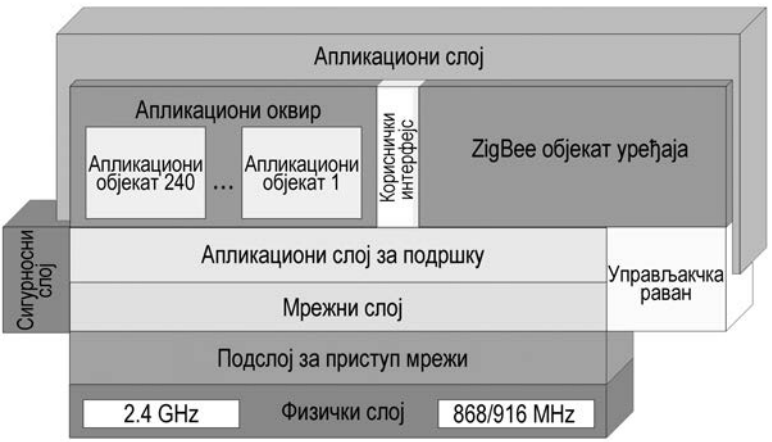
Слика 13.24 Међусобни рад и коегзистенција

би се обезбедио међусобни рад (интероперабилност) са другим ZigBee крајњим производима. Производи који користе од стране произвођача одређене профиле[1], који ће радити као затворени системи[2], тестирани су да би се обезбедио истовремени рад (коегзистенција) са другим ZigBee системима (слика 13.24).

13.8.2 Архитектура Zigbee протокола

ZigBee спецификацијом дефинисани су слојеви архитектуре протокола изнад физичког слоја и MAC подслоја (слика 13.25).

Апликациони слој (APL[3]) представља највиши слој ZigBee скупа протокола. Састоји се од:



Слика 13.25 ZigBee скуп протокола

[1] *Manufacturer-specific profiles*
[2] *Embeded*
[3] *Application*

- апликационог оквира (AF[1]) чија је основна функција да обезбеди опис како да се направи профил ZigBee скупу протокола (како би се осигурало да профили могу бити генерисани на конзистентан начин);
- ZigBee објекта уређаја (ZDO[2]) који дефинише улогу уређаја у оквиру мреже, иницира и/или одговара на захтеве повезивања или откривања путање (руте) и успоставља сигурне релација између мрежних уређаја;
- апликационог подслоја за подршку (APS[3]) који нуди сличну подршку апликационим услугама као што то чини TCP протокол у TCP/IP мрежама. Разлика ипак постоји: овај подслој не обезбеђује контролу тока који нуди TCP протокол;
- апликационог објекта (APS[4]) који представља софтвер крајњих тачака које контролишу рад уређаја.

ZDO управљачка раван (MN[5]) омогућава комуникацију између апликационог и мрежног слоја са ZigBee објектима уређаја (ZDO).

Сигурносни слој (SSP[6]) обезбеђује сигурносне механизме за слојеве који користе шифровање (мрежни и апликациони).

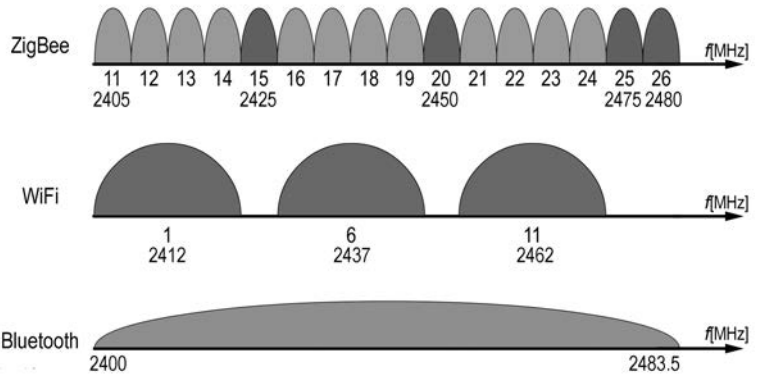
Мрежни слој (NWK[7]) води рачуна о мрежним адресама и рутирању. Задаци овог слоја су и покретање мрежног координатора, придруживање мрежних адреса, додавање и уклањање мрежних уређаја, откривање руте, прослеђивање рутирајућих порука и примена сигурносних мера.

IEEE802.15.4 MAC подслој одговоран је за пружање поуздане комуникације између уређаја и његовог непосредног суседа, имплементацијом CSMA/CA протокола. MAC подслој је такође одговоран за састављање и декомпоновање рамова података.

IEEE802.15.4 физички слој обезбеђује интерфејс за приступ физичком преносном медијуму.

Физички слој се састоји од два слоја која раде у два одвојена

[1] *Application Framework*
[2] *ZigBee Device Object*
[3] *Application Support Sublayer*
[4] *Application Support Object*
[5] *Management Plane*
[6] *Security Service Provider*
[7] *NetWorK layer*



Слика 13.26 Фреквенцијски спектар и канали: ZigBee, WiFi и Bluetooth

фреквенцијска опсега. Нижа учестаност физичког слоја покрива два опсега: 868MHz (Европа) и 915MHz (САД и Аустралија). Виши опсег учестаности (2,4GHz) користи се у готово целом свету. Радио фреквенцијски спектар и доступни канали за ZigBee (IEEE802.15.4) и Wi-Fi (IEEE802.11b/g/n) се преклапају. Канали су обележени (слика 13.26) тако да:

- канал 0 припада опсегу 868MHz,
- канали 1-10 припадају опсегу 915MHz а
- канали 11-26 припадају опсегу 2,4GHz.

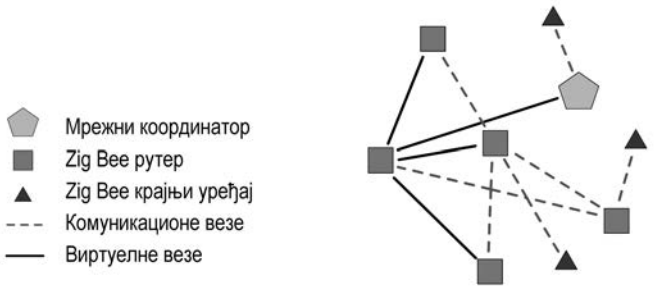
Избором канала могу се избећи сметње за оне који користе слободни простор између два суседна IEEE802.11 канала плус канали 25 и 26.

Канали 1, 6 и 11 обично се користе код WiFi мрежа у САД. Друге земље препоручују друге канале. У том случају је за све ZigBee производе најбоље користити канале који се најмање преклапају са наведеним. Са слике 13.26 види се да су то канали 15, 20, 25. и 26. Користе се следеће модулације: проширивање спекта директном секвенцом (DSSS[1]), квадратурна фазна модулација (QPSK[2]) и амплитудска модулација (ASK[3]).

13.8.3 Врсте ZigBee уређаја

У ZigBee мрежи дефинисане су три врсте уређаја: координатор, рутер и крајњи уређај (слика 13.27).

[1] *Direct Sequence Spread Spectrum*
[2] *Quadrature Phase Shift Keying*
[3] *Amplitude Shift Keying*



Слика 13.27 Врсте ZigBee уређаја и врсте њихових међусобних веза

Координатор је уређај који покреће и контролише мрежу. Координатор складишти информације о мрежи, које омогућују да делује као центар од поверења[1] и као складиште за сигурносне кључеве.

Рутери проширују област покривања мреже, омогућују динамичке руте око препрека, као и резервне руте у случају загушења мреже или отказу уређаја. Они се могу повезати са координатором и другим рутерима, као и обезбедити подршку уређајима за које су задужени.

Крајњи уређаји могу да преносе или примају поруке, али не могу да обављају функције рутирања. Они морају бити повезани на координатор или рутер, а не могу пружити подршку уређају детету.

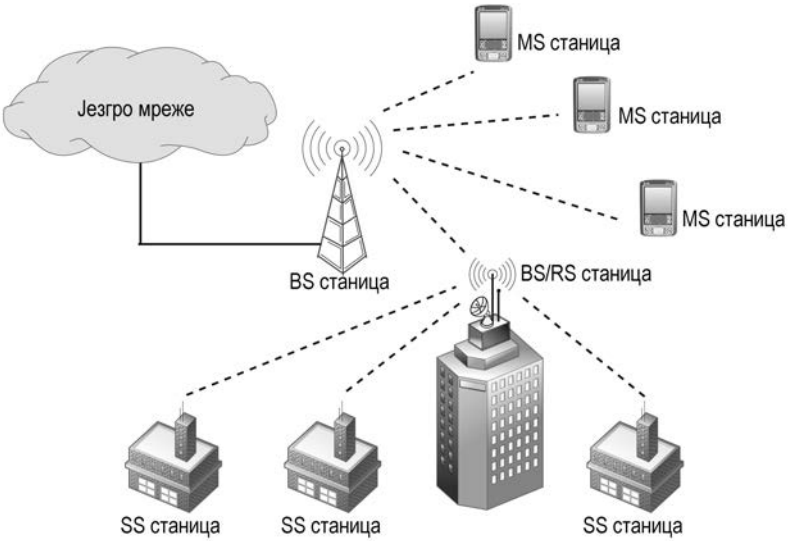
13.9 Бежичне мреже градског подручја

У овом одељку биће описане основе мрежне топологије бежичних мрежа градског подручја, архитектура протокола стандарда IEEE 802.16, као и разлике (где постоје) између различитих верзија стандарда.

13.9.1 Преглед IEEE 802.16 стандарда и начина повезивања

Архитектура мрежа реализованих по IEEE 802.16x стандарду (означавају се и као WiMAX мрежна топологија) је ћелијска. Мрежа је предвиђена да обезбеди приступ у великим градским или руралним областима. Као што је приказано на слици 13.28. WiMAX мрежна топологија састоји се од једне или више базних станица (BS[2] станица)

[1] *Trust Center*
[2] *Base Station*



Слика 13.28 Архитектура мреже реализоване по IEEE 802.16x
(WiMAX мрежна топологија)

скупа мобилних станица (MS[1] станица) и/или претплатничких станица (SS[2] станица). У зависности од верзије стандарда IEEE 802.16 и учестаности које се користе станица корисника може али не мора да буде у линији видљивости (LoS[3]) базне станице. Да би се мрежа проширила и повезала на преносни систем[4] користе се станице обнављивачи (RS[5] станице).

Спецификација IEEE802.16 је еволуирала у протеклом периоду. Првобитна спецификација је прихваћена 2001. године као бежични стандард за мреже градског подручја (MAN[6] стандард). Развијен је са намером да обезбеди велику брзину података и тачка-тачка комуникацију (PTP[7]) између фиксираних станица корисника. Стандард је

[1] *Mobile Stations*
[2] *Subscriber Stations*
[3] *Line-of-Sight*
[4] *Backhaul*
[5] *Repeater Station*
[6] *Metropolitan Area Network*
[7] *Point To Point*

предвидео коришћење лиценцираног опсега 10GHz до 66GHz, тако да је интерференција смањена. Пошто се користе радио-сигнали мале таласне дужине захтева се линија видљивости између станица. Уобичајена област примене је повезивање фиксиране тачке преносног система до локација које су повезане на жичну мрежу. Користе се усмерене антене на оба краја везе. Подржано је фреквенцијско дуплирање (FDD[1]) и временско дуплирање (TDD[2]).

Објављена су два проширења (амандмана[3]) стандарда. Прво је децембра 2002. год. усвојено проширење IEEE802.16c. У овој верзији стандарда додати су детаљни профили система или типична примена за опсег 10-66GHz и исправљене грешке претходне верзије стандрада. Друго проширење стандарда је IEEE802.16a. Оно је проширило употребу WiMAX система и на лиценцирани опсег учестаности испод 11GHz.

Уобичајено фреквенцијски опсег 2-11GHz садржи лиценциране и нелиценциране учестаности. Одговарајуће таласне дужине омогућавају сигналима да заобилазе препреке. Дозвољено је вишелинијско простирање и простирање ван линије видљивости (NLOS[4]). Поред подршке за тачка-тачка (PTP) пренос додата су два режима рада тј. начина повезивања станица. Први је тачка-више тачака (P2MP[5]) где скуп корисничких станица (SS станица) може да се повеже на једну базну станицу (BS станица). У другом начину - међусобно повезаних станица[6] корисничка станица не мора да буде повезана само на базну станицу већ може да шаље и суседној корисничкој станици. На тај начин се проширује област покривања мреже и умањује могућност отказа. Интернет посредницима погодује IEEE802.16a технологија пошто омогућава повезивање руралних области нарочито уколико жична инфраструктура не обезбеђује одговарајућу брзину приступа Интернету. Спецификација захтева коришћење омнидирекционих антена и динамичко бирање учестаности (DFS[7]). Ова техника дозвољава да се интерференција спречи

[1] *Frequency Division Duplexing*
[2] *Time Division Duplexing*
[3] *Amendment/s*
[4] *Non- Line-of-Sight*
[5] *Point to Multi-Point*
[6] *Mesh*
[7] *Dynamic Frequency Selection*

динамичким пребацивањем на другу радио-учестаност (RF[1]) а на основу мерења неких параметара нпр. односа сигнала и шума. Ширина радио-канала који се користи је од 1,25MHz до 28MHz. Мреже реализоване по стандарду IEEE802.16a, које користе ортогонални фреквенцијски мултиплекс (OFDM[2]), омогућавају пренос података брзином до 75Mb/s. Величина ћелије може да буде и пречнику до 50km али је уобичајена величина 5km до 10km.

Стандарде који носи ознаку IEEE802.16-2004 замена је и унапређење проширења стандарда IEEE802.16, IEEE802.16a, IEEE802.16c и IEEE802.16d. Проширење стандарда IEEE802.16d, које се сматра првом верзијом WiMAX стандарда и основом за WiMAX компатибилност, довело је до употребе WiMAX система у фиксним мрежама. У овој верзији стандарда уведен је нов лиценцирани опсег учестаности. Користе се два опсега учестаности: 2GHz-11GHz и 10GHz-66GHz. Фреквенцијски опсег канала је променљив и креће се од 1,25MHz до 28MHz. Брзина преноса података је 75Mb/s ако се користи канал ширине 20MHz. Спецификација IEEE802.16-2004 предвиђа коришћење секторских омнидирекционих антена као замену за омнидирекционе антене. Уведен је двоструки механизам за аутентификацију између базне и корисничке станице.

Све претходно описане верзије IEEE802.16 стандарда подразумевале су само рад са непокретним (стационарним) станицама и нису узимале у обзир механизме и фунцкионалност која се захтева код покретних (мобилних) система. У ствари није подржан хендовер начин рада а модулационе шеме нису пројектоване тако да се „носе“ са мобилним окружењем и са променљивим карактеристикама канала. Најбитније проширење (амандман) WiMAX системима јесте подршка за мобилност корисника при брзини возила од 120 km/h. Основе овог система описане су стандарду IEEE802.16e који се среће и под ознаком IEEE802.16-2005. Стандард предвиђа: рад у фреквенцијском опсегу од 2GHz до 6GHz, примену протокола који дозвољавају мобилност у мрежи, и хендовер механизам подржан аутентификацијом. Са становишта сигурности подржана је сигурна размена кључева за време

[1] *Radio Frequency*

[2] *Orthogonal Frequency Division Multiplexing*

аутентификације и шифровања података коришћењем DES[1] или AES[2] алгоритама.

Да би се поправила рањивост мрежа по стандарду IEEE802.16-2004 уведени су нови механизми сигурности. Спецификација IEEE802.16e предвиђа неколико побољшања на физичком и MAC слоју, подржава ширину канала од 1,25MHz до 20MHz и обезбеђује брзине (нпр. <15Mb/s за канал од 5MHz) за мобилне и роминг[3] кориснике. У поређењу са IEEE802.16d број корисника које мрежа може да подржи је повећан а и квалитет услуге је побољшан. Пречник ћелије је у опсегу од 2km до 5km. Стандард IEEE 802.16e омогућава приступ широкопојасном Интернету за преносиве уређаје са интегрисаним WiMAX адаптерима. Код мобилних WiMAX система неколико додатних мрежних механизама је подржано као што су: међусобна аутентификација између мобилног корисника и мрежа, трансфер квалитета услуга и сигурносних механизама за време хендовера.

Покренуто је неколико додатних стандардизационих пројеката: IEEE802.16f/l за подршку мобилним MIB[4] базама података, стандарди IEEE802.16j и IEEE802.16m за нови интерфејс. Детаљније о свим пројектима може се наћи у табели 13.4 која се налази на крају овог поглавља.

13.10 Концепција WiMAX мрежа

Услуге које пружа WiMAX мрежа су различите. Мрежа може да подржи стационарне и мобилне кориснике у градским подручјима. Саобраћај може да буде уобичајено претраживање Интернета или саобраћај у реалном времену који захтева строга ограничења у квалитету услуга (QoS). Компоненте WiMAX мреже треба да са сигурношћу све захтеве испуне. Да би се достигао тај циљ захтева се имплементација и координација свих делова скупа протокола са циљем да се обезбеде сигурност, успостављање везе, брзи хендовер и одговарајући нивои квалитета.

---

[1] *Data Encryption Standard*

[2] *Advanced Encryption Standard*

[3] *Roaming*

[4] *Management Information Base*

Скуп IEEE802.16 стандарда усмерен је на радио и MAC слој и ни на који начин не указује како се управља вишим слојевима. Као што је познато WiMAX мрежа се може састојати од већег броја различитих елемената: мобилних станица, базних станица, међумрежних пролаза, сервера за аутентификацију који су потребни у мрежама са више корисника, више терминала и чак и више оператора.

У циљу дефинисања организације мреже где је обезбеђен међусобни рад свих компонената WiMAX Forum[1] је објавио скуп докумената који специфицирају мрежне елементе, организацију и конфигурацију која се мора применити у WiMAX мрежи[2].

13.10.1 Референтни модел WiMAX мреже

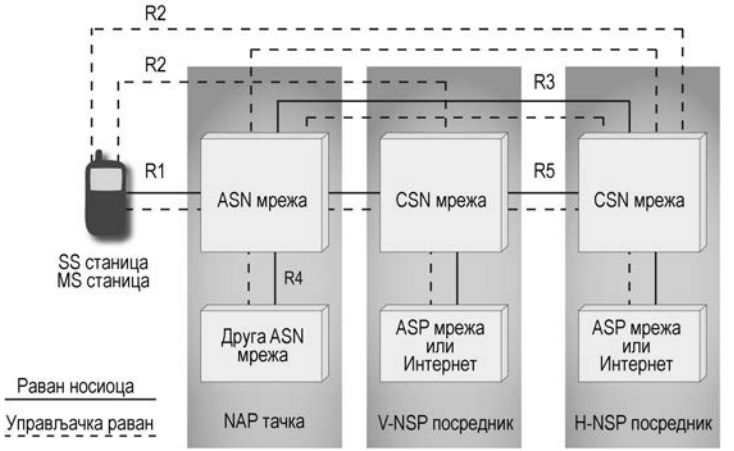
У циљу разумевања међусобних веза мрежних компоненти анализираћемо логичку презентацију WiMAX мреже. Ову шему објавио је WiMAX Forum и означена је као мрежни референтни модел (NRM[3]). На шеми (слика 13.29) могу се уочити логички домени[4], функционалне целине[5] и референтне тачке (RP[6]). Циљ је дозволити различита решења и истовремено одржавати свеобухватну интероперабилност између различитих реализација функционалних целина. Због тога никакве претпоставке нису направљене у имплементацији функционалних целина. За неке од њих одреднице су обезбеђене преко тзв. профила. У поглављу које следи биће описане релевантне компоненте мрежног референтног модела.

13.10.2 Функционалне целине

Главне функционалне целине скициране на слици 13.29 су:

- Мобилна станица (MS[7] станица). Мобилна станица је уређај који користи претплатник да би приступио WiMAX мрежи. Исти уређај може да користи један или више корисника и исти корисник може да приступи мрежи користећи више од једне мобилне станице.

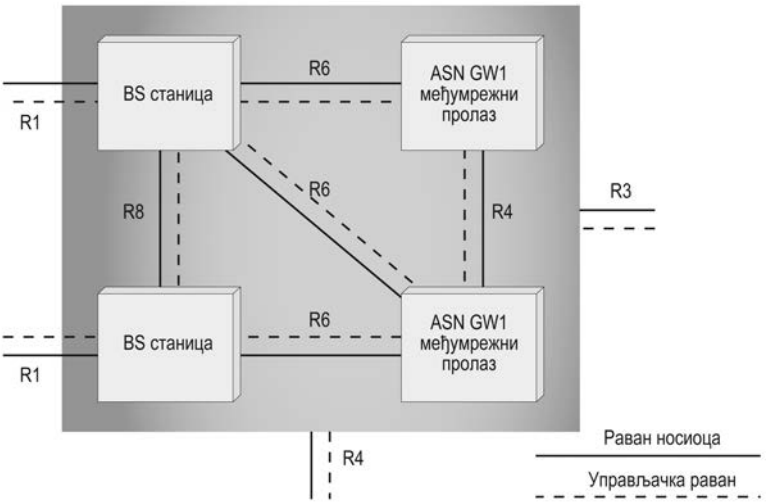
[1] Може се наћи на адреси [20].
[2] Може се наћи у документима [21] и [22].
[3] *Network Reference Model*
[4] *Logical domains*
[5] *Functional entities*
[6] *Reference Points*
[7] *Mobile Station*



Слика 13.29 WiMAX референтни модел[6]

- Мрежа за приступ услузи (ASN[1] мрежа). Представља границу функционалне интероперабилности WiMAX клијента са WiMAX услугом повезивања[2]. Она је највише одговорна за вођење рачуна о повезивању на слоју 2, преусмеравању свих AAA[3] порука ка матичном интернет посреднику (H-NSP[4] посредник), преусмеравање порука 3. слоја (нпр. порука DHCP протокола и мобилног IP протокола). Логичка декомпозиција ASN мреже приказана је на слици 13.30. Две најважније компоненте ASN мреже су радио базне станице (BS станица) и међумрежни пролаз ASN мреже (ASN-GW[5]). ASN-GW је међумрежни пролаз ка IP мрежи и крајњи терминал од RP3 референтне тачке што ће касније бити описано.
- Мрежа за пружање услуге повезивања је (CSN[6] мрежа). CSN мрежа је целина (субјекат) која има право на управљање везом на IP слоју претплатничких терминала. Прецизније она обухвата следеће задатке:

[1] *Access Service Network*
[2] *Connectivity services*
[3] *Authentication, Authorization and Accounting*
[4] *Home Network Service Provider*
[5] *Access Service Network Gateway*
[6] *Connectivity Service Network*



Слика 13.30 WiMAX референтни модел, декомпозиција ASN мреже

- обезбеђивање IP адреса,
- међумрежни пролаз ка другим мрежама,
- реализацију AAA функција,
- вођење рачуна о мобилности у оквиру ASN мреже коришћењем мобилног IP протокола.

MS станица, ASN мрежа или CSN мрежа дефинисане су преко логичких функционалних целина које се могу реализовати у једној крајњој станици или се могу дистрибуирати на више крајњих станица.

13.10.3 Логички домени

Логички домени могу се посматрати као група функција које могу бити повезане у један домен. На слици 13.29 представљена су три логичка домена:

- Тачка приступа мрежи (NAP[1] тачка). Представља физичку тачку коју користи претплатнички треминал када приступа мрежи. Са логичке тачке гледишта ASN мрежа која опслужује MS станицу део је тачке приступа мрежи.
- Матични интернет посредник (H-NSP посредник). H-NSP посредник је WiMAX добављач интернет услуга са којим WiMAX

[1] *Network Access Point*

претплатник има уговор о нивоу услуга[1]. Пословна целина аутентификује и ауторизује претплатничке сесије и одговорна је за процедуре израчунавања и тарифирања чак и при преласку из сопствене (матичне) мреже у другу мрежу (роминг).

- Посећени интернет посредник (V-NSP[2] посредник). V-NSP посредник је WiMAX добављач интернет услуга који претплатник користи да би приступио мрежи када је у ромингу без обзира што нема потписан уговор о нивоу услуга са V-NSP посредником. Опсег услуга који V-NSP посредник обезбеђује претплатнику зависи од роминг везе између V-NSP и H-NSP посредника.

13.10.4 Референтне тачке

Посматрајући слику 13.29 референтна тачка (Rx[3] тачка) је крајња тачка комуникације између функционалних целина и чини стандардни интерфејс који се мора користити за постизање укупне интероперабилности између компонената различитих произвођача. Спецификација WiMAX Forum дефинише неколико референтних тачака у мрежном референтном моделу и такође обезбеђује три примера профила. Сваки профил укључује само подскуп ових референтних тачака. У даљем излагању биће описане најбитније референтне тачке које су дефинисане у мрежном референтном моделу. То су:

- Референтна тачка R1 - састоји се од протокола и процедура које се користе на ваздушном интерфејсу између MS станице ASN мреже. То је радио и MAC WiMAX интерфејс;
- Референтна тачка R3 - представља интерфејс између ASN мреже и CSN мреже (којом управља или H-NSP или V-NSP посредник). Ове две функционалне целине користе овај интерфејс за пренос AAA порука и порука везаних за управљање мобилним IP протоколом. У пракси овај интерфејс је линк са IP протоколом који даје подршку RADIUS[4] протоколу;

---

[1] *Service Level Agreement*

[2] *Visited Network Service Provider*

[3] *Reference Point*

[4] RADIUS (*Remote Authentication Dial In User Service*) је мрежни протокол који обезбеђује централизовано управљање аутентификацијом, ауторизацијом и обрачунавањем (AAA - *Authentication, Authorization, Accounting*) за рачунаре који се повезују на мрежу и користе је.

- Референтна тачка R4 - одговорна је за комуникацију између ASN мрежа којима управља иста CSN мрежа. На пример када MS станица обавља хендовер[1] између две ASN мреже у истој NSP мрежи AAA фаза може се избећи коришћењем комуникације преко овог интерфејса;
- Референтна тачка R5 - дефинише и обезбеђује међумрежне функције између CSN мрежа којима управљају H-NSP или V-NSP посредници. Овај интерфејс би у пракси била IP путања која би могла да се рутира кроз додељене[2] или чак и јавне мреже;
- Референтна тачка R8 – користи се у оквиру ASN мреже са више базних станица (BS станица) R8 да подржи брз и неприметан хендовер мобилних станица.

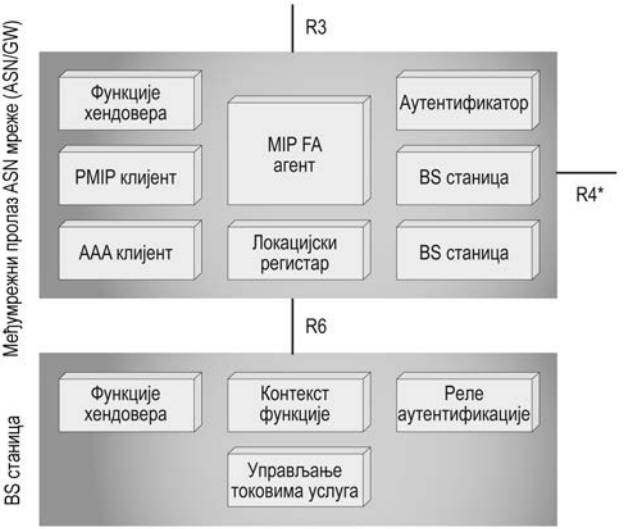
13.10.5 Профили ASN мреже

Посматрајући логичку архитектуру приказану на слици 13.30 могуће имплементације су предложене и описане у спецификацији организације WiMAX Forum. Ове опције називају се профили[3] и разликују се за различите референтне тачке. На сликама 13.31 и 13.32 представљена су два ASN профила (A и B) које је предложио WiMAX Forum са циљем да се усмери ASN имплементација ка двема најчешћим конфигурацијама.

Профил A (слика 13.31) ће се користити у конфигурацији где један међумрежни пролаз ASN мреже (ASN-GW) управља са више базних станица. Оваква конфигурација је погодна када је потребно покрити већа пространства (области) или када се ради са великим бројем корисника. Мобилношћу потпуно руководи међумрежни пролаз ASN мреже (ASN-GW) који управља хендовером претплатника преко R4 референтне тачке[4].

Профил B приказан на слици 13.32 представља конфигурацију са физичком колокацијом[5] међумрежног пролаза ASN мреже (ASN-GW) и базне станице (BS станица). Све функције ASN мреже укључене су у једну целину што као предност даје смањење комплексности. Недостатак овог решења је 1:1 однос између међумрежног пролаза ASN мреже (ASN-GW)

[1] *Handover*
[2] *Dedicated*
[3] *Profiles*
[4] *ASN-anchored mobility*
[5] *Co-location*



* Мобилност када је MS станица учвршћена за ASN мрежу могућа је преко R6 и R4 тачака.
* Подршка за ову функцију је опциона

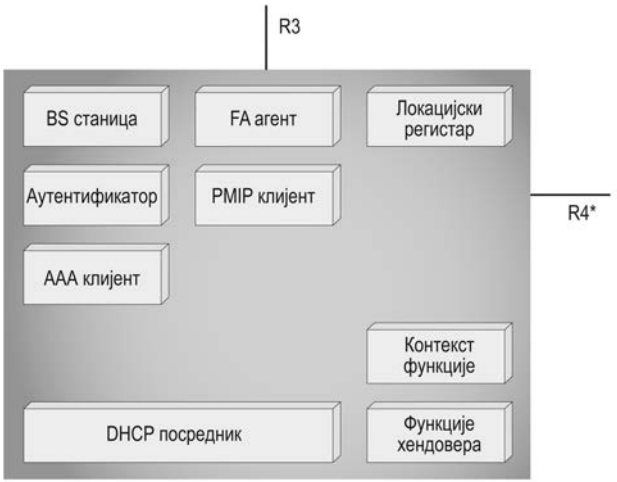
Слика 13.31 WiMax профил за A имплементацију ASN мреже

и базне станице (BS). Због тога ово решење одговара само у случају малог броја корисника дислоцираних у локализованој области.

Функционална декомпозиција скицирана на сликама 13.31 и 13.32 указује да су ASN мрежи присутне целине потребне за подршку AAA процедурама, MIP[1] усаглашеној мобилности и DHCP[2] протоколу. Различита локација ових целина је најважнији елемент на који треба указати када се упоређују ова два профила.

Избор профила треба да буде у зависности од потреба за облашћу покривености и њеног утицаја на цену хардвера и софтвера потребног за то решење. На пример, када је примењен профил A може се користити хардвер и софтвер различитих произвођача што може довести до неконзистентности и некомпатибилности. Када је примењен профил B ови проблеми се избегавају (R1 и R4 су стандардни и широко примењени интерфејси) али је ограничена могућност проширења.

[1] *Mobile IP*
[2] *Dynamic Host Confguration Protocol*



Слика 13.32 WiMax профил за B имплементацију ASN мреже

13.10.6 Хендовер процедура у WiMAX мрежи

Мобилна станица у WiMAX мрежи може да мења тачке прикључивања на мрежу и реализује хендовер. Две различите врсте хендовера су могуће: када је мобилна станица учвршћена за ASN мрежу[1] или учвршћена за CSN мрежу[2]. Посматрајући слику 13.29 у првом случају базне станице са којима се мобилна станица повезује при кретању налазе се у истој ASN мрежи. Међумержни пролаз (ASN-GW) са којим су базне станице повезане је исти, тако да не постоји потреба да се иновирају табеле рутирања или мрежне адресе. У овом случају хендовером управљају целине ASN мреже и немају никаквог утицаја на IP слој. У другом случају мобилна станица (MS станица) креће се из једне ASN мреже у другу. CSN мрежа на коју су повезане ASN мреже може да буде иста уколико нова ASN мрежа припада H-NSP посреднику или друга уколико нова ASN мрежа припада V-NSP посреднику.

У другом случају нова CSN мрежа ће се понашати као посредник и резултат је суштински исти. Подсетимо се да ASN мрежа није монолитна компонента већ се састоји од већег броја подмодула. Ови подмодули

[1] *ASN anchored*
[2] *CSN anchored*

могу да буду физички смештени у истој крајњој станици (хосту) као што је то представљено на слици 13.32 или одвојени као што је представљено на слици 13.31. Промена тачке прикључивања мобилне станице значи промену базне станице али у зависности од изабране конфигурације може да значи промену других функционалних целина као што су аутентификатор или мобилни спољни агент (MIP FA[1]). У одељку који следи сматраћемо да је реч о мобилној станици учвршћеној за CSN мрежу пошто она директно утиче на мрежни слој. WiMAX мрежа требало би да обезбеди роминг корисницима који желе да задрже активне сесије у случају хендовера.

Два су предуслова да се сесија одржи активном: задржавање исте IP адресе код хендовера и иновирање реверзне руте од удаљеног одредишта у мобилној станици. Овај задатак се може одрадити употребом мобилног IP протокола за мобилну станицу који подржава овај стандард. Уколико мобилна станица користи само DHCP протокол, онда се исти резултат може добити комбиновањем DHCP протокола за мобилну станицу и MIP протокола који се користи само за крајње станице[2].

WiMAX Forum је обратио пажњу на неколико проблема умрежавања користећи структуру стандардних протокола прилагођених за специфичне проблеме WiMAX мрежа. Један од проблема је да не постоји стандард и уобичајени начин за пребацивање конфигурационих параметара са централизованог сервера у мобилну станицу. У принципу када мобилна станица улази у мрежу (по први пут или током хендовера) процедура аутентификације је једина фаза у којој су мобилни терминал, ASN мрежа и CSN мрежа присиљени да комуницирају. То је једини тренутак када се аутентификација, умрежавање и IP параметри могу пребацити са позадинског управљања ка остатку мрежа. У пракси ово се дешава по следећој шеми:

- Параметри за одређени терминал или крајњег корисника чувају се у бази података којој RADIUS сервер има приступ;
- Ови параметри се премештају у аутентификатор користећи атрибуте RADIUS протокола преко референтних тачака R3 и R5;

[1] *Mobile IP Foreign Agent*

[2] *Back-end*

- Од аутентификатора, користећи прилагођене протоколе, премештају се на друге елементе мреже. На пример QoS параметри су премештени у базну станицу заједно са MSK[1] кључем који је договорен у току аутентификације;
- Ако неки од њих морају да се преносе у мобилну станицу онда се они убацују у DHCP екстензију која ће бити коришћена за мобилну станицу преко R1 референтне тачке.

Треба уочити да је RADIUS протокол једини стандардни начин којим се премештају параметри из CSN мреже у ASN мрежу. У ASN мрежи крајња тачка која са њима управља је IEEE802.1x аутентификатор.

13.10.7 DHCP протокол и WiMAX мобилни системи

Мобилна станица која не подржава MIP протокол може да има статичку преконфигурисану IP адресу или може да користи DHCP протокол да добије нову IP адресу на сваком хендоверу (приликом сваког преузимања станице). Чак и у првом случају хендовер[2] не може да буде транспарентно за IP слој пошто је потребно остале мрежне параметре иновирати. Посебно на страни клијента мобилна станица ће морати да ажурира своју путању (руту) ка новом подразумеваном међумрежном пролазу или IP адреси неког DNS сервера присутног у новој мрежи. Не постоји други стандардни протокол који би могао да пребацује ове параметре ка мобилној станици осим DHCP протокола. Тако да ће мобилна станица, чак и ако има статички конфигуриасану IP адресу, имати потребу да користи DHCP протокол.

На крају успешне аутентификације ASN мрежи још увек није познато да ли мобилна станица јесте MIP компатибилна или није. Разлика је заснована на активности мобилне станице. Ако је први пакет који шаље мобилна станица DHCP DISCOVER рам, ASN мрежа ће закључити да је DHCP компатибилна станица. Ако је рам MIP захтев за регистрацију ASN мрежа ће закључити да је реч о MIP компатибилној станици. У првом случају ће морати да активира DHCP процедуре а можда и PMIP[3] процедуру.

---

[1] *Master Session Key* - привремени криптографски кључ из кога се изводе остали кључеви који служе за шифровање и дешифровање порука у току трајања једне сесије.

[2] Преузимање станице, прелазак станице, предаја станице

[3] *Proxy-MIP*

DHCP протокол може бити конфигурисан у WiMAX мрежи на два начина. Може се користити субјекат (целина[1]) који игра улогу DHCP агента за прослеђивање или DHCP посредника. У првом случају одговарајућа целина у ASN мрежи се не понаша као класичан DHCP сервер већ само прослеђује захтеве удаљеном серверу. У другом случају сам ASN је тај који одговара на DHCP REQUEST рам и додељује IP адресу мобилној станици. Треба имати на уму да, уколико желимо да сесија остане активна, IP адреса мора бити иста пре и после предаје мобилне станице.

Ако се DHCP сервер понаша као реле (агент за прослеђивање) његова улога биће да проследи DHCP REQUEST на други сервер који мора да буде исти онај који је користила мобилна станица у својој првој аутентификацији. DHCP реле (агент за прослеђивање) не може знати IP адресу овог сервера, тако да ова информација мора бити укључена, на неки начин, у информације које ASN мрежа прима на крају аутентификације, као RAIDUS атрибут. Овај параметар може бити унапред конфигурисан (преконфигурисан) у ASN мрежи, али такав приступ није употребљив у мрежама са више оператера[2] у које клијент може да дође из спољне мреже[3].

Слично томе, ако је ASN мрежа конфигурисана да се понаша као DHCP посредник, она ће морати да одговори на DHCP DISCOVER поруку са истом IP адресом коју је користила мобилна станица пре хендовера (преузимање станице). Порука DHCP DISCOVER може да носи одређену опцију којом клијент тражи да му се додели IP адреса коју он већ користи. Из сигурносних разлога ова одлука не може бити заснована на IP адреси коју мобилна станица за себе захтева пошто су DHCP рамови неаутентификовани и мобилна станица можда покушава да добије туђу адресу. Опет, IP адреса која се додељује мобилној станици биће укључена у одређени RADIUS атрибут који је укључен у приступи/прихвати пакет који долази из CSN мреже.

[1] *Entity*
[2] *Multi-operator networks*
[3] *Foreign network*

13.10.8 Мобилни IP протокол и WiMAX мобилни системи

Мобилни IP протокол је стандардни комуникациони протокол[1] организације IETF[2] чија је основна намена да омогући кориснику да има сталну IP адресу док се креће и мења своју тачку приступа мрежи. Ова функција веома је корисна за све оне везе код којих је потребна стална веза током корисниковог роминга кроз мрежу. У претходном одељку објаснили смо како је могуће користити DHCP да се задржи иста IP адреса за време хендовера. Ово је неопходно да се задржи сесија активном, али то није довољно. Други крај комуникационе везе ће примати IP пакете, али његови одговори ће бити усмерени ка сопственој мрежи чвора. Достављање пакете из сопствене мреже у гостујућу мрежу[3] обезбеђује мобилни IP протокол.

Мобилни IP протокол решава проблем одржавања сесије помоћу две адресе за сваки терминал: сопствене адресе и заштитне адресе (COA[4] адреса). Сопствена адреса је IP адреса чвора коју је добио од свог сопственог агента (HA[5] агента) на својој сопственој мрежи. Ова IP адреса је она која ће остати непромењена (фиксна) за време роминга. COA[6] адреса је адреса коју је чвору дао спољни агент (FA[7] агент), агент мобилности који је задужен да обезбеди IP адресе терминалима који су прикључени на V-NSP мрежу.

Начин на који мобилни IP (MIP) протокол ради може се укратко представити:

1. У време своје прве аутентификације мобилна станица добија IP адресу од свог HA агента у својој сопственој мрежи. Докле год је станица покривена овом мрежом пакети ка и од тог чвора усмеравају се једноставно користећи ову адресу и мобилни IP протокол се не користи.
2. Када се мобилни терминал помери изван своје сопствене (матичне) мреже ка другој мрежи он трага за FA агентом и добија COA адресу од њега.

[1] Описан у документима RFC5944 и RFC4721.
[2] *Internet Engineering Task Force*
[3] *Visited network*
[4] *Care-of-Address*
[5] *Home Agent*
[6] *Care-of-Address*
[7] *Foreign Agent*

3. Након доделе COA адресе мобилна станица започиње MIP регистрацију са HA агентом користећи стандардну процедуру слањем захтева за регистрацију (RRQ[1]). Ова регистрација има намеру да обавести HA агента о тренутној COA адреси која треба да се користи да би се дошло до терминала на посећеној (гостујућој) мрежи. Да би потврдио да је регистрацију добио HA агент шаље назад одговор (RRP[2]).
4. Пут до и од мобилне станице мора да се промени након регистрације у спољном домену. Када мобилна станица жели да комуницира са удаљеном станицом она користи директну путању из своје гостујуће мреже. Када удаљена станица жели да комуницира са мобилном станицом она мора да користи технику троугластог рутирања[3]. Удаљена станица шаље свој пакет (IP датаграм) на матичну адресу мобилне станице. HA агент ће добити овај пакет, одредити тренутну COA адресу за мобилну станицу (на основу последњег поступка регистрације), послати пакет правом FA агенту који прослеђује IP датаграм мобилној станици.

Стандард IEEE	Опис	Статус
802.16-2001	Бежични широкопојасни фиксни приступ (*Fixed Broadband Wireless Access*) 10 - 63 GHz	Замењен
802.16.2-2001	Препоручена пракса за коегзистенцију (*Recommended practice for coexistence*)	Замењен
802.16-2004	Ваздушни интерфејс за бежични широкопојасни фиксни приступ (*Air Interface for Fixed Broadband Wireless Access System*) Збирно 802.16 - 2001, 802.16a, 802.16c и P802.16d)	Замењен
802.16f-2005	MIB база података (*Management Information Base*) за 802.16-2004	Замењен
802.16e-2005	Мобилни бежични широкопојасни приступни систем (*Mobile Broadband Wireless Access System*)	Замењен

[1] *Registration Request*
[2] *Registration Reply*
[3] *Triangular routing*

Стандард IEEE	Опис	Статус
802.16k-2007	Премошћавање (*Bridging*) за 802.16 (проширење за IEEE802.1D)	Важећи
P802.16i	Мобилна MIB база (*Mobile Management Information Base*) Пројекат спојен са 802.16-2009	Припојен
802.16-2009	Ваздушни интерфејс за системе са фиксним и мобилним широкопојасним приступом (*Air Interface for Fixed and Mobile Broadband Wireless Access System*) Збирно: 802.16-2004, 802.16-2004/Cor 1, 802.16e, 802.16f, 802.16g и P802.16i.	Важећи
802.16j-2009	Преусмеравање у више скокова (*Multihop relay*)	Важећи
802.16h-2010	Механизам за унапређену коегзистенцију за рад код изузетих лиценци (*Improved Coexistence Mechanisms for License-Exempt Operation*)	Важећи
802.16m-2011	Унапређен ваздушни интерфејс са брзинама од 100Mb/s за мобилне и од 1Gb/s за фиксне кориснике. (*Advanced Air Interface with data rates of 100 Mbit/s mobile and 1Gbit/s fixed*) Познат као *Mobile WiMAX Release 2* или *WirelessMAN-Advanced* са циљем да испуни ITU-R IMT-Advanced захтеве за 4G системе.	Важећи
P802.16n	Мреже високе поузданости (*Higher Reliability Networks*)	У току је израда

Табела 13.6 Опис и статус стандарда IEEE802.16x

13.11 Питања и задаци

1. Описати намену регулаторних тела за бежичне системе.
2. Које асоцијације произвођача бежичних система постоје?
3. Који се опсези без посебних дозвола користе код бежичних система?
4. Категоризовати бежичне рачунарске мреже по физичкој топологији?
5. Набројати предности испреплетаних мрежа.
6. На коликом су растојању носиоци и колика је ширина канала код IEEE 802.11?
7. Који непреклапајући канали могу истовремено да се користе у бежичним локалним рачунарским мрежама Европе а који у САД?

8. Описати основни принцип рада протокола CSMA/CA.
9. Описати проблем скривеног терминала и начин разрешавања.
10. Нацртати и описати архитектуру протокола по стандарду IEEE802.11-2012. Које су измене предложене проширењем IEEE802.11s?
11. Описати приступне категорије и њихову намену.
12. Описати времена дефинисана стандардом IEEE802.11-2012.
13. Нацртати и описати компоненте инфраструктурне IEEE802.11 мреже.
14. Како се повезују инфраструктурна и испреплетана мрежа?
15. Описати MIMO технологију и њену примену.
16. Која су основна обележја Bluetooth технологије.
17. Направити компаративну анализу Bluetooth верзија стандарда.
18. Која је намена Bluetooth профила? Навести један пример.
19. Нацртати и описати архитектуру протокола Bluetooth система.
20. Упоредити Bluetooth и IEEE802.15 архитектуре протокола?
21. Који се IEEE стандард односи на сензорске мреже?
22. Нацртати и описати ZigBee архитектуру протокола.
23. Које врсте чворова постоје у ZigBee мрежама?
24. Који фреквенцијски опсези се користе код сензорских мрежа?
25. Који се алгоритми за рутирање користе код сензорских мрежа?
26. Која фреквенцијска подручја користи WiMAX Forum за тестирање?
27. Упоредити карактеристике TDM и OFDM система
28. Набројати и описати правце развоја стандарда IEEE802.16.
29. Описати принцип рада MIMO система.
30. Описати принцип рада мобилних система у бежичним IP мрежама.

Индекс скраћеница

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

V

W

Z

Индекс појмова

А

Б

В

Г

Д

Е

Ж

З

И

Ј

К

Л

М

Н

О

П

Р

С

T

Ћ

У

Ф

Х

Ц

Ч

Ш

Литература

[1]. Васиљевић, В., *Рачунарске мреже*, ВИШЕР, Београд, 2008.

[2]. Goralski, W., *The Illustrated Network: How TCP/IP Works in a Modern Network*, Morgan Kaufmann Publishers, Burlington, MA, 2010.

[3]. Dijkstra, E., *A Note on Two Problems in Connexion with Graphs Numer*, Math., vol. 1, стр. 269-271, 1959.

[4]. Jacobson, V. , *Congestion avoidance and control*, Proceedings of SIGCOMM '88 (Aug. 1988), ACM.

[5]. Karn, P., Partridge, C., *Estimating round-trip times in reliable transport protocols*, In Proceedings of SIGCOMM '87, 1987.

[6]. Клајн, И., Шипка, М., *Велики речник страних речи и израза*, Прометеј, Нови Сад, 2007.

[7]. Kleinrock, L., Kamoun, F., *Hierarchical routing for large networks*, Computer Networks, vol. 1, no. 3, pp. 155–174, 1977.

[8]. Kurose, J., Ross, K., *Computer Networking, a Top-Down Approach,* Addison-Wesley, Boston, MA, 2012.

[9]. Papadopuouls, C., Parulkar, M. , *Experimental Evolution of SunOS IPC and TCP/IP Protocol Implementation*, IEEE/ACM Transaction on Networking, vol. 1, no. 2, pp. 199-216, 1993.

[10]. Perkins, E., Royer, E., *The Ad Hoc On-Demand Distance-Vector Protocol*, In Ad Hoc Networking, Ed. C. Perkins, Boston, Addison-Wesley, 2001.

[11]. Stallings, W., *Computer Networking With Internet Protocols and Technology*, Pearson Education, Inc., Upper Saddle River, New Jersey, 2004.

[12]. Stallings, W., *Data and Computer Communications*, Pearson Education, Inc., Upper Saddle River, New Jersey, 2004.

[13]. Stevens, R. , *TCP/IP Ilustrated, Volume 1, the Protocols*, Addison-Wesley, Boston, MA, 2011.

[14]. Tanenbaum, A., *Computer Network*, Pearson Education, Inc., Upper Saddle River, New Jersey, 2003.

[15]. Tang, S. , Möller, P., Sharif, H., Ed., *Wimax Security And Quality Of Service*, 2010 John Wiley & Sons Ltd., Chichester, West Sussex, 2010.
[16]. Tse, D., Viswanath, P., *Fundamentals of Wireless Communication*, Cambridge University press, Cambridge, 2005.
[17]. Chiu, D., Jain, R., *Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks*, доступно на: http://home.ie.cuhk.edu.hk/~dmchiu/aimd.pdf
[18]. WiMAX Forum, *WiMAX Forum Network Architecture. Stage 2: Architecture Tenets, Reference Model and Reference Points*. V. 1.2, WiMAX Forum Std., 2009.
[19]. WiMAX Forum, *WiMAX Forum Network Architecture. Stage 3: Detailed Protocols and Procedures,* WiMAX Forum Std., 2009.
[20]. http://www.ieee.org
[21]. http://www.wikipedia.org
[22]. http://www.wimaxforum.org

ISBN-978-86-7982-188-1

Висока школа електротехнике и рачунарства
струковних студија