Računarske komunikacije - Prvi kolokvijum

Poglavlje I

R16. *Zamislite slanje više paketa od izvornog do odredišnog računara nepromenjivom putanjom. Navedite od čega se sve sastoji ukupno kašnjenje paketa od jednog do drugog kraja. Koja su kašnjenja uvek ista, a koja promenjiva?*

Odgovor:
Kašnjenje paketa se sastoji od kašnjenja usled obrade, kašenjenja usled prenosa, kašnjenja usled prostiranja i kašnjenja usled čekanja u redu. Sva kašnjenja osim kašnjenja usled čekanja u redu su ista.

***

R19. *Pretpostavimo da računar A želi da pošalje veliku datoteku do računara B. Putanja između računara A i računara B ima tri linka, brzina R_1 = 500 kb/s, R_2 = 2 Mb/s i R_3 = 1 Mb/s.*

a) *Pretpostavljajući da u mreži nema drugog saobraćaja, koja je propusna moć za prenos ove datoteke?*
b) *Pretpostavimo da datoteka ima 4 miliona bajtova. Koliko će, približno, trajati prenos te datoteke do računara B?*
c) *Ponovite (a) i (b), ali sada sa brzinom R_2 smanjenom na 100 kb/s.*

Odgovor:
a) Propusnu moć predstavlja najmanja brzina linka u mreži, a to je ovde **500 kb/s**.
b) Vreme prenosa datoteke : Veličina podataka / Propusna moć => **L/R**
L = 4 mil. bajtova = 32 mil. bitova = 32 * 10^6 b
R = 500 kb/s = 5 * 10^5 b/s

$$\frac{32 * 10^6 \text{ b}}{5 * 10^5 \text{ b/s}} = \frac{320 \text{ s}}{5} = 64\text{s}$$

c) Isti postupak kao iznad, samo što je ispod 1 * ... usled nove propusne moći => 320 s / 1 = **320 sekundi**.

***

R22. *Navedite pet zadataka koje jedan sloj može da obavlja. Da li je moguće da jedan od tih zadataka (ili više njih) obavljaju dva sloja (ili više njih)?*

Odgovor:
Kontrola grešaka, kontrola protoka, segmentacija i reasemblovanje (ponovno sastavljanje), multipleksovanje i ostvarivanje konekcije. Da, ovi zadaci se mogu nalaziti na više slojeva, na primer kontrola grešaka se često nalazi na više slojeva.

R23. *Kojih su pet slojeva u skupu internet protokola? Šta su osnovna zaduženja svakog od tih slojeva?*

Odgovor:
Pet slojeva, od vrha ka dnu, su:

- **Aplikativni sloj** - Na ovom sloju se nalaze mrežne aplikacije i njihovi protokoli, kao i HTTP, FTP...
- **Transportni sloj** - Prenosi poruke aplikativnog sloja između krajnjih tačaka aplikacije. TCP i UDP.
- **Mrežni sloj** - Prenosi pakete mrežnog sloja *(datagram)* od jednog računara do drugog. IP protokol.
- **Sloj veze** - Prenosi pakete od jednog čvora do sledećeg. Wi-Fi, Ethernet…
- **Fizički sloj** - Zadužen za prenošenje pojedinačnih bitova unutar istog okvira između susednih čvorova.

***

R24. *Šta je poruka aplikativnog sloja? Segment transportnog sloja? Datagram mrežnog sloja? Okvir sloja veze?*

Odgovor:
- Podaci koje aplikacija želi da pošalje i koji bivaju prosleđeni transportnom sloju.
- Stvoren od strane transportnog sloja, enkapsulira poruku apl. sloja sa zaglavljem transportnog sloja.
- Enkapsulira segment transportnog sloja sa zaglavljem mrežnog sloja.
- Enkapsulira datagram mrežnog sloja sa zaglavljem sloja veze.

--- Zadaci ---

P10. *Zamislite paket dužine L koji nastaje u krajnjem sistemu A, putuje preko tri linka do odredišnog krajnjeg sistema. Ova tri linka su povezana pomoću dva komutatora paketa. Označimo redom sa: d_i , s_i i R_i dužinu, brzinu prostiranja i brzinu prenosa linka i, za i = 1, 2, 3. Kašnjenje obrade komutatora paketa je d_{obrada}. Pod pretpostavkom da ne postoji kašnjenje usled čekanja u redu, koliko je ukupno kašnjenje paketa od jednog do drugog kraja u zavisnosti od d_i s_i R_i (i = 1, 2, 3) i L?*

*Pretpostavimo sada da je paket veličine 1500 bajtova, brzina prostiranja sva tri linka jednaka 2,5 * 10^8 m/s, brzina prenosa sva tri linka je 2 Mb/s, kašnjenje usled obrade komutatora paketa je 3 msec, dužina prvog linka je 5000 km, a dužina drugog linka je 4000 km, dok je dužina poslednjeg linka 1000 km. Koliko je kašnjenje od jednog do drugog kraja za ove vrednosti?*

Odgovor:

1. Prvom krajnjem sistemu *(A)* je potrebno L/R_1 vremena da pošalje paket na prvi link *(skroz levo isprek. crte)*.
2. Paket se prostire kroz prvi link za d_1/s_1 vremena.
3. Komutator (*S1)* dodaje kašnjenje usled obrade podataka od d_{obrada} vremena.
4. Nakon primanja celog paketa, *S1* je potrebno L/R_2 vremena da prenese paket na drugi link.
5. Paket se prostire kroz drugi link *(isprekidane crtice na sredini)* za d_2/s_2 vremena.
6. Na isti način kao i pre možemo naći kašnjenja prouzrokovana od *S2* i trećeg linka: d_{obrada} , L/R_3 i d_3/s_3.

Kombinujući sva ova kašnjenja dobijamo sledeću računicu:

$d_{od\text{-}A\text{-}do\text{-}B} = L/R_1 + d_1/s_1 + d_{obrada} + L/R_2 + d_2/s_2 + d_{obrada} + L/R_3 + d_3/s_3$

$d_{od\text{-}A\text{-}do\text{-}B} = L/R_1 + L/R_2 + L/R_3 + d_1/s_1 + d_2/s_2 + d_3/s_3 + d_{obrada} + d_{obrada}$

** Imamo samo 2 kašnjenja usled obrade jer B ne obrađuje paket već ga prima.*

Rešenje zadatka:

L = 1500 B = 1500 * 8 b = 12000 b = 12 * 10^3 b

$s_1 = s_2 = s_3$ = 2,5 * 10^8 m/s = 25 * 10^7 m/s

$R_1 = R_2 = R_3$ = 2 Mb/s = 2 * 10^6 b/s

d_{obrada} = 3 msec = 3 * 10^{-3} s

d_1 = 5000 km = 5 * 10^3 km = 5 * 10^6 m

d_2 = 4000 km = 4 * 10^3 km = 4 * 10^6 m

d_3 = 1000 km = 1 * 10^3 km = 1 * 10^6 m

$d_{od\text{-}A\text{-}do\text{-}B}$ = 6 + 6 + 6 + 20 + 16 + 4 + 3 + 3 = 18 + 40 + 6 = **64 ms**

$\frac{12 * 10^3 b}{2 * 10^6 b/s} = 6 * 10^{-3} s = 6 ms$ $\quad \frac{L}{R_1} = \frac{L}{R_2} = \frac{L}{R_3}$

$\frac{d_1}{s_1} = \frac{5 * 10^6 m}{25 * 10^7 m/s} = 0.2 * 10^{-1} s = 20 ms$ $\quad 2 * 10^{-2} s$

4/25 = 0.16

1/25 = 0.04

P24. *Pretpostavimo da hitno želite da isporučite 40 terabajtova podataka iz Bostona do Los Anđelesa. Na raspolaganju vam je posvećen link brzine 100 Mb/s za prenos podataka. Da li ćete se odlučiti za prenos podataka putem ovog linka, ili ćete koristiti FedEx-ovu isporuku preko noći? Objasnite.*

Odgovor:
40 terabajta = 40 * 10^{12} bajtova = 320 * 10^{12} bitova
100 Mb = 100 * 10^{6} bitova.

Vreme prenosa korišćenjem posvećenog linka : *veličina podataka / propusni opseg*
L = 320 * 10^{12} b
R = 100 * 10^{6} b/s
T = L/R

$$\frac{320 * 10^{12}\ b}{100 * 10^{6}\ b/s} = 3,2 * 10^{6}\ s = 3\,200\,000\ s$$

3 200 000 s / 60 = 53333 min = 888 h = 37 dana.

Logično, iskoristićemo FedEx isporuku jer je jedna noć mnogo kraće vreme od 37 dana koje bismo imali ukoliko bismo prenosili putem linka.

Poglavlje II

R2. *U čemu je razlika između arhitekture mreže i arhitekture aplikacije?*

- Arhitektura mreže se odnosi na organizaciju komunikacionih procesa u slojeve *(pr. petoslojna Internet arhitektura)*. Arhitektura aplikacije je dizajnirana od strane developera i određuje strukturu te aplikacije *(pr. klijent-server ili P2P)*.

***

R3. *Kada par procesa međusobno komuniciraju, koji od ta dva procesa je klijent, a koji server?*

- Proces koji započinje komunikaciju je klijent. Proces koji čeka da bude kontaktiran je server.

***

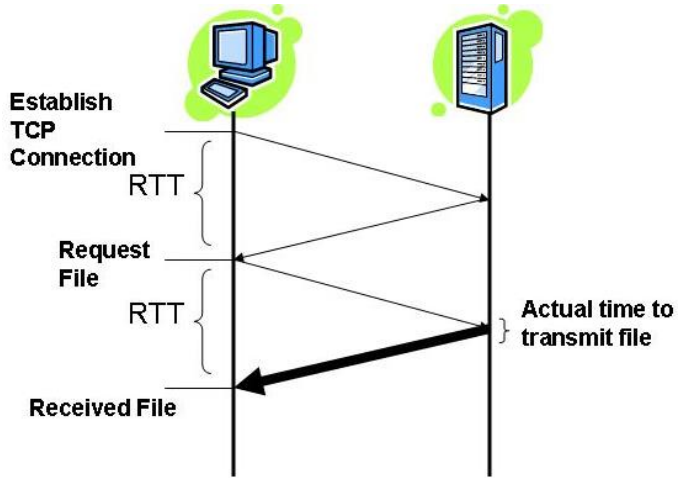
R5. *Pomoću kojih informacija proces koji se izvršava na jednom računaru prepoznaje proces koji se izvršava na drugom računaru?*

- IP adresa odredišnog računara i broj porta soketa odredišnog procesa.

***

R6. *Pretpostavimo da želite da obavite prenos podataka sa udaljenog klijenta na server, što je brže moguće. Da li ćete koristiti protokol UDP ili protokol TCP? Zašto?*

- UDP. Uz UDP prenos može biti završen u jednom roundtrip time - RTT *(vreme povratnog putovanja, dužina vremena koja je potrebna da bi se signal poslao plus dužina vremena potrebnog da se potvrdi da se signal primio).* Klijent pošalje zahtev za prenos u UDP soket, i server šalje odgovor nazad klijentovom UDP soketu. Sa TCP, minimum 2 RTT su potrebna - jedan da uspostavi TCP konekciju, i drugi za klijenta da pošalje zahtev, i za server da pošalje nazad odgovor.



R8. *Navedite četiri opšte vrste usluga koje transportni protokoli nude. Za svaku od tih usluga navedite koju nudi protokol UDP; a koje protokol TCP (ili oba).*

a) Pouzdan prenos podataka - TCP nudi, UDP ne.
b) Garantovanje propusne moći - Nijedan.
c) Ispunjenje vremenskih ograničenja - Nijedan.
d) Bezbednost (*uz pomoć enkripcije/šifrovanja*) - Nijedan.

***

R9. *Sećate se da se protokol TCP može poboljšati SSL uslugom kojom se obezbeđuje bezbednost između procesa, uključujući šifrovanje. Da li SSL radi na transportnom sloju ili na aplikativnom sloju? Ukoliko programer aplikacije želi da koristi protokol TCP poboljšan uslugom SSL, šta mora da uradi?*

- SSL radi na aplikativnom sloju. SSL soket uzima neenkriptovane podatke iz aplikativnog sloja, šifruje ih i prosleđuje TCP soketu. Ukoliko programer želi da TCP poboljša pomoću SSL-a, on mora da unese SSL kod u aplikaciju.

***

R10. *Šta znači protokol usaglašavanja (eng. handshaking)?*

- Protokol koristi usaglašavanje ukoliko dva uređaja koja komuniciraju prvo razmene kontrolne pakete pre nego što pošalju podatke jedan drugom. SMTP koristi handshaking na aplikativnom sloju dok HTTP ne.

***

R12. *Zamislite veb lokaciju za elektronsku trgovinu koja želi da čuva podatke o svakom svom korisniku. Opišite kako je to moguće učiniti pomoću kolačića.*

- Kada korisnik prvi put poseti sajt, server pravi jedinstveni identifikacioni broj, tj. pravi unos u back-end bazi podataka, i vraća ovaj identifikacioni broj kao broj kolačića. Ovaj broj kolačića se skladišiti na korisnikovom uređaju i biva ažuriran od strane pretraživača. Prilikom svake naredne posete, pretraživač šalje broj kolačića nazad sajtu. Tako sajt zna kada ovaj korisnik (*ili preciznije, pretraživač*) posećuje sajt.

***

R13. *Opišite kako veb keširanje smanjuje kašnjenje pri dobijanju zahtevanog objekta. Da li veb keširanje smanjuje kašnjenje za sve zahtevane objekte od strane korisnika, ili samo za neke objekte? Zašto?*

- Veb keširanje dovodi željeni sadržaj korisnika ,,*bliže*'' njemu. Veb keširanje može da smanji kašnjenje za sve objekte, čak i one koji nisu keširani, pošto keširanje smanjuje saobraćaj na linkovima.

R15. *Zašto se kaže da protokol FTP šalje kontrolne informacije ,,izvan opsega''?*

- FTP koristi dve paralelne TCP konekcije, jednu konekciju za slanje kontrolnih informacija *(kao što je zahtev za prenos fajla/datoteke)* i dodatnu konekciju za slanje te datoteke. Pošto kontrolne informacije nisu poslate preko iste konekcije kao i datoteka, FTP šalje kontrolne informacije *,,izvan opsega''*.

***

R16. *Pretpostavimo da Alisa sa svog naloga za e-poštu, zasnovanom na vebu (recimo Hotmail ili gmail), želi da pošalje poruku Bobu koji svom serveru za e-poštu pristupa preko protokola POP3. Opišite put ove poruke od Alisinog do Bobovog računara. Obavezno navedite sve protokole aplikativnog sloja koji se koriste za prenos poruke sa jednog na drugi računar.*

- Poruka je prvo poslata sa Alisinog uređaja njenom mail serveru preko HTTP-a. Potom Alisin mail server šalje poruku Bobovom mail serveru preko SMTP-a. Bob na kraju prenosi poruku sa svog mail servera na svoj uređaj preko POP3.

***

R19. *Da li je moguće da veb server i server za e-poštu neke organizacije imaju istovetan pseudonim za imena svojih računara (recimo foo.com)? Kog tipa je RR zapis u kome se nalazi ime računara servera za e-poštu?*

- Da, veb server i mail server neke organizacije mogu da imaju isti alias za naziv host-a. MX zapis se koristi za mapiranje naziva hosta mail servera za njegovu IP adresu.

***

R25. *Navedite barem četiri aplikacije kojima prirodno odgovara P2P arhitektura (Pomoć: Distribucija datoteka i instant razmena poruka su dve).*

- Distribucija datoteka, instant razmena poruka, video streaming, raspodeljeno izračunavanje.

***

R26. *UDP serveru koji je opisan u odeljku 2.7 bio je potreban samo jedan soket, dok su TCP server bila potrebna dva. Zašto? Kada bi TCP server morao da podrži n istovremenih veza, od kojih je svaka sa različitog računara klijenta, koliko bi mu soketa za to bilo potrebno?*

- Kod UDP servera ne postoji soket *,,dobrodošlice''*, i svi podaci od različitih klijenata ulaze u server putem ovog jednog soketa. Kod TCP servera, postoji soket dobrodošlice, i svaki put kada klijent započne konekciju sa serverom, novi soket se kreira. Samim tim, da bi podržao *n* istovremenih veza, serveru je potrebno *n+1* soketa.

R27. *Zbog čega u klijentsko-serverskoj aplikaciji, koja koristi protokol TCP iz odeljka 2.7, serverski program mora da se pokrene pre klijentskog? Zbog čega u klijentsko-serverskoj aplikaciji za protokol UDP klijentski program može da se pokrene pre serverskog?*

- Čim se klijent pokrene, TCP aplikacija će pokušati da uspostavi TCP konekciju sa serverom. Ako TCP server nije pokrenut, onda klijent neće uspeti da uspostavi vezu. Kod UDP aplikacija klijent ne uspostavlja konekcije odmah nakon pokretanja.

--- Zadaci ---

P1. *Tačno ili netačno?*

a) *Korisnik zahteva veb stranu koja se sastoji od nekog teksta i tri slike. Za ovu stranu klijent šalje jednu poruku zahteva, a prima četiri poruke odgovora.* => **Netačno**.
b) *Dve različite veb strane (na primer, www.mit.edu/research.html i www.mit.edu/students.html) mogu da se pošalju istom postojanom vezom.* => **Tačno**.
c) *Sa nepostojanom vezom između pretraživača i izvornog servera moguće je da jedan TCP segment prenosi dve različite HTTP poruke zahteva.* => **Netačno**.
d) *Zaglavlje Date: u HTTP poruci odgovora pokazuje kada je objekat koji se nalazi u tom odgovoru poslednji put izmenjen.* => **Netačno.**
e) *HTTP poruka odgovora nikad nema prazno telo poruke*. => **Netačno.**

***

P3. *Zamislite HTTP klijenta koji želi da preuzme veb dokument sa određene URL adrese. IP adresa odgovarajućeg HTTP servera je u početku nepoznata. Koji su protokoli transportnog i aplikativnog sloja, pored protokola HTTP, potrebni u ovom slučaju?*

- Slojevi aplikativnog sloja : DNS i HTTP
  Slojevi transportnog sloja : UDP za DNS; TCP za HTTP.

P4. *Posmatramo sledeći niz ASCII znakova koje je pokupio program Wireshark pošto je pretraživač poslao HTTP GET poruku (u stvari, ovo je stvarni sadržaj HTTP GET poruke). Karakteri <cr><lf> su znakovi za novi red i povratak na početak reda (odnosno <cr> znak u tekstu koji sledi predstavlja znak za novi red koji se nalazi na tom mestu u zaglavlju HTTP poruke). Odgovorite na sledeća pitanja, navodeći gde ste u HTTP GET poruci pronašli odgovor.*

```
GET /cs453/index.html HTTP/1.1<cr><lf>Host: gai
a.cs.umass.edu<cr><lf>User-Agent: Mozilla/5.0 (
Windows;U; Windows NT 5.1; en-US; rv:1.7.2) Gec
ko/20040804 Netscape/7.2 (ax) <cr><lf>Accept:ex
t/xml, application/xml, application/xhtml+xml, text
/html;q=0.9, text/plain;q=0.8,image/png,*/*;q=0.5
<cr><lf>Accept-Language: en-us,en;q=0.5<cr><lf>Accept-
Encoding: zip,deflate<cr><lf>Accept-Charset: ISO
-8859-1,utf-8;q=0.7,*;q=0.7<cr><lf>Keep-Alive: 300<cr>
<lf>Connection:keep-alive<cr><lf><cr><lf>
```

a) Koja je URL adresa dokumenta koju zahteva pretraživač?

- URL adresa zahtevanog dokumenta je *http://gaia.cs.umass.edu/cs543/index.html*. Polje *Host:* navodi naziv servera i */cs543/index.html* navodi naziv dokumenta.

b) Koju verziju protokola HTTP koristi pretraživač?

- Pretraživač koristi HTTP verziju 1.1, što se može videti pre prvog <cr><lf> para.

c) Da li pretraživač traži nepostojanu ili postojanu vezu?

- Pretraživač zahteva postojanu vezu, kao što navodi *Connection: keep-alive* deo.

d) Koja je IP adresa računara na kome se izvršava pretraživač?

- Ovo je trik pitanje. Ova informacija se ne nalazi nigde u HTTP poruci, tako da je nemoguće otkriti IP adresu tog računara samo na osnovu HTTP poruka. Za odgovor na ovo pitanje morali bismo da gledamo u IP datagrame.

e) Koji tip pretraživača započinje ovu poruku? Zašto je potreban tip pretraživača u HTTP poruci zahteva?

- Mozilla/5.0. Ova informacija je potrebna serveru kako bi poslao različite verzije istog objekta različitim tipovima pretraživača.

P5. *Tekst u nastavku pokazuje odgovor koji je poslao server na HTTP GET poruku iz prethodnog pitanja. Odgovorite na sledeća pitanja, navodeći gde ste u poruci ispod, pronašli odgovor.*

```
HTTP/1.1 200 OK<cr><lf>Date: Tue, 07 Mar 2008
12:39:45GMT<cr><lf>Server: Apache/2.0.52 (Fedora)
<cr><lf>Last-Modified: Sat, 10 Dec2005 18:27:46
GMT<cr><lf>ETag: "526c3-f22-a88a4c80"<cr><lf>Accept-
Ranges: bytes<cr><lf>Content-Length: 3874<cr><lf>
Keep-Alive: timeout=max=100<cr><lf>Connection:
Keep-Alive<cr><lf>Content-Type: text/html; charset=
ISO-8859-1<cr><lf><cr><lf><!doctype html public "-
//w3c//dtd html 4.0 transitional//en"><lf><html><lf>
<head><lf> <meta http-equiv="Content-Type"
content="text/html; charset=iso-8859-1"><lf> <meta
name="GENERATOR" content="Mozilla/4.79 [en] (Windows NT
5.0; U) Netscape]"><lf> <title>CMPSCI 453 / 591 /
NTU-ST550A Spring 2005 homepage</title><lf></head><lf>
<much more document text following here (not shown)>
```

a) *Da li je server uspešno pronašao odgovarajući dokument ili nije? Kada je odgovor sa dokumentom obezbeđen?*
   - Statusni kod *200* i reč *OK* nam govore da je server uspešno pronašao dokument. Odgovor je obezbeđen 7. marta 2008., u 12:39:45 GMT.

b) *Kada je dokument poslednji put menjan?*
   - Dokument *index.html* je poslednji put menjan Sat, 10 Dec 2005., u 18:27:46 GMT.

c) *Koliko bajtova ima dokument koji se vraća?*
   - 3874 bajtova.

d) *Koji su prvih 5 bajtova dokumenta koji se vraća u odgovoru? Da li se server složio da uspostavi postojanu vezu?*
   - Prvih pet bajtova vraćenog dokumenta su : *<!doc*. Server se složio sa uspostavljanjem postojane veze (*persistent connection*), što se da videti pomoću *Connection: Keep-Alive* polja.

***

P13. *U čemu je razlika između zaglavlja MAIL FROM: u protokolu SMTP i zaglavlja FROM: u samoj poruci?*
- *MAIL FROM:* u SMTP je poruka SMTP klijenta koja identifikuje pošaljioca poruke SMTP serveru. *FROM:* iz same poruke **nije** SMTP poruka, već samo jedan red u telu mail poruke.

***

P14. *Kako protokol SMTP obeležava kraj tela poruke? A kako protokol HTTP? Da li za obeležavanje kraja tela poruke protokol HTTP može da koristi istu metodu kao i protokol SMTP? Objasnite.*

- SMTP koristi red koji sadrži samo tačku da označi kraj tela poruke. HTTP koristi *Content-Length* polje zaglavlja da označi dužinu tela poruke. HTTP ne može da koristi SMTP metodu, jer HTTP poruku mogu činiti binarni podaci, dok kod SMTP-a telo poruke mora biti u 7-bitnom ASCII formatu.

P31. *Instalirajte i kompajlirajte Python programe TCPClient i UDPClient na jedom računaru, a TCPServer i UDPServer na drugom.*

a) Pretpostavimo da program TCPClient izvršavate pre izvršavanja programa TCPServer. Šta se dešava? Zašto?

b) Pretpostavimo da program UDPClient izvršavate pre izvršavanja programa UDPServer. Šta se dešava? Zašto?

c) Šta se dešava, ako koristite različite brojeve portova za klijentsku i serversku stranu?

a) Ako prvo pokrenemo TCPClient, onda će klijent pokušati da uspostavi TCP konekciju sa nepostojećim procesom servera. TCP konekcija neće biti ostvarena.

b) UDPClient ne uspostavlja TCP konekciju sa serverom, tako da bi sve trebalo da radi bez problema ako pokrenemo UDPServer pre nego što unesemo neke podatke na tastaturi.

c) Ako koristimo različite brojeve portova, onda će klijent pokušati da uspostavi TCP konekciju sa pogrešnim ili nepostojećim procesom. Događaće se greške.

***

P32. *Pretpostavimo da u programu UDPClient.py, nakon što kreiramo soket, dodamo još jedan red: clientSocket.bind((‘’, 5432))*
Da li će biti neophodno da promenite program UDPServer.py? Koji su brojevi portova za sokete u programu UDPClient i UDPServer? Koji su to brojevi bili pre ove izmene?
- U originalnom programu, UDPClient ne specifira broj porta kada kreira soket. U ovom slučaju, kod dopušta operativnom sistemu da odabere broj porta. Sa dodatnim redom, kada UDPClient biva pokrenut, nastaje UDP soket sa portom broj 5432.
UDPServer mora da zna klijentov broj porta kako bi mogao da pošalje pakete nazad njegovom soketu. UDPServer određuje klijentov broj porta otvarajući datagram koji primi od klijenta, samim tim UDP server će raditi sa bilo kojim klijentovim brojem porta, uključujući 5432. UDPServer ne mora da bude modifikovan.
Pre : Soket klijenta = x (Odabran od strane OS) ; Soket servera = 9876
Posle : Soket klijenta = 5432

***

P33. *Da li možete da konfigurišete vaš veb pretraživač, tako da otvori više istovremenih veza ka veb lokaciji? Koje su prednosti i nedostaci većeg broja istovremenih TCP veza?*
- Da, možemo. Prednost je da ćemo potencijalno brže preuzeti datoteku. Nedostatak je da ćemo uzeti mnogo bandwith-a, time značajno usporiti preuzimanja ostalih korisnika koji dele naše linkove.

Poglavlje III

R3. *Uzmimo TCP vezu između računara A i računara B. Pretpostavimo da TCP segmenti koji putuju od računara A prema računaru B imaju broj izvornog porta X, a broj odredišnog porta Y. Koji su brojevi izvornog i odredišnog porta za segmente koji putuju od računara B prema računaru A?*
- Izvorišni port broj Y i odredišni port broj X.

R4. *Opišite zašto bi se programer odlučio da se aplikacija izvršava preko protokola UDP, a ne preko protokola TCP.*
- Programer možda ne želi da njegova aplikacija koristi TCP-ovu kontrolu zagušenja, koja može da uspori brzinu slanja podataka aplikacije u vreme zagušenja. Često programeri VoiP i video stream aplikacija biraju da pokreću svoje aplikacije preko UDP-a jer žele da izbegnu TCP-ovu kontrolu zagušenja. Takođe, nekim aplikacijama nije neophodan pouzdan prenos podataka koji TCP osigurava.

R5. *Zbog čega se za prenos glasa i video zapisa u savremenom internetu sve više koristi protokol TCP umesto protokola UDP (Savet: odgovor koji tražimo nema nikakve veze sa mehanizmom kontrole zagušenja protokola TCP.)*
- Pošto je većina firewall-ova podešena da blokiraju UDP saobraćaj, korišćenjem TCP-a za video i zvuk saobraćaj dopušta saobraćaj da prođe kroz firewall.

R6. *Da li je moguće da aplikacija koristi prednosti pouzdanog prenosa podataka, iako se izvršava protokolom UDP? Ako jeste, na koji način?*
- Da, moguće je, programer može da ubaci pouzdan prenos podataka u protokol aplikacionog sloja. Ipak, ovo bi zahtevalo mnogo vremena i dosta debagovanja.

R7. *Pretpostavimo da neki proces na računaru C ima UDP soket sa brojem porta 6789. Pretpostavimo da računari A i B pošalju po jedan UDP segment računaru C sa brojem odredišnog porta 6789. Da li će oba ta segmenta biti upućena na isti soket na računaru C? Ako je tako, kako će proces na računaru C znati koji od tih segmenata potiče od kog računara?*
- Da, oba segmenta će biti upućena na isti soket. Za svaki primljeni segment operativni sistem će obezbediti procesu IP adrese kako bi on odredio izvore pojedinih segmenata.

R8. *Pretpostavimo da se neki veb server izvršava na računaru C na portu 80. Pretpostavimo da taj veb server koristi trajne veze i da trenutno prima zahteve od dva različita računara, A i B. Da li se svi zahtevi upućuju kroz isti soket na računaru C? Ukoliko bi se prenosili kroz različite sokete, da li bi oba ta soketa imala broj porta 80? Objasnite i obrazložite svoj odgovor.*
- Za svaku trajnu vezu (konekciju), veb server pravi odvojene *,,sokete veze''.* Svaki soket veze je definisan četvoro-kombinacijom (source IP adresa, source port broj, destination IP adresa, destination port broj). Kada host C primi IP datagram, on proverava ova četiri polja u njemu da bi odredio kojem soketu bi trebao da prosledi sadržaj TCP segmenta. Samim tim, zahtevi iz A i B prolaze kroz različite sokete. Oba zahteva imaju destination port broj 80, ali imaju različite vrednosti za source IP adrese što ih razlikuje.

R9. *Zašto je bilo neophodno da u protokol **rdt** uvedemo redne brojeve?*
- Oni su neophodni kako bi primalac saznao da li pristižući paket sadrži nove podatke ili je retransmisija.

R10. *Zašto je bilo neophodno da u protokol **rdt** uvedemo tajmere?*
- Da bismo obradili gubitke u kanalu. Ako ACK za poslati paket nije primljen unutar vremena tajmera za taj paket, paket *(ili njegov ACK/NACK)* se smatra izgubljenim i dolazi do retransmisije paketa.

R14. *Tačno ili netačno?*

a) *Računar A šalje veliku datoteku računaru B preko TCP veze. Broj nepotvrđenih bajtova koje A šalje ne može da premaši veličinu prijemnog bafera. =>* **Netačno.**
b) *Veličina TCP prijemnog prozora rwnd se nikada ne menja tokom trajanja veze. =>* **Netačno.**
c) *Pretpostavimo da računar A šalje veliku datoteku računaru B preko TCP veze. Broj nepotvrđenih bajtova koje A šalje ne može da premaši veličinu prijemnog bafera. =>* **Tačno.**
d) *Pretpost. da računar A šalje veliku datoteku računaru B preko TCP veze. Ako je redni broj nekog segmenta u ovoj vezi m, onda je redni broj sledećeg seg. obavezno m + 1. =>* **Netačno.**
e) *TCP segment ima u svom zaglavlju polje za rwnd. =>* **Tačno.**
f) *Pretpostavimo da je poslednja vrednost SampleRTT u nekoj TCP vezi jednaka 1 sekund. Trenutna vrednost TimeoutInterval za tu vezu svakako je >= 1 sekund. =>* **Netačno.**
g) *Pretpostavimo da računar A šalje jedan segment sa rednim brojem 38 i 4 bajta podataka, preko TCP veze računaru B. Broj potvrde prijema koji se nalazi u tom segmentu mora da bude 432. =>* **Netačno.**

R15. *Pretpostavimo da računar A šalje dva TCP segmenta, jedan za drugim, računaru B preko TCP veze. Prvi segment ima redni broj 90; drugi ima redni broj 110.*

a) *Koliko podataka sadrži prvi segment?*
   - 20 bajtova.
b) *Pretpostavimo da se prvi segment izgubi, ali da drugi segment stigne do B. Koji je broj potvrde koji računar B šalje računaru A?*
   - ack number = 90.

R16. *Uzmimo primer sa Telnet-om, opisan u odeljku 3.5. Nekoliko sekundi nakon slova ,,C'', korisnik upiše slovo ,,R''. Koliko se segmenata šalje posle upisivanja slova ,,R'' i šta se stavlja u polja rednog broja i broja potvrde prijema ovih segmenata?*
- 3 segmenata.
Prvi segment : seq = 43, ack = 80;
Drugi segment : seq = 80, ack = 44;
Treći segment : seq = 44, ack = 81.

R17. *Pretpostavimo da na nekom linku, koji je usko grlo brzine R b/s, postoje dve TCP veze. Svaka od njih šalje veliku datoteku (u istom smeru, preko linka koji je usko grlo). Prenos datoteka počinje istovremeno. Koju brzinu prenosa bi TCP pokušao da dodeli svakoj od ovih veza?*
- Sa obzirom da imamo dve veze koje dele jedan link, R/2.

R18. *Tačno ili netačno? Posmatramo kontrolu zagušenja protokola TCP. Kada kod pošaljioca istekne tajmer, vrednost* ***ssthresh*** *postavlja se na jednu polovinu njegove prethodne vrednosti.*
- Netačno, postavlja se na polovinu trenutne vrednosti prozora zagušenja.

--- Zadaci ---

P2. *Pogledajte sliku 3.5. Koje su vrednosti brojeva izvornog i odredišnog porta u segmentima, koji teku od servera prema klijentskim procesima? Koje su IP adrese u datagramima mrežnog sloja, koji prenose segmente transportnog sloja?*

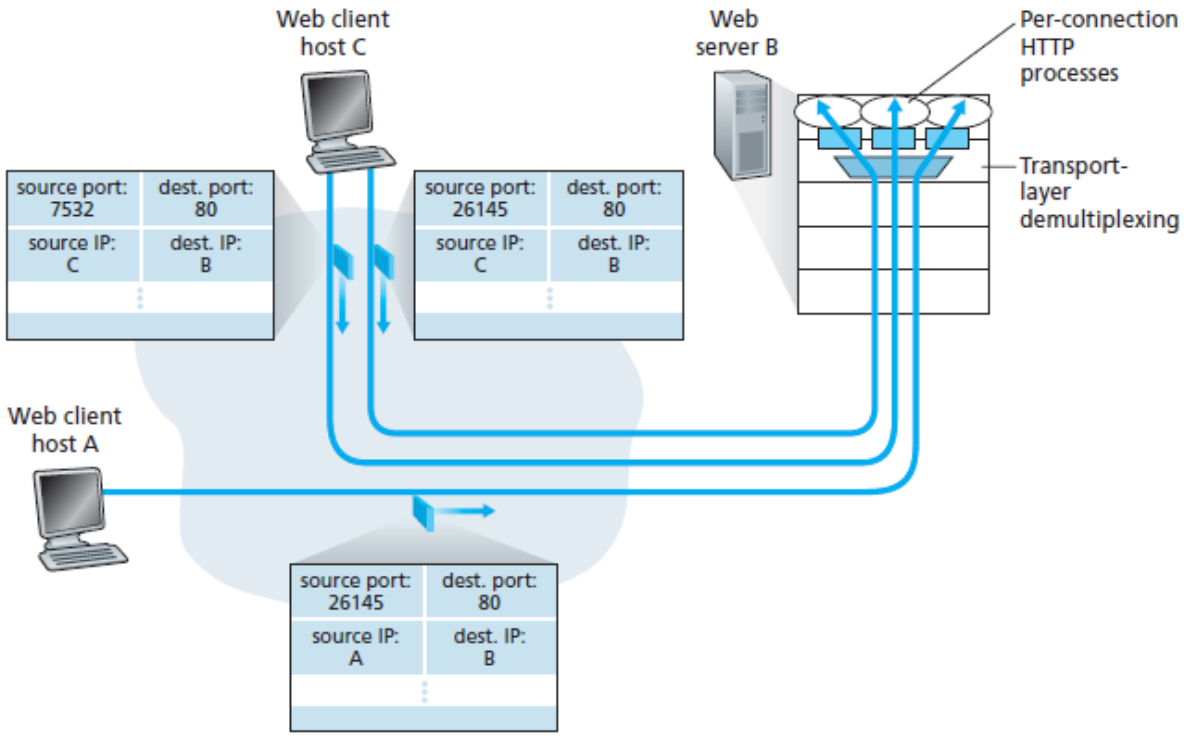


Figure 3.5 ♦ Two clients, using the same destination port number (80) to communicate with the same Web server application

- U segmentima koji teku od servera, izvorišni port sva tri segmenata je 80, a odredišni 7532 za jedan paket host C-a, a 26415 i za drugi paket host C-a i za host A.
Izvorna IP adresa u sva tri slučaja je B, dok je odredišna C za segmente hosta C i A za host A.

P3. *UDP i TCP za svoje kontrolne zbirove koriste prvi komplement. Pretpostavimo da imate sledeća tri 8-bitna bajta: 01010011, 01100110, 01110100. Koji je prvi komplement zbira ovih 8-bitnih bajtova? (Obratite pažnju na to da, iako UDP i TCP koriste 16-bitne reči kada izračunavaju kontrolni zbir, za ovaj problem se zahteva da radite sa sabircima od 8 bitova.) Prikažite šta ste radili. Zašto UDP koristi prvi komplement za zbir; tj. zašto jednostavno ne koristi sam zbir? Kako primalac korišćenjem prvog komplementa otkriva greške? Da li je moguće da greška u jednom bitu ostane neprimećena? A greška u dva bita?*

0 1 0 1 0 0 1 1	1 0 1 1 1 0 0 1
+ 0 1 1 0 0 1 1 0	+ 0 1 1 1 0 1 0 0
1 0 1 1 1 0 0 1	0 0 1 0 1 1 1 0

Prvi komplement zbira ova tri bajta : 1 1 0 1 0 0 0 1.
Radi detektovanja greški, primalac sabira četiri reči *(eng.* ***WORD*** *- 16bits/2bytes)* - tri originalne reči u checksum *(provera ispravnosti poruke)*. Ukoliko zbir sadrži nulu, primalac zna da je došlo do greške. Sve jednobitne greške će biti detektovane, ali dvobitne greške se mogu provući *(npr. ako se poslednja cifra prve reči konvertuje u 0 i poslednja cifra druge reči se konvertuje u 1).*

P4.

a) *Pretpostavimo da imate sledeća dva bajta: 01011100 i 01100101. Koji je prvi komplement zbira ova dva bajta?*
   - Sabiranjem dva bajta dobijamo 11000001. Prvi komplement je 00111110.
b) *Pretpostavimo da imate sledeća dva bajta: 11011010 i 01100101. Koji je prvi komplement zbira ova dva bajta?*
   - Sabiranjem dva bajta dobijamo 01000000. Prvi komplement je 10111111.
c) *Za bajtove u delu (a) navedite primer u kome se jedan bit promeni u svakom od ova dva bajtova, a da se prvi komplement ne promeni.*
   - Prvi bajt = 0101**0**100; drugi bajt = 0110**1**101.

P5. *Pretpostavimo da UDP primalac izračunava internet kontrolni zbir za primljeni UDP segment i da pronađe da se poklapa sa vrednošću koja se nalazi u polju kontrolnog zbira. Može li primalac da bude potpuno siguran da nije došlo do greške u bitovima? Objasnite.*

- Ne može, usled načina na koji se checksum za paket izračunava. Ako su odgovarajući bitovi (koji se poklapaju i međusobno sabiraju) od dve 16-bit reči 0 i 1, onda čak i kada bi se oni obrnuli (prvi postao 1 a drugi 0) rezultat bi ostao isti. Samim tim, prvi komplement koji primaoc izračuna ostaje isti. Ovo znači da će checksum biti verifikovan čak iako se dogodila greška u prenosu.

P25. *Rekli smo da aplikacija može da izabere UDP za transportni protokol zato što UDP nudi bolju kontrolu aplikacije (u odnosu na TCP) toga koji podaci se šalju u segmentima i kada se šalju.*

a) Zašto aplikacija ima veću kontrolu toga koji se podaci šalju u segmentima?
b) Zašto aplikacija ima veću kontrolu toga kada se šalju segmenti?

a) Razmotrimo slanje poruke aplikacije preko transportnog protokola. Uz TCP, aplikacija ispisuje podatke u *connection send buffer* i TCP će grabiti bajtove bez nužnog ubacivanja bilo kakve poruke unutar TCP segmenta; TCP može da stavi više ili manje od jedne poruke u segment. Sa druge strane, UDP enkapsulira unutar segmenta šta god da mu aplikacija da, tako da ako aplikacija da UDP-u poruku aplikacije, ova poruka će biti u telu UDP segmenta. Samim tim, uz UDP, aplikacija ima veću kontrolu podataka koji se šalju u segmentu.
b) Uz TCP, usled kontrole toka i kontrole zagušenja, postoji mogućnost velikih kašnjenja od vremena kada aplikacija ispisuje podatke u sopstveni *send buffer* do trenutka kada su podaci dati mrežnom sloju. UDP nema kašnjenja usled kontrole toka i kontrole zagušenja.

Poglavlje IV

R3. *Kakva je razlika između rutiranja i prosleđivanja?*
- Prosleđivanje ima veze sa pomeranjem paketa sa ruterovog ulaznog porta ka odgovarajućem ruterovom izlaznom portu. Rutiranje ima veze sa određivanjem krajnjih putanja/ruta između izvora i odredišta.

R12. *Da li ruteri imaju IP adrese? Ako da, koliko ih imaju?*
- Da, imaju. Imaju po jednu adresu za svaki interfejs *(izlaz)*.

R13. *Koji je 32-bitni binarni ekvivalent IP adrese 223.1.3.27?*
- 11011111.00000001.00000011.00011011

R15. *Pretpostavimo da između izvornog i odredišnog računara postoje tri rutera. Ako zanemarimo fragmentaciju, preko koliko interfejsa prelazi IP datagram koji se pošalje od izvornog do odredišnog računara? Koliko tabela prosleđivanja će se indeksirati, da bi se datagram preneo od izvora do odredišta?*
- 8 interfejsa (izlaz A, ulaz R1, izlaz R1, ulaz R2, izlaz R2, ulaz R3, izlaz R3, ulaz B). 3 tabele prosleđivanja.

--- Zadaci ---

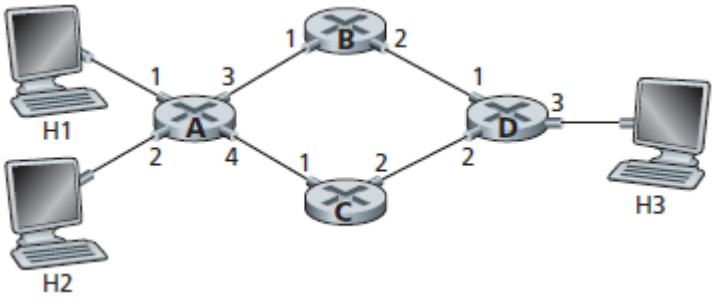
P4. *Uzmimo za primer mrežu dole navedenu.*

a) Pretpostavimo da je ova mreža datagramska mreža. Prikažite tabelu prosleđivanja u ruteru A, tako da se sav saobraćaj, upućen na računar H3, prosleđuje preko interfejsa 3.

b) Pretpostavimo da je ova mreža datagramska mreža. Da li možete da upišete tabelu prosleđivanja u ruter A, tako da sav saobraćaj od H1 koji se upućuje na računar H3, bude prosleđen preko interfejsa 3; a sav saobraćaj usmeren sa računara H2 do računara H3, bude prosleđen preko interfejsa 4? (Savet: ovo je trik pitanje.)

c) Pretpostavimo da je ova mreža, mreža virtuelnog kola, i da se između računara H1 i H3 odvija jedna veza, a druga između računara H2 i H3. Upišite tabelu prosleđivanja u ruter A, tako što se sav saobraćaj, koji je usmeren od računara H1 do računara H3, prosleđuje preko interfejsa 3, dok se sav saobraćaj, usmeren od računara H2 do računara H3, prosleđuje preko interfejsa 4.

d) Pretpostavite isti scenario kao pod stavkom (c), upišite tabelu prosleđivanja u čvorove B, C i D.



a) Podaci upućeni računari H3 se prosleđuju kroz interfejs 3.

Destination Address	Link Interface
H3	3

b) Ne, zato što pravilo prosleđivanja je isključivo zasnovano na odredišnoj adresi.

c) *Ovo nemam pojma.*

d) Ni ovo.

P10. *Zamislimo mrežu sa datagramima u kojoj se koristi 32-bitno adresiranje (32-bit host addresses). Pretpostavimo da ruter ima četiri linka, označena od 0 do 3, a da bi pakete trebalo proslediti interfejsima linkova na sledeći način:*

Raspon odredišnih adresa	Interfejs linka
11100000 00000000 00000000 00000000 do 11100000 00111111 11111111 11111111	0
11100000 01000000 00000000 00000000 do 11100000 01000000 11111111 11111111	1
11100000 01000001 00000000 00000000 do 11100001 01111111 11111111 11111111	2
Ostalo	3

a) Napravite tabelu prosleđivanja koja ima četiri stavke, koristi preklapanje najdužeg prefiksa i prosleđuje pakete na odgovarajuće interfejse linkova.

b) Opišite kako bi vaša tabela prosleđivanja odredila odgovarajući interfejs linka za datagrame sa sledećim odredišnim adresama:
11001000 10010001 01010001 01010101
11100001 01000000 11000011 00111100
11100001 10000000 00010001 01110111

a) *(Ovde gledamo koji bitovi od IP adrese su isti kao do bitovi IP adrese)*

Poklapajući prefiks	Interfejs linka
11100000 00	0
11100000 01000000	1
1110000	2
Ostalo	3

b) Interfejs 3 (ostalo, jer se ništa ne poklapa), interfejs 2 (1110000 se poklapa), interfejs 2.
** Da smo imali 11100001 1 - 3 kao dodatnu stavku u tabeli, onda bi treća adresa išla na interfejs 3 jer je preciznije.*

P13. *Zamislimo ruter koji povezuje tri podmreže: podmreža 1, podmreža 2 i podmreža 3. Pretpostavimo da se zahteva da svi interfejsi, u sve tri podmreže, imaju prefiks 223.1.17/24. Takođe pretpostavimo da se zahteva da podmreža 1 podržava minimalno 60 interfejsa, podmreža 2 minimalno 90 interfejsa, a podmreža 3 najmanje 12 interfejsa. Odredite tri mrežne adrese (oblika a.b.c.d/x) koje zadovoljavaju postavljene zahteve.*

- PC1 - 60, PC2 - 90, PC3 - 12. Prva koristiva adresa je 223.1.17.0, poslednja 223.1.17.255.
223.1.17.0/26
223.1.17.128/25
223.1.17.192/28

P15. *U problemu P10 traženo je da napravite tabelu prosleđivanja (koristeći preklapanje najdužeg prefiksa). Prepišite tu tabelu prosleđivanja u obliku a.b.c.d/x umesto u obliku binarnih nizova.*

Poklapajući prefiks	Interfejs linka
224.0/10	0
224.64/16	1
224/8	2
Ostalo	3

P17. *Uzmimo na primer topologiju prikazanu na slici 4.17. Označimo tri podmreže sa računarima (počevši u smeru kazaljke na satu od 12:00) kao mreže A, B i C. Označite podmreže bez računara, kao mreže D, E i F.*

a) Dodelite mrežne adrese svakoj od ovih šest podmreža uz sledeća ograničenja:

1. *Sve adrese bi trebalo dodeliti iz opsega 214.97.254/23.*
2. *Podmreža A bi trebalo da ima dovoljno adresa da podrži 250 interfejsa.*
3. *Podmreža B bi trebalo da ima dovoljno adresa da podrži 120 interfejsa.*
4. *Podmreža C bi trebalo da ima dovoljno adresa da podrži 120 interfejsa.*
5. *Podmreže D, E i F moraju biti u stanju da podrže po dva interfejsa.*

- Svaku mrežu adrese treba dodeliti u obliku a.b.c.d/x ili a.b.c.d/x - e.f.g.h/y.

b) *Na osnovu odgovora pod (a), napravite tabele prosleđivanja (koristeći preklapanje najdužeg prefiksa) za sva tri rutera.*

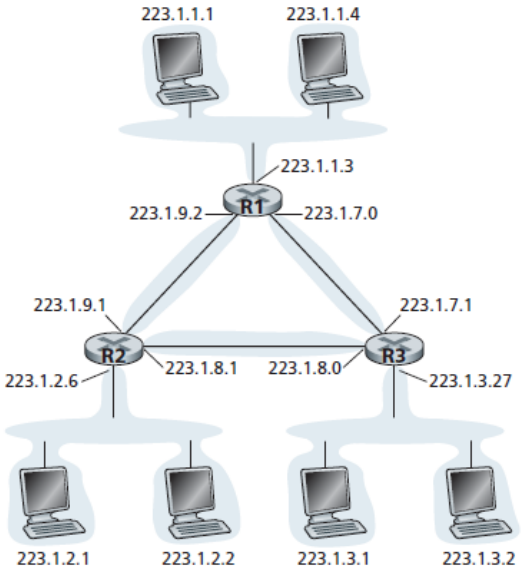


Figure 4.17 ♦ Three routers interconnecting six subnets

a)

Opseg mreže 214.97.254/23			Maska
Naziv podmreže	Network ID	Broadcast	Zaokružen br.
PMA - 250	214.97.254.0/24	214.97.254.255	256
PMB - 120	214.97.255.0/25	214.97.255.127	128
PMC - 120	214.97.255.128/25	214.97.255.255	128
PMD - 2	Za ove mreže nema mesta		4
PME - 2	Loša postavka zadatka		4
PMF - 2	/23 = 9 slobodnih bitova = 512 adr.		4

b) Jedan prefiks je 214.97.254/24, drugi 214.97.255.0/25, treći 214.97.255.128/25. Ne može lepo da se uradi jer postavka zadatka nije dobra, ali bi suština bila takva.

P20. *Pretpostavimo da su između izvornog računara A i odredišnog računara B datagrami ograničeni na 1 500 bajtova (uključujući zaglavlje). Ako pretpostavimo da je IP zaglavlje 20 bajtova, koliko bi datagrama bilo potrebno da se pošalje jedan MP3 od 5 miliona bajtova? Objasnite kako ste izračunali odgovor.*

- Veličina MP3 fajla = 5 miliona bajtova = $5 * 10^6$ bajtova.

Pretpostavljamo da se podaci prenose u TCP segmentima, gde svaki TCP segment ima zaglavlje od 20 bajtova. To znači da svaki datagram može da sadrži 1500 - 40 = 1460 bajtova MP3 fajla.

Broj neophodnih datagrama za prenos :

$$\frac{5 * 10^6 \text{ b}}{1460 \text{ b}} = \frac{5000 * 10^3 \text{ b}}{1460 \text{ b}} = 3424$$

BONUS :

Popuni priloženu tabelu. U kolonu 3 upiši redne brojeve uloga iz pomoćne tabele Tab3.

TCP/IP	1	2	3
Sloj aplikacije	Podaci	HTTP, FTP, DNS	1
Sloj prezentacije	Podaci	•SSL	2, 3
Sloj sesije	Podaci	•NetBIOS, PPTP	4
Transportni sloj	Segment	TCP	5, 6
Sloj mreže	Paket	IPv6	7
Sloj podataka	Okvir	•ARP, DHCP	8, 9, 10
Fizički sloj	Biti	•Fizička topologija	11, 12

Tab3

1	•Pruža korisnički interfejs
2	•Predstavlja podatke
3	•Vrši obradu kao što je šifrovanje
4	•Održava odvojenim podatke različitih aplikacija
5	•Pruža pouzdanu ili nepouzdanu isporuku
6	•Vrši ispravku grešaka pre novog prenosa
7	•Pruža logičko adresiranje, koje koriste ruteri za određivanje putanje
8	•Kombinuje pakete u bajtove i bajtove u okvire
9	•Pruža pristup mediju koristeći MAC adrese
10	•Vrši detekciju grešaka
11	•Premešta bitove između uređaja
12	•Određuje napon, brzinu prenosa podataka i način priključivanja kablova

Zadatak :

Za IP adresu 192.168.55.32/20 izračunati sledeće :

- Subnet masku, wildcard masku, IP adresu mreže, Broadcast adresu, opseg korisnih adresa

_____

** Subnet maska razdvaja network deo mreže (deo rezervisan za mreže) od host dela mreže (deo rezervisan za krajnje uređaje). Bitovi 1 u subnet masci predstavljaju network deo mreže. Ova subnet maska rezerviše 20 bitova (od 32) za network deo.*

Subnet maska : Uzimamo broj desno od / iz zapisa IP adrese. Toliko prvih bitova ima vrednost 1.
11111111.11111111.11110000.00000000
Subnet maska : 255.255.240.0

Wildcard maska : Izvrnuta subnet maska.
00000000.00000000.00001111.11111111
Wildcard maska : 0.0.15.255

IP adresa mreže : Svuda gde je subnet maska 0 u IP adresi ispunimo nulama.
** Drugim rečima, host deo ispunimo nulama.*
IP adresa iz zadatka : 11000000.10101000.0011**0111.00100000**
Subnet maska : 11111111.11111111.1111**0000.00000000**
11000000.10101000.0011*0000.00000000*
IP adresa mreže : 192.168.48.0

Broadcast adresa : Svuda gde je subnet maska 0 u IP adresi ispunimo jedinicama.
11000000.10101000.0011*1111.11111111*
Broadcast adresa : 192.168.63.255

Opseg korisnih adresa : $2^{\text{broj bitova u host delu (tj. svuda gde je subnet maska 0)}} - 2$
Opseg korisnih adresa : $2^{12} - 2 = 4096 - 2 = 4094$
** - 2 se koristi zato što krajnjim uređajima (računarima, smartphone-ovima koji žele da se povežu na mrežu i sl.) se ne mogu dodeliti broadcast adresa niti IP adresa mreže (takođe znana kao Network ID).*

Zadatak :

***Odrediti klasu adrese, broj mreže, broj mogućih podmreža i broj mogućih hostova*.**

IP adresa : 96.96.11.78

Subnet maska : 255.255.252.0

_____

- Subnet maska razdvaja mrežni deo IP adrese (deo rezervisan za mreže) od host dela (deo rezervisan za krajnje uređaje). Tamo gde je bit 1 kod maske predstavlja network deo.

Pr. klasa A - 11111111.00000000.00000000.00000000. - Prvi oktet je rezervisan za mreže.

- IP adrese klase A (0 - 126 prvi oktet) imaju default subnet masku od 255.0.0.0. tj. /8
- IP adrese klase B (128 - 191 prvi oktet) imaju default subnet masku od 255.255.0.0 tj. /16
- IP adrese klase C (192 - 223 prvi oktet) imaju default subnet masku od 255.255.255.0 tj. /24

01100000.01100000.00001011.01001110

Network. Host . Host . Host

11111111.00000000.00000000.00000000

11111111.11111111.11111100.00000000

Klasa A	Network	*Host*	*Host*	*Host*
	1-126	0-255	0-255	0-255
Klasa B	Network	Network	*Host*	*Host*
	128-191	0-255	0-255	0-255
Klasa C	Network	Network	Network	*Host*
	192-223	0-255	0-255	0-255

IP Adresa

Klasa A subnet maska - Network oktet - Host okteti

Data subnet maska u zadatku

Gledamo samo deo date subnet maske koji je obuhvaćen host oktetima default subnet maske zadatate IP adrese.

11111111.00000000.00000000.00000000

11111111.11111111.11111100.00000000

Broj mogućih podmreža : $2^{\text{broj 1 u maski}} - 2$

Broj mogućih hostova : $2^{\text{broj 0 u maski}} - 2$

Rešenje :

Address Class: _____**A**_____

Network Number: _____**96.0.0.0**_____

Possible # of Subnets: _____**$2^{14} - 2$**_____

Possible # of Hosts:_____**$2^{10} - 2$**_____

- Dakle, da je adresa pripadala klasi B (npr. adresa 165.217.11.43), gledali bismo samo poslednja dva okteta date subnet maske i u njima brojali broj jedinica za broj mogućih podmreža i broj nula za broj mogućih hostova.

** Za network number uzmemo IP adresu i u njoj ispunimo nulama sve što se poklapa sa host delom klasne IP maske.*

Primeri za vežbu :

IP adresa : 107.21.54.100	| 171.91.51.1	|195.91.5.1
Subnet maska : 255.255.255.224	| 255.255.252.0	|255.255.255.240

Zadatak :

Odrediti network broj podmreže i njenu broadcast adresu, kao i broj mogućih podmreža i hostova.

IP adresa : 201.19.100.140

Subnet maska : 255.255.255.224

_____

IP adresa : **11001001.00010011.01100100.100**01100

Subnet maska : **11111111.11111111.11111111.111**00000

Crveni deo IP adrese je nepromenjiv. *Tamo gde su bitovi subnet maske 1 znači da ti bitovi ostaju isti u IP adresi.*

Broj podmreže *(tj. Subnet number)* : Podesiti sve bitove u IP adresi koji se poklapaju sa **0** u subnet maski na **0**.

Broadcast adresa : Podesiti sve bitove u IP adresi koji se poklapaju sa **0** u subnet maski na **1**.

Subnet number *(tj. Network ID)* : 11001001.00010011.01100100.100**00000** -- 201.19.100.128

Broadcast address : 11001001.00010011.01100100.100**11111** -- 201.19.100.159

Broj mogućih podmreža : $2^{\text{broj 1 u maski kod delova date maske koji se poklapaju sa host delom klasne maske}}$ - 2

Broj mogućih hostova *(tj. krajnjih uređaja)* : $2^{\text{broj 0 u maski kod delova date maske koji se poklapaju sa host delom klasne maske}}$ - 2

- IP adresa 201.19.100.140 pripada klasi **C**, jer prvi oktet pripada opsegu 192 - 223.

Maska klase C : 255.255.255.0 - Tamo gde je 1 je **network deo**, tamo gde je 0 je **host deo**.

Data subnet maska u zadatku : 255.255.255.224

11111111.11111111.11111111.**00000000**

11111111.11111111.11111111.11100000

Alternativno : *Ako naučimo sve podmaske klasa, znamo da je maska klase C /24, tj. da ima 24 bitova. U ovom zadatku subnet maska je /27 (vidi crveno ofarban deo pri vrhu, 27 jedinica). 27 - 24 = 3 => 2^3 - 2 je broj mogućih podmreža.*

Rešenje :

Subnet Number: ____**201.19.100.128** _____

Broadcast Address: ____**201.19.100.159** _____

Possible # of Subnets: ____**2^3 - 2** _____ ***(6)***

Possible # of Hosts: ____**2^5 - 2** _____ ***(30)***

Primeri za vežbu :

IP adresa : 75.60.11.5	190.101.2.140	182.191.25.11
Subnet maska : 255.255.255.0	255.255.255.192	255.255.255.224

Zadatak :

IP adresa hosta je 195.5.6.17, a mrežna maska 255.255.255.240. Odrediti:

a) adresu mreže kojoj host pripada

195.5.6.17 = 11000011.00000101.00000110.0001**0001**

M. maska = 11111111.11111111.11111111.1111**0000**

Host : 11000011.00000101.00000110.0001**0000**, *tj.* **195.5.6.16**

b) mrežni prefiks

Mrežna maska ima 28 jedinica, tako da je prefiks /28.

c) broadcast adresu

195.5.6.17 = 11000011.00000101.00000110.0001**0001**

M. maska = 11111111.11111111.11111111.1111**0000**

Broadcast : 11000011.00000101.00000110.0001**1111**, *tj.* **195.5.6.31**

d) maksimalan broj hostova koji se mogu adresirati u toj mreži

$2^{\text{broj bitova u host delu maske (boldovani deo)}} - 2 = 2^4 - 2 = 16 - 2 =$ **14**

e) raspon adresa koje se mogu dodeliti hostovima

- Sve koje su između adrese mreže (network ID) i broadcast adrese tj. (network ID + 1) - (broadcast - 1)

195.5.6.17 - 195.5.6.30

Opcionalno, ako je previše komplikovano preskoči

Zadatak 16.

Za IP adresu 192.168.214.0/29 odrediti:
1) broj bita koji određuje podmrežu (u odnosu na difolt masku podmreže),
2) broj mogucih IP adresa u svakoj podmreži,
3) masku podmreže, i
4) opseg IP adresa za prvu i poslednju podmrežu.

Rešenje:
Za IP adresu 192.168.214.0/29 :
1) broj bita koji određuje podmrežu (u odnosu na difolt masku podmreže) je 5 (difolt mreža je klasa C, tj. 192.168.214.0/24);
2) broj mogucih IP adresa u svakoj podmreži je 6 (2^3-2=6, preostaje 3 bita za adrese hostova), ili 254-248=6;
3) maska podmreže je 255.255.255.248;
4) IP adrese (host-ova) u prvoj podmreži su 192.168.214.9 do 192.168.214.14, a IP adrese u poslednjoj podmreži su 192.168.214.249 do 10.255.214.254.

Zadatak :

IP adresa hosta je 190.58.0.0/21. Odrediti:

a) mrežnu masku

/21 znači da prvih 21 bitova ima vrednost 1

11111111.11111111.11111000.00000000 = **255.255.248.0**

b) broadcast adresu

Svuda gde mrežna maska ima vrednost 0 ubaci 1 u IP adresi

190.58. 00000**000.00000000**

255.255. 11111**000.00000000**

Broadcast adresa : 190.58. 00000111.11111111 = **190.58.7.255**

c) maksimalan broj hostova koji se mogu adresirati u toj mreži

$2^{\text{broj bitova koji su 0 u mrežnoj masci (tj. host deo maske)}} - 2 = 2^{11} - 2 = 2048 - 2 =$ **2046**

d) raspon adresa koje se mogu dodeliti hostovima

Sve između network ID-a (IP adresa svuda 0 gde je mrežna maska 0) i broadcast adrese (svuda 1)

*Network ID : 190.58. 00000**000.00000000** = 190.58.0.0*

190.58.0.1 - 190.58.7.254

Vezano za zadatak na narednoj strani, pogledati tek na kraju zadatka da bi razumeli

* Prva IP podmreža po redosledu se ne može koristiti
– 156.78.0.0 *(subnetid*=00000)
* Prva IP podmreža koja se može koristiti
– 156.78.8.0 *(subnetid*=00001)
* Zadnja IP podmreža koja se može koristiti
– 156.78.240.0 *(subnetid*=11110)
* Poslednja IP podmreža po redosledu se ne može koristiti
– 156.78.248.0 *(subnetid*=11111)

U prvoj IP korišćenoj IP podmreži (156.78.8.0)
– Prva IP adresa po redosledu je adresa IP podmreže i ne može se dodeliti računaru
* 156.78.8.0 *(hostid*=00000000000)
– Prva IP adresa koja se može dodeliti računaru je druga adresa po redosledu
* 156.78.8.1*(hostid*=00000000001)

* U poslednjoj korišćenoj IP podmreži (156.78.240.0)
– Poslednja IP adresa koja se može dodeliti računaru je pretposlednja adresa po redosledu
* 156.78.247.254 *(hostid*=11111111110)
– Poslednja IP adresa po redosledu je broadcast adresa i ne može se dodeliti računaru
* 156.78.247.255 *(hostid*= 11111111111)

Zadatak :

Zadata je IP mreža klase B 156.78.0.0

Napraviti podmrežavanje tako da postoji najmanje 20 podmreža, a da svaka može da ima najmanje 1000 računara.

Klasa B : maska je **/16**, što znači da network deo adrese (netid) je 16 bita, a host deo 16 bita.
- 156.78 ne diramo, to uvek ostaje isto.

- Najmanje 20 podmreža : treba nam 2^x da bude veće ili jednako broju 20.
$2^4 = 16$, 16 je manje od 20 i ne može; $2^5 = 32 \Rightarrow 32 > 20$, može.
Pošto je eksponent broj **5**, to znači da najmanje toliko bitova uzimamo za network deo podmreža (*subnetid*)

- Najmanje 1000 računara : isto kao i iznad, treba nam 2^x da bude veće ili jednako broju 1000.
$2^9 = 512$, 512 je manje od 1000 i ne može; $2^{10} = 1024 > 1000$, može.
Pošto je eksponent broj **10**, to znači da najmanje toliko bitova uzimamo za host deo podmreža (*hostid*)

Formula : subnetid + hostid mora biti jednak (32 - maska klase date IP adrese)
U ovom slučaju subnetid + hostid = 16 (32 - 16)
- subnetid mora biti najmanje 5, hostid mora biti najmanje 10.
- Moguće su nam kombinacije 6 + 10 i 5 + 11.
** Uzimamo varijantu gde je subnetid = 5, a hostid = 11*
Mrežna maska naših podmreža : klasna maska + subnetid = 16 + 5 = /21 tj. 255.255.248.0

Broj koristivih podmreža : razlika između $2^{\text{prefiks naše maske}}$ i $2^{\text{prefiks klasne maske}}$ - 2.
$2^{21} - 2^{16} - 2 = 2^5 - 2 = 32 - 2 =$ **30 podmreža koje možemo koristiti**.
** Prefiks samo znači da npr. pišemo /21 umesto binarnog zapisa i sl.*

Broj koristivih IP adresa u svakoj od podmreža : $2^{32} - 2^{\text{prefiks naše maske}} - 2$
$2^{32} - 2^{21} - 2 = 2^{11} - 2 = 2048 - 2 =$ **2046 koristivih IP adresa u svakoj podmreži.**
Novo, mnogo brže rešenje : $2^{\text{subnetid}} - 2 = 2^5 - 2 = 32 - 2 =$ **30 podmreža ...** $2^{\text{hostid}} - 2 = 2048 - 2 =$ **2046 adr.**
Alternativno :
Poredimo našu mrežnu masku sa maskom klase.
maska kl. B = 11111111.11111111.00000000.00000000
/21 maska = 11111111.11111111.11111000.00000000
Broj koristivih podmreža : $2^{\text{tamo gde je 1 u našoj maski i 0 u maski klase}} - 2$: $2^5 - 2 = 32 - 2 = 30$
Broj koristivih adresa u svakoj podmreži : $2^{\text{tamo gde je 0 u našoj maski i 0 u maski klase}} - 2$: $2^{11} - 2 = 2048 - 2 = 2046$

Opseg koristivih podmreža *(sve osim prve (sve plavo tj. subnetid 0) i poslednje (sve plavo tj. subnetid 1) zbog kojih smo i radili - 2)*:
Prva IP podmreža koja ne može da se koristi : 11111111.11111111.00000000.00000000 - **156.78.0.0**
Prva IP podmreža koja može da se koristi : 11111111.11111111.00001000.00000000 - **156.78.8.0**
Posl. IP podmreža koja može da se koristi : 11111111.11111111.11110000.00000000 - **156.78.240.0**
Posl. IP podmreža koja ne može da se koristi : 11111111.11111111.11111000.00000000 - **156.78.248.0**
- U prvoj koristivoj podmreži (156.78.8.0) opseg je između 00001000.00000001 i 00001111.11111110.
- U posl. kor. podmr. (156.78.240.0) broadcast adr. je 11111111.11111111.11110111.11111111 - 156.78.247.255

Zadatak :

ABC kompanija poseduje adresu klase B, 172.16.0.0. Kompanija treba da napravi šemu za podmrežavanje da obezbedi sledeće : 36 subneta sa barem 100 hostova, 24 subneta sa barem 255 hostova, 10 subneta sa barem 50 hostova

_____

a) Koliko je podmreža potrebno za ovu mrežu?

36 + 24 + 10 = **70 podmreža**.

b) Koji je minimalan broj bita koji može biti pozamljen?

2^7 = 128 > 70 : **7 bita.**

c) Koja je subnet maska za ovu mrežu?

Maska klase B : 11111111.11111111.00000000.00000000

Naša maska : 11111111.11111111.1111111**0**.00000000

Subnet maska : 255.255.254.0

d) Koliko postoji korisnih podmreža?

2^7 - 2 = **126 podmreža**.

e) Koliko ima korisnih hostova po subnetu?

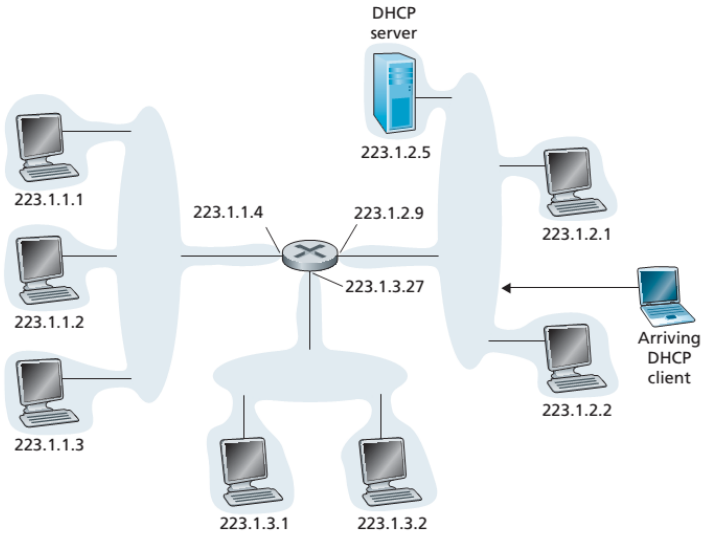
2^9 - 2 = **510 hostova.**

Računarske komunikacije - Dodatak za ispit 2017

DHCP - Protokol za dinamičko konfigurisanje računara

DHCP je protokol između klijenta i servera. Klijenti su obično novi računari u mreži koji žele da dobiju informacije o konfiguraciji mreže i svoju IP adresu. Obično svaka podmreža ima svoj DHCP server, ali ukoliko on nije prisutan onda je potreban agent za prenos DHCP (obično ruter) koji zna adresu DHCP servera za tu mrežu.

Na slici je prikazan DHCP server koji je u podmreži 223.1.2/24, i ruter koji služi kao DHCP agent uređajima iz mreža 223.1.1/24 i 223.1.3/24.



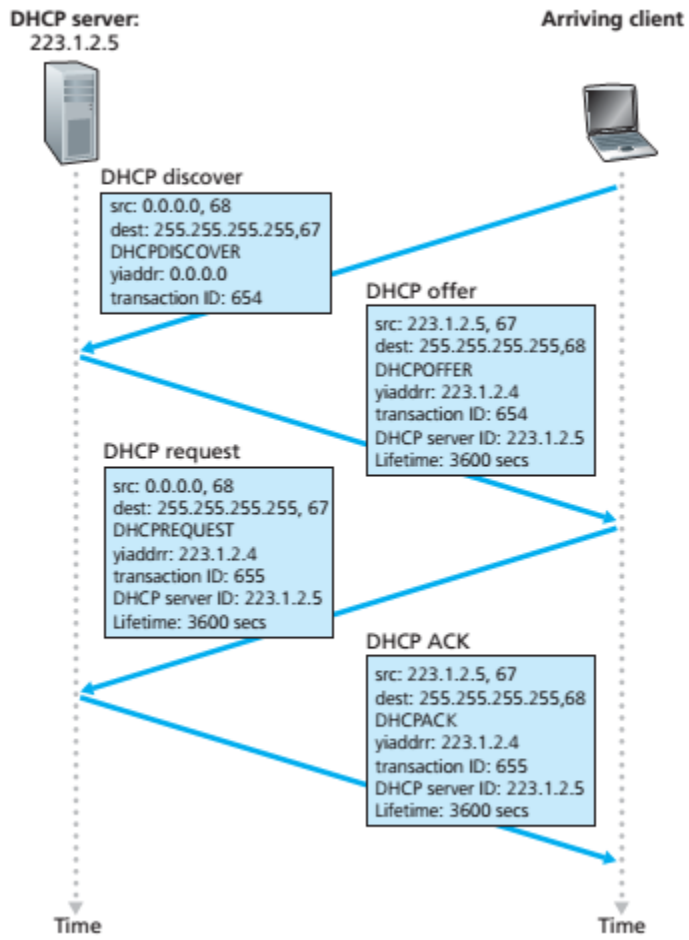
Protkol DHCP se odvija u četiri koraka :

1. **Otkrivanje DHCP-a** - Novopridošli računar prvo mora da pronađe DHCP server sa kojim treba da komunicira. To se radi upotrebom **DHCP poruke za otkrivanje**, koju klijent šalje unutar UDP paketa na portu 67. DHCP klijent enkapsulira tu poruku u IP datagram uz broadcast odredišnu IP adresu 255.255.255.255. i izvornu IP adresu 0.0.0.0. DHCP klijent potom prosleđuje taj datagram svom adapteru koji ga enkapsulira u okvir sloja veze (*frame*) koji u odredišnoj adresi ima broadcast MAC adresu (FF-FF-FF-FF-FF-FF). DHCP klijent potom šalje taj okvir (koji sadrži poruku za otkrivanje) u mrežu, njega primaju svi adapteri u mreži. Ukoliko je DHCP server unutar iste podmreže on će obraditi tu poruku; a ako je DHCP agent priključen onda će on proslediti okvir mreži sa DHCP serverom (taj preneseni okvir će imati različitu izvornu MAC adresu). Poruka za otkrivanje sadrži **ID transakcije** koji omogućava odgovorima da budu usklađeni sa zahetvom za otkrivanje.

2. **Ponuda DHCP servera** - DHCP server koji prima DHCP poruku za otkrivanje odgovara klijentu **DHCP poruku za ponudu**. U mreži je moguće da bude prisutno više DHCP servera, te je moguće da klijent bude u stanju da bira između više ponuda. Svaka poruka servera za ponudu sadrži **ID transakcije** primljene poruke za otkrivanje, predloženu IP adresu (prvo radi ARP sa ovom adresom da vidi da li je slobodna), mrežnu masku i **vreme zakupa IP adrese** - vreme za koje će IP adresa biti važeća. Okvir sloja veze koji sadrži IP datagram koji sadrži UDP segment koji sadrži DHCP poruku za ponudu se onda šalje klijentu.

3. **DHCP zahtev** - Novoprispeli klijent će izabrati između jedne ili više ponuda servera i odgovoriće na izabranu ponudu **DHCP porukom za zahtev**, vraćajući nazad parametre konfiguracije.

4. **DHCP ACK** - Server odgovara na DHCP poruku za zahtevom porukom **DHCP ACK**, potvrđujući zahtevane parametre. Kada klijent primi DHCP ACK, interakcija je upotpunjena i klijent može da koristi IP adresu dodeljenu od DHCP servera za dato vreme zakupa. Postoje i mehanizmi za produžavanje vremena zakupa IP adrese.



Fragmentacija IP datagrama

Ne mogu svi protokoli sloja veze da prenose pakete iste veličine, npr. Ethernet paketi ne mogu da prenesu više od 1500 bajtova podataka. Maksimalna količina podataka koju može da prenosi paket sloja veze naziva se maksimalna jedinica za transfer (**maximum transfer unit**, **MTU**). Problem nastaje kada linkovi na ruti između pošaljioca i odredišta možda koriste različite protokole sloja veze i svaki od tih protokola može da ima drugačiji MTU.

Pretpostavimo da imamo ruter koji povezuje nekoliko linkova od kojih svaki izvršava drugačiji protokol sloja veze sa različitim MTU-ovima. Ruter prima IP datagram jednog linka, proverava u svojoj tabeli prosleđivanja koji je izlazni link i utvrđuje da taj izlazni link ima manju MTU od dužine primljenog IP datagrama. Kako stisnuti veliki IP paket u polje korisnih podataka sloja veze? Rešenje ovog problema je da se podaci iz IP datagrama **fragmentiraju** na više manjih IP datagrama, a zatim se ti manji datagrami šalju preko izlaznog linka. Svaki od ovih manjih datagrama naziva se **fragment.**

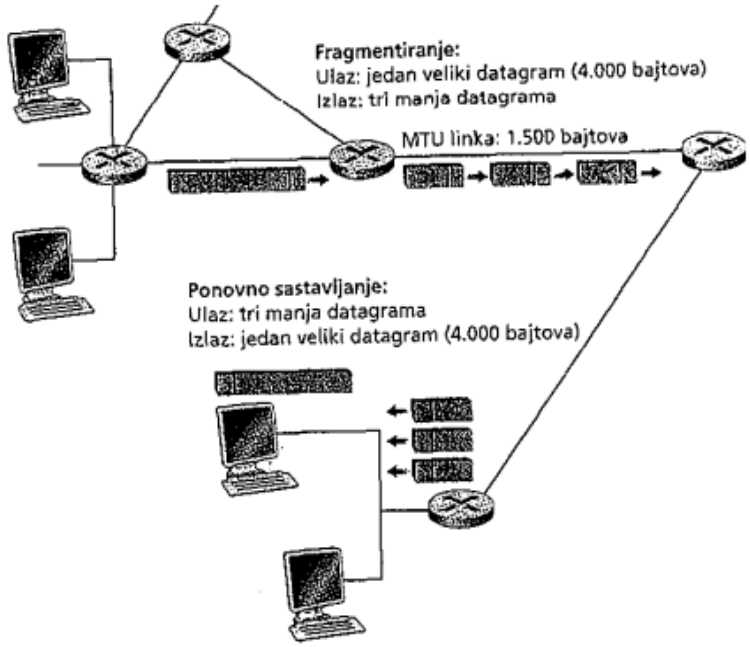
Fragmenti moraju ponovo da se sastave pre nego što se predaju transportnom sloju na odredištu. Zadatak ponovnog sastavljanja obvaljaju krajnji sistemi (a ne ruteri). Kada odredišni računar primi niz datagrama sa istog izvora, on prvo utvrđuje da li oni pripadaju nekom većem datagramu. Ako utvrdi da su neki datagrami u stvari fragmenti, on mora da utvrdi kada je primio poslednji fragment i kako bi primljene fragmente trebalo ponovo sastaviti da bi se dobio prvobitni datagram. Projektanti IPv4 protokola stavili su u IP datagram **polja identifikacija, oznake i ofseta fragmentacije**. Kada se datagram pravi, izvorni računar mu osim izvorne i odredišne adrese dodeljuje **identifikacioni broj**, taj broj on povećava za svaki poslati datagram. Kada ruter mora da fragmentira datagram, svaki novi datagram (tj. fragment) označava se izvornom adresom, odredišnom adresom, i **identifikacionim brojem** prvobitnog datagrama.

Kada odredište primi niz datagrama sa istog izvornog računara, on može da ispita **identifikacione brojeve** datagrama i utvrdi koji su datagrami u stvari fragmenti jednog istog većeg datagrama. Pošto je IP nepouzdana usluga, jedan ili više fragmenata možda neće stići na odredište. Zbog toga, da bi odredišni računar bio siguran da je primio poslednji fragment prvobitnog datagrama, poslednji fragment ima u **bitu oznake vrednosti 0**, dok ostali fragmenti imaju vrednost **1**. Osim toga, da bi odr. računar mogao da utvrdi da li neki fragment nedostaje (i takođe da bi znao da ih sastavi u pravilnom redosledu), on koristi **polje ofseta** da odredi lokaciju fragmenta u prvobitnom IP datagramu.

Datagram od 4000 bajtova (20 bajtova IP zaglavlja + 3980 bajtova IP korisnih podataka) stiže na ruter i mora da se prosledi na link čiji MTU iznosi 1500 bajtova. To znači da 3980 bajtova podataka iz prvobitnog datagrama treba podeliti u tri fragmenata (koji su svi takođe datagrami). Pretpostaviti da prvobitni datagram ima **identifikacioni broj 777**. Broj korisnih bajtova osim poslednjeg mora biti deljiv sa 8 i vrednosti ofseta se navode u jedinicama od po 8 bajtova.

Korisni podaci iz datagrama se predaju transportnom sloju na odredištu tek pošto mrežni (IP) sloj potpuno rekonstruiše prvobitni datagram. Ako jedan ili više fragmenata ne stigne do odredišta, datagram se odbacuje i ne predaje se transportnom sloju.

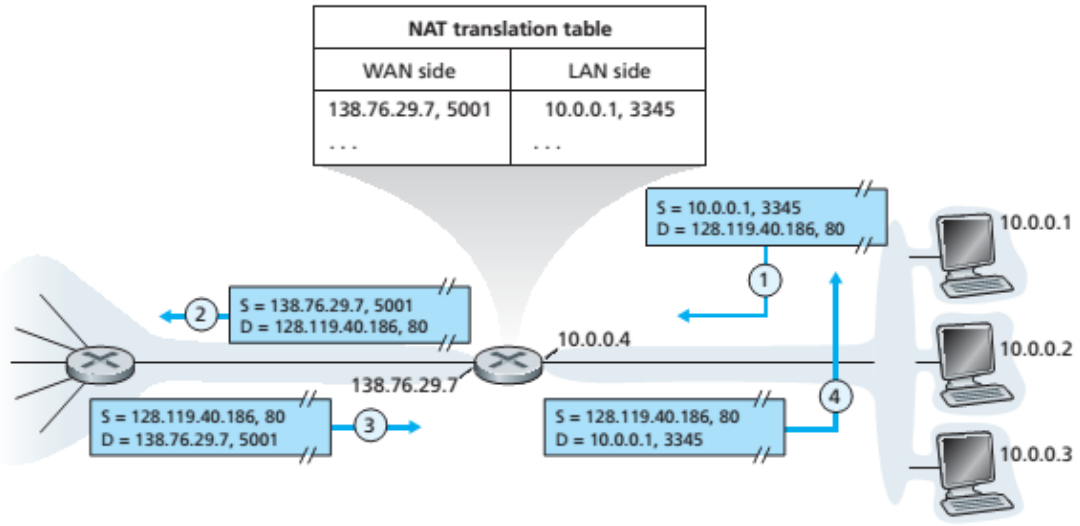
Fragment	Veličina podataka	ID	Ofset	Oznaka
Prvi	1.480 bajtova	777	0 (znači da podatke treba umetnuti od pozicije 0)	1 (znači da ima još)
Drugi	1.480 bajtova	777	185 (znači da podatke treba umetnuti počev od pozicije 185-8=1480)	1 (znači da ima još)
Treći	1.020 bajtova (3980 –1480 – 1480)	777	380 (znači da podatke treba umetnuti počev od pozicije 380-8=2960)	0 (znači da je poslednji)



NAT - Prevodioci mrežnih adresa

Kako ne bi bili u obavezi da kućnoj mreži sa 30 računara ISP obezbedi 30 različitih IP adresa, napravljen je sistem gde imamo privatne i javnu IP adresu. Kada se paketi šalju iz naše mreže, oni na Internetu izgledaju kao da je svima njima ista izvorna adresa (a to je javna adresa), tj. kao da je u pitanju jedan uređaj. U suštini NAT ruter krije detalje kućne mreže od spoljnog sveta. NAT ruter u sebi ima tabelu prevođenja. Kada mu iz kućne mreže stigne paket, on zapamti izvornu adresu tog paketa i port koji se koristio i ubacuje ga u svoju tabelu, a potom taj paket nastavlja dalje sa javnom IP adresom i sa nekim nasumičnim (novim) portom koji mu je NAT ruter dodelio. Na ovaj način, kada NAT ruteru stigne odgovor od tog nekog uređaja kome je poslat paket, taj uređaj će kao odredišni port staviti port koji je NAT ruter dodelio, i samim tim NAT ruter će znati kojem uređaju unutar kućne mreže je pristigli paket namenjen.

NAT ruter za spoljni svet ne izgleda kao ruter. Umesto toga, NAT ruter se prema spoljnom svetu ponaša kao jedan uređaj sa jednom IP adresom. Ruter dobija svoju adresu od DHCP servera kod posrednika Internet usluga, a isti kućni ruter izvršava DHCP server koji obezbeđuje adrese za računare u adresnom prostoru kućne mreže pod kontrolom NAT DHCP rutera.



Računar 10.0.0.1 u kućnoj mreži zahteva veb stranicu od nekog veb servera (port 80) sa IP adresom 128.119.40.186. Računar 10.0.0.1 paketu dodeljuje (proizvoljan) broj izvornog porta 3345 i šalje datagram u LAN. NAT ruter prima datagram, pravi novi broj izvornog porta 5001 za taj datagram, zamenjuje izvornu IP adresu svojom IP adresom za WAN 138.76.29.7 i zamenjuje prvobitni broj izvornog porta 3345 sa 5001 (može da izabere bilo koji port koji nije trenutno u tabeli prevođenja), i ubacuje tu stavku u svoju tabelu.

Veb server ni ne zna da je pristigli datagram koji sadrži HTTP zahtev prepravljen u NAT ruteru i odgovara datagramom čija je odredišna IP adresa NAT rutera, a odredišni port 5001. Kada datagram stigne u NAT ruter, on gleda u svoju tabelu i pronalazi odgovarajuću IP adresu (10.0.0.1) i odgovarajući broj porta (3345) i prosleđuje.

Prosleđivanje i rutiranje

Uloga mrežnog sloja je jednostavna - prenošenje paketa od izvornog do odredišnog računara. U tom poslu mogu se uočiti dve značajne funkcije ovog sloja :

Prosleđivanje - Kada paket stigne na ruterov ulazni link, on mora da ga premesti na odgovarajući izlazni link. Na primer, paket koji od računara H1 stigne u ruter R1 mora da se prosledi sledećem ruteru na putanji prema H2.

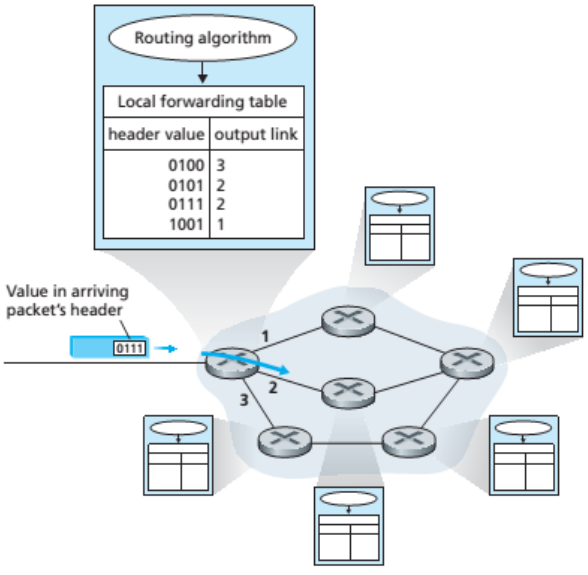
Rutiranje - Mrežni sloj mora da utvrdi rutu/putanju kojom paketi idu od izvora do oredišta. Algoritmi rutiranja definišu te putanje.

Prosleđivanje znači lokalnu aktivnost rutera prilikom prenosa datagrama sa interfejsa ulaznog linka u odgovarajući interfejs izlaznog linka. Rutiranje znači sveukupni proces određivanje putanje sa kraja na kraj kroz celu mrežu kojom će datagrami ići, od izvora do odredišta.

Svaki ruter ima tabelu prosleđivanja. Ruter prosleđuje paket tako što ispituje vrednost jednog polja u zaglavlju pristiglog paketa i tu vrednost koristi kao indeks za svoju tabelu. Rezultat dobijen iz tabele ukazuje na interfejs ruterovog linka na koji treba proslediti paket.

U ruter stiže paket koji u polju zaglavlja ima vrednost 0111. Ruter koristi tu vrednost kao indeks za tabelu prosleđivanja i utvrđuje da je interfejs izlaznog linka za ovaj paket interfejs 2. Ruter tada interno prosleduje paket interfejsu 2. Tabele prosleđivanja se konfigurišu na osnovu algoritama rutiranja.

Algoritam rutiranja može da bude centralizovan (kada se algoritam izvršava na jednoj centralnoj lokaciji a informacije za rutiranje se preuzimaju na svim ruterima) ili decentralizovan (kada se na svakom ruteru izvršava deo distribuiranog algoritma rutiranja). U oba slučaja, ruter prima poruke protokola rutiranja koje se koriste za konfigurisanje tabele prosleđivanja.



ARP - Address Resolution Protocol

ARP je protokol koji služi za pretvaranje logičkih (IP) adresa u fizičke (MAC). Host (čvor) u mreži koji želi dobiti neku fizičku adresu broadcast-uje ARP zahtev u mrežu. Čvor na mreži koji ima adresu iz zahteva u odgovoru šalje svoju fizičku adresu. Drugim rečima, ARP omogućuje nekom računaru da pronađe fizičku adresu ciljnog računara unutar iste mreže ako je data samo IP adresa ciljnog računara.

Postoji i inverzni ARP (Reverse ARP - RARP) koji se koristi za otkrivanje sopstvene IP adrese. Host šalje u mrežu (broadcast) svoju fizičku adresu, a RARP server mu u odgovoru šalje njegovu IP adresu.

ARP klijent i server procesi se izvršavaju na svakom računaru koji koristi IP protokol unutar mreže. Ovo znači da svaki računar može biti i klijent i server, u zavisnosti od situacije.

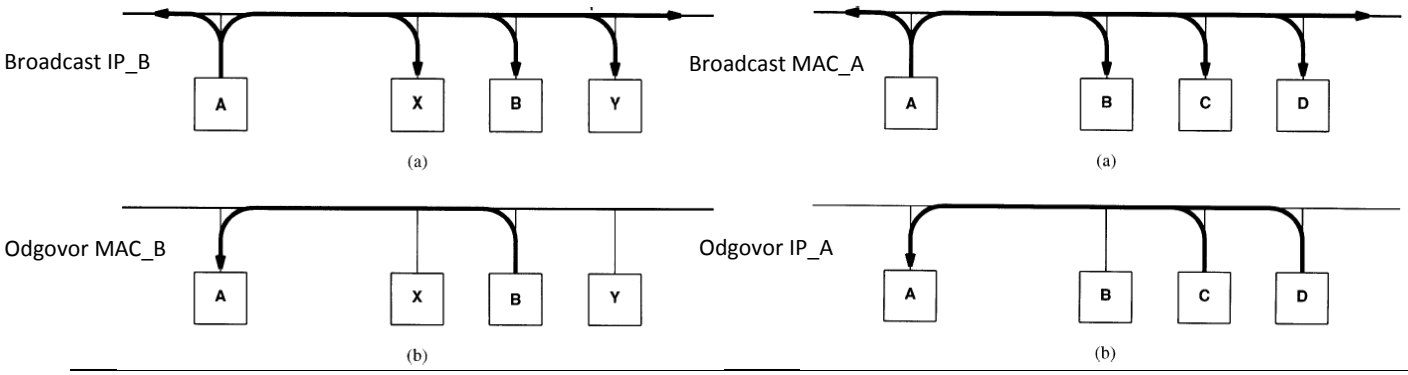
Da bi se smanjio broj zahteva za određivanje fizičke adrese, "klijent" proces ubacuje već razrešene adrese u memoriju koja se naziva ARP keš (ARP cache).

Postoje četiri tipa poruka koje se šalju ARP protokolom: ARP request / reply, RARP request / reply.

Poruka sa ARP zahtevom (*"Y.Y.Y.Y pita: ko je X.X.X.X ?",* gde su X.X.X.X i Y.Y.Y.Y IP adrese) šalje se kao broadcast. Pošto su broadcast primili svi sistemi u mreži, ciljani sistem odgovara a ostali bez obaveštenja ignorišu ovaj ARP zahtev.

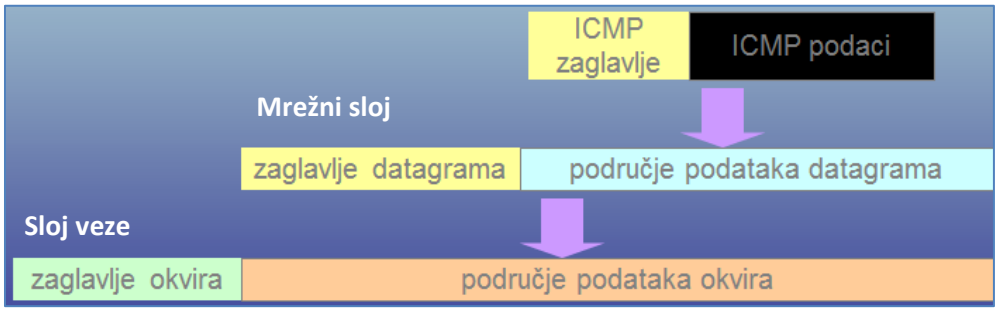
Sistem na kojeg se odnosi poruka kreira ARP odgovor (*"X.X.X.X je hh:hh:hh:hh:hh:hh"*, gde je hh:hh:hh:hh:hh:hh fizička adresa računara čija je IP adresa X.X.X.X). Ovaj paket se šalje direktno na adresu računara koji je uputio zahtev (u ovom slučaju Y.Y.Y.Y). Po prijemu paketa sistem sa IP adresom Y.Y.Y.Y u svoj ARP keš upisuje utvrđenu IP i fizičku adresu, a isto tako je ARP server upisao podatke računara koji je poslao zahtev u svoj keš kako bi se smanjilo optrećenje na mreži u budućnosti.

Primeri ARP-a (levo) i RARP-a (desno):



ICMP - Internet Control Message Protocol

IP protokol služi za prenos krajnjih podataka. Ostali protokoli mrežnog sloja su ARP, RARP, DHCP, BOOTP, ICMP. **ICMP je protokol za upravljanje porukama na Internetu** - služi za izveštavanje o neočekivanim događajima (greškama) ili za testiranje Interneta (stanje IP mreže). Ruteri ga koriste da bi poslali obaveštenje o problemima, a računari da bi proverili da li može da se stigne do odredišta. ICMP poruke su sastavni deo IP datagrama. ICMP o greškama izveštava izvor datagrama, a ne posredne rutere.



Ispod ovoga ide fizički sloj kojim se šalju podaci.

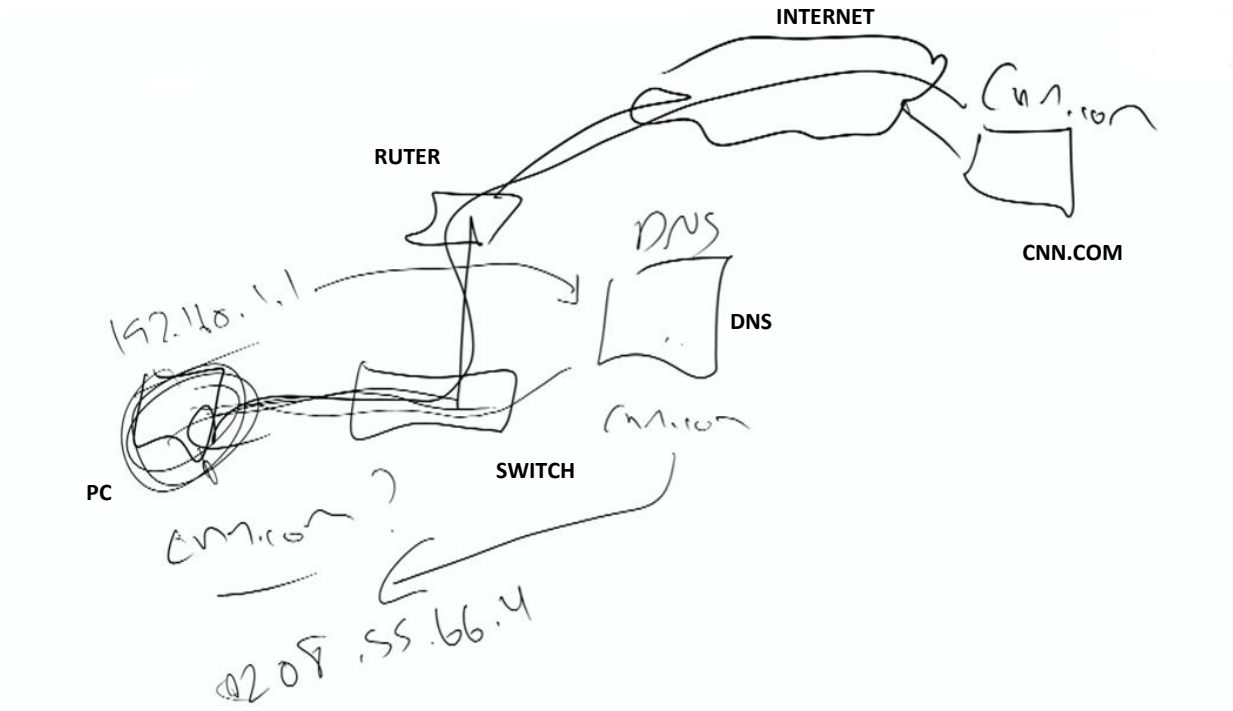
Vrsta poruke	Opis
Destination unreachable	Paket se ne može isporučiti
Time exceeded	Polje životni vek je na nuli
Parameter problem	Neispravno polje u zaglavlju
Source quench	Prigušivanje izvora
Redirect	Obavestenje o pogrešnom rutiranju
Echo	Provera aktivnosti računara
Echo reply	Potvrda aktivnosti računara
Timestamp request	Eho sa vremenskom oznakom
Timestamp reply	Potvrda sa vrem. oznakom

- Odredište nedostupno - Ruter ne može da locira odredište; na putu paketa je podmreža koja ne dozvoljava fragmentaciju datagrama.
- Vreme isteklo - Generiše se kada se paket odbaci jer mu je životni vek (TTL - Time to Live) istekao.
- Greška u parametrima - U polju zaglavlja je otkrivena nedozvoljena vrednost.
- Prigušivanje izvorišta - Danas se retko koristi, problem zagušenja se rešava na transportnom sloju.
- Preusmeravanje - Obaveštava se pošaljioc da je paket pogrešno usmeren.
- EHO i odgovor na EHO - Utvrđivanje da li je određeno odredište dostupno i aktivno. Na poruku EHO odredište **odgovara.**
- EHO i ODGOVOR sa vremenskom potvrdom - Testiranje performansi mreže.

DNS - Domain Name System

DNS je servis servera koji mapira nazive domena sa IP adresama. Mnoge stvari koje mi ljudi nalazimo lakim je računarima beskorisno. Kada računari pokušavaju da pristupe nečemu na lokalnoj mreži ili Internetu, oni to rade koristeći IP adresu. Nama ljudima je veoma teško zapamtiti IP adrese, i zato koristmo nazive domena, imena kompjutera, imena hostova, kao što su ***Server***, ili ***cnn.com***. Problem je što računaru cnn.com ne znači ništa, njemu treba IP adresa servera koji sadrži cnn.com. Kada dobije tu IP adresu onda on može da rutira naš pretraživač do te IP adrese koji može da uzme tu Internet stranicu i da nam je prikaže. Isto tako je kada želimo da pristupimo deljenim resursima. Računaru ne znači ništa Server, njemu treba IP adresa tog računara. Kada sazna IP adresu, onda će poslati nas na tu IP adresu koja sadrži folder *Share* i onda možemo da pristupimo deljenim resursima. **DNS je to što povezuje nazive domena sa IP adresama**, kada on ne bi postojao onda bi mi ljudi morali da pamtimo sve te IP adrese što bi bilo mnogo teško.

Problem većini početnika je što misle da servisi rade mnogo više nego što stvarno rade pa se zbune. Bitno je napomenuti da je DNS samo jedan od servisa koji se koriste na TCP/IP protokolu i jedino što radi je mapiranje naziva domena sa IP adresama. **DHCP** servis je ono što dinamično dodeljuje IP adrese računarima kada se povežu na mrežu, ono je potpuno druga stvar od DNS-a i može biti skladištena na nekom drugom serveru. U praksi su uglavnom DHCP i DNS na istom serveru, uglavnom svi ti servisi su ugrađeni u današnje rutere.

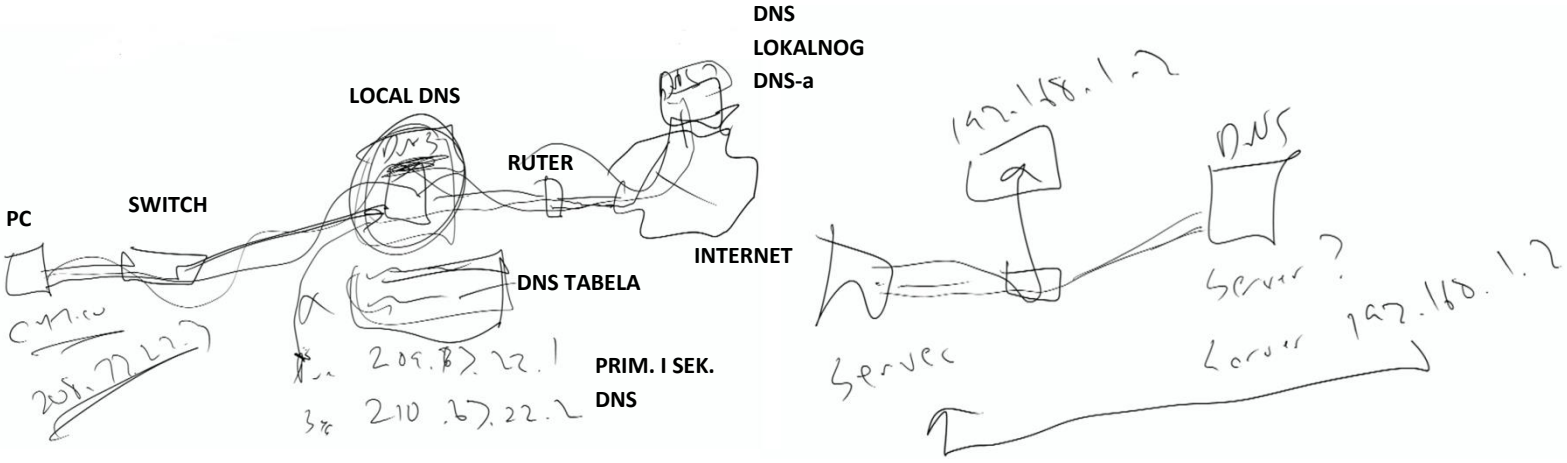


Lokalna mreža

- Naš računar je povezan na naš switch. Kada on hoće da ode na neku stranicu tipa cnn.com, računaru to uopšte nema smisla. Unutar našeg računara DNS server je već konfigurisan, npr. 192.168.1.1 je default DNS server konfigurisan na ovom računaru. Naš PC će otići na tu adresu (koja je unutar lokalne mreže, ide kroz switch) i reći ,,želim da odem na cnn.com''. DNS server mu onda odgovara da se cnn.com nalazi na 208.55.66.4. Nakon toga će naš računar otići kroz switch->ruter na Internet i potražiti tu IP adresu, ta adresa će ga odvesti na cnn.com, i onda će cnn.com poslati stranicu kao odgovor.

Internet

- Treba napomenuti da ako je lokalni DNS server u pitanju, onda će imati samo rezultate za lokalnu mrežu, znaće samo za računare unutar mreže. Kada pitamo za cnn.com, to će proći kroz switch, doći do lokalnog DNS servera koji je povezan na ruter koji je povezan na Internet. Unutar našeg lokalnog DNS servera, naš DNS server će takođe imati sopstvene DNS podatke. Dakle, imaće svoje podatke, ali ako unutar tih podataka ne može da nađe odgovor na upit imaće svoj DNS kojeg treba da upita ako nema odgovora. Npr. 208.77.22.1 mu je primarni DNS server, 210.67.22.2 mu je sekundarni. Ako pokušamo da odemo na cnn.com, prvo ćemo ići kroz switch do lokalnog DNS servera, koji će pretražiti sopstvenu bazu da vidi ako ima odgovor. Kada vidi da ne može da nađe tu informaciju, proveriće koji mu je glavni DNS server i otići će na Internet ili gde god je taj primarni DNS server lociran i pitati taj DNS server koja bi IP adresa cnn.com-a trebala da bude. Ako on bude imao tu informaciju, on će je vratiti našem lokalnom DNS serveru koji će je vratiti nama (208.77.22.3) i onda ćemo mi moći da pristupimo cnn.com koristeći tu IP adresu. Dakle, unutar naše lokalne mreže, ne samo da naš DNS ima sopstvenu tabelu IP adresa i naziva domena, već ima i sopstveni DNS server kome će slati upit ako nema odgovor unutar svoje tabele.



- Slična priča kada ga pitamo ,,gde se nalazi računar *Server*?'' (sl. desno), on će proveriti svoje tabele i dati nam odgovor tj. IP adresu, i onda mi preko te IP adrese kroz switch pristupamo tom računaru.

1. Naš PC pita DNS gde je računar *Server*
2. DNS odgovara *"Server se nalazi na 192.168.1.2"*
3. Sada kada zna IP adresu, naš PC šalje podatke preko switch-a računaru *Server*