Glava 5: Mrežni nivo

5. 1 Uvod

5.2 IP (*Internet Protocol*)

- Format datagrama
- IP adresiranje

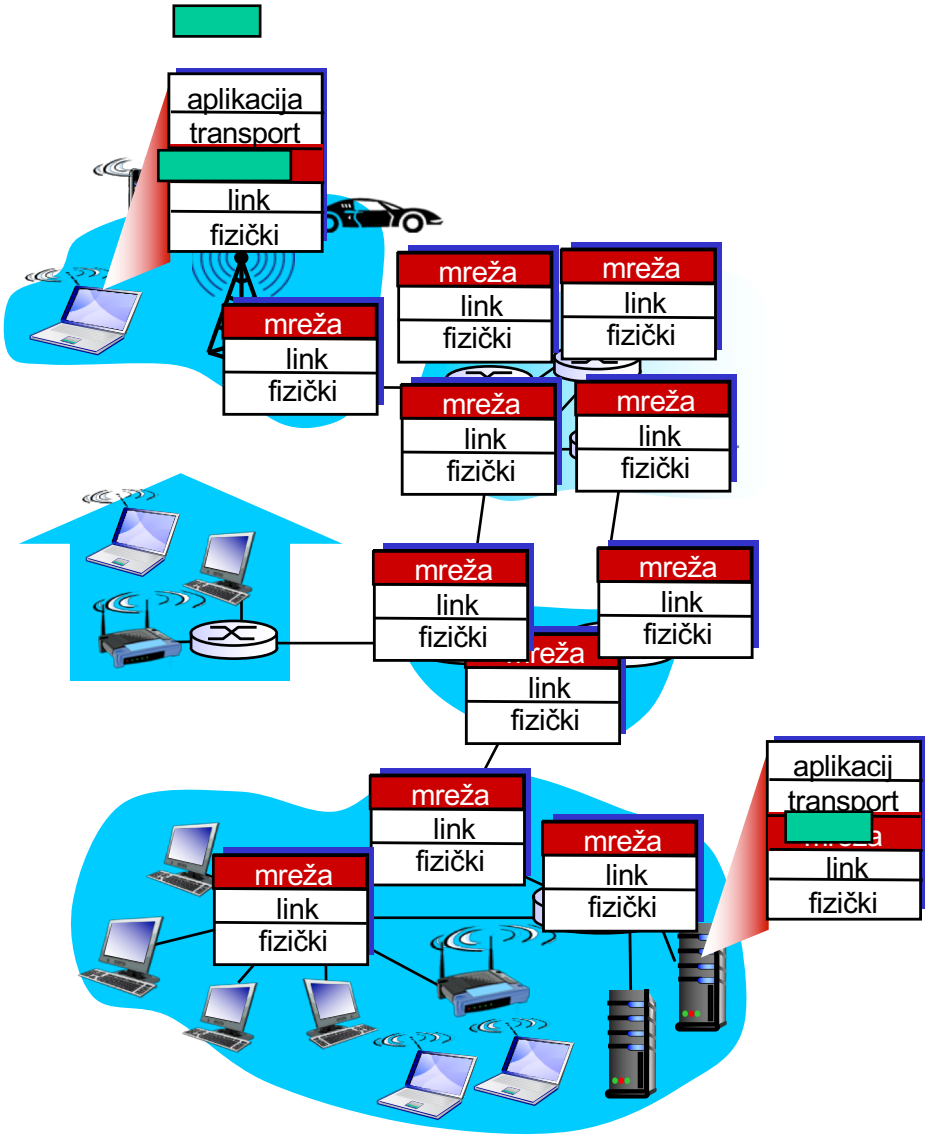
5.3 Rutiranje

- *Link state*
- *Distance Vector*
- Hijerarhijsko rutiranje
- Protokoli rutiranja

5.4 Ruter

Mrežni nivo

- Prenos segmenta od pošiljaoca do odredišta
- Na strani koja šalje enkapsuliraju se segmenti u datagrame
- Na strani prijema predaja segmenata transportnom nivou
- Protokoli mrežnog nivoa su implementirani u *svakom* hostu, ruteru
- Ruter ispituje polja zaglavlja svakog IP datagrama kojeg prosleđuje



Ključne funkcije mrežnog nivoa

- *prosleđivanje:* prenoe paketa sa ulaza rutera na odgovarajući izlaz
- *rutiranje:* izbor rute kojom se paketi prenose od izvora do destinacije.
  - *Algoritmi rutiranja*

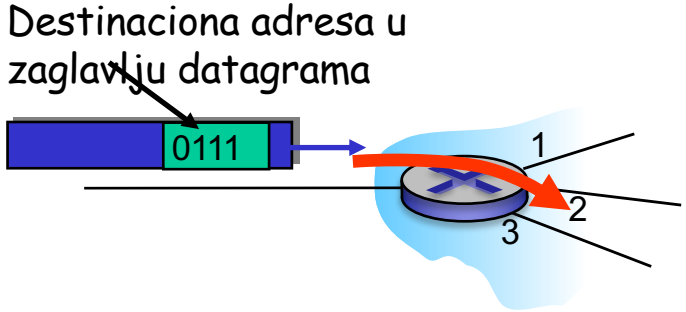
analogija:

- rutiranje: proces planiranja putovanja
- prosleđivanje: proces prolaska kroz jednu raskrsnicu

Mrežni nivo: ravan podataka, ravan kontrole

Ravan podataka

- Lokalna funkcija rutera
- Determiniše kako se datagram koji dolazi na ulazni port rutera prosleđuje na izlazni port
- Funkcija prosleđivanja

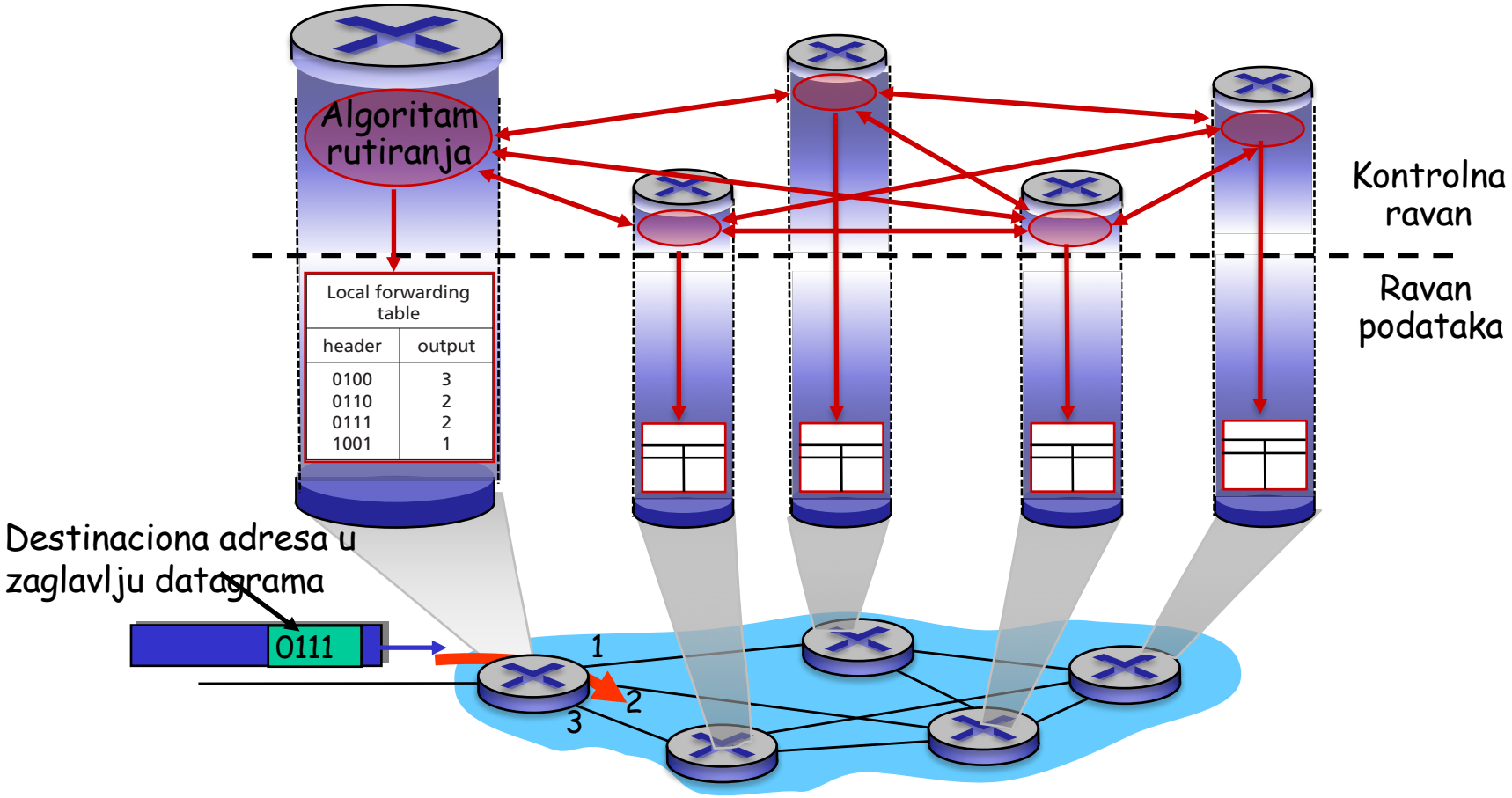


Kontrolna ravan

- Mrežna logika
- Određuje kako se datagram rutira duž putanje od kraja do kraja od izvorišnog do odredišnog hosta
- Dva pristupa:
  - *Tradicionalni algoritmi rutiranja:* implementirani u ruterima
  - *Software Defined Networking (SDN)*: implementirani u udaljenim serverima

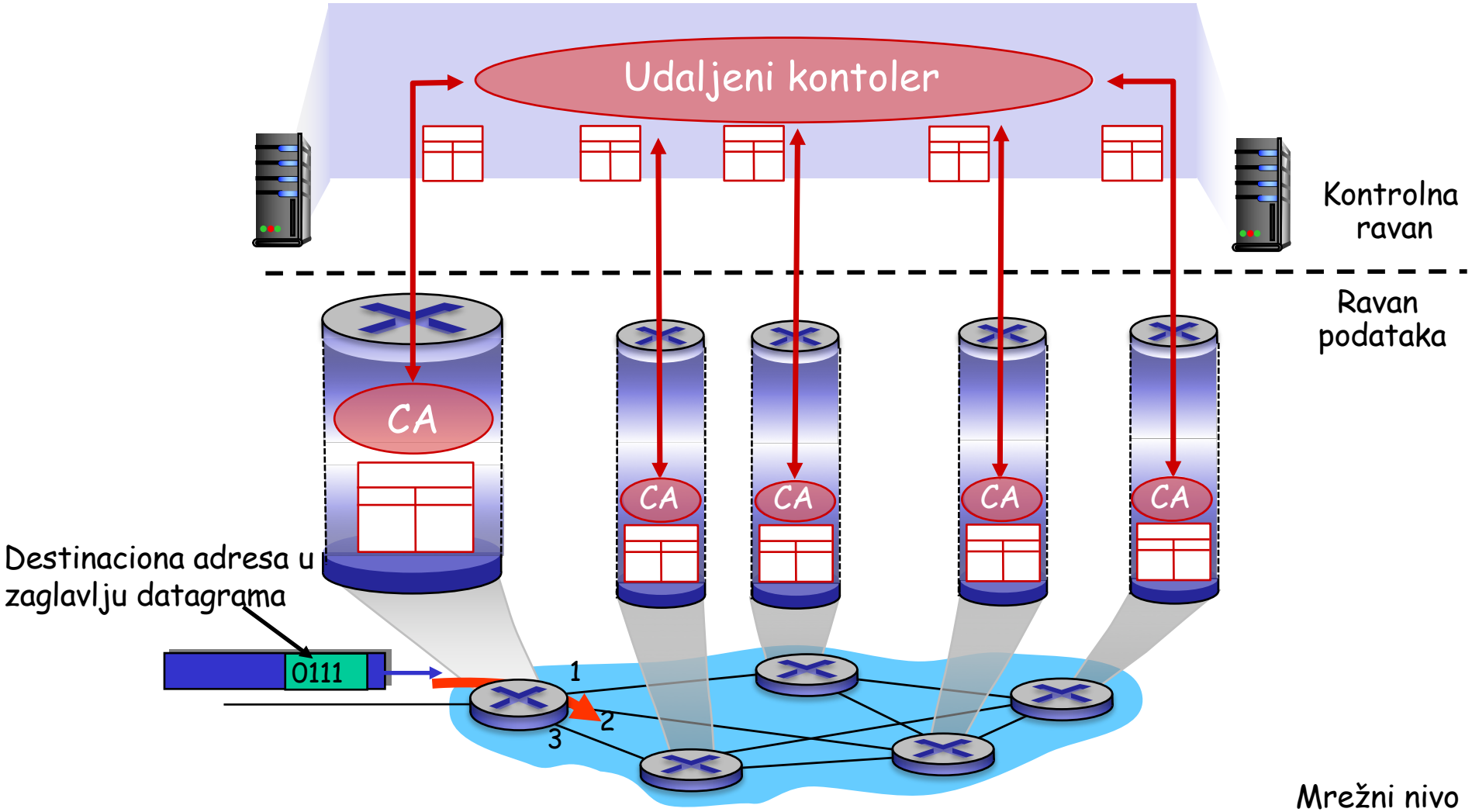
Distribuirana kontrolna ravan

Individualni algoritmi rutiranja se izvršavaju samostalno *u svakom ruteru* i interaguju u kontrolnoj ravni



Centralizovana kontrolna ravan

Udaljeni kontroler interaguje sa lokalnim kontrolnim agentima (CAs)



Mrežni servisni model

Pitanje: Koji *servisni model* nudi "kanal" koji transportuje datagrame od pošiljaoca do prijemnika?

Primjer servisa za individualne datagrame:

- Garantovana predaja
- Garantovana predaja sa kašnjenjem manjim od određene vrijednosti (recimo 100ms)

Primjer servisa za tok datagrama:

- Redosledna predaja datagrama
- Garantovani minimalni protok toka
- Ograničene promjene u međupaketskim intervalima
- Nivo zaštite

Modeli servisa mrežnog nivoa:

Mrežna Arhitektura	Model Servisa	Garantovani ?				"Congestion Feedback"
		Opseg	Gub.	Red.	Tajm.	
Internet	*best effort*	bez	ne	ne	ne	ne (preko gubitaka)
ATM	CBR	konstantna brzina	da	da	da	nema zagušenja
ATM	VBR	garantov. brzina	da	da	da	nema zagušenja
ATM	ABR	garantov. minimum	ne	da	ne	da
ATM	UBR	bez	ne	da	ne	ne

- Internet model se proširuje sa: Intserv, Diffserv

Glava 5: Mrežni nivo

5. 1 Uvod

5.2 IP (*Internet Protocol*)

- Format datagrama
- IP adresiranje

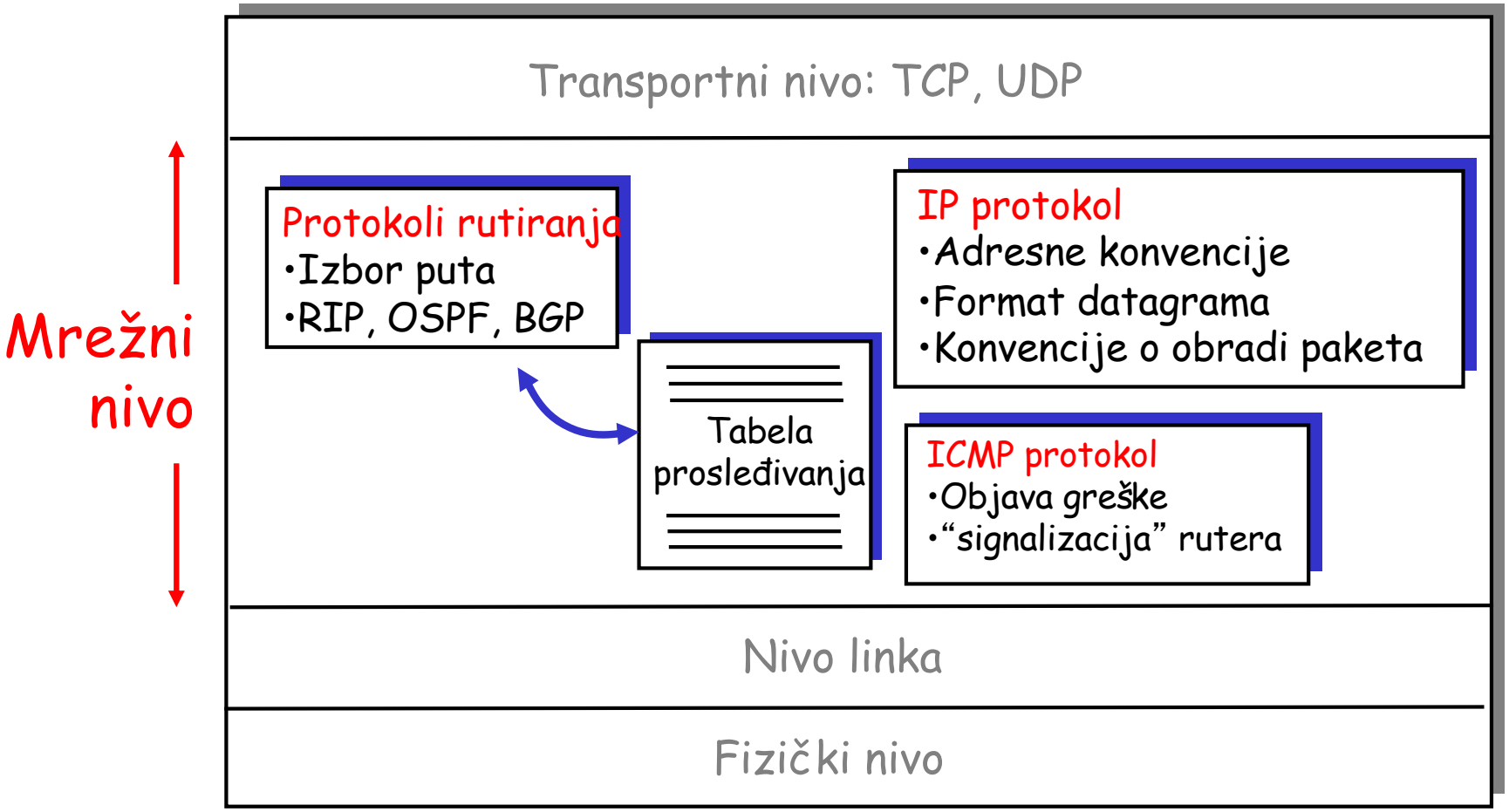
5.3 Rutiranje

- *Link state*
- *Distance Vector*
- Hijerarhijsko rutiranje
- Protokoli rutiranja

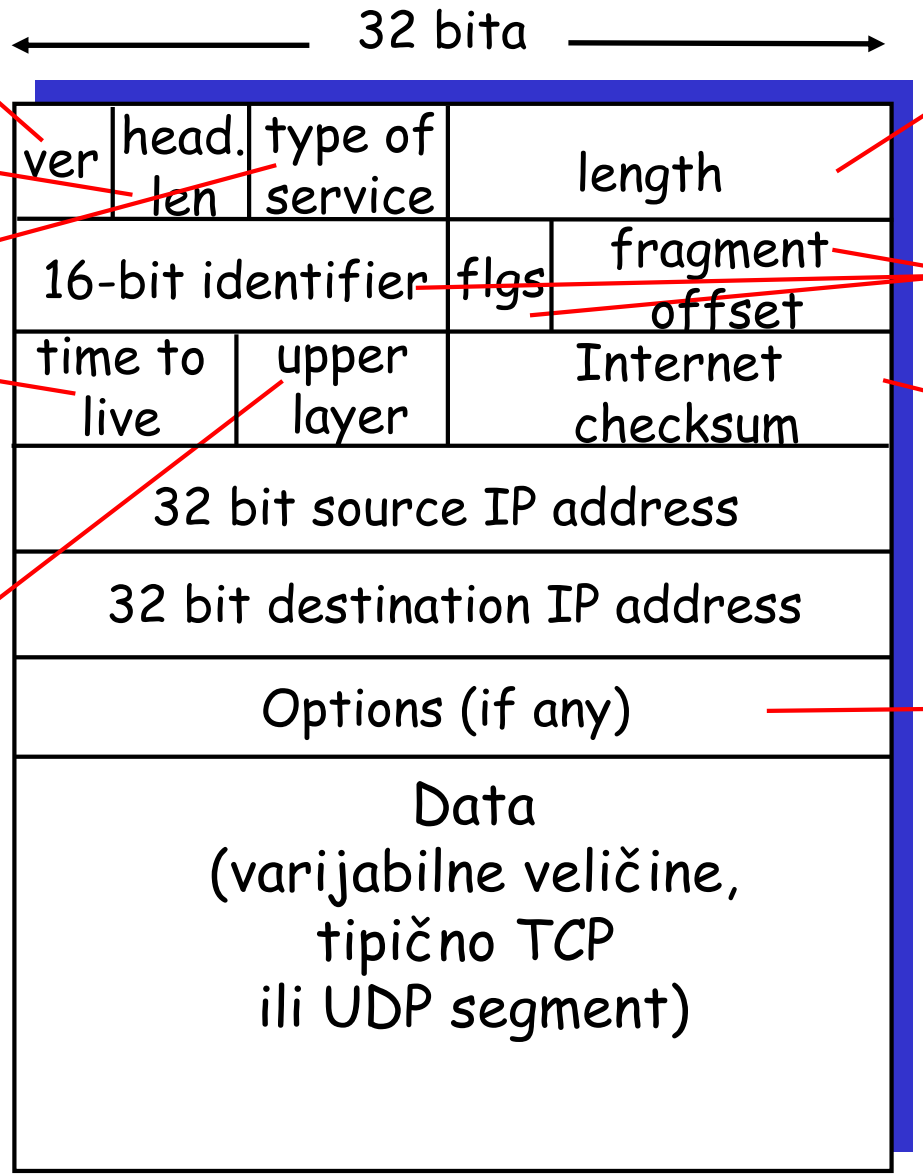
5.4 Ruter

Internet mrežni nivo

Host, ruter funkcije mrežnog nivoa:



Format IP datagrama



Verzija IP protokola

Veličina zaglavlja (u bajtima)

"tip" podataka

Maksimalan broj preostalih hopova (dekrementira se u svakom ruteru)

Protokol višeg nivoa kome treba predati podatke TCP 6, UDP 17

Ukupna veličina datagrama (u bajt.)

za fragmentaciju/ defragmentaciju

Samo za zaglavlje

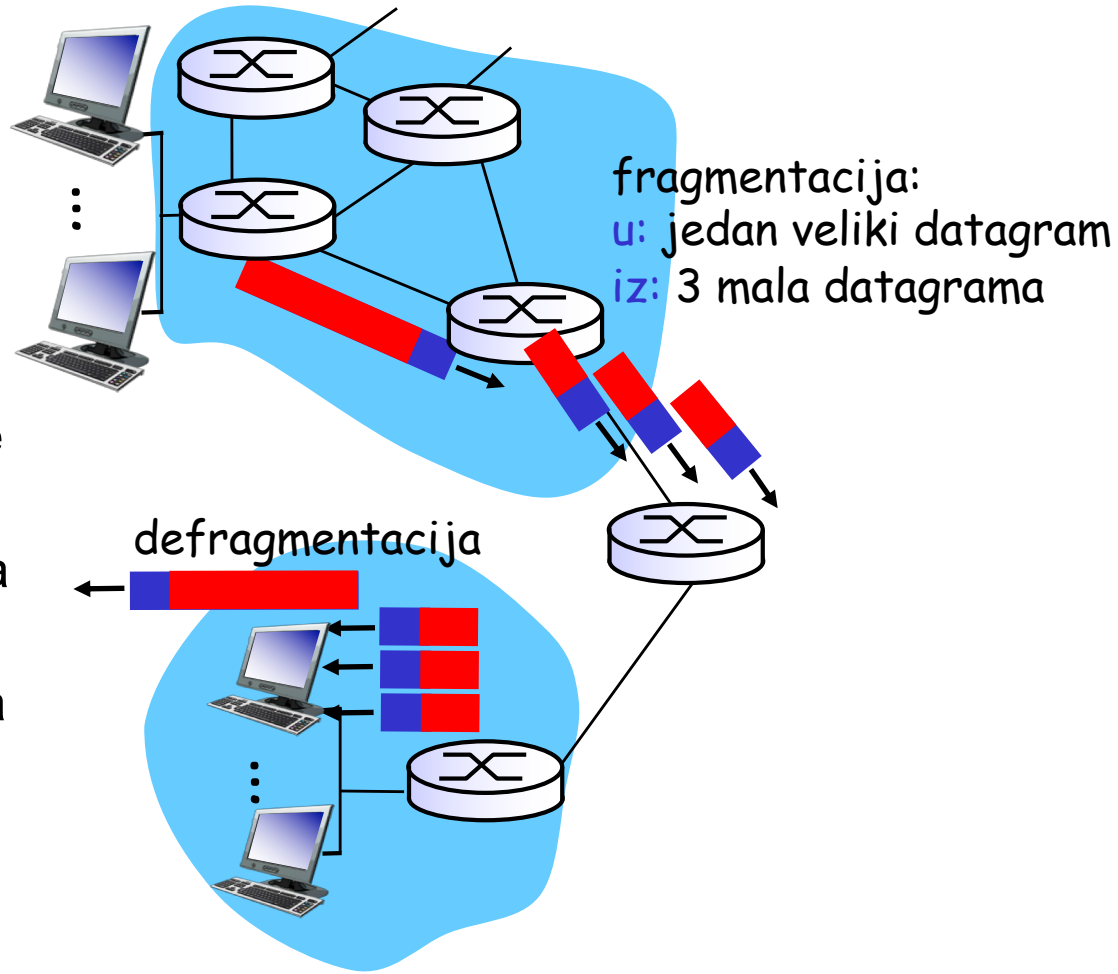
Npr. "timestamp" Definisanje rute, specificira listu rutera koje treba posjetiti.

Koliko zaglavlje sa TCP?

- 20 bajtova TCP-a
- 20 bajtova IP-a
- = 40 bajtova + zaglavlje nivoa apl.

IP Fragmentacija & Defragmentacija

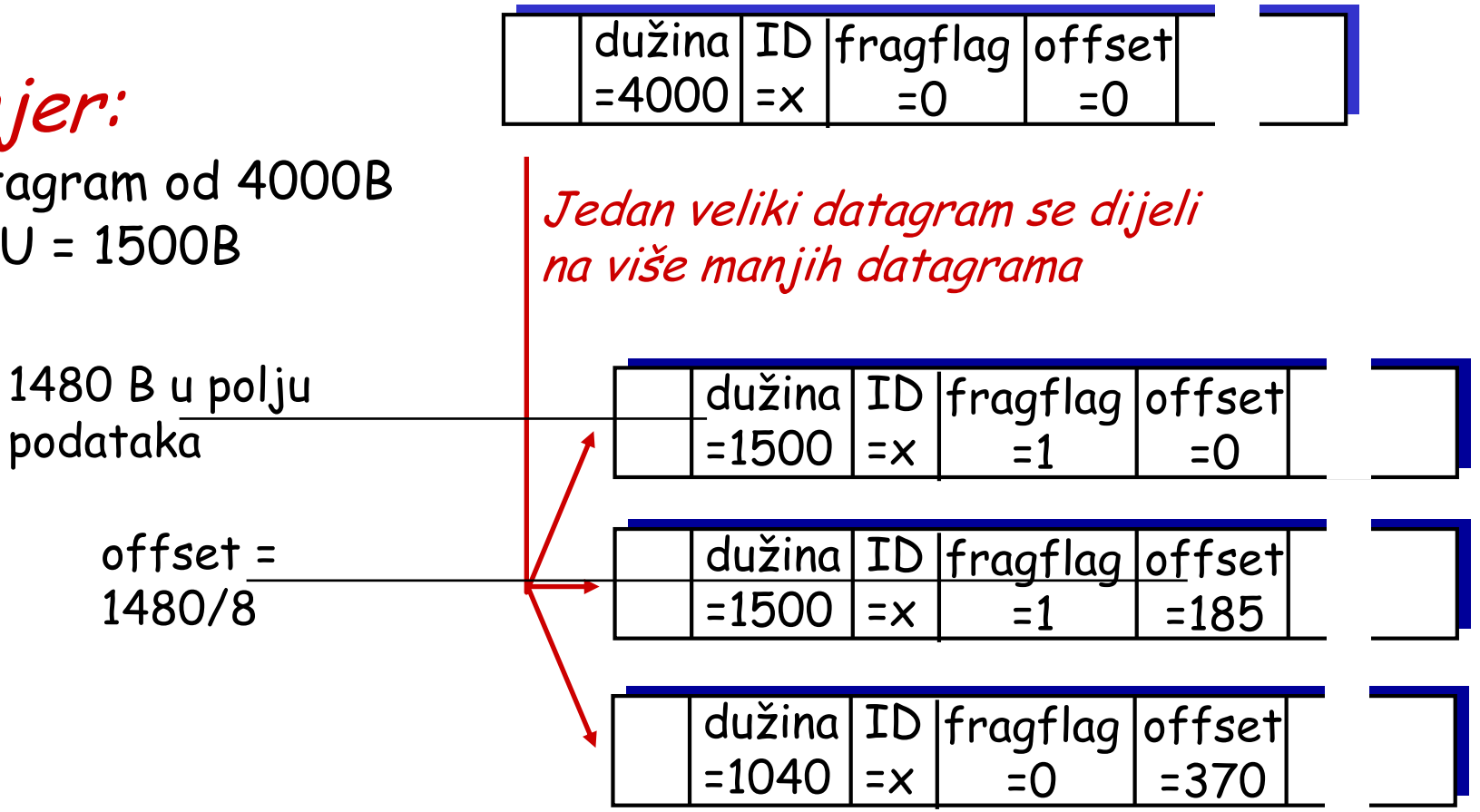
- Mrežni linkovi imaju MTU (*max.transfer size*) ili najveći mogući okvir nivoa linka.
  - Različiti tipovi linkova, različiti MTU-ovi
- veliki IP datagram se dijeli ("fragmentira") u okviru mreže
  - jedan datagram postaje više datagrama
  - "defragmentira" se samo na konačnoj destinaciji
  - IP biti zaglavlja se koriste za identifikaciju redosleda vezanog za fragment



IP fragmentacija, defragmentacija

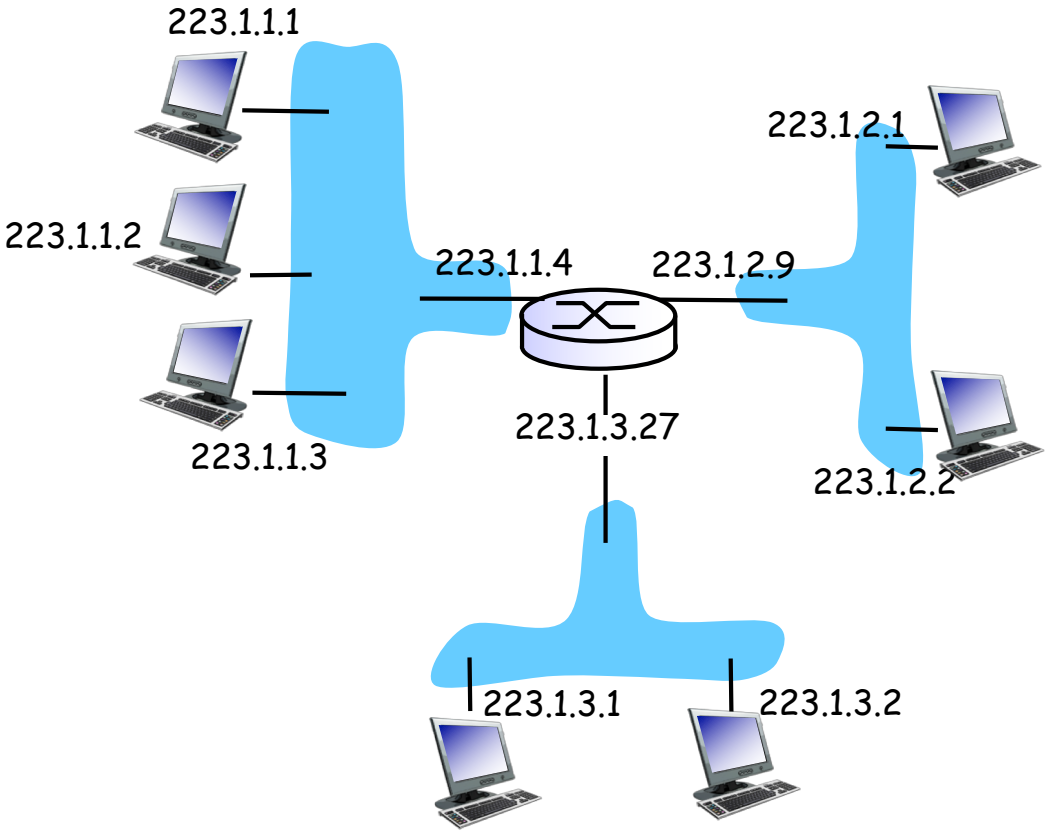
Primjer:

- Datagram od 4000B
- MTU = 1500B



IP Adresiranje: uvod

- IP adresa: 32-bitni identifikator *interfejsa hosta ili rutera*
- *interfejs:* veza između host/rutera i fizičkog linka
  - ruteri tipično imaju više interfejsa
  - i host može imati više interfejsa
  - IP adrese su vezane za svaki interfejs

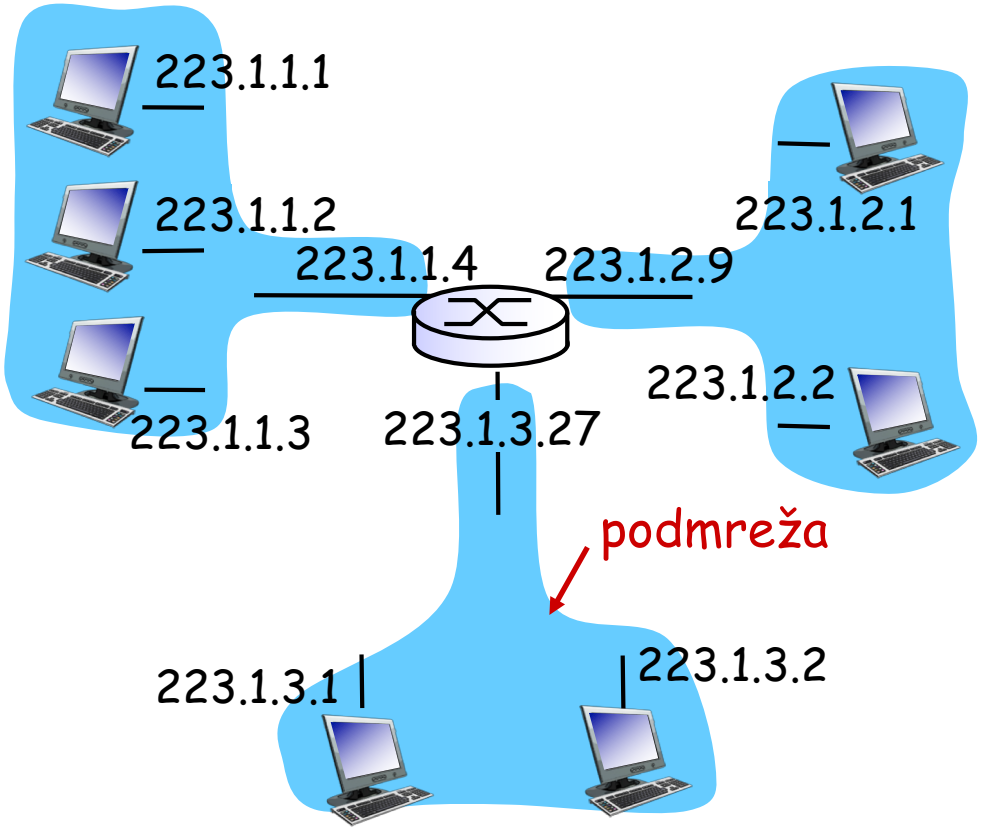


223.1.1.1 = 11011111 00000001 00000001 00000001

223 1 1 1

IP Adresiranje

- IP adresiranje:
  - Mrežni dio (biti višeg reda)
  - Dio hosta (biti nižeg reda)
- *Šta je mreža?* (iz perspektive IP adrese)
  - Interfejsi uređaja sa istim mrežnim dijelom IP adrese
  - mogu fizički dosegnuti jedni druge bez učešća rutera

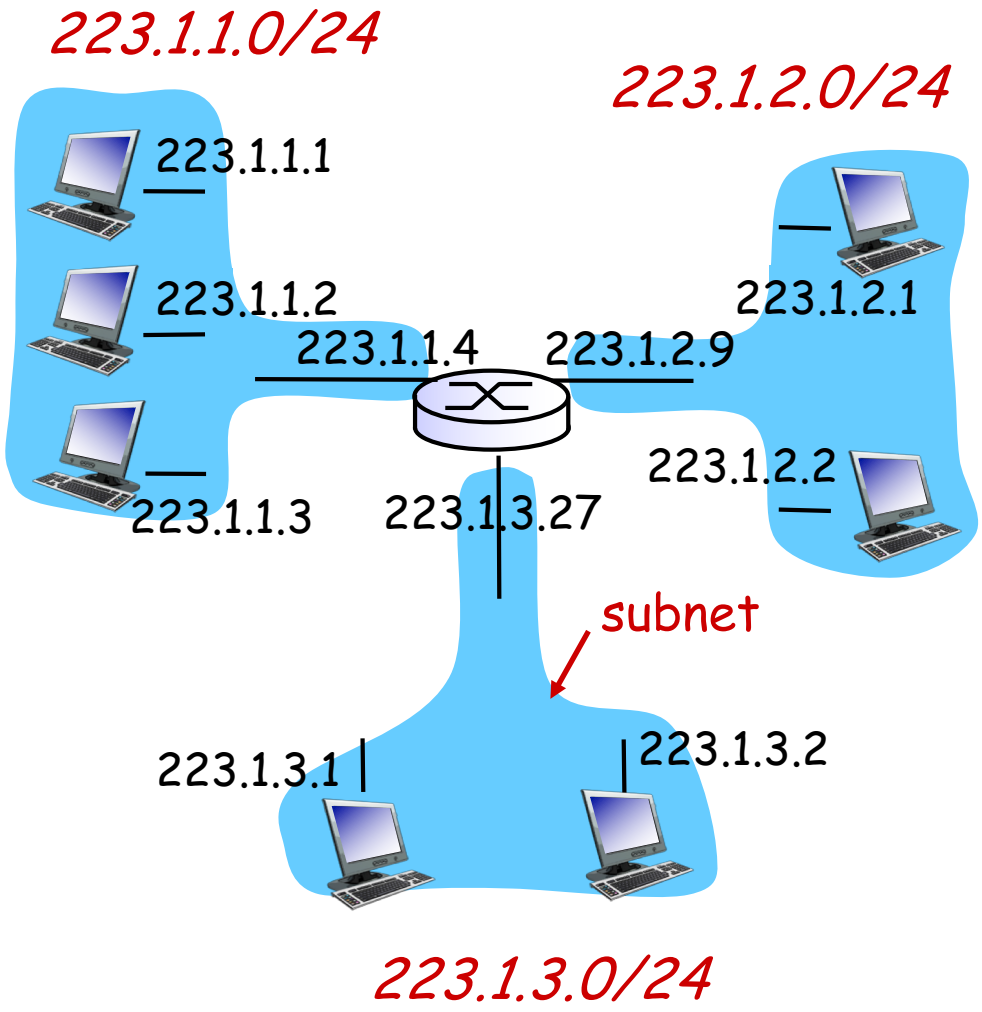


Mreža se sastoji od 3 IP podmreže (prvih 24 bita su mrežna adresa)

Podmreža

Napomena

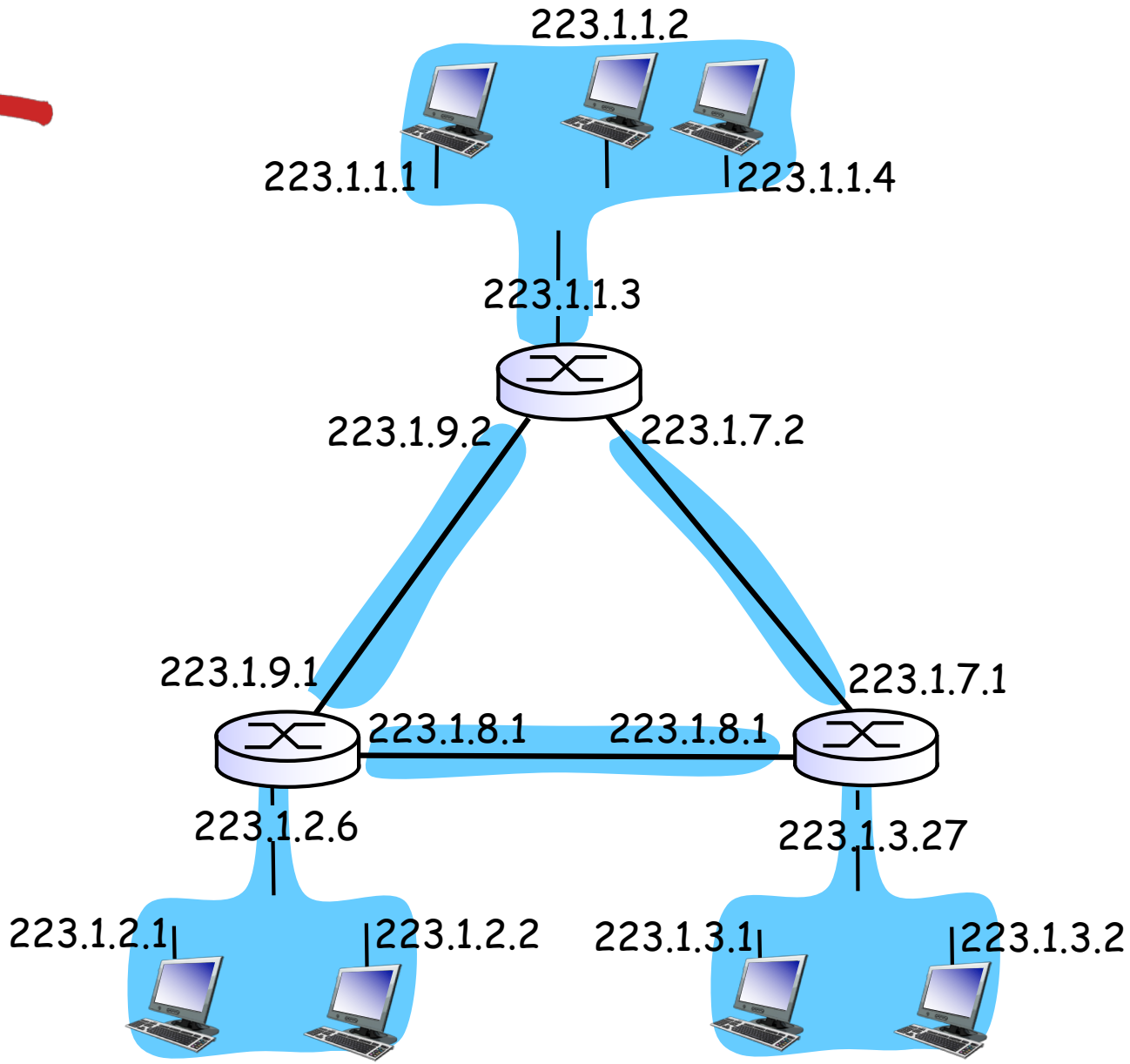
- Da bi odredili podmreže, treba razdvojiti svaki interfejs od njegovog hosta ili rutera, kreirajući ostrva izolovanih mreža. Svaka izolovana mreža se zove podmreža.



Maska podmreže: /24

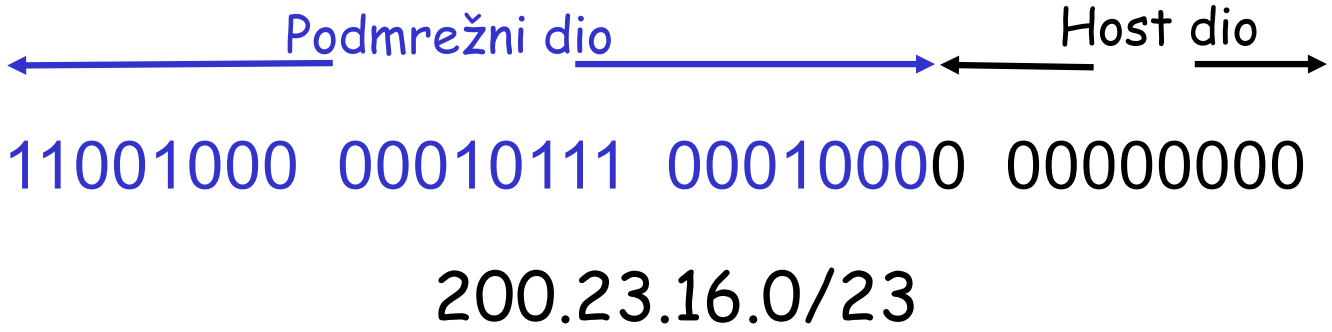
Podmreže

Koliko ih je?



IP adresiranje: CIDR

- CIDR: Classless InterDomain Routing
  - Podmrežni dio adrese je proizvoljne veličine
  - Format adrese: a.b.c.d/x, gdje je x broj bita u mrežnom dijelu adrese



IP adrese: kako dobiti IP adresu?

P: Kako *host* dobija IP adresu?

- od strane sistem administratora
  - Winl: control-panel->network->configuration->tcp/ip->properties
  - UNIX: /etc/rc.config
- dinamički od DHCP servera
  - DHCP (Dynamic Host Configuration Protocol)
  - *plug-and-play*

DHCP: Dynamic Host Configuration Protocol

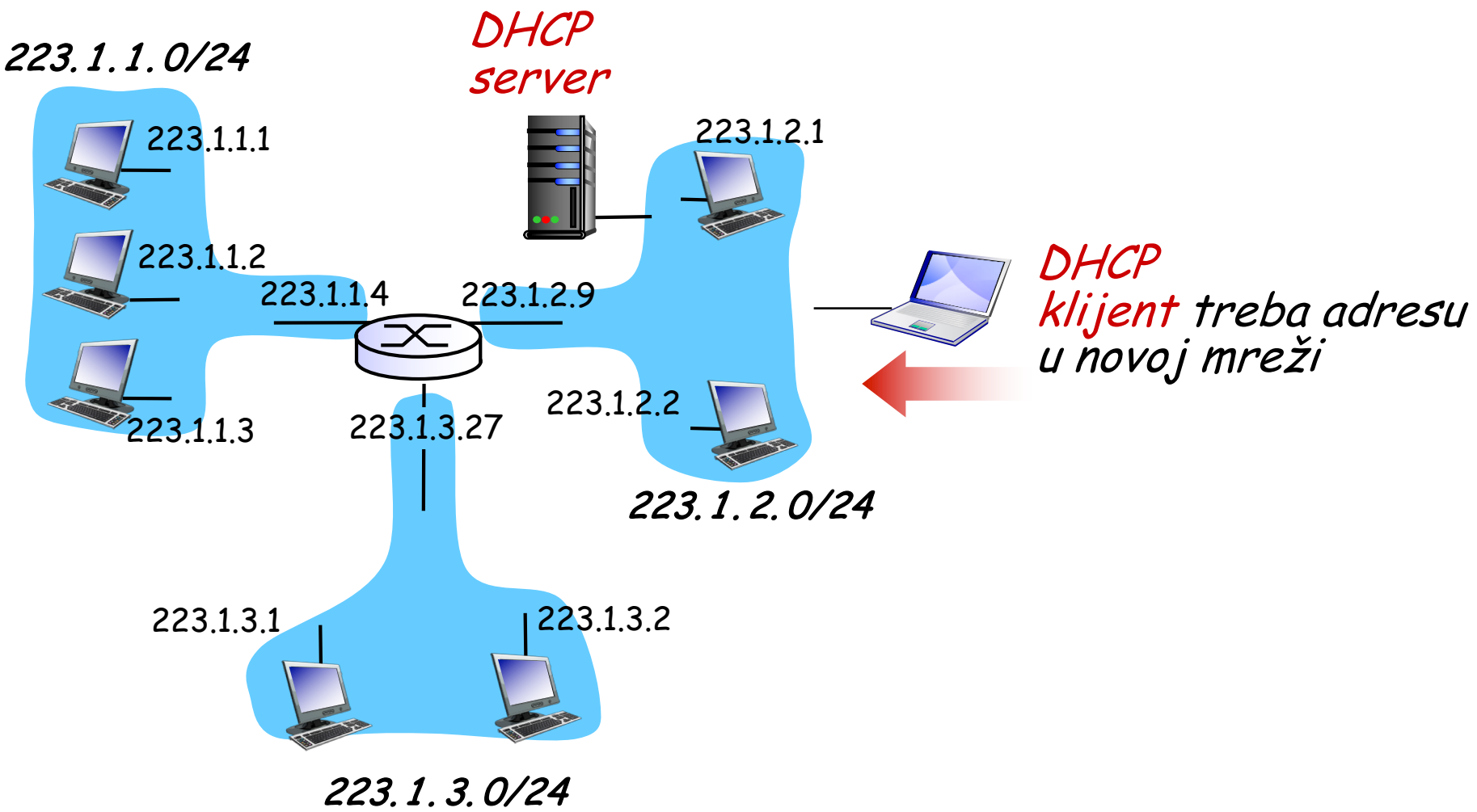
Cilj: omogućiti hostu dinamičko dobijanje adresa, prilikom povezivanja na mreću, od DHCP servera

- Može obnoviti adresu koju je već koristio
- Omogućava "reuse" adresa (host zadržava adresu dok je uključen)
- Olakšava pristup mobilnim korisnicima koji se pridružuju mreži

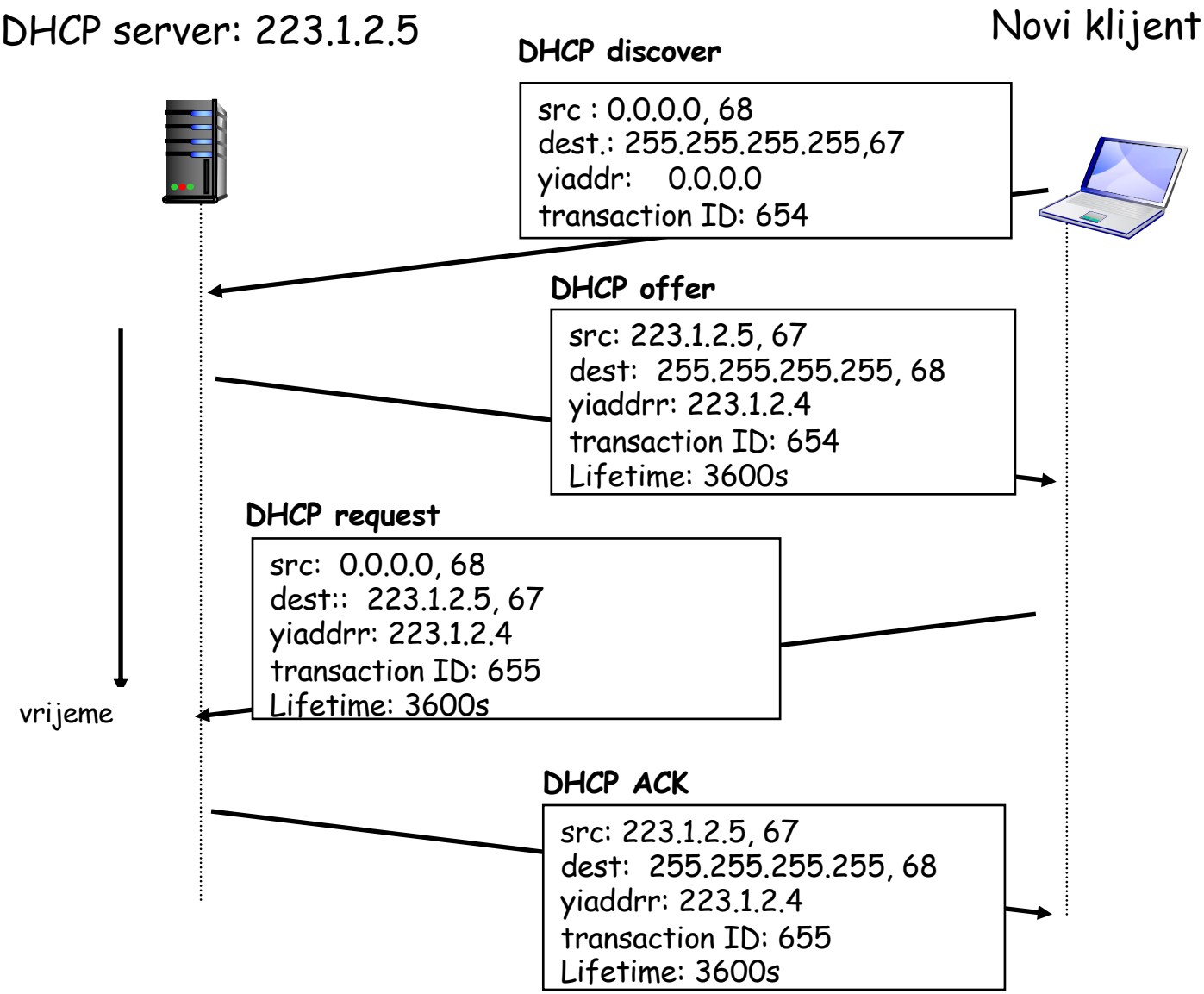
Pregled DHCP:

- host svima šalje "DHCP discover" poruku (UDP segment na port 67)
- DHCP server odgovara "DHCP offer" porukom
- host zahtijeva IP adresu "DHCP request" porukom
- DHCP server šalje adresu "DHCP ack" porukom

DHCP klijent-server scenario



DHCP klijent-server scenario

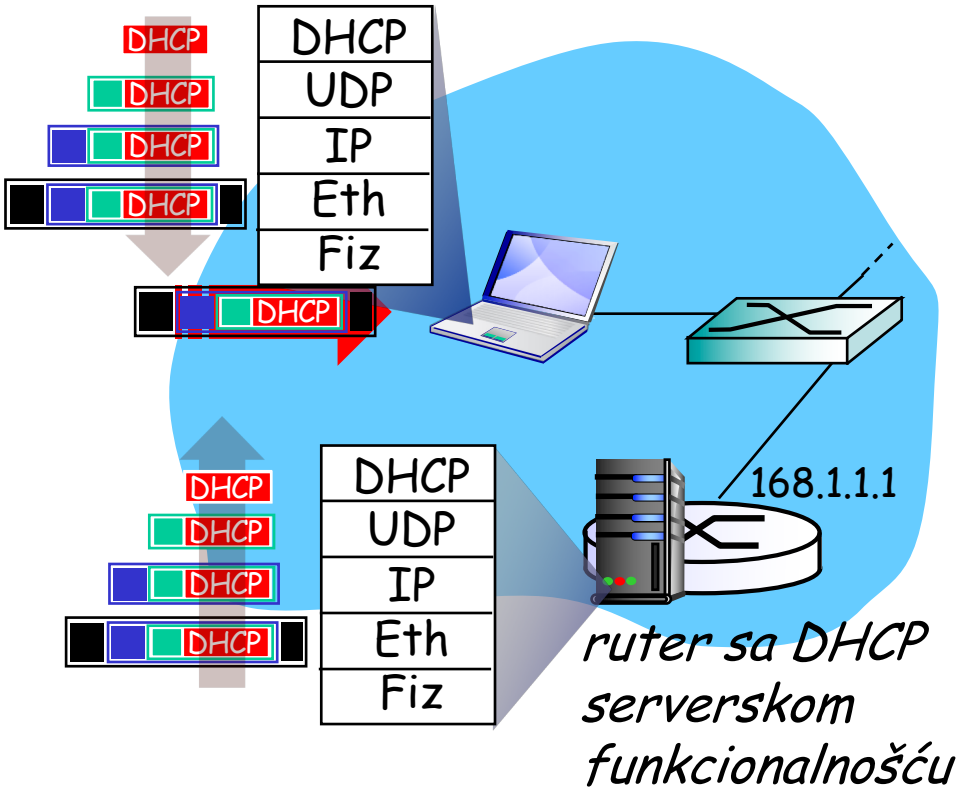


DHCP nudi više od IP adrese

DHCP obezbjeđuje više od same alokacije IP adrese u podmreži:

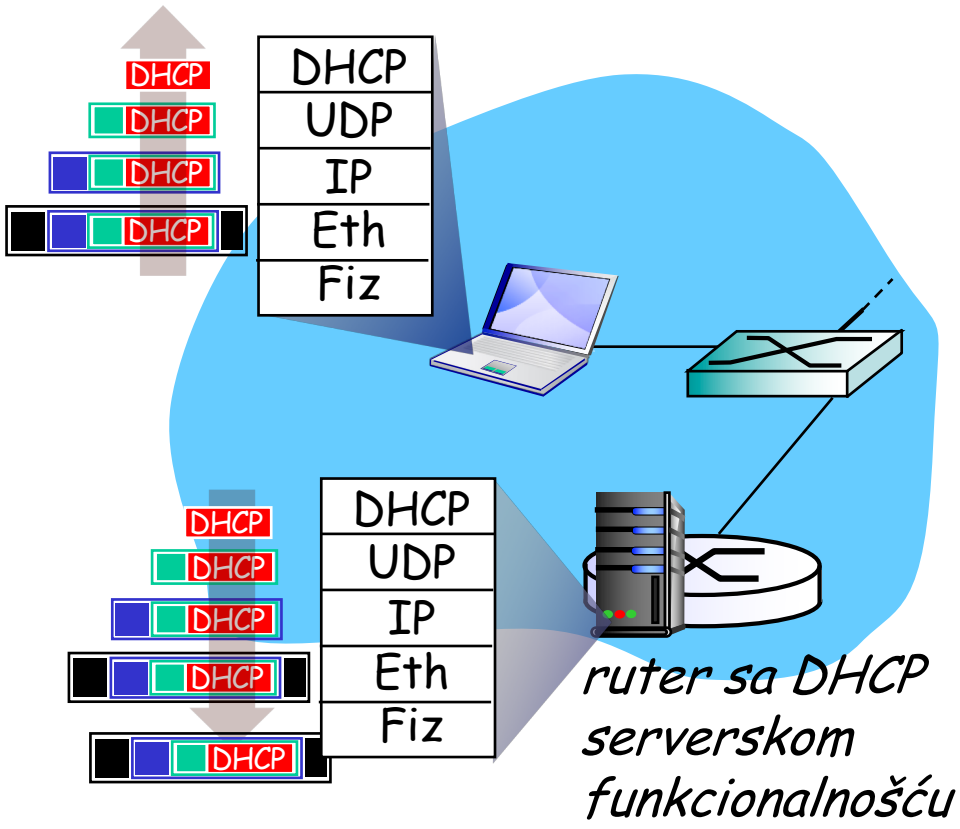
- Adresu *gateway* rutera podmreže
- Ime i IP adresu DNS servera
- *Subnet* masku (indicira mrežni dio adrese)

DHCP: primjer



- Laptopu je potrebna IP adresa, adresa gateway-a i adresa DNS servera
- koristi DHCP
- DHCP zahtjev se enkapsulira u UDP segment, pa u IPdatagram, a zatim u IEEE802.3 Ethernet frejm
- Ethernet fejm se šalje svim interfejsima u LAN-u i prima od strane DHCP servera
- Obavlja se suprotan proces enkapsulaciji

DHCP: primjer



- DHCP server kreira DHCP potvrdu koja sadrži klijentsku IP adresu, IP adresu *gateway* rutera, ime i IP adresu DNS servera
- Frejm se prosleđuje do klijenta koji ga otvara
- Klijentu je poznata IP adresa, ime i IP adresa DSN servera, IP adresa *gateway* rutera

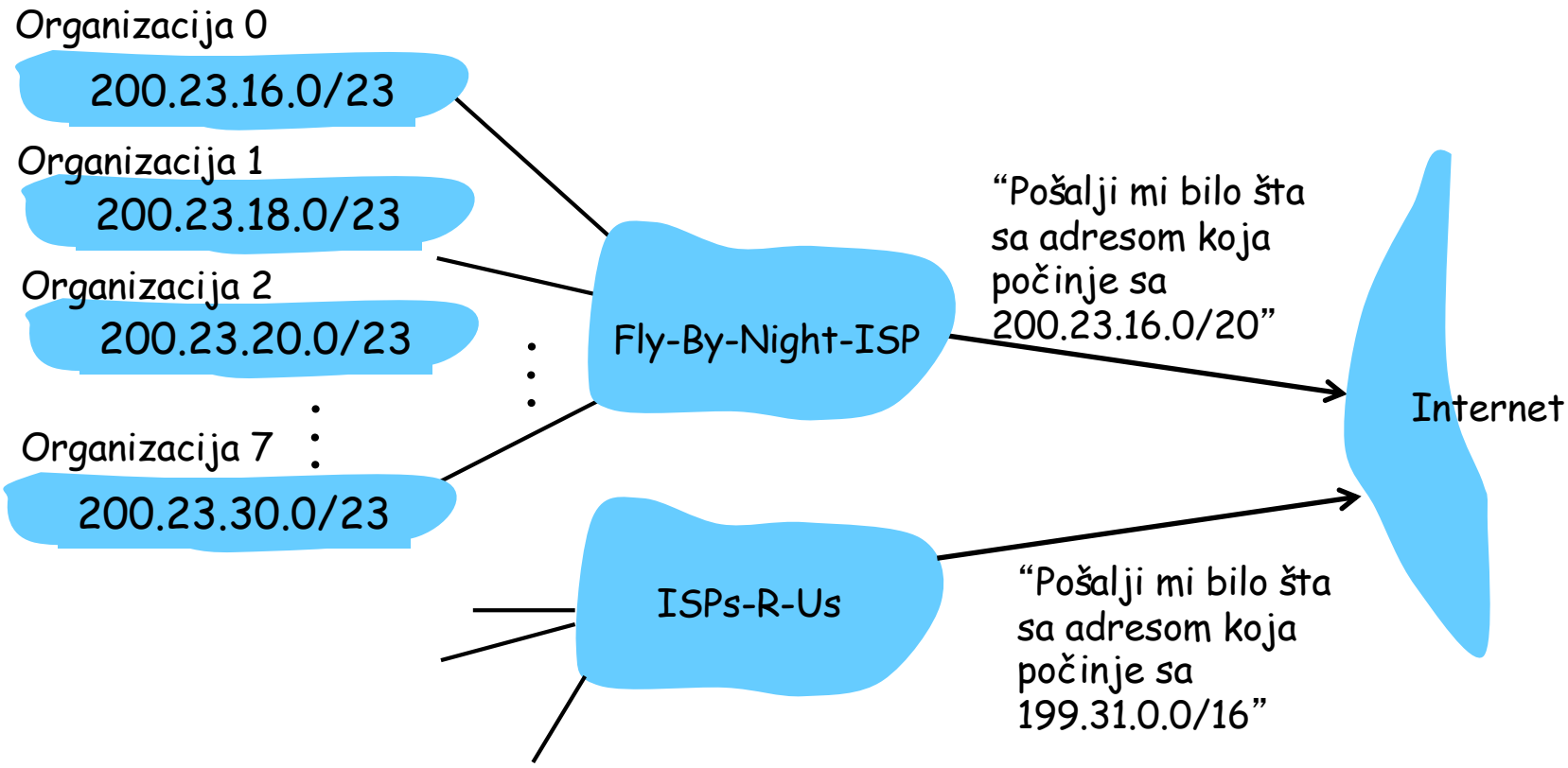
IP adrese: kako dobiti IP adresu?

P: Kako mreža dobija podmrežni dio IP adrese?

ISP-ov blok	11001000 00010111 00010000 00000000	200.23.16.0/20
Organizacija 0	11001000 00010111 00010000 00000000	200.23.16.0/23
Organizacija 1	11001000 00010111 00010010 00000000	200.23.18.0/23
Organizacija 2	11001000 00010111 00010100 00000000	200.23.20.0/23
...	.....	....
Organizacija 7	11001000 00010111 00011110 00000000	200.23.30.0/23

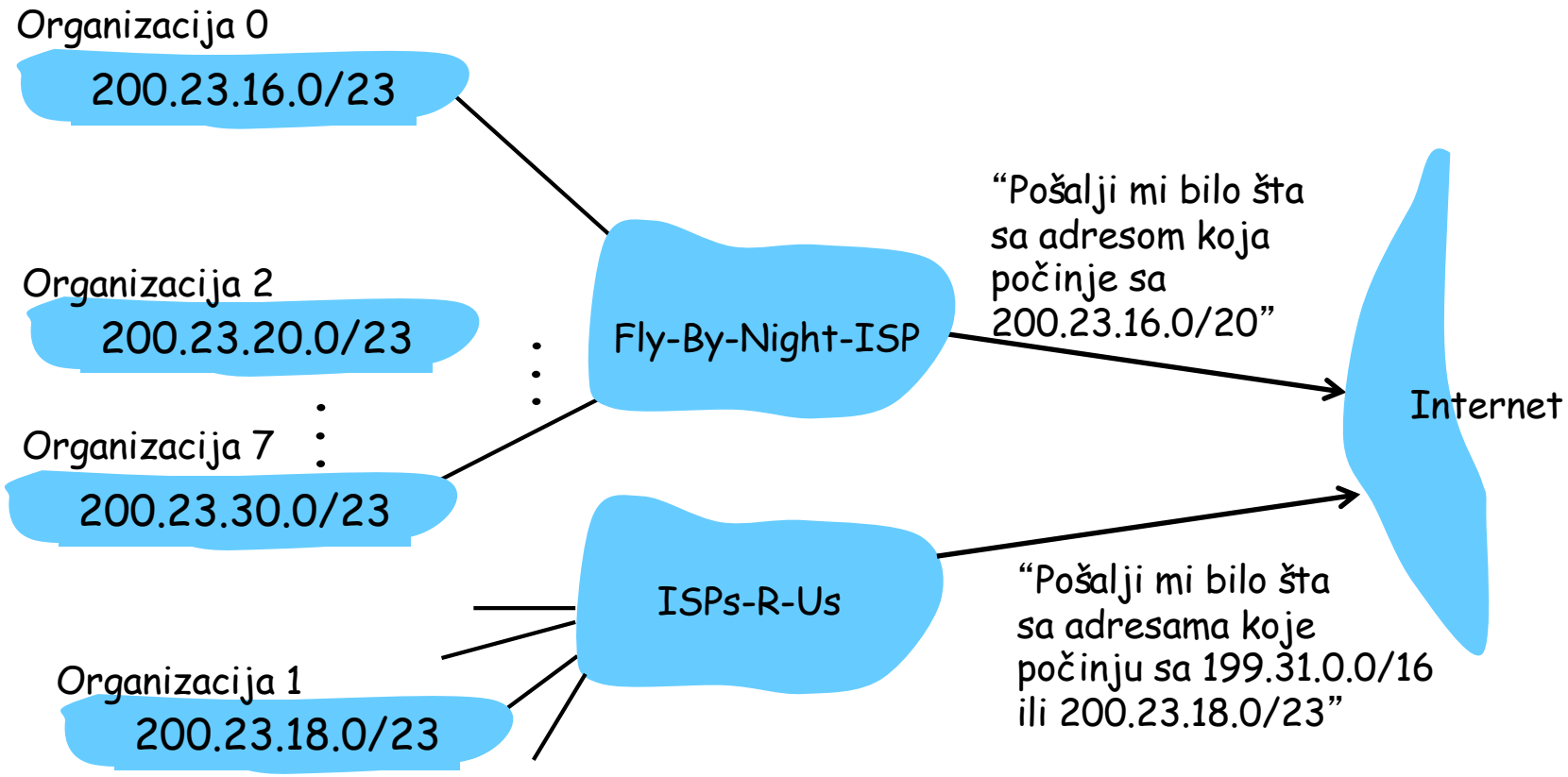
Hijerarhijsko adresiranje: agregacija ruta

Hijerarhijsko adresiranje dozvoljava efikasno oglašavanje informacije potrebne za rutiranje:



Hijerarhijsko adresiranje: specifičnije rute

ISPs-R-Us ima više specifičnih ruta do Organizacije 1



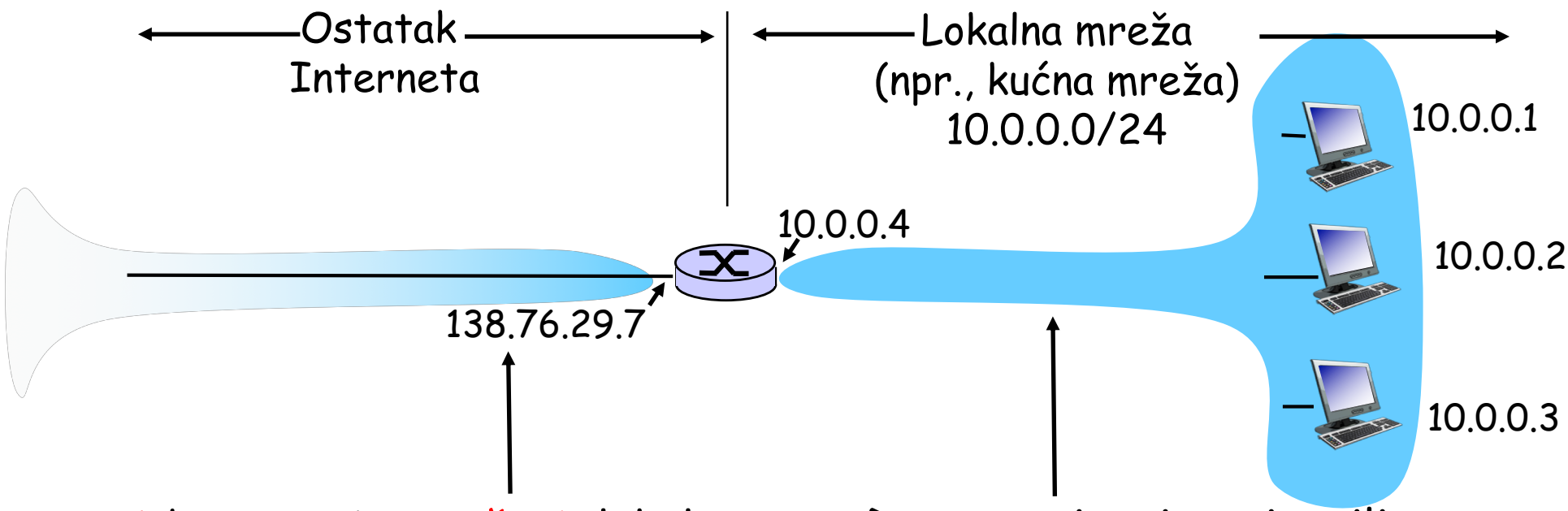
IP adresiranje: poslednja riječ...

P: Kako ISP dobija svoj blok adresa?

O: ICANN: Internet Corporation for Assigned Names and Numbers

- Dodjeljuje adrese
- Upravlja DNS
- Dodjeljuje imena domena
- Razrešava sporove
- Dodjeljuje adrese lokalnim regionalnim Internet registrima (ARIN, RIPE, APNIC, LACNIC i AFRINIC)

NAT: Network Address Translation



Svi datagrami *napuštaju* lokalnu mrežu imajući istu jedinstvenu izvorišnu adresu NAT IP: 138.76.29.7, Različiti brojevi izvorišnih portova

Datagrami sa izvorima ili destinacijama u ovoj mreži imaju 10.0.0.0/24 adresu za izvor, destinaciju (kao što je uobičajeno)

NAT: Network Address Translation

- Motivacija: lokalna mreža koristi samo jednu IP adresu:
  - Nema potrebe za dodjelu opsega adresa od strane ISP (samo jedna IP adresa se koristi za sve uređaje)
  - Mogu mijenjati adrese uređaja u lokalnim mrežama bez obavještavanja "ostatka svijeta"
  - Mogu mijenjati ISP bez mijenjanja adresa uređaja u lokalnim mrežama
  - Uređaji unutar mreže se eksplicitno ne adresiraju, na vidljiv način "ostatku svijeta" (plus u smislu zaštite).

NAT: Network Address Translation

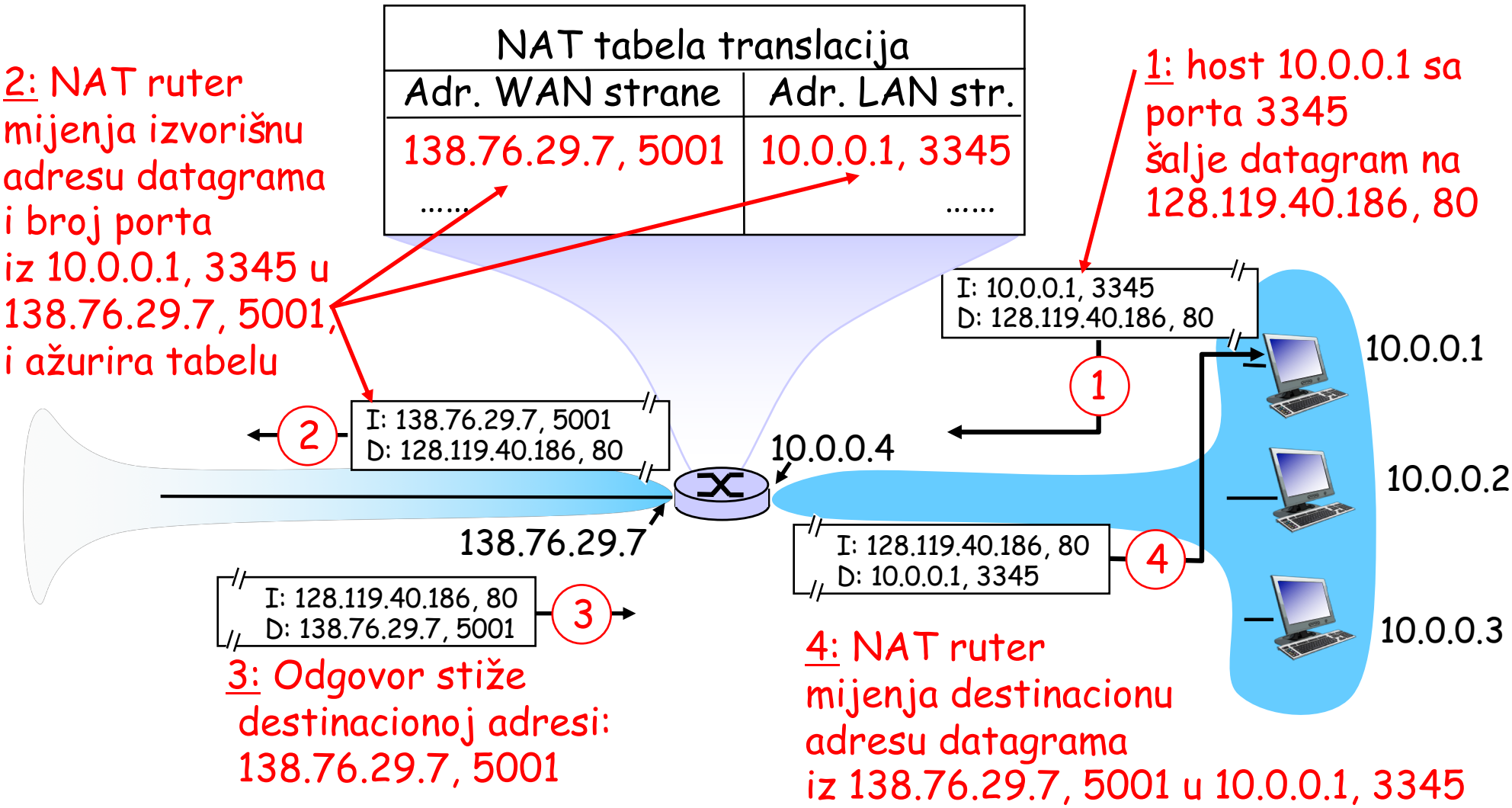
Implementacija: NAT ruter mora:

- *odlazni datagrami:* zamijeniti (izvorišnu IP adresu, broj port) svakog odlaznog datagrama sa (NAT IP adresom, novim brojem porta)
  . . . udaljeni klijenti/serveri će odgovoriti korišćenjem (NAT IP adrese, novi broj porta) kao adrese destinacije.

- *zapamtiti (u NAT tabeli translacija)* svaki (izvorišna IP adresa, broj porta) i (NAT IP adresa, novi broj porta) u vidu translaticionog para

- *dolazeći datagrami:* zamijeniti (NAT IP adresu, novi broj porta) u polju destinacije svakog dolaznog datagrama sa odgovarajućim (izvorišna IP adresa, broj porta) smještenim u NAT tabeli

NAT: Network Address Translation



NAT: Network Address Translation

- 16-bitno polje broja porta:
  - 65536 simultanih veza sa jednom adresom sa LAN strane!
- NAT je kontraverzan:
  - Ruteri bi trebali vršiti obradu samo do nivoa 3
  - Narušava prirodu od kraja do kraja
    - NAT mora biti uzet u obzir od strane dizajnera aplikacija, npr., P2P aplikacija
  - Oskudica adresa se može ublažiti i prije upotrebe IPv6
  - Broj porta se posredno koristi za adresiranje računara

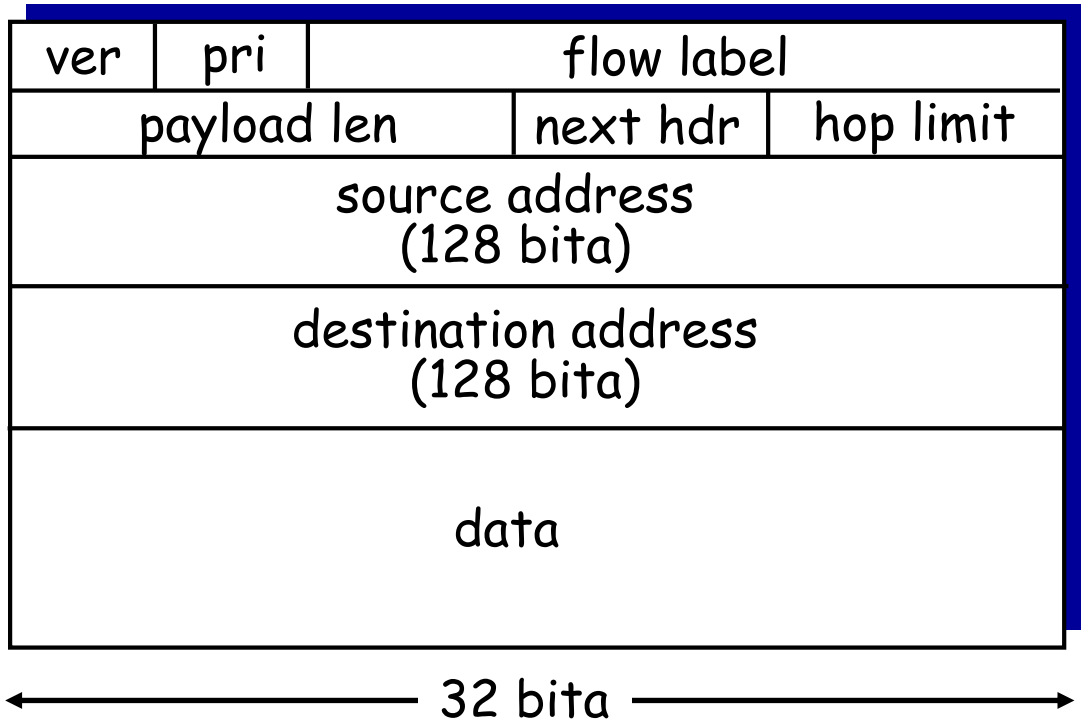
IPv6

- Inicijalna motivacija: 32-bitni adresni prostor će vrlo brzo u potpunosti biti dodijeljen.
- Dodatna motivacija:
  - Format zaglavlja pomaže obradi/prosleđivanju
  - Promjene zaglavlja uključuju QoS

- IPv6 format datagrama:
  - Zaglavlje fiksne-dužine od 40B
  - Nije dozvoljena fragmentacija

IPv6 zaglavlje (nastavak)

Priority: identifikuje prioritet između datagrama u "toku"
Flow Label: identifikuje datagrame u istom "toku".
(koncept "toka" nije precizno definisan).
Next header: identifikuje protokola višeg nivoa za podatke



Druge izmjene u odnosu na IPv4

- *Checksum:* *potpuno uklonjena kako bi se smanjila obrada na svakom* hopu
- *Options:* dozvoljene, ali van zaglavlja, indicirano sa "Next Header" poljem
- *ICMPv6:* nova verzija ICMP
  - dodatni tipovi poruka, npr. "Packet Too Big"
  - funkcija upravljanja multicast grupama

Glava 5: Mrežni nivo

5. 1 Uvod

5.2 IP (Internet Protocol)

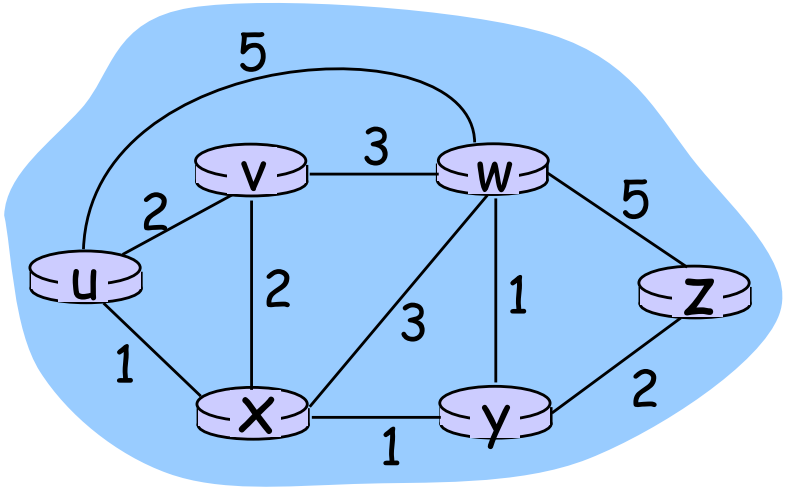
- Format datagrama
- IP adresiranje

5.3 Rutiranje

- *Link state*
- *Distance Vector*
- Hijerarhijsko rutiranje
- Protokoli rutiranja

5.4 Ruter

Abstrakcija pomoću grafa



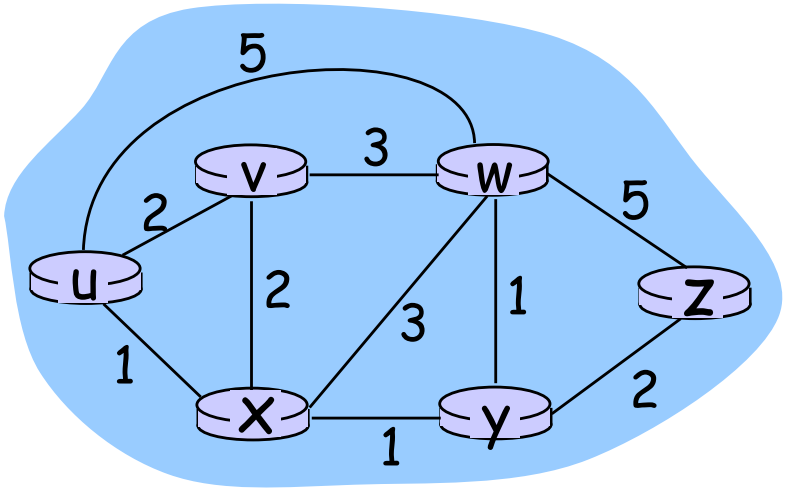
Graf: G = (N,E)

N = skup rutera = { u, v, w, x, y, z }

E = skup linkova ={ (u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) }

Napomena: Abstrakcija pomoću grafa je korisna i u drugim mrežnim kontekstima.
Primjer: P2P, gdje je N skup peer-ova, a E skup TCP konekcija

Abstrakcija pomoću grafa: troškovi



- c(x,x') = težinski faktor (cost) linka (x,x')
  - npr., c(w,z) = 5
- težinski faktor može biti uvijek 1, ili recipročan protoku, ili recipročan zagušenju

Težinski faktor puta $(x_1, x_2, x_3, \ldots, x_p) = c(x_1,x_2) + c(x_2,x_3) + \ldots + c(x_{p-1},x_p)$

Pitanje: Koji je put sa najmanjim težinskim faktorom između u i z ?

Algoritam rutiranja je algoritam koji pronalazi put sa najmanjim težinskim faktorom

Klasifikacija algoritama rutiranja

Globalna ili decentralizovana informacija?

Globalna:

- svi ruteri posjeduju kompletnu topologiju mreže sa informacijama o težinskim faktorima linkova
- *link state* algoritmi

Decentralizovani:

- ruter poznaje fizički povezane susjede i težinske faktore linkova do susjeda
- iterativni proces izračunavanja, razmjena informacija sa susjedima
- *distance vector* algoritmi

Statički ili dinamički?

Statički:

- Rute se sporo mijenjaju

Dinamički:

- Rute se mijenjaju mnogo brže
  - periodični *update*
  - kao odgovor na promjene težinskih faktora linkova

Link-State algoritam rutiranja

Dijkstra algoritam

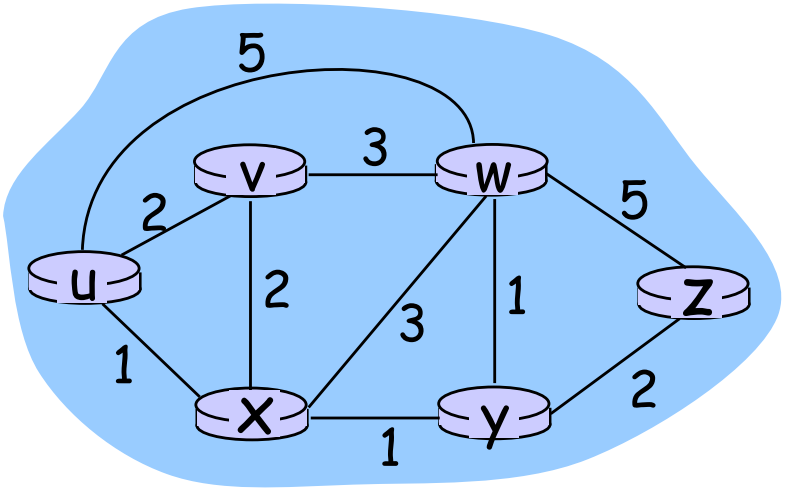
- Mrežna topologija, težinski faktori linkova poznati svim čvorištima
  - Dobijeni preko *link state broadcast* poruke
  - Sva čvorišta imaju istu informaciju
- Proračunava puteve najmanjih težinskih faktora od jednog čvorišta ("izvor") do svih ostalih čvorova
  - generiše tabelu rutiranja za to čvorište
- iterativni: poslije k iteracija, poznat je put sa najmanjim težinskim faktorom do k destinacija

Notacija:

- c(x,y): težinski faktor linka od čvorišta x do y su beskonačni ukoliko čvorišta nijesu susjedi
- D(A): trenutna vrijednost težinskog faktora puta od izvorišta do destinacije A
- p(A): sledeće čvorište duž puta od izvorišta do čvorišta A, koje je susjed A
- N': skup čvorišta čiji su najniži težinski faktori puta poznati

Dijsktra Algoritam (na čvoru u)

1 ***Inicijalizacija:***
2 N' = {u}
3 Za sva čvorišta A
4 Ako je A susjed čvorištu u
5 tada D(A) = c(u,A)
6 else D(A) = ∞
7
8 ***Petlja***
9 Pronaći B koje nije u N' tako da je D(B) minimalno
10 dodati B skupu N'
11 update D(A) za sve A susjede B koji nijesu u N' :
12 D(A) = min(D(A), D(B) + c(B,A))
13 /* novi težinski faktori za A su ili stari težinski faktori za A ili su poznati
14 najmanji težinski faktori puta do B plus težinski faktori od B do A */
15 ***dok sva čvorišta ne postanu članovi skupa N'***

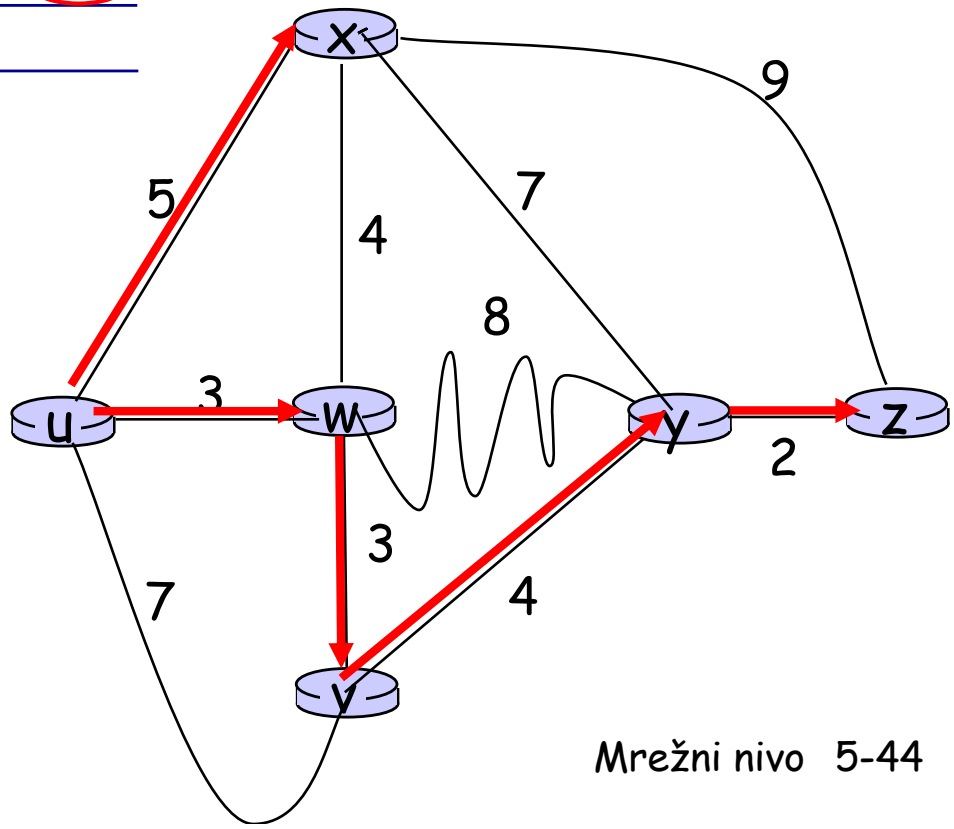


Dijkstra algoritam: primjer

Step	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx	6,w			11,w	14,x
3	uwxv				10,v	14,x
4	uwxvy					12,y
5	uwxvyz					

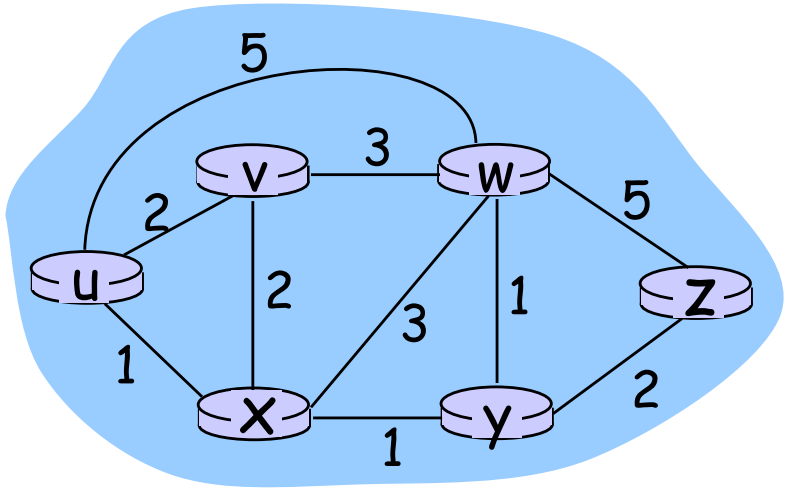
Napomene:

- Pronaći najkraći put praćenjem prethodnih čvorišta
- Linkovi mogu biti prekinuti



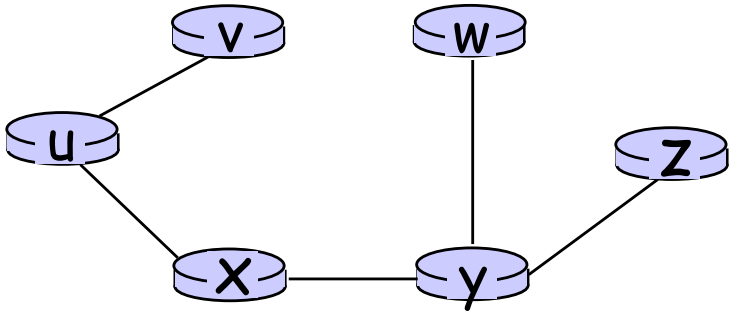
Dijkstra algoritam (čvorište u)

Korak	N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1,u	∞	∞
1	ux	2,u	4,x		2,x	∞
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y
5	uxyvwz					



Dijkstra algoritam (čvorište u)

Rezultantna *shortest-path* topologija iz čvorišta u:



Rezultantna tabela prosleđivanja u čvorištu u:

destinacija	link
v	(u,v)
x	(u,x)
y	(u,x)
w	(u,x)
z	(u,x)

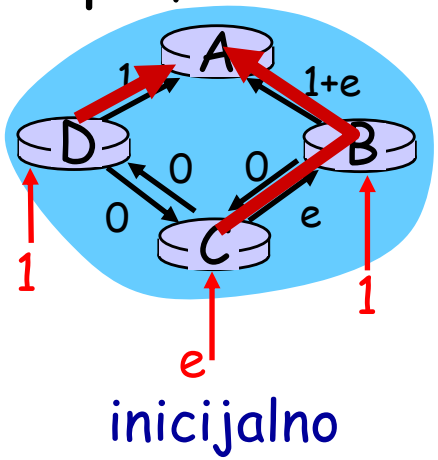
Dijkstra's algoritam, diskusija

Kompleksnost algoritma: n čvorišta

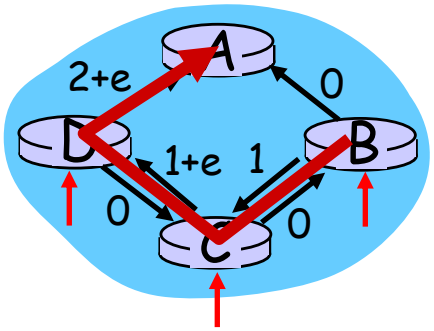
- Svaka iteracija: potrebno da provjeri sva čvorišta koja nijesu u N'
- n*(n+1)/2 komparacija: $O(n^2)$
- Moguće su efikasnije implementacije: O(nlogn)

Moguće su oscilacije:

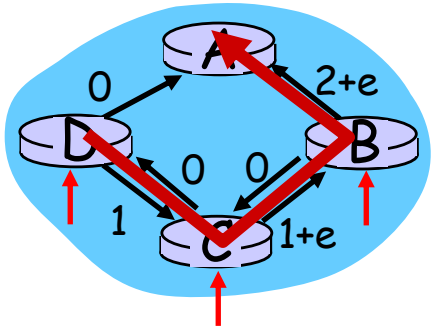
- npr., težinski faktor linka = količina prenešenog saobraćaja



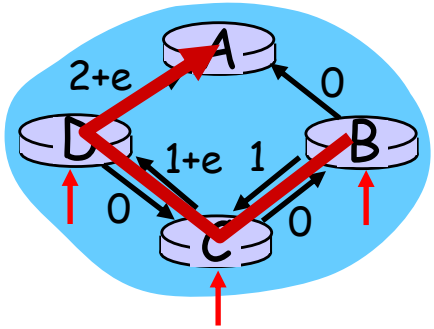
inicijalno



izračunavanje



izračunavanje



izračunavanje

Distance Vector algoritam (1)

Bellman-Ford jednačina (dinamičko programiranje)

Definišimo

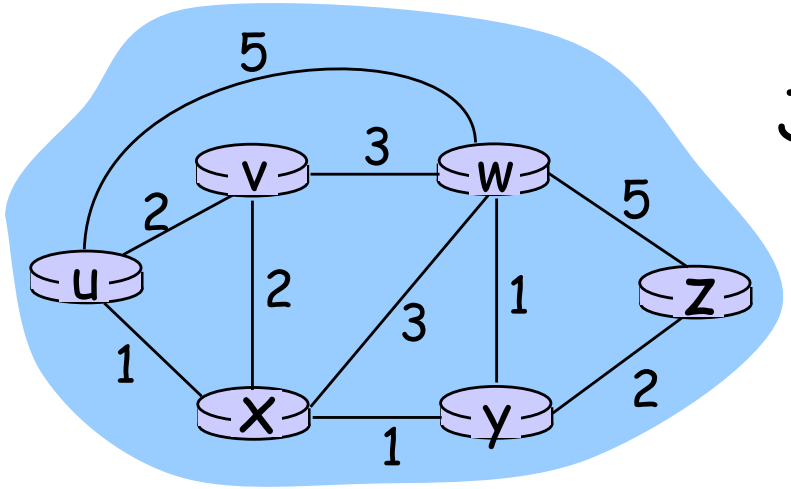
$d_x(y)$:= težinski faktor puta sa najmanjim troškovima od x do y

Tada

$$d_x(y) = \min_v \{c(x,v) + d_v(y)\}$$

Gdje se $\min_v$ uzima u odnosu na sve susjede x

Distance Vector algoritam (2)



Jasno, $d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$

B-F jednačina kaže:

$$d_u(z) = \min \{ c(u,v) + d_v(z), \; c(u,x) + d_x(z), \; c(u,w) + d_w(z) \}$$
$$= \min \{2 + 5, \; 1 + 3, \; 5 + 3\} = 4$$

Čvorište koje dostigne minimum je sledeći korak (hop) u najkraćem putu ➜ tabela prosleđivanja

Distance Vector algoritam (3)

- $D_x(y)$ = estimira najmanji težinski faktor od x do y
- *Distance vector*: $\mathbf{D}_x = [D_x(y): y \in N]$
- Čvorište x poznaje težinske faktore do svakog svog susjeda v: $c(x,v)$
- Čvorište x nadzire $\mathbf{D}_x = [D_x(y): y \in N]$
- Čvorište x također nadzire *distance vector*-e svojih susjeda
  - Za svakog susjeda v, x nadzire $\mathbf{D}_v = [D_v(y): y \in N]$

Distance vector algoritam (4)

Osnovna ideja:

- Svako čvorište periodično šalje estimaciju svog *distance vector*-a svojim susjedima
- Kada čvorište x primi novu DV estimaciju od svog susjeda v, ažurira svoj sopstveni DV korišćenjem B-F jednačine:

$$D_x(y) \leftarrow \min_v\{c(x,v) + D_v(y)\} \quad \text{za svako čvorište } y \in N$$

- U većem broju slučajeva, pod normalnim okolnostima, procjena $D_x(y)$ *konvergira stvarnom najmanjem težinskom faktoru* $d_x(y)$

Distance Vector algoritam (5)

Iterativni, asinhron:

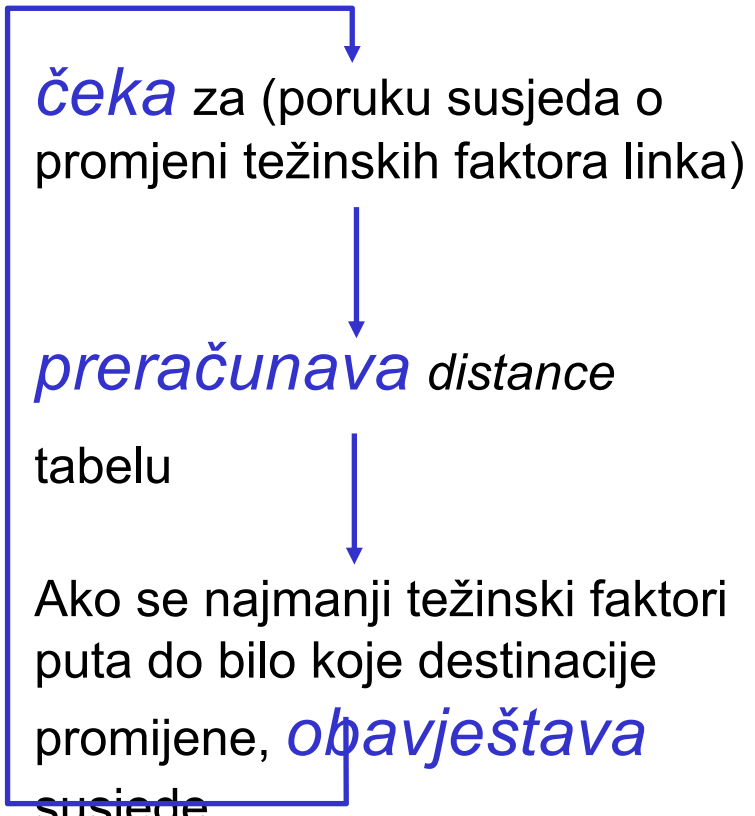
svaka lokalna iteracija je uzrokovana:

- lokalni težinski faktori linka su promjenljivi
- porukama od susjeda: najmanji težinski faktori puta su promijenjeni

Distribuiran:

- svako čvorište obavještava susjeda *samo* kada se njegov put sa najmanjim težinskim faktorom promijeni
  - susjedi informišu susjede ako je to potrebno

Svako čvorište:



$D_x(y) = \min\{c(x,y) + D_y(y), c(x,z) + D_z(y)\} = \min\{2+0 , 7+1\} = 2$

$D_x(z) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\} = \min\{2+1 , 7+0\} = 3$

Tabela čvorišta x

Tež. fakt. do

od	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

Tež. fakt. do

od	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

Tež. fakt. do

od	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

Tabela čvorišta y

Tež. fakt. do

od	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

Tež. fakt. do

od	x	y	z
x	0	2	7
y	2	0	1
z	7	1	0

Tež. fakt. do

od	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

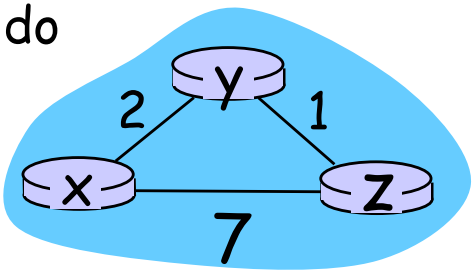


Tabela čvorišta z

Tež. fakt. do

od	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

Tež. fakt. do

od	x	y	z
x	0	2	7
y	2	0	1
z	3	1	0

Tež. fakt. do

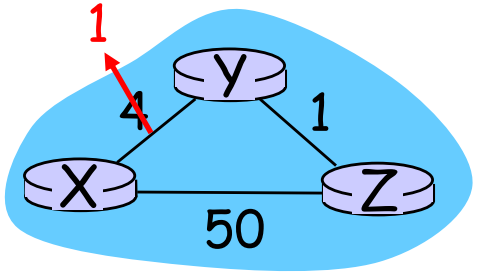
od	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

vrijeme

Distance Vector: promjene težinskih faktora linka

Promjene težinskih faktora linka:

- Čvorište detektuje lokalne promjene težinskih faktora linka
- ažuriranje *distance* tabele
- Ako se težinski faktori promijene na putu sa najmanjim težinskim faktorom, obaviještava susjede



"dobre vijesti brzo putuju"

U trenutku t_0, *y* detektuje promjenu težinskog faktora linka, ažurira njegov DV, i informiše susjede.

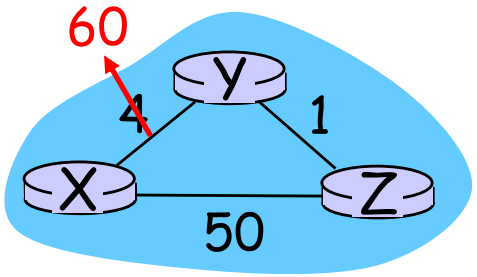
U trenutku t_1, *z* prima update od *y* i ažurira svoju *distance* tabelu. Izračunava novi najmanji težinski faktor do *x* i šalje svojim susjedima svoj DV.

U trenutku t_2, *y* prima od *z update* i ažurira svoju *distance* tabelu. *Y*-ov najmanji težinski faktor se ne mijenja i stoga *y* ne šalje nikakvu poruku ruteru *z*.

Distance Vector: promjene težinskih faktora linka

Promjene troškova linka:

- dobre vijesti se brzo prenose
- loše vijesti se sporije prenose – problem "brojanje do ∞"!
- 44 iteracije do stabilizacije algoritma



Tehnika split-horizon

- Update ruta šalje težinske faktore ruta koje se mogu doseći preko drugih portova.

Poissoned reverse (lažno rastojanje):

- Ako Z rutira preko Y do X :
  - Z govori Y da je njegova (Z-ova) udaljenost do X beskonačna (tako da Y ne bi rutirala do X preko Z)
- Da li će to riješiti problem brojanja do beskonačnosti?

Poređenje LS i DV algoritama

Kompleksnost poruke

- LS: sa n čvorišta, E linkova, O(nE) poruka šalje svaki čvor
- DV: razmjena samo između susjeda
  - Konvergencija varira u vremenu

Brzina konvergencija

- LS: $O(n^2)$ algoritam zahtijeva O(nE) poruka
  - Mogu imati oscilacije
- DV: konvergencija varira u vremenu
  - Može biti petlji
  - Problem brojanja do ∞

Robustnost: šta se dešava kada ruter otkaže?

LS:

- Čvorište može objaviti težinski faktor neispravnog *linka*
- Svako čvorište proračunava svoju sopstvenu tabelu

DV:

- DV čvorište može objaviti težinski faktor neispravnog *linka*
- Tabelu svakog čvorišta koriste drugi
  - Greška se prenosi kroz mrežu

Skalabilno rutiranje

Prethodna analiza je bila - idealizacija

- Svi ruteri su identični
- *flat* mreža

... praksa je drugačija

veličina: nekoliko stotina miliona destinacija:

- ne mogu se sve destinacije smjestiti u tabele rutiranja!
- razmjena tabela rutiranja može oboriti linkove!
- LS može izazvati potiskivanje saobraćaja na račun *broadcasta* tabela
- DV teško može konvergirati

administrativna autonomija

- internet = mreža svih mreža
- svaki mrežni administrator želi
  - kontrolu rutiranja u svojoj mreži
  - sakriti mrežnu organizaciju od ostalih

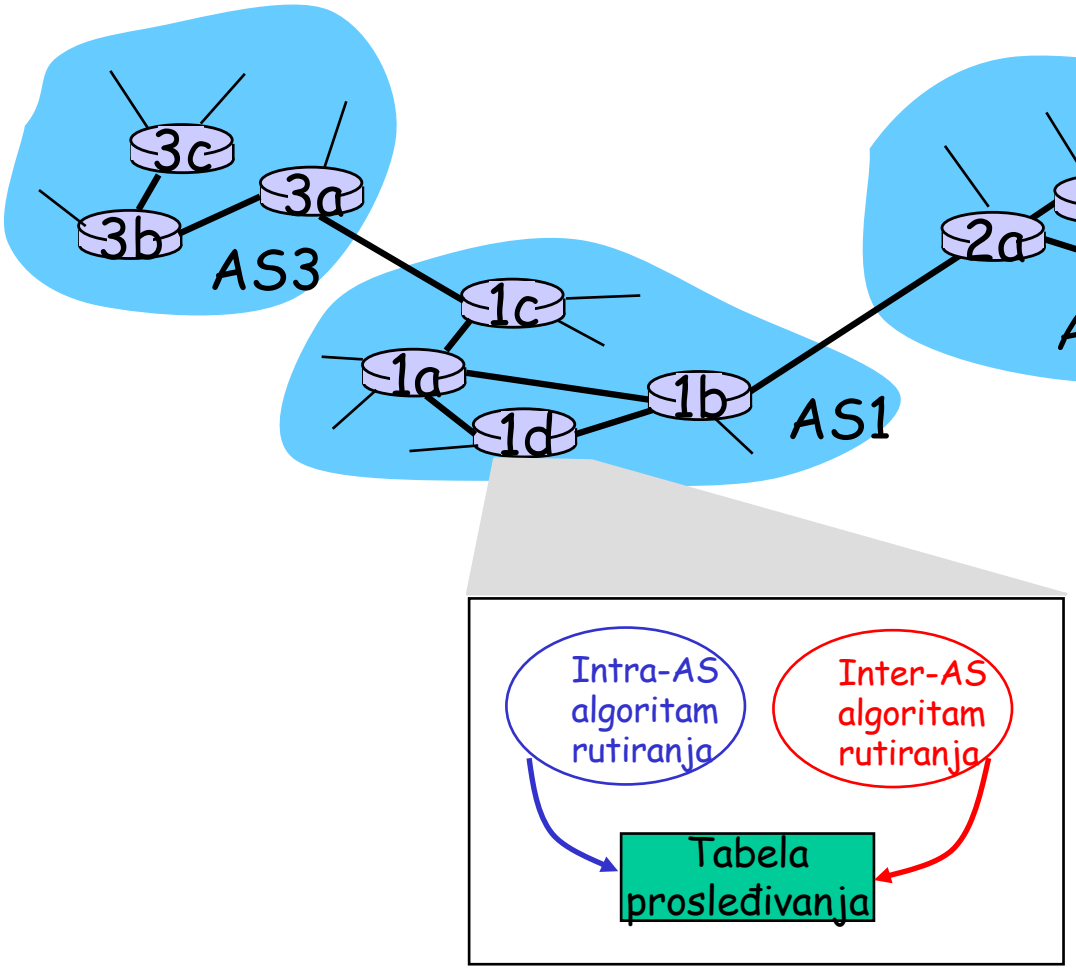
Internet pristup skalabilnom rutiranju

- grupiše rutere u regione, "autonomni sistemi" (AS)
- ruteri u istom AS izvršavaju isti protokol rutiranja
  - "intra-AS" protokol rutiranja se slično ponaša objašnjenim idealizovanim modelima
  - ruteri u različitim AS mogu izvršavati različite intra-AS protokole rutiranja

Gateway ruter

- Prosleđuje datagrame van AS

Međupovezivanje AS-ma



- Tabela prosleđivanja se konfiguriše i sa intra- i sa inter-AS algoritmom rutiranja
  - Intra-AS setuje sadržaje za interne destinacije
  - Inter-AS & Intra-AS setuju sadržaje za eksterne destinacije

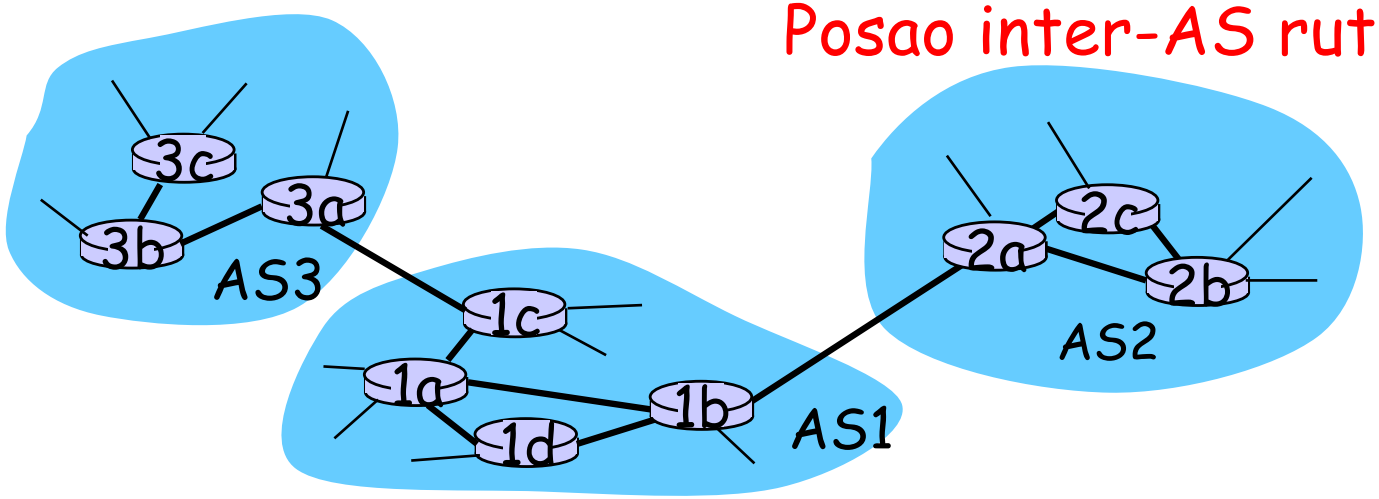
Inter-AS zadaci

- Pretpostavimo da ruter u AS1 primi datagram za koji je destinacija van AS1
  - Ruter bi trebao proslijediti paket prema *gateway* ruteru. Kojem?

Ruteri AS1 treba:

1. da nauče koje su destinacije dostižne preko AS2, a koje preko AS3
2. da proslijede tu informaciju o mogućnosti dosezanja do svih rutera u AS1

Posao inter-AS rutiranja!



Intra-AS Rutiranje

- Poznato kao Interior Gateway Protocols (IGP)
- Najpozantiji Intra-AS protokoli rutiranja:
  - RIP: Routing Information Protocol
  - OSPF: Open Shortest Path First
  - IGRP: Interior Gateway Routing Protocol (vlasništvo kompanije Cisco)
  - IS-IS: Intermediate system to intermediate system

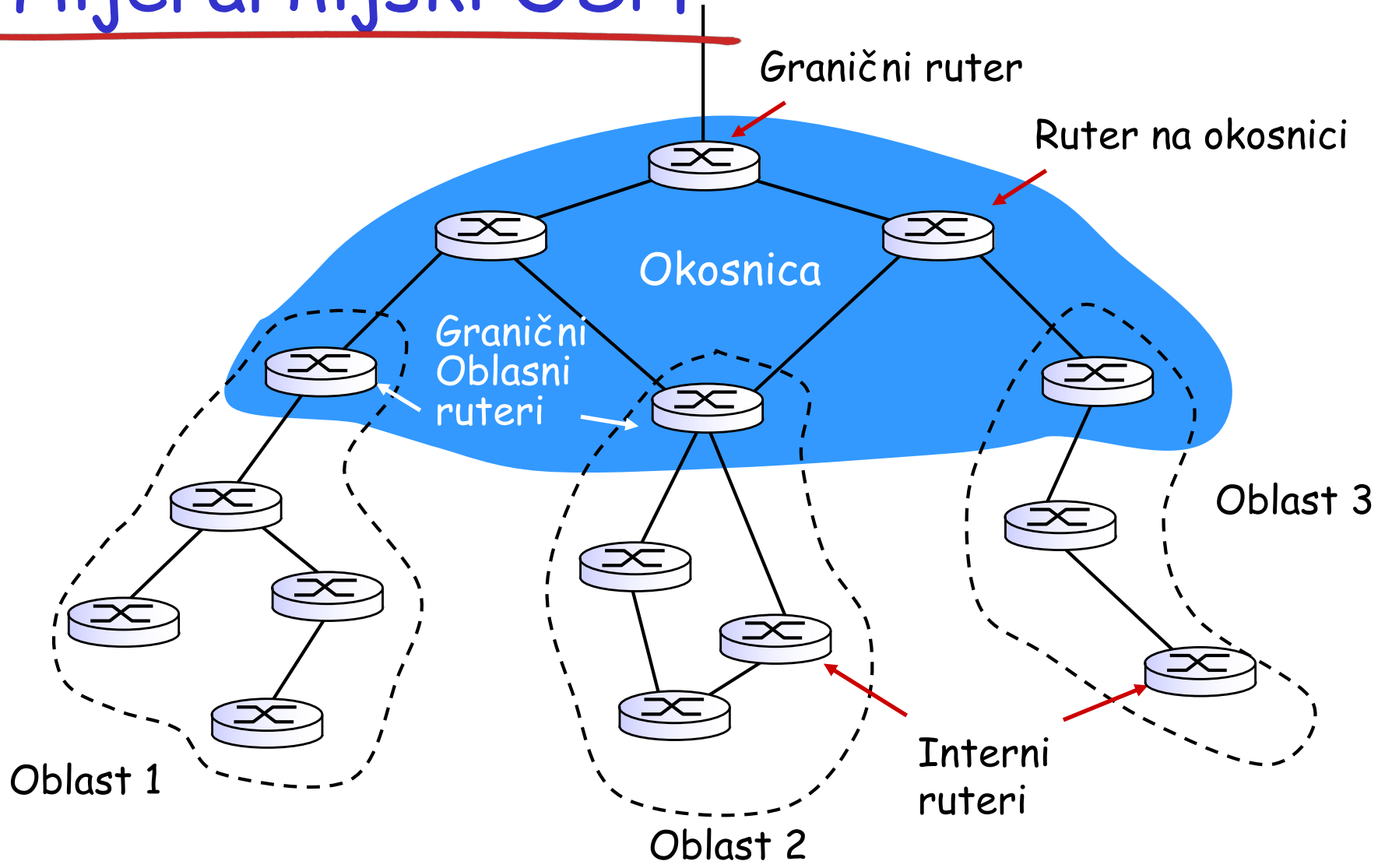
OSPF (Open Shortest Path First)

- Interior Gateway Protocol (IGP)
- "open": javno dostupan
- Verzija 2 (RFC 2328) iz 1998
- Verzija 3 (RFC2740) iz 1999 podržava IPv4 i IPv6
- Koristi se u velikim kompanijskim mrežama zbog brze konvergencije, rješavanja problema petlji i balansiranja saobraćaja, dok operatori koriste IS-IS koji je pogodan za stabilne mreže
- Koristi "Link State" algoritam
  - LS širenje paketa
  - Mapa topologije na svakom čvorištu
  - Proračun rute korišćenjem Dijkstra algoritma
  - Broadcast svakih 30min
- OSPF oglašavanja nose po jednu informaciju po susjednom ruteru
- Širenje oglašavanja preko čitavog AS ("flooding")
  - Nose se u OSPF porukama direktno preko IP (a ne preko TCP ili UDP) pri čemu potrebne kontrole obavlja OSPF
- Radi smanjenja saobraćaja može se koristiti koncept DR (*designated router*) i *multicast* tabele.

Napredne OSPF karakteristike (nema ih RIP)

- Sigurnost: za sve OSPF poruke se mora znati izvor (prevencija malicioznih aktivnosti) pri čemi se koriste lozinke ili MD5 kodiranje
- Više puteva sa istim troškovima je dozvoljeno (samo jedan put u RIP)
- Za svaki link, više metrika troškova za različiti TOS (npr., troškovi satelitskog linka su podešeni na "nisko" za *best effort*; visoko za servis u realnom vremenu)
- Integrisana uni- i multicast podrška:
  - *Multicast* OSPF (MOSPF) koristi istu bazu podataka o topologiji kao OSPF
- Hijerarhijski OSPF u velikim domenima.

Hijerarhijski OSPF



Hijerarhijski OSPF

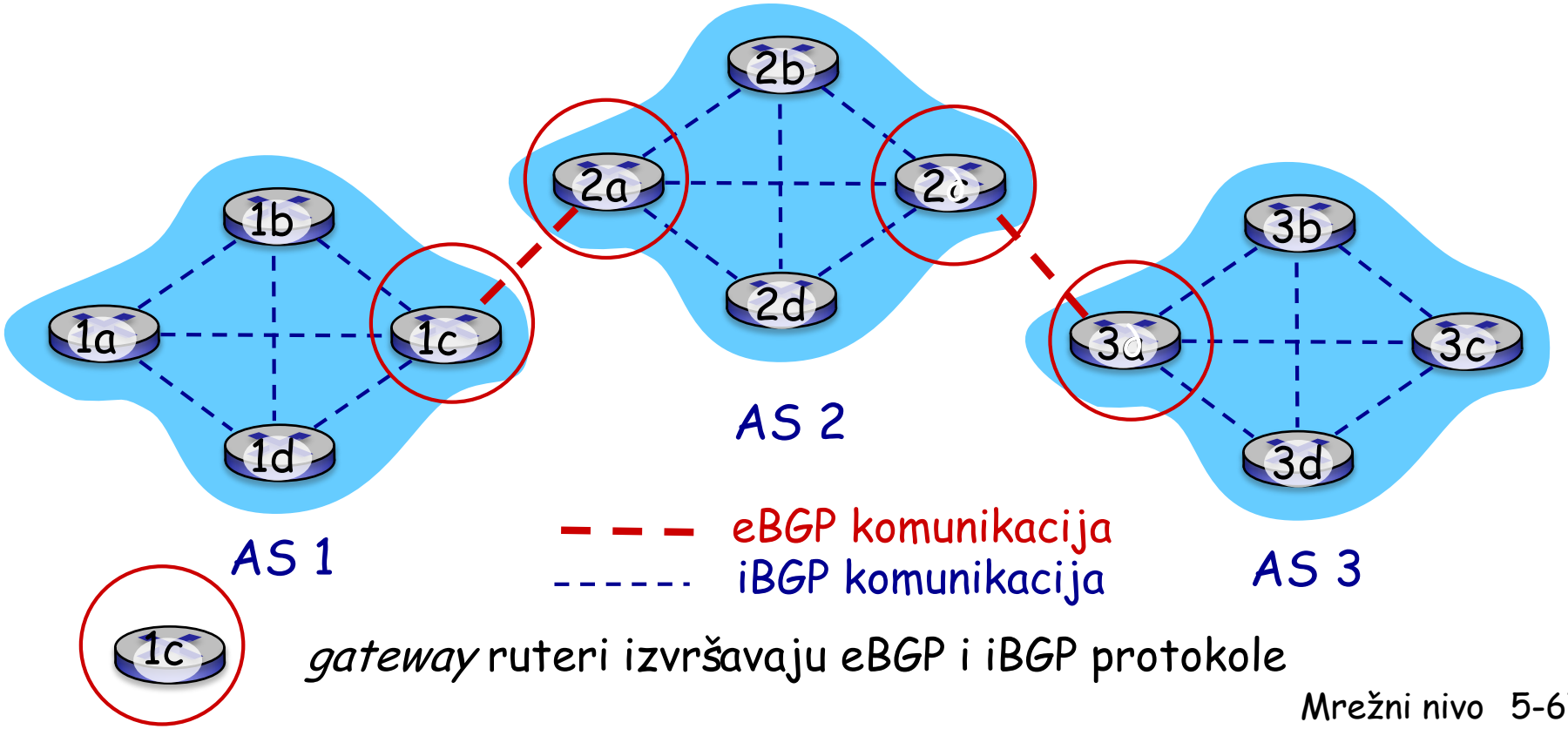
- Hijerarhija u dva nivoa: lokalna mreža (oblast) i okosnica.
  - Oglašavanja o stanju linka samo u lokalnoj mreži (oblasti)
  - Svako čvorište ima detaljnu topologiju mreže; samo poznaje najkraći put do mreža u drugim mrežama.
- **Ruter na granici lokalne mre□e:** "sumira" rastojanja do mreža u sopstvenoj zoni odgovornosti i to oglašava drugim ruterima na granicama lokalnih mreža.
- **Ruteri okosnice:** izvršavaju OSPF rutiranje samo na okosnici.
- **Grani□ni ruteri:** povezivanje na druge AS.

Internet inter-AS rutiranje: BGP

- BGP (Border Gateway Protocol): *de facto* standard
- Verzija 4 (RFC1771) iz 1994. je doživjela preko 20 korekcija, pri čemu je zadnja RFC4271 (iz 2006.)
- CIDR i agregacija ruta
- Naslijedio EGP čime je napravljena potpuna decentralizacija Interneta
- Mogu ga koristiti i kompanije kada OSPF nije dovoljno dobar i kada se radi o *multihomed* mreži (bolja redundansa).
- BGP omogućava svakom AS:
  1. Dobijanje informacije o dostižnosti sa susjednih AS-ova.
  2. Prosleđivanje prethodne informacije svim ruterima u okviru AS.
  3. Utvrđivanje "dobre" rute do podmreža baziranih na informaciji o dostižnosti i politici.
- Dozvoljava podmreži oglašavanje svog prisustva ostatku Interneta: *"Ovdje sam"*

BGP osnove

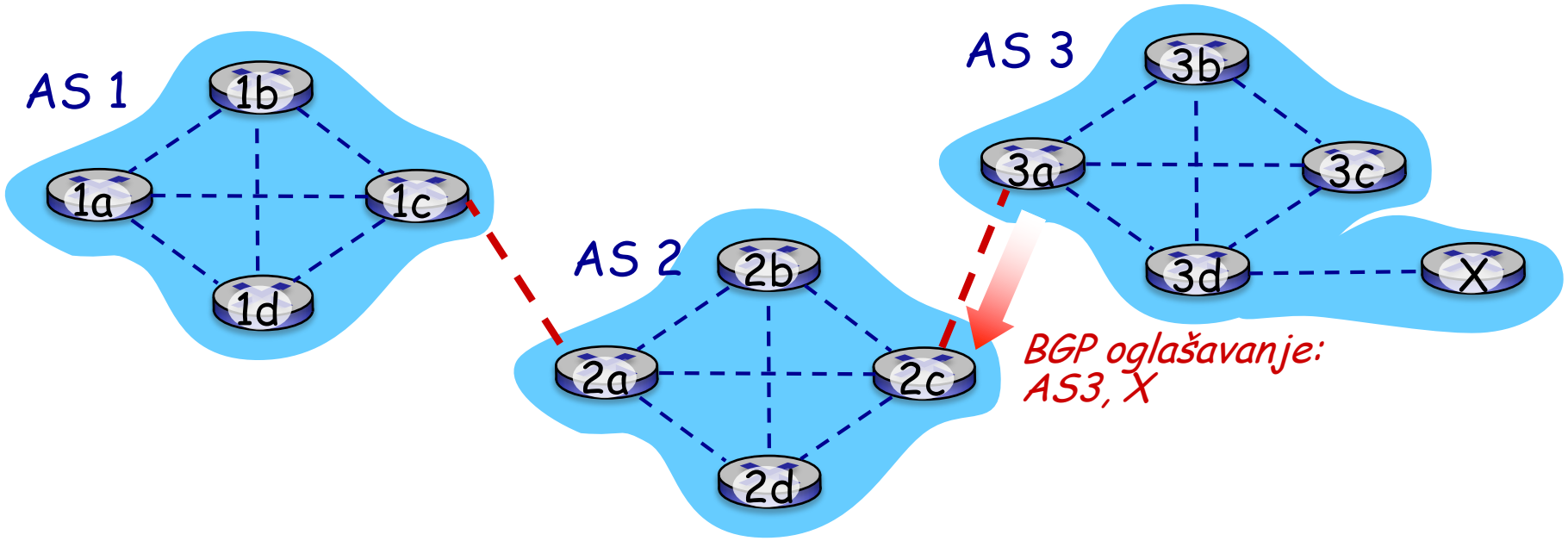
- Parovi rutera (BGP peer-ovi) razmjenjuju informaciju rutiranja preko semi-permanentne TCP konekcije (port 179): BGP sesije
- Svakih 60s šalje keep alive poruku
- Napomena: BGP sesije ne odgovaraju fizičkim linkovima.
- Kada AS2 oglasi prefiks do AS1, AS2 *obećava* da će proslijediti bilo koji datagram koji je adresiran do tog prefiksa preko sebe.
  - AS2 može agregirati prefikse u oglašavanjima



gateway ruteri izvršavaju eBGP i iBGP protokole

Distribuirana informacija o dostižnosti

- Sa eBGP sesijom između 3a i 1c, AS3 šalje informaciju o dostižnosti prefiksa do AS1.
- 1c može tada koristiti iBGP za distribuciju ove nove informacije o dostizanju prefiksa do svih rutera u AS1
- 1b može tada ponovo oglasiti novu informaciju o dostizanju do AS2 preko 1b-2a eBGP sesije
- Kada ruter stekne znanje o novom prefiksu, kreira sadržaj za taj prefiks u tabeli rutiranja.



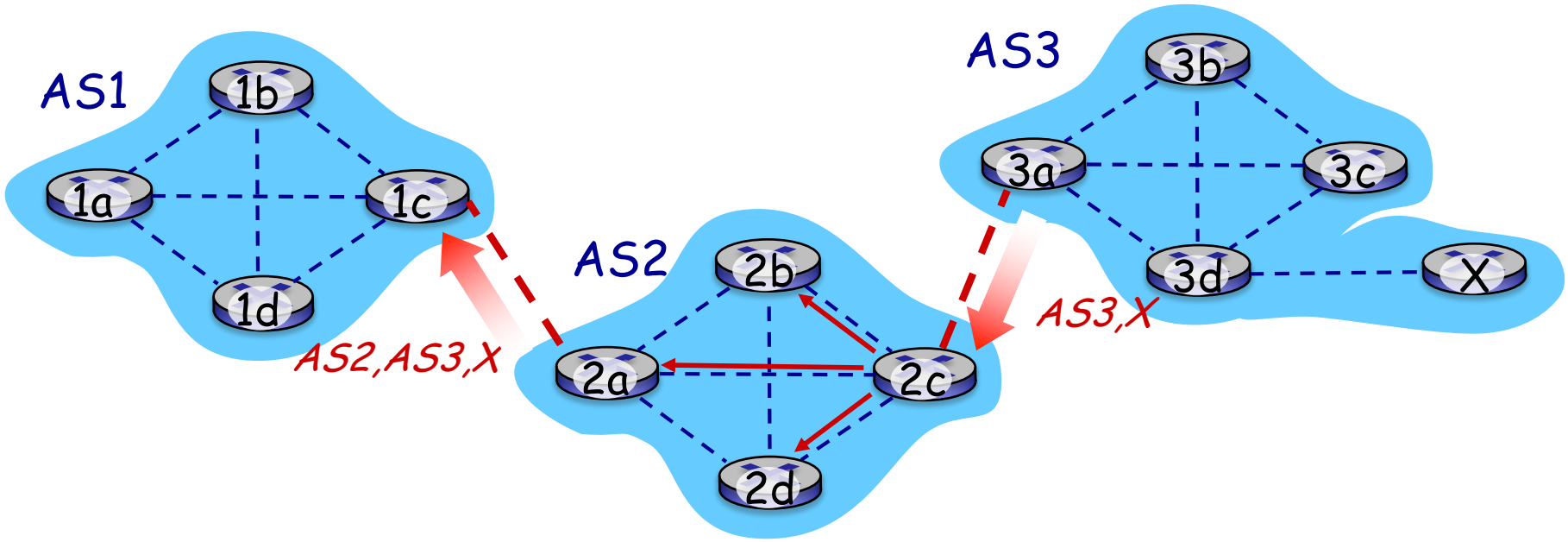
Atributi puta & BGP rute

- Kada oglašava prefiks, oglašavanje uključuje BGP atribute.
  - prefix + atributi = "ruta"
- Dva važna atributa:
  - AS-PATH: sadrži AS-ove preko kojih je oglašavanje prefiksa prošlo: AS67 AS17
  - NEXT-HOP: Indicira specifični interni-AS ruter do *next-hop* AS. (Može biti više linkova od trenutne AS do *next-hop* AS.)
- Kada *gateway* ruter primi oglašavanje rute, koristi politiku importovanja za potvrdu/odbijanje.

BGP izbor rute

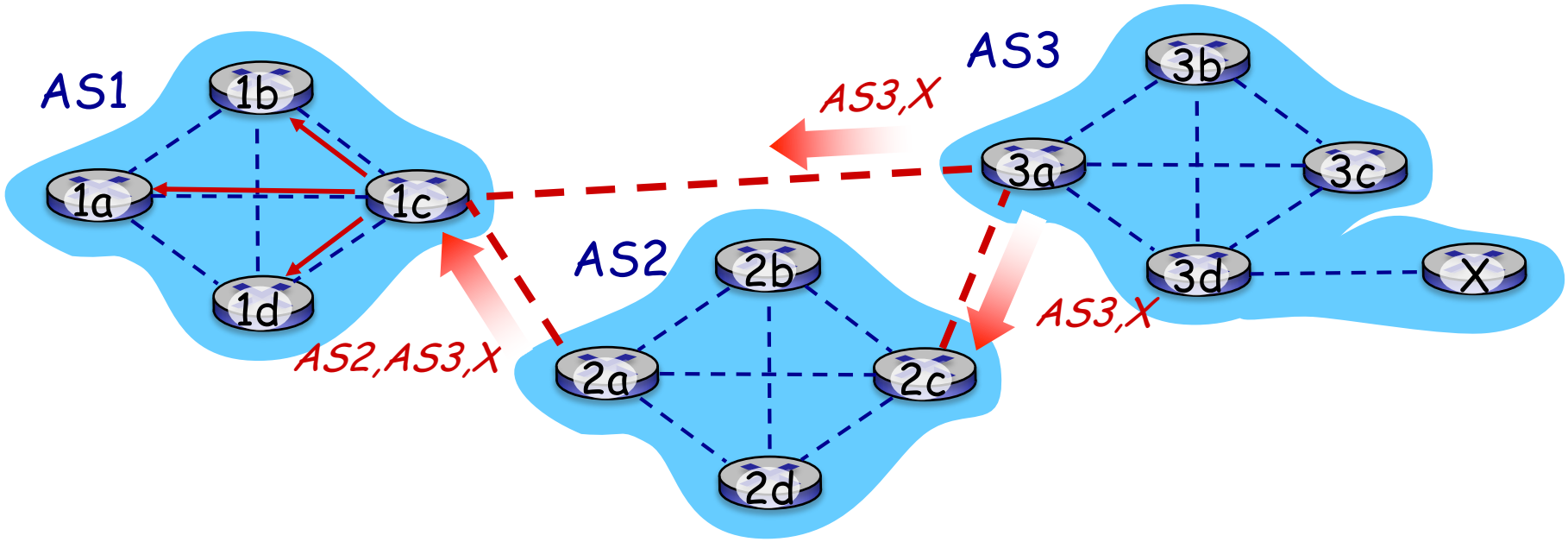
- Ruter može naučiti više od jedne rute do istog prefiksa. Ruter mora odabrati rutu.
- Pravila eliminacije:
  1. Vrijednost atributa lokalne reference: odluka politike
  2. Najkraći AS-PATH
  3. Najbliži NEXT-HOP ruter: "vrući krompir" rutiranje
  4. Dodatni kriterijum

BGP oglašavanje puta



- AS2 ruter 2c dobija oglašavanje puta AS3,X (preko eBGP) od AS3 rutera 3a
- Bazirano na AS2 politici, AS2 ruter 2c prihvata put AS3,X, prosleđuju (preko iBGP) do svih AS2 rutera
- Bazirano na AS2 politici, AS2 ruter 2a oglašava (preko eBGP) put AS2, AS3, X do AS1 rutera 1c

BGP oglašavanja ruta



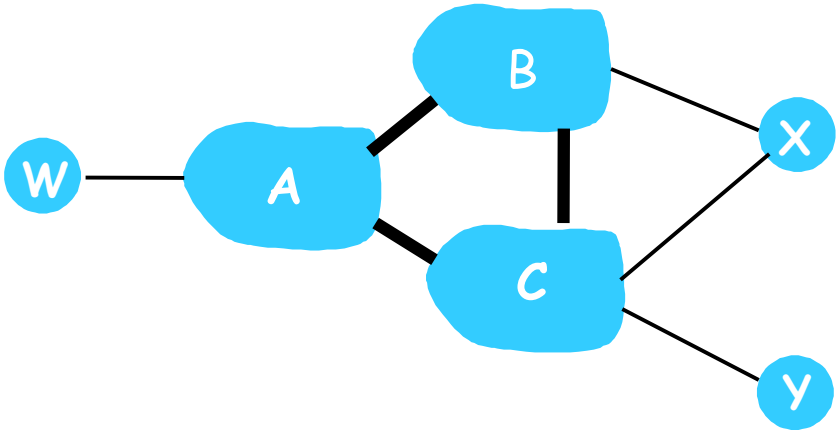
gateway ruter može naučiti više puteva do destinacija:

- AS1 gateway ruter 1c uči put *AS2,AS3,X* od 2a
- AS1 gateway ruter 1c uči put *AS3,X* od 3a
- Bazirano na politici, AS1 gateway ruter 1c bira put *AS3,X, i oglašava put kroz AS1 preko iBGP*

BGP poruke

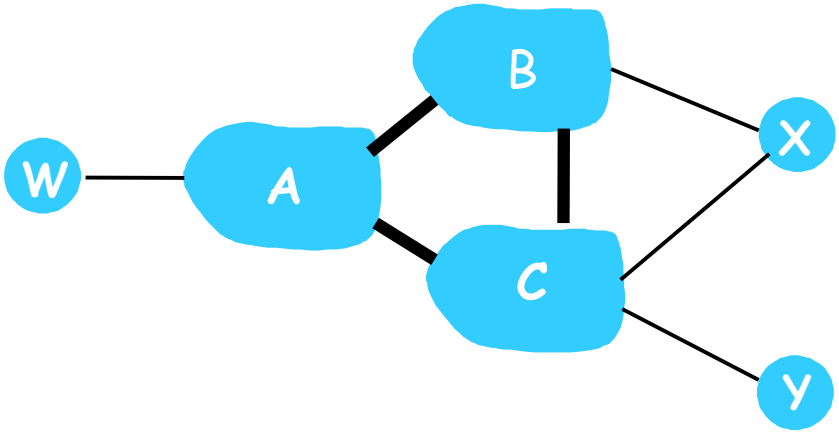
- BGP poruke se razmjenjuju preko TCP.
- BGP poruke:
  - OPEN: otvara TCP vezu sa peer i vrši identifikaciju pošiljaoca
  - UPDATE: oglašava novi put (ili odbacuje stari)
  - KEEPALIVE održava vezu u odsustvu UPDATE poruka, a potvrđuje i OPEN zahtjev
  - NOTIFICATION: izvještava o greškama u prethodnoj poruci, a takođe se koristi za raskidanje veze

BGP politika rutiranja



- A,B,C su mreže provajdera
- x,w,y su korisnici (mreža provajdera)
- x je *dual-homed*: povezan na dvije mreže
  - x ne želi da se saobraćaj rutira od B preko x do C
  - .. tako x neće oglašavati B rutu do C

BGP: kontroliše ko rutira do tebe



- A oglašava B put Aw
- B oglašava X put BAw
- Da li će B oglašavati C put BAw?
  - Neće! B ne dobija "profit" za rutiranje CBAw pošto w i C nisu B-ovi korisnici
  - B želi da prinudi C da rutira do w preko A
  - B želi da rutira *samo* do/od njegovih korisnika!

Zašto različito Intra- i Inter-AS rutiranje ?

Politika:

- Inter-AS: administrator želi kontrolu nad načinom rutiranja saobraćaja i time ko rutira kroz njegovu mrežu.
- Intra-AS: jedan administrator, nema potrebe za političkim odlukama

Veličina:

- hijerarhijsko rutiranje čuva veličinu tabele, smanjuje saobraćaj koji se odnosi na ažuriranje

Performanse:

- Intra-AS se može fokusirati na performanse
- Inter-AS politika može dominirati u odnosu na performanse

Glava 5: Mrežni nivo

5. 1 Uvod

5.2 IP (Internet Protocol)

- Format datagrama
- IP adresiranje

5.3 Rutiranje

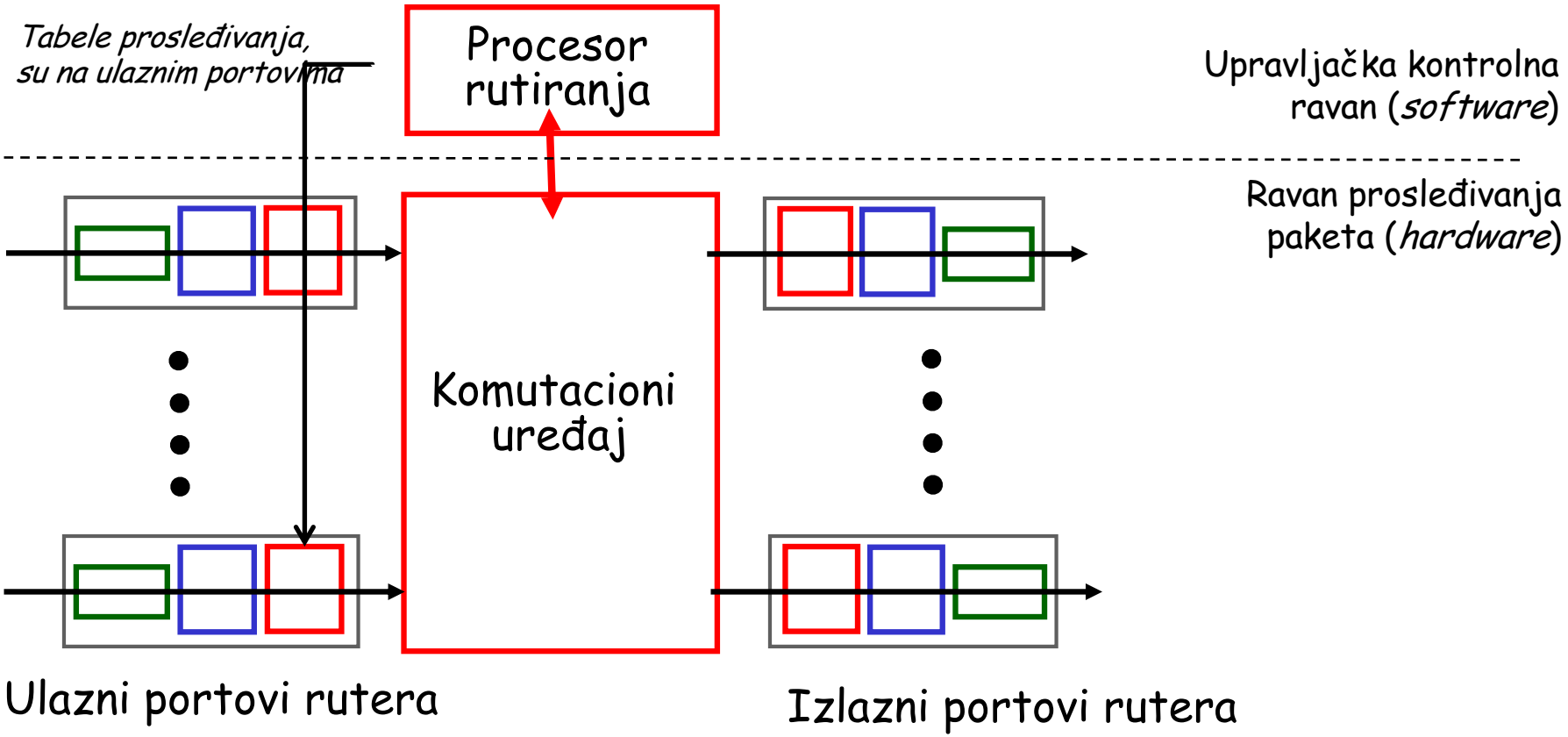
- *Link state*
- *Distance Vector*
- Hijerarhijsko rutiranje
- Protokoli rutiranja

5.4 Ruter

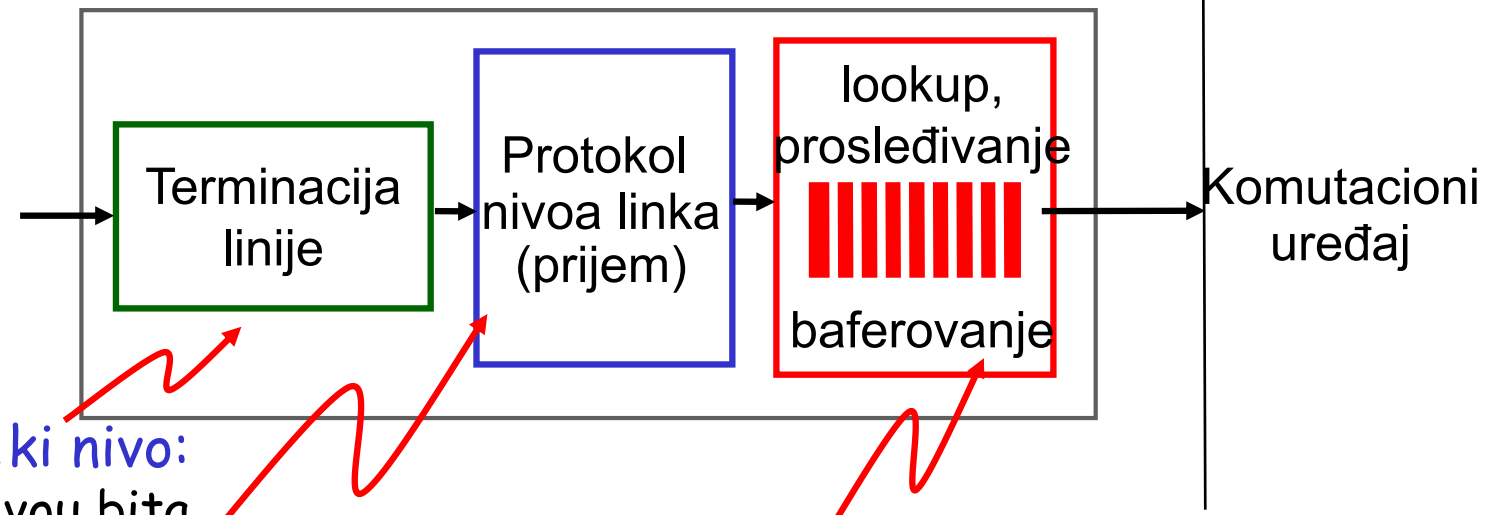
Pregled arhitekture rutera

Dvije ključne funkcije rutera:

- Izvršava algoritme/protokole rutiranja (RIP, OSPF, BGP)
- *Prosleđuje (komutira)* datagrame sa ulaznog na izlazni link



Funkcije ulaznog porta



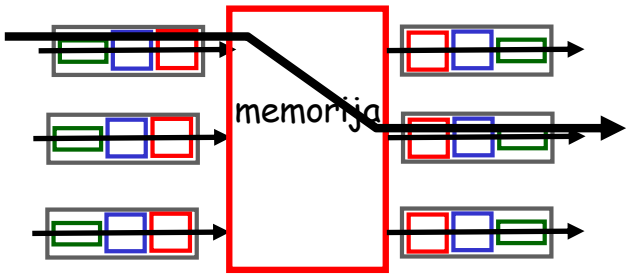
Fizički nivo:
Prijem na nivou bita

Nivo linka:
npr., Ethernet

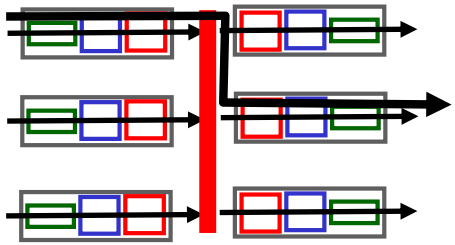
Decentralizovana komutacija:

- Na osnovu destinacione adrese datagrama, traži izlazni port korišćenjem tabele rutiranja u memoriji ulaznog porta
- Cilj: kompletirati obradu na ulaznom portu u skladu sa brzinom na linku
- Red čekanja (bafer): ako datagrami pristižu brže nego što je brzina prosleđivanja u komutacionom uređaju

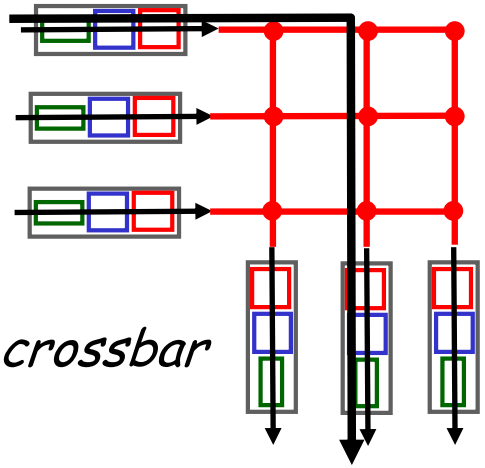
Tri tipa komutacionih uređaja



memorija



magistrala

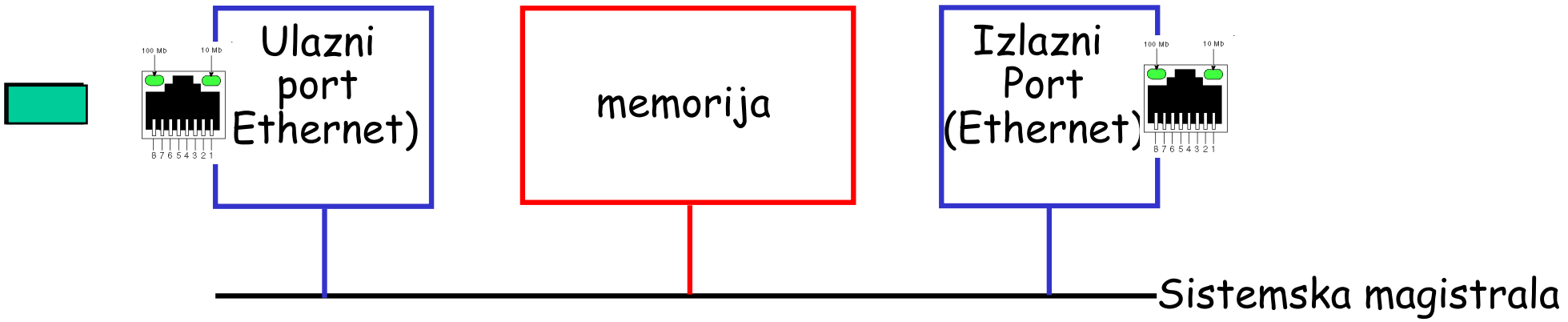


crossbar

Komutacija preko zajedničke memorije

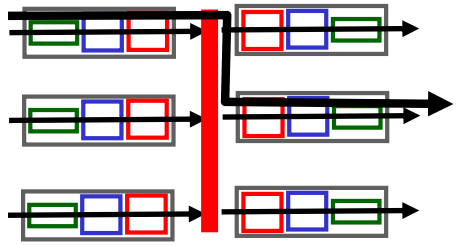
Prva generacija rutera:

- tradicionalni računari sa komutacijom pod direktnom kontrolom CPU
- paketi se smještaju u memoriju sistema
- brzina ograničena brzinom memorije (svaki datagram se mora dva puta prenijeti preko magistrale)
- Cisco Catalyst *switch* serije 8500 (specifično rješenje)



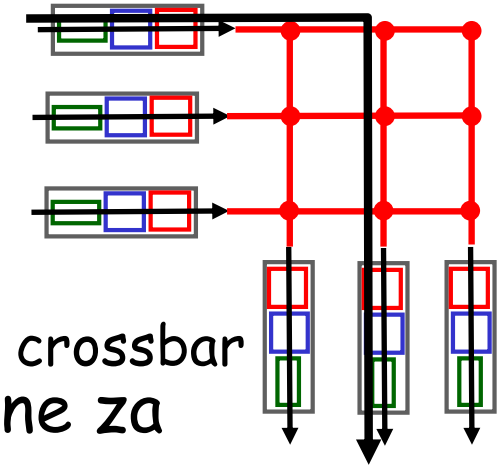
Komutacija preko zajedničke magistrale

- Datagram se sa memorije ulaznog porta do memorije izlaznog porta prenosi preko zajedničke magistrale bez učešća procesora
- Kolizija na magistrali: brzina komutacije je ograničena kapacitetom magistrale
- Cisco 5600:
  - 32 Gb/s magistrala,
  - pristupni i kompanijski ruteri (neregionalne ili na okosnici)



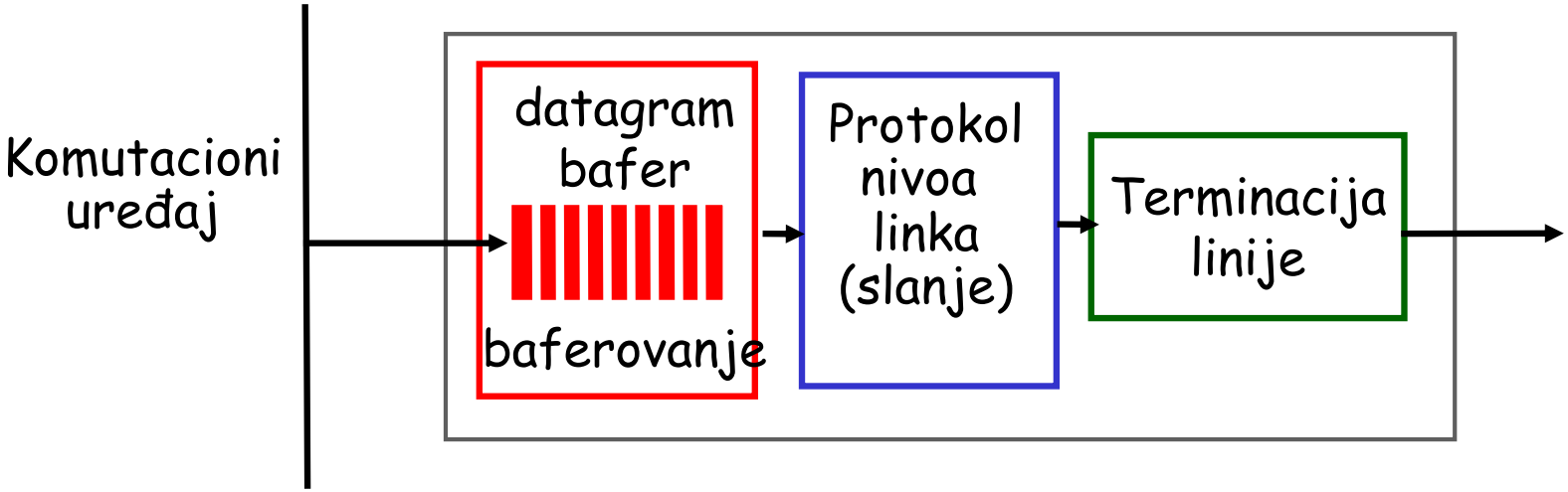
magistrala

Prostorni komutatori



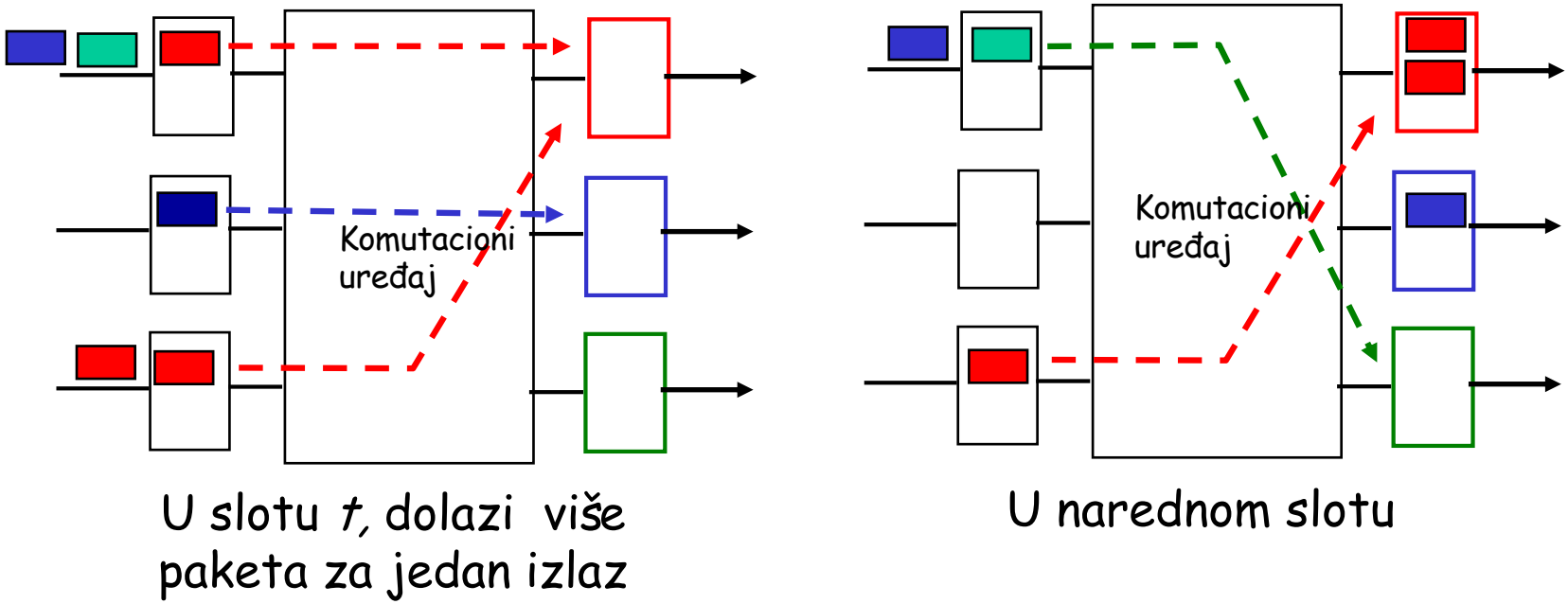
- Komutacione strukture su inicijalno razvijene za povezivanje procesora u multiprocesorsku aritekturu
- Prevazilazi ograničenja kapaciteta magistrale
- Nudi više puteva između skupa ulaza i skupa izlaza
- Crossbar topologija
- Napredan dizajn: fragmentacija datagrama u ćelije fiksne dužine, komutiranje ćelija kroz uređaj.
- Cisco 12000:
  - komutira do 60Gb/s kroz komutacionu matricu
- Banyan, Clos, paralelni...
- Komutiraju pakete fiksne dužine

Izlazni portovi



- *Baferovanje* se zahtijeva kada datagrami stižu iz uređaja većom brzinom nego što je brzina prenosa datagrama
- *Disciplina raspoređivanja (Scheduling)* bira za prenos datagrame u redovima čekanja

Izlazno baferovanje



U slotu *t*, dolazi više paketa za jedan izlaz

U narednom slotu

- Baferovanje kada je dolazna brzina veća od odlazne brzine
- *baferovanje (kašnjenje) i gubici zbog zauzetosti bafera na izlaznom portu!*

Veličina bafera?

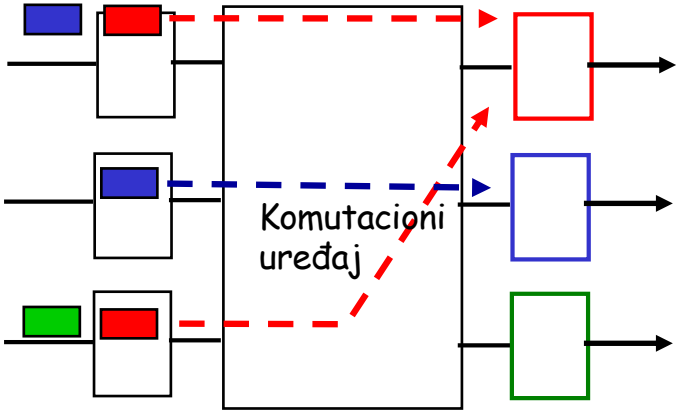
- RFC 3439 (*rule of thumb*) pravilo: srednja veličina bafera je jednaka "prosječno" RTT (npr. 250ms) pomnoženo sa kapacitetom linka *C*

$$RTT \cdot C$$

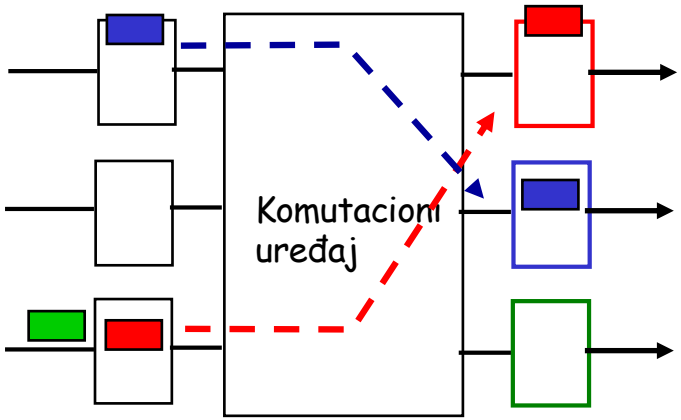
  - npr., *C* = 10Gb/s link: 298MB
- Neke preporuke ukazuju da su moguće i manje memorije: za *N* tokova, potrebna veličina bafera je $\frac{RTT \cdot C}{\sqrt{N}}$

Red čekanja na ulaznom portu

- Uređaj je sporiji od ulaznih portova -> redovi čekanja se mogu pojaviti na ulazima
- *Head-of-the-Line* (HOL) blokiranje: smještanje datagrama u ulazne redove čekanja sprečava druge datagrame u redovima čekanja da se prosleđuju dalje
- *Kašnjenje u redovima čekanja i kašnjenje zbog prepunog ulaznog bafera!*



Izlazna kolizija u trenutku t (samo jedan crveni paket može biti proslijeđen)



Zeleni paket doživljava HOL blokiranje