Računarske mreže

Protokoli transportnog sloja

Uvod

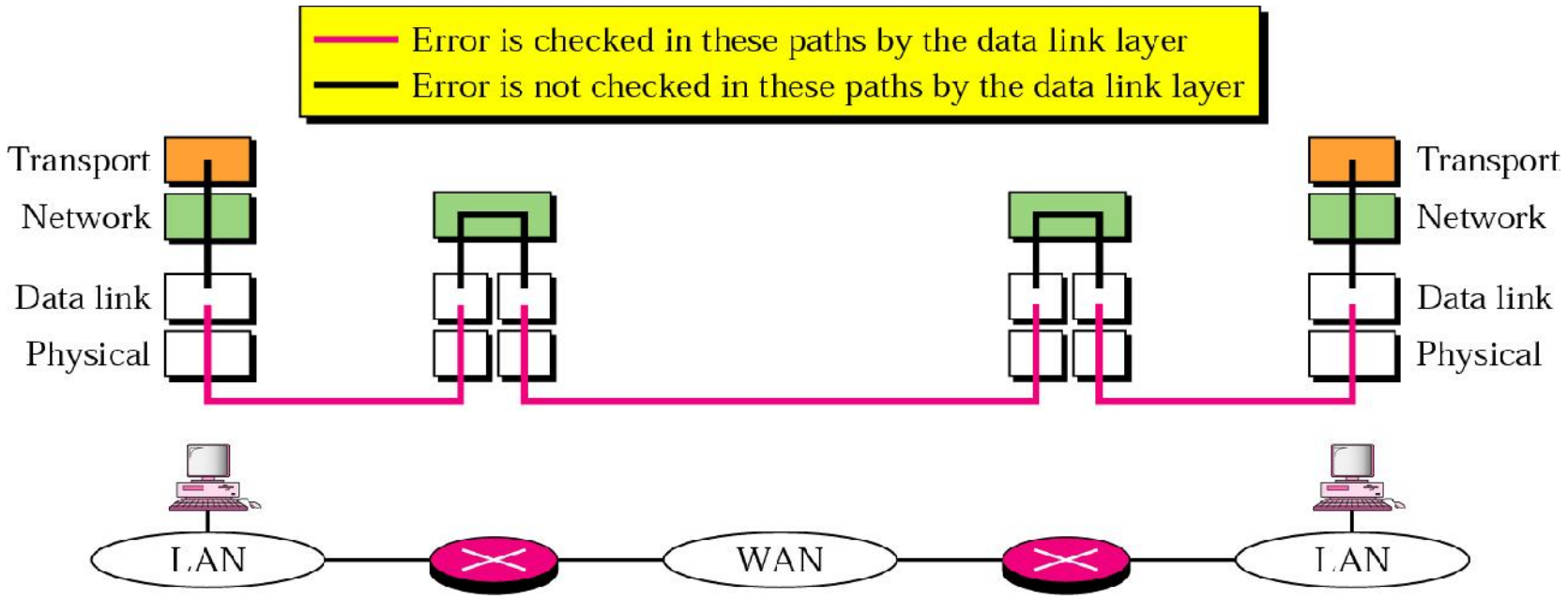
- Internet ima dva glavna protokola u transportnom sloju
  - UDP:
    - radi bez uspostavljanja veze
    - segmenti nisu numerisani, mogu da kasne i da stižu preko reda
  - TCP:
    - protokol sa uspostavljanjem veze pre prenosa podataka

Uvod

- Pouzdani i nepouzdani servisi
- Pouzdani servisi :
  - Kontrola greške (*error control*)
  - Kontrola toka (*flow control*)
  - Kontrola zagušenja
  - TCP
- Nepouzdani servisi
  - UDP

Uvod

- Da li je potreban piouzdan transportni servis, ako je na sloju veze podataka primenjena pouzdana usluga?



UDP

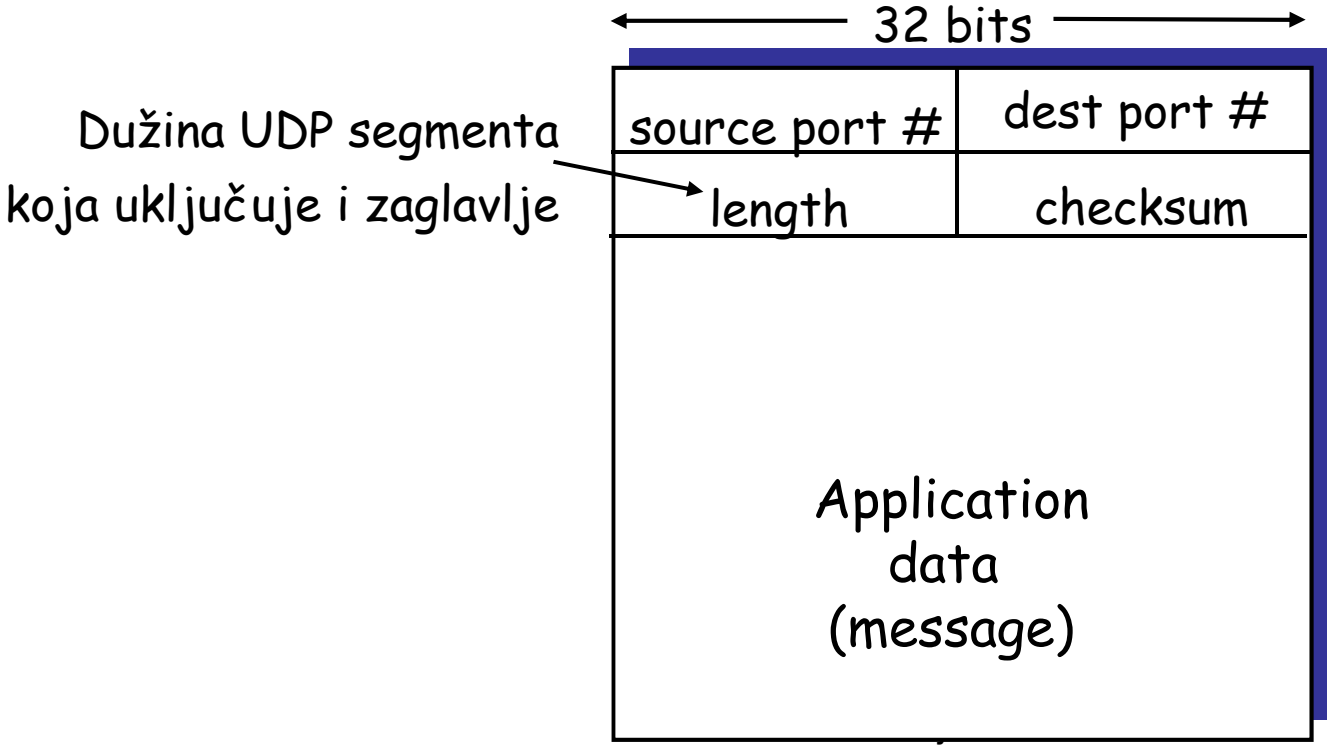
- UDP – *User Datagram Protocol*
- Nepouzdani protokol, bez uspostavljanja direktne veze
  - Omogućava aplikacijama da šalju kapsulirane IP datagrame za koje ne moraju prethodno da uspostavljaju vezu
- Opisan je u dokumentu RFC 768

UDP

- Potrebe za ovim servisom
  - Mali (jednostavan) header
  - Pogodan za prenos kratkih poruka
  - Ne zahteva preveliku saradnju pošiljaoca i primaoca
  - Omogućava multipleksiranje/demultipleksiranje
    - *Multicasting*
  - Prenos informacija o rutiranju (RIP)
  - Prenos podataka u realnom vremenu
    - *Real Time Transport Protocol* (RTP)

UDP

- UDP prenosi segmente koji se sastoje od 8-bajtnog zaglavlja i korisničkih podataka



UDP

- Portovi identifikuju dva kraja veze
  - Od 0 do 65535
- Koristan sadržaj UDP segmenta se predaje procesu koji je pridružen odredišnom portu
- Izvorišni port se koristi za slanje odgovora
  - oznaka procesa na izvorištu kome se odgovara
- Kontrolni zbir UDP paketa
  - Nije obavezan – prikazuje se nulom kada nije izračunat
    - Isključuje se npr. kod digitalizovanog govora – povećava se kvalitet kodovanih podataka
  - Ako je *chacksum* izračunat kao 0 prikazuje se svim 1

UDP

- Šta UDP ne radi:
  - nema uspostavljanja/raskidanja veze
  - nema potvrde prijema
  - nema ispravljanja redosleda
  - ne upravlja tokom i zagušenjem
  - ne kontroliše greške
  - ne vrši retransmisiju izgubljenih segmenata
- Sve prepušta korisničkim aplikacijama
- UDP je koristan za klijent-server arhitekturu
  - Klijent šalje kratak zahtev i očekuje kratak odgovor
  - Ako se UDP segment izgubi, aktiviraće se tajmer koji obaveštava aplikaciju da ga pošalje ponovo
  - Pojednostavljuje programski kod (potrebno je manje poruka nego kod prethodnog uspostavljanja veze)

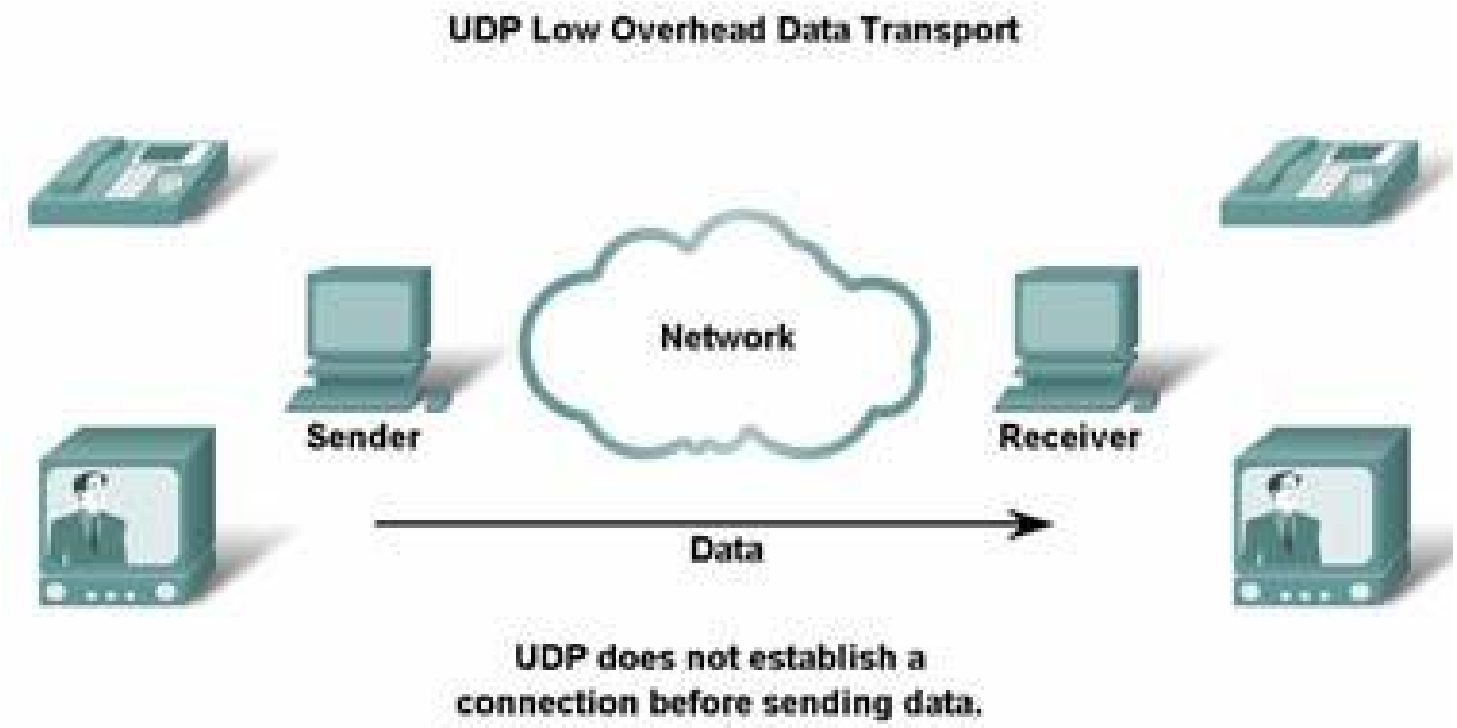
UDP

- Karakteristike UDP protokola
- – *Connectionless*
  - ne uspostavlja konekciju pre slanja podataka
  - svaki datagram se nezavisno prenosi
- *Unreliability – nepouzdan*
  - izgubljeni ili oštećeni paketi (datagram) se ne mogu retransmitovati
  - nema kontrole i provere redosleda pristizanja paketa
  - ako je potrebno, aplikacija koja koristi UDP mora da implementira ove funkcije
- *Low overhead*
  - brz, jednostavan, ne zahteva velike mrežne i procesorske resurse

UDP primena

- Mnoge aplikacije ne zahtevaju pouzdanost prenos i UDP funkcije su dovoljne (npr. periodičan prenos manje količinu podataka)
- Primeri
  - DNS – *Domain Name System*
  - SNMP – *Simple Network Management Protocol*
  - DHCP – *Dynamic Host Configuration Protocol*
  - RIP–*Routing Information Protocol*
  - TFTP – *Trivial File Transfer Protocol*
- Nekim aplikacijama bi TCP predstavljao problem
  - veliko zaglavlje, a podaci se kontinualno prenose u malim količinama
  - kontrola greške, retransmisija, dinamički prozor mogu da uspore podatke i degradiraju prenos
    - *Real-Time saobraćaj*
    - VoIP–*Voice Over IP*
    - *Video Streaming*

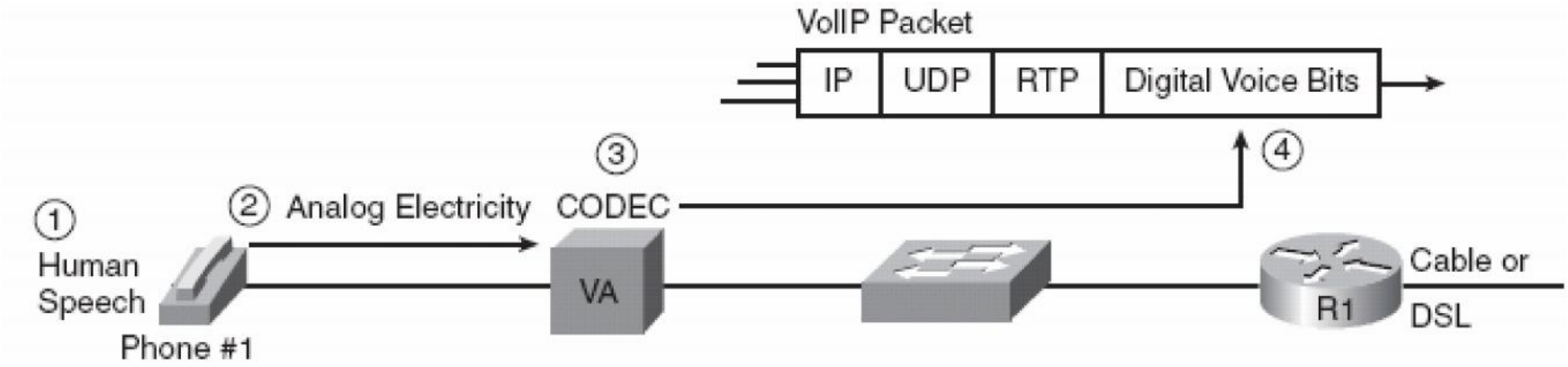
UDP primena


UDP Low Overhead Data Transport
Network
Sender
Receiver
Data
UDP does not establish a
connection before sending data.

UDP saobraćaj

- Primer: VoIP, IP Telefonija, IP TV – interaktivan prenos zvuka i slike
- zahtevi
  - malo kašnjenje (*delay)* – do 200 ms za interaktivan razgovor
  - mala varijacija kašnjenja - džiter (*jitter) +/- 30 ms*
  - manji gubitak paketa se može tolerisati
    - ako se izgubi neki paket, retransmisija bi narušila kašnjenje i džiter
    - povremeni gubitak nekog paketa nije primetna za učesnike u komunikaciji - kvalitet se nenarušava značajno

UDP saobraćaj



2	2	4	4	4 bits	6 bits	6 bits	2	2	2	3	1
Source Port	Dest. Port	Sequence Number	Ack. Number	Offset	Reserved	Flags	Window Size	Checksum	Urgent	Options	PAD

TCP Header

2	2	2	2
Source Port	Dest. Port	Length	Checksum

UDP Header

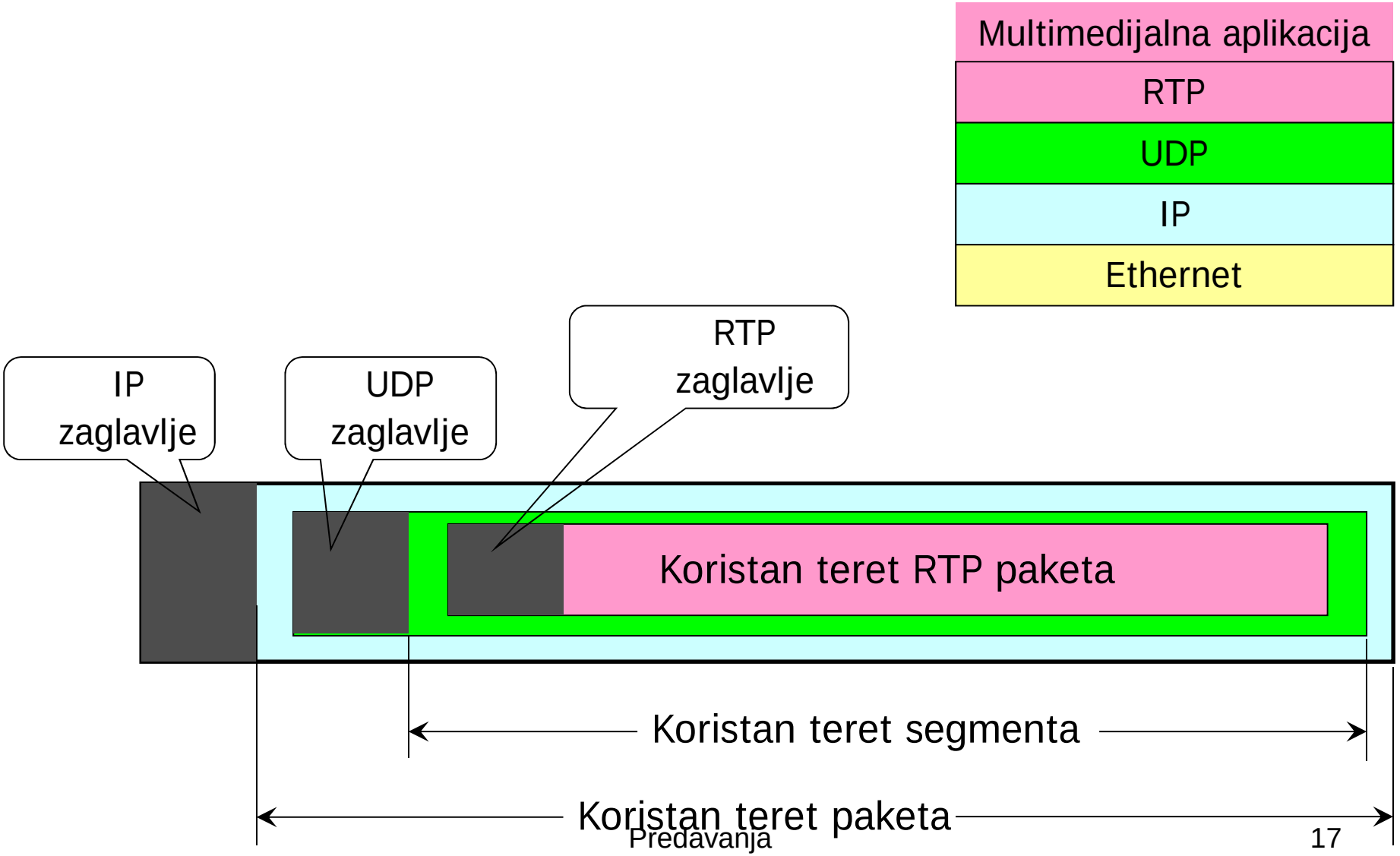
UDP

- UDP je interfejs ka IP protokolu
- Primena npr. kod DNS-a
  - Sistem imena domena (*Domain Name System*)
- Npr. program koji treba da pronađe IP adresu za www.singidunum.ac.rs
  - Šalje se UDP paket sa logičkim imenom DNS serveru
  - Server odgovara UDP paketom sa IP adresom
    - Preko mreže su razmenjene dve jednostavne poruke
    - Nema potrebe za prethodnim uspostavljanjem veze

UDP i on-line multimedijske aplikacije

- RTP protokol (*Real Time Protocol*)
- Multimedijske aplikacije: Internet radio, IP telefonija, Video konferencije, muzika i video na zahtev itd.
  - Sastoje se iz više tokova
- RTP u realnom vremenu multipleksira različite tokove i kodira ih u jedinstven tok UDP paketa
- RTP transportni protokol je ugrađen u sloj aplikacije

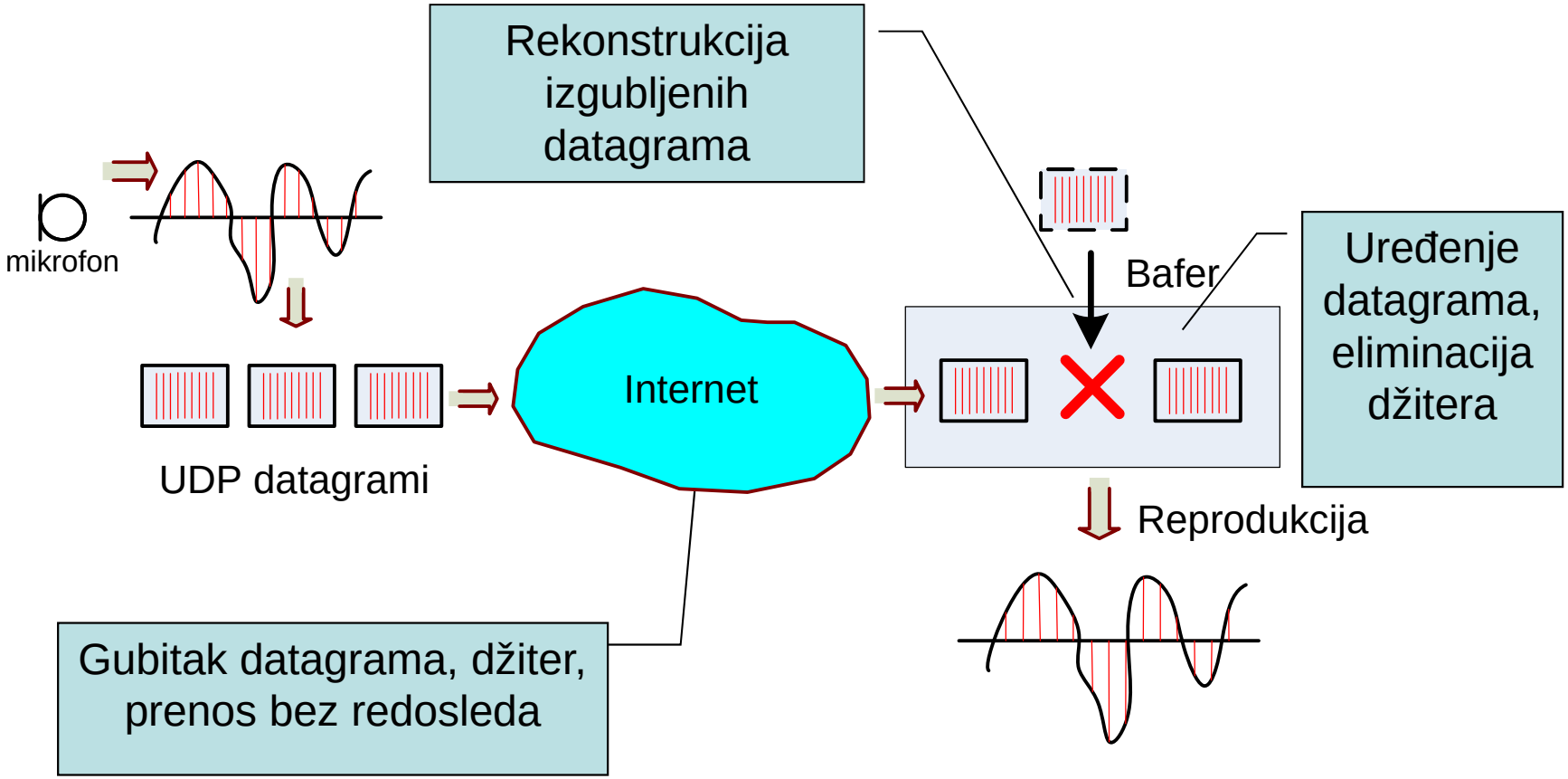
UDP i on-line multimedijske aplikacije



RTP i UDP

- Svaki RTP paket u UDP toku ima redni broj
- Ako neki paket nedostaje na odredištu on se interpolira
- Kod RTP nema:
  - upravljanja tokom, potvrđivanja paketa, retransmisije paketa
- RTP paket sadrži način kodiranja
  - Način kodiranja se može menjati vremenom
- RTP sadrži vremensko označavanje:
  - Ublažavanje efekta neravnomernosti stizanja paketa (džiter)
  - Sinhronizacija istovremenih tokova (slika i zvuk)

RTP i UDP


Rekonstrukcija izgubljenih datagrama
mikrofon
Bafer
Uređenje datagrama, eliminacija džitera
Internet
UDP datagrami
Reprodukcija
Gubitak datagrama, džiter, prenos bez redosleda

TCP

- TCP – *Transmission Control protocol*
- Protokol sa uspostavljanjem direktne veze (tzv. Protokol za upravljanje prenosom)
- Obezbeđuje pouzdan tok bajtova s kraja na kraj veze kroz nepouzdanu raznovrsnu mrežu
  - Različite topologije, propusni opsezi, kašnjenja, veličine paketa i sl.
- Projektovan je tako da se može dinamički prilagoditi promenljivim karakteristikama Interneta
- Održi pouzdanu vezu čak i u slučajevima pojave raznih vrsta otkaza u mrežnoj infrastrukturi
- Definisan je u dokumentu RFC 793, ispravke grešaka u RFC 1122, dopune u RFC 1323

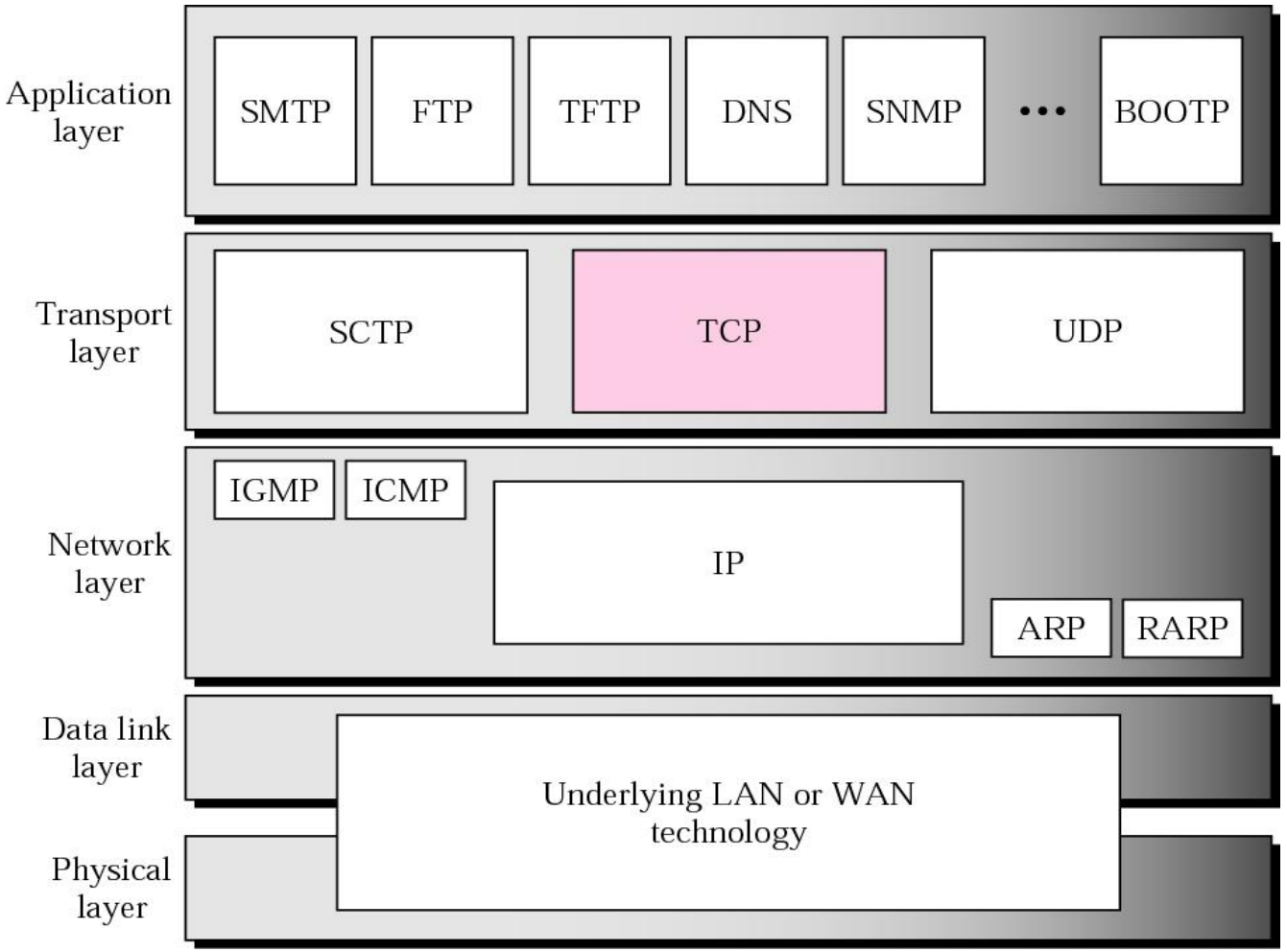
TCP

- Funkcije:
  - segmentacija podataka
  - upravljanje vezom
  - potvrda prijema
  - ispravljanje redosleda
  - upravljanje tokom i zagušenjem

TCP

- TCP prihvata tokove korisničkih podataka i deli ih u segmente
  - Max. veličina segmenta je 64 KB, a najčešće je to 1460 bajtova zbog Ethernet okvira
- Segmenti se šalju kao zasebni IP datagrami
- TCP obezbeđuje pouzdanost
  - TCP mora da brine o tome da datagram bude ispravno isporučen (IP sloj ne brine o tome)
  - Datagrami mogu stići preko reda – TCP treba da ih ispravno složi

TCP


Application layer
SMTP
FTP
TFTP
DNS
SNMP
• • •
BOOTP
Transport layer
SCTP
TCP
UDP
Network layer
IGMP
ICMP
IP
ARP
RARP
Data link layer
Underlying LAN or WAN technology
Physical layer

Model TCP Usluge

- Usluge koje TCP pruža aplikacijama
  - Proces-proces komunikacija
  - Orijentacija na tok (prenos toka podataka, a ne pojedinačnih poruka)
  - Prenos podataka u punom dupleksu
  - Konekcioni servis (uspostavljanje veze, prenos podataka, raskidanje veze)
  - Pouzdani servis (pouzdan prenos podataka je odgovornost TCP-ja, a ne aplikacije)

Karakteristike TCP Usluge

- TCP veze su u punom dupleksu, tipa od tačke do tačke
- TCP ne podržava višesmerno niti neusmereno (difuzno) emitovanje
- Kada TCP primi podatke od aplikacije, on može da čeka da se sakupi dovoljno podataka (efikasnost prenosa)
- Nekada aplikacije zahtevaju da se uneseni podaci moraju odmah slati
  - Npr. prijavljivanje na računar

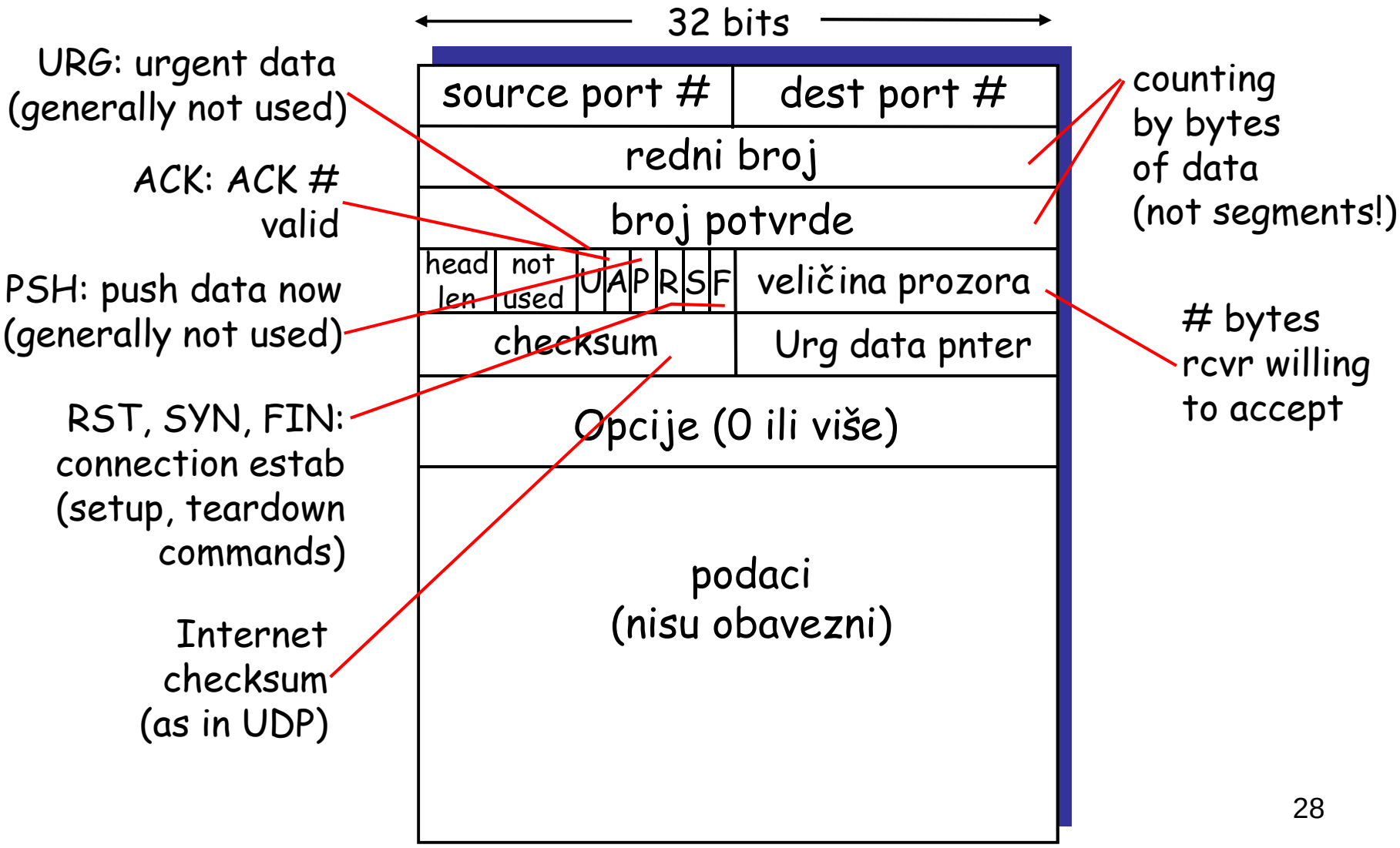
Karakteristike TCP Usluge

- Hitno slanje podataka
- U slučajevima kada korisnik interaktivno radi sa udaljenom aplikacijom
  - Npr. <Ctrl><C> - za prekid započete aplikacije
  - Na predaji se ovakav zahtev odmah prosleđuje (nema daljeg skladištenja)

TCP protokol

- Pretpostavka za TCP
  - Svaki bajt na TCP vezi ima svoj 32-bitni redni broj
- Dve transportne jedinice razmenju podatke u obliku segmenata
- Segment sadrži
  - 20-bajtno zaglavlje
  - Podatke (kojih i ne mora biti)

TCP protokol



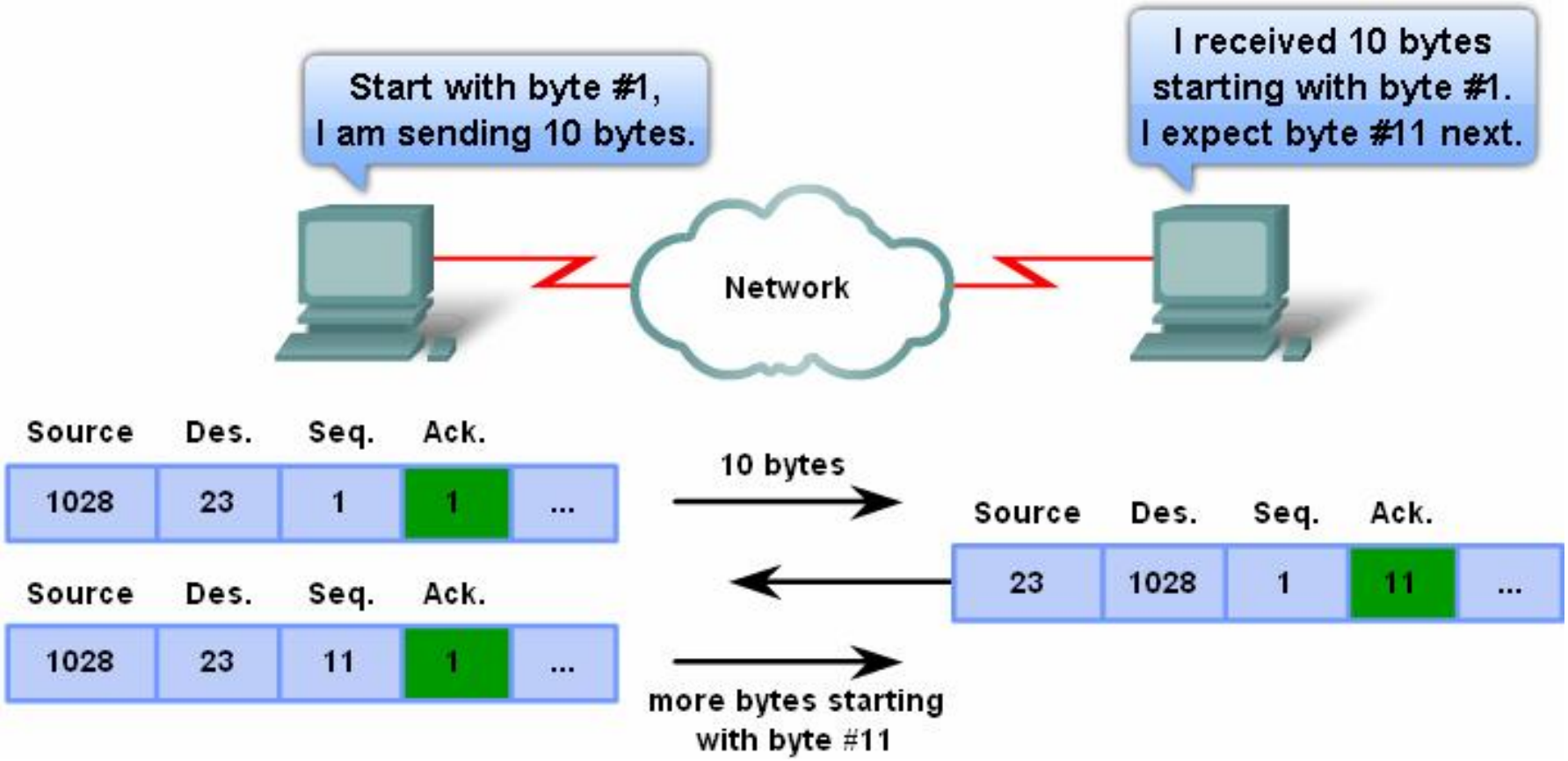
TCP heder

- *Source Port* - izvorišni port – slučajno dodeljen broj veći od 1023, koji će da identifikuje aplikaciju inicijatora komunikacije (klijenta)
- *Destination Port* – odredišni port – poznati port koji identifikuje serversku aplikaciju. Predstavlja protvrdu da su se svi prethodni okteti uspešno primili
- *Header Length* – dužina hedera
- *Code bits (Flags)* – flegovi koji označavaju posebne tipove paketa u održavanju sesije
- *Window* – veličina dinamičkog TCP prozora – koliko okteta može biti poslato pre nego što se dobije njihova potvrda (*Ack)*
- *Checksum* – provera bitskih grešaka, 16 bita, komplement sume TCP hedera, TCP podataka i TCP pseudo-hedera
- TCP *pseudo-heder* – polja: *src* i *dst* IP adrese, protokol i dužina TCP segmenta
  - donekle se krši princip slojeva, radi uključivanja podataka IP nivoa, koji nema proveru greške

TCP potvrda prenosa segmenta

- *Sequence Number (SEQ)* – broj prvog bajta aplikativnih podatka u segmentu koji se šalje
- *Acknowledgement Number (ACK)* – broj sledećeg bajta koji se očekuje da se primi
- U fazi uspostavljanja sesije uzima se slučajno izabrana vrednost SEQ za početnu vrednost
- Svaki pojedinačni bajt u nizu aplikativnih podataka ima svoju redni broj relativno u odnosu na inicijalni SEQ
- Prijem kontinualnog niza segmenata (svi bajtovi od početka aplikativnih podataka) se potvrđuje sa vrednošću ACK za jedan većom od rednog broja poslednjeg primljenog bajta u nizu
- značenje – “ovo je sledeći bajt koji se očekuje za prijem, svi prethodni su uspešno primljeni”

TCP potvrda prenosa segmenta



TCP protokol

Code bits – Kontrolni biti

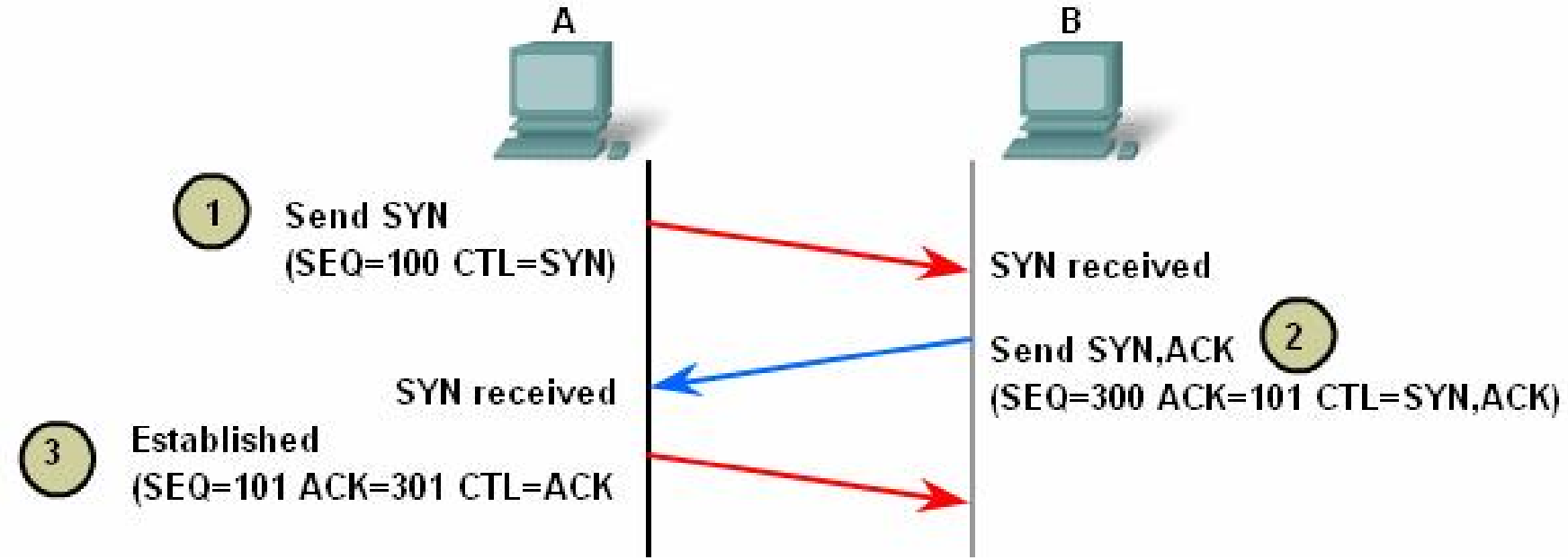
- Kontrolni biti (*Code bits, Flags)*
  - nose informacije o kontrolnim podacima i aktiviraju pojedina polja
- Najznačajniji flegovi
  - SYN – sinhronizacija brojne sekvence (*Sequence Number)*
  - FIN – označava poslednji segment koji se šalje – završava se sesija
  - ACK – polje *Acknowledgement Number* je aktivno – potvrđuje se prijem do određenog bajta u segmentu

Bit	Značenje
URG	Polje *URGENT POINTER* je validno
ACK	Polje *ACKNOWLEDGMENT NUMBER* je validno
PSH	Segment zahteva *push*
RST	Resetuj konekciju
SYN	Sinhronizuje brojeve sekvence
FIN	Poslednji segment u sekvenci za slanje

TCP uspostavljanje sesije

- TCP – *connection oriented protocol*
  - dvosmerna komunikacija – dve odvojene komunikacione sesije, u oba smera po jedna
  - uspostavljanje, održavanje i raskidanje obe komunikacione sesije
    - različite vrednosti SEQ i ACK
- Uspostavljanje TCP sesije u tri koraka - *Three-way handshake*
- 1. korak
  - inicijator slučajno bira vrednost za *Sequence Number – SEQ (u primeru SEQ=100)*
  - inicijator šalje paket sa SYN flagom i izabranom vrednošću *Sequence Number*
- 2. korak
  - druga strana prepoznaje i prihvata inicijativu za uspostavljanje sesije, i o tome obaveštava inicijatora - postavlja se ACK flag, a vrednost ACK polja se postavlja na SEQ+1 – sledeći bajt koji se očekuje za prijem
  - u istom paketu se inicira sesija u suprotnom smeru, na isti način kao 1. korak (SEQ=300)
- 3. korak
  - inicijator prihvata sesiju u suprotnom smeru i obaveštava drugu stranu (ACK=101)
- Rezultat: uspostavljene sesije u oba smera

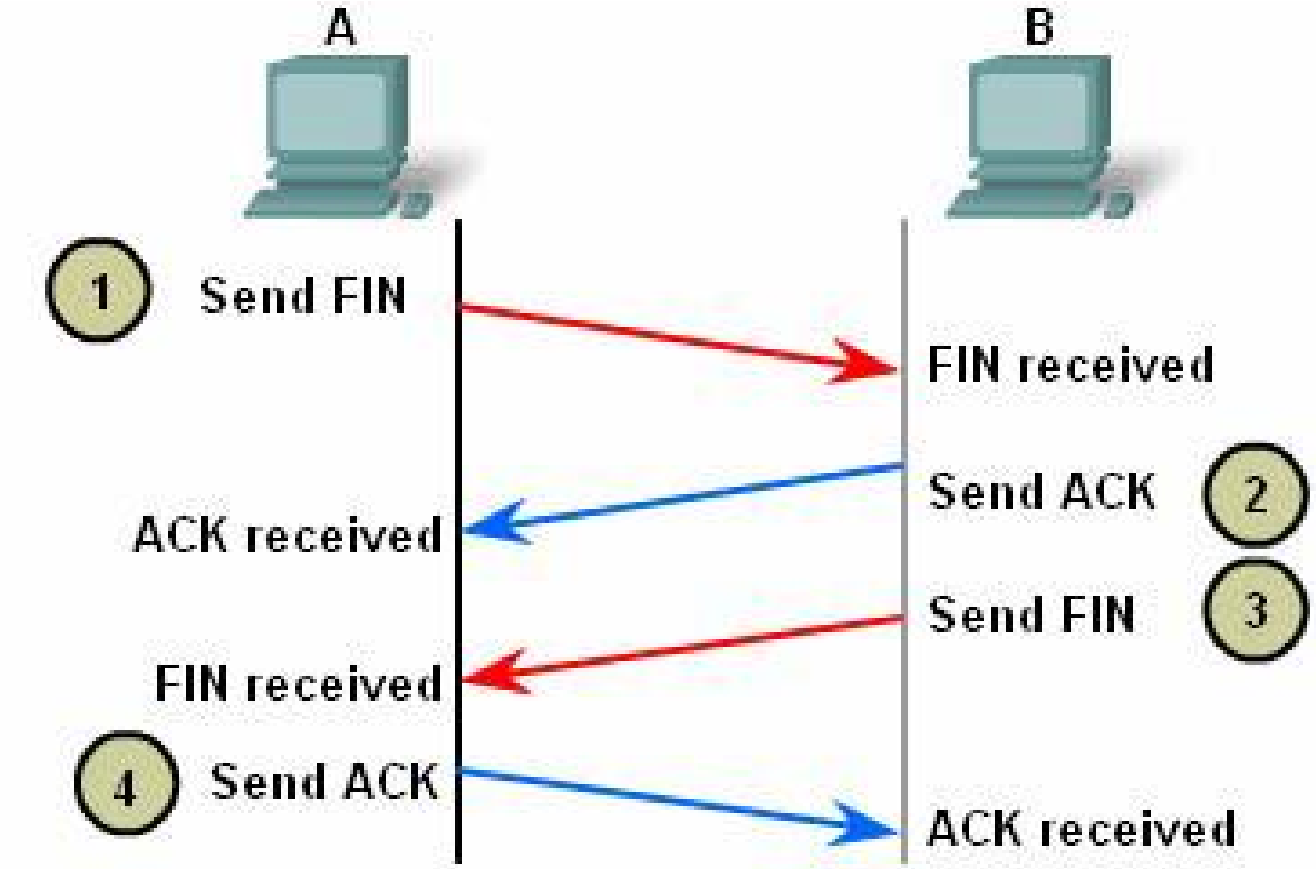
TCP uspostavljanje sesije



TCP raskidanje sesije

- Raskidanje TCP sesije u dva koraka u oba smera – 2 x *Two-way handshake*
- 1. korak
  - kada jedna strana nema više paketa za slanje, šalje se FIN flag u poslednjem paketu
- 2. korak
  - druga strana potvrđuje prijem poslednjeg paketa slanjem ACK za taj paket
  - sesija u tom smeru je zatvorena
- 3. korak
  - druga strana nastavlja da šalje preostale pakete, dok ne dođe do poslednjeg paketa u poslednjem paketu se postavlja FIN flag
- 4. korak
  - prva stran potvrđuje prijem poslednjeg paketa slanjem odgovarajuće ACK potvrde
- Rezultat: TCP sesija je prekinuta
- U slučaju da druga stana nema paketa za slanje, 2. i 3. korak se mogu objediniti dobija se *Three-way handshake*

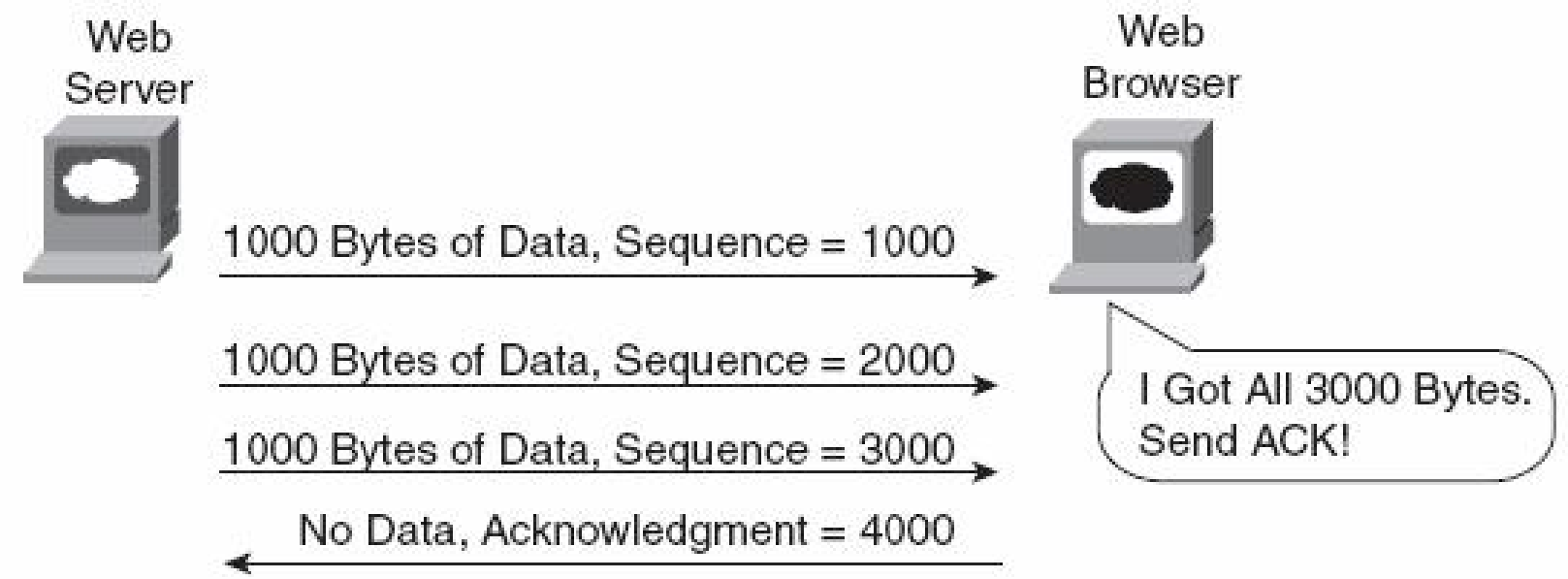
TCP raskidanje sesije



TCP pouzdan prenos

- Pouzdan prenos (*Reliable delivery)*
  - Svaki poslati segment zahteva potvrdu (*Ack)* da je uspešno pristigao na strani primaoca
- Potvrda se čeka određeno vreme – *timeout*
  - za svaki poslati segment startuje se poseban tajmer
- Vreme tajmera nešto veće od vremena putovanja segmenta od izvorišta do odredišta i nazad – RTT (*Round Trip Time)*
  - *RTT = a*OldRTT+(1*-a)*NewRTTsample
  - Timeout = b*RTT , b=2
- Ako nema grešaka, primalac će da pošalje potvrdu za poslednji bajt poslednjeg segmenta
  - potvrda da su svi segmenti uspešno primljeni
- Pošiljalac nastavlja sa slanjem novih segmenata
  - čeka se nova potvrda

TCP pouzdan prenos

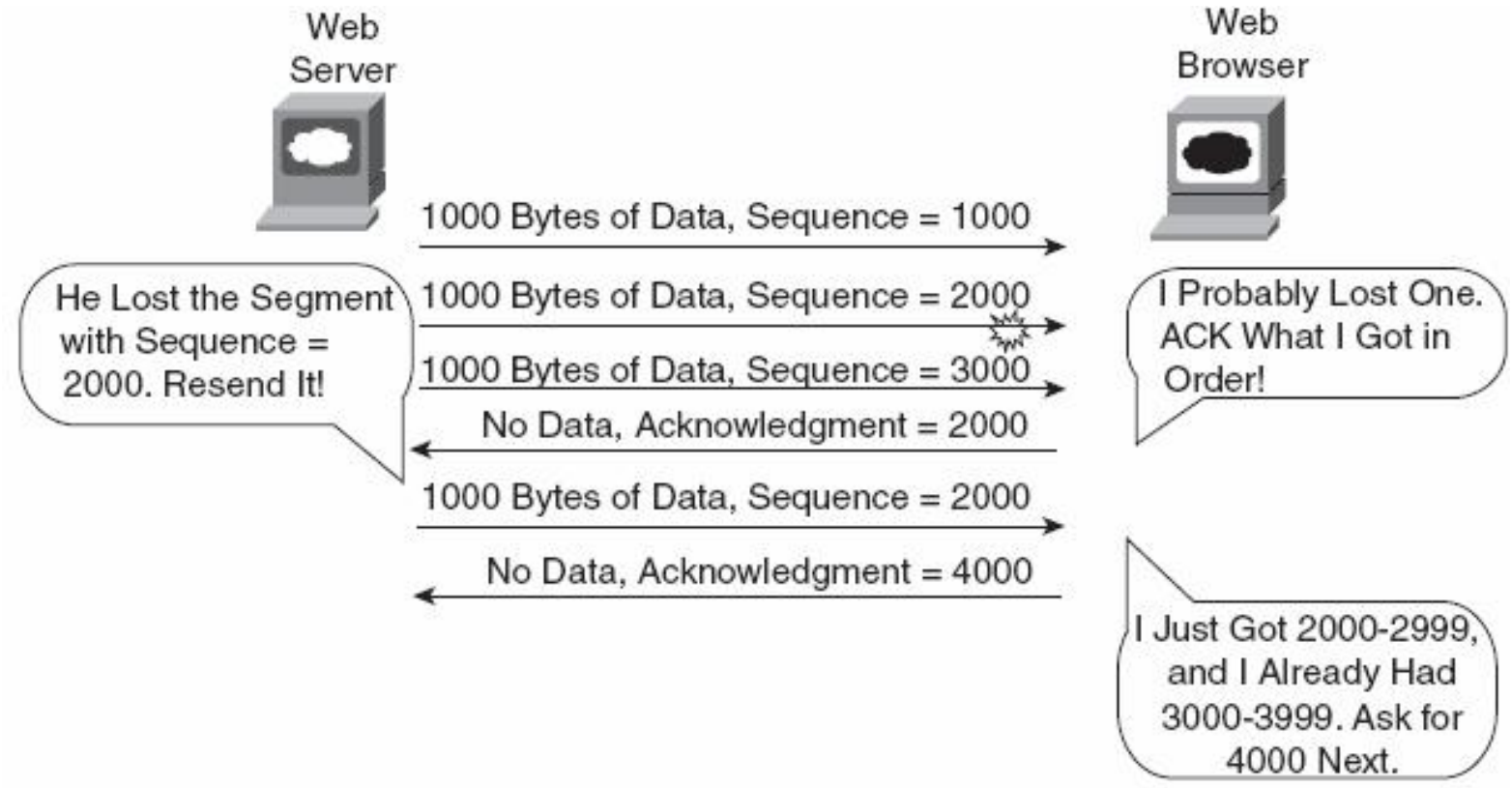


TCP oporavak od greške

Oporavak od greške (*Error Recovery)*

- Primalac je dobio poslednji segment, ali prepoznaje da je izostao jedan prethodni segment
  - potvrđuje se prijem poslednjeg ispravnog segmenta u nizu
  - poslednji primljeni segment, koji je diskontinualan sa prethodno primljenim segmentima se ne potvrđuje
- Ako istekne tajmer nekog segmenata, a ne stigne njegova potvrda, sprovodi se retransmisija samo tog segmenta, u nadi da su naredni segmenti uspešno pristigli
- Kada primalac dobije retransmitovani segment, popuniće “rupu” u nizi primljenih podataka i potvrdiće poslednji ispravan segment

TCP oporavak od greške

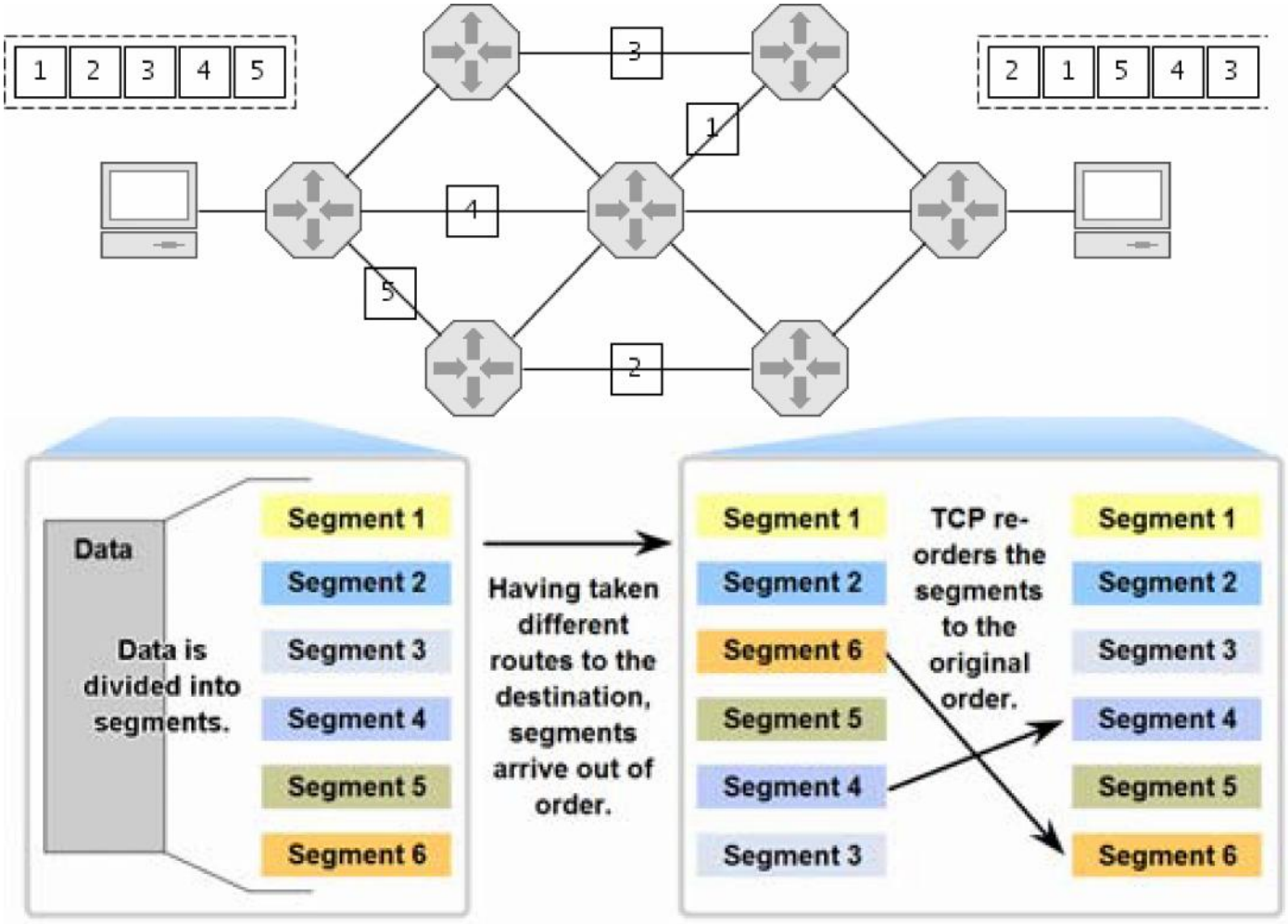


TCP rekonstrukcija redosleda segmenata

Rekonstrukcija redosleda segmenata (*segmets reordering)*

- U slučaju više putanja između izvorišta i odredišta različiti segmenti uzimaju različite putanje (*Load Balancing)*
- Može da dođe do permutacije segmenata – promene redosleda prijema
- Prijemna strana će da rekonstruiše originalni redosled na osnovu SEQ vrednosti primljenih segmenata

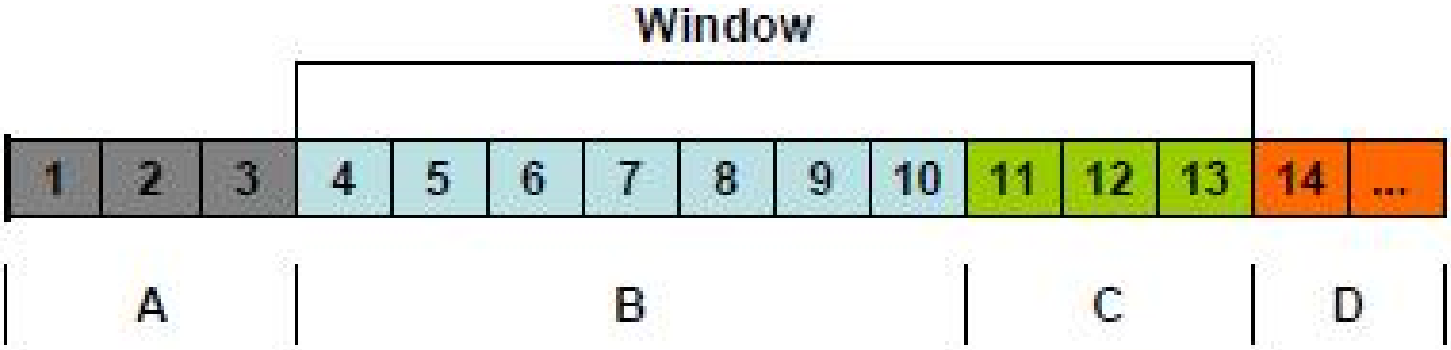
TCP rekonstrukcija redosleda segmenata



TCP kontrola toka

- **Kontrola toka (*Flow control)* –** mehanizam TCP prozora ***(Window)***
- Polje *Window* u TCP zaglavlju
  - označava ukupan broj bajtova koji se mogu poslati pre nego što se čeka na potvrdu
- Kada se popuni prozor
  - obustavlja se sa slanjem novih segmenata
  - čeka se potvrda prethodnih segmenata sa početka prozora
  - potvrđeni segmenti se oslobađaju i oslobađa se prostor u prozoru za slanje novih segmenata

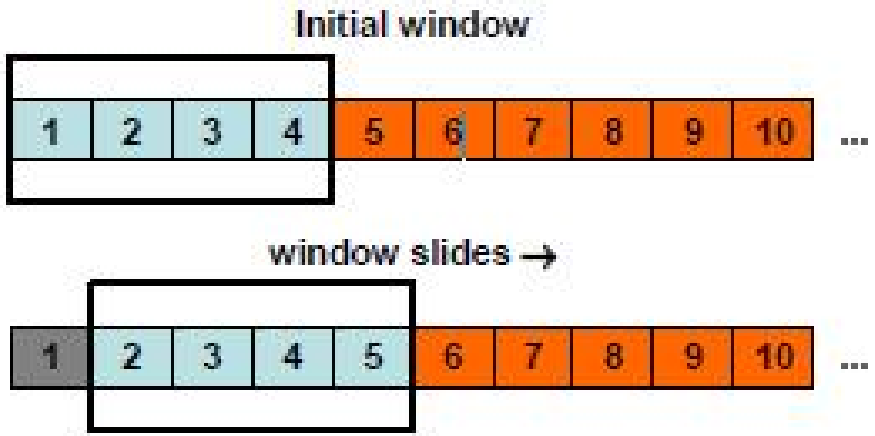
TCP kontrola toka



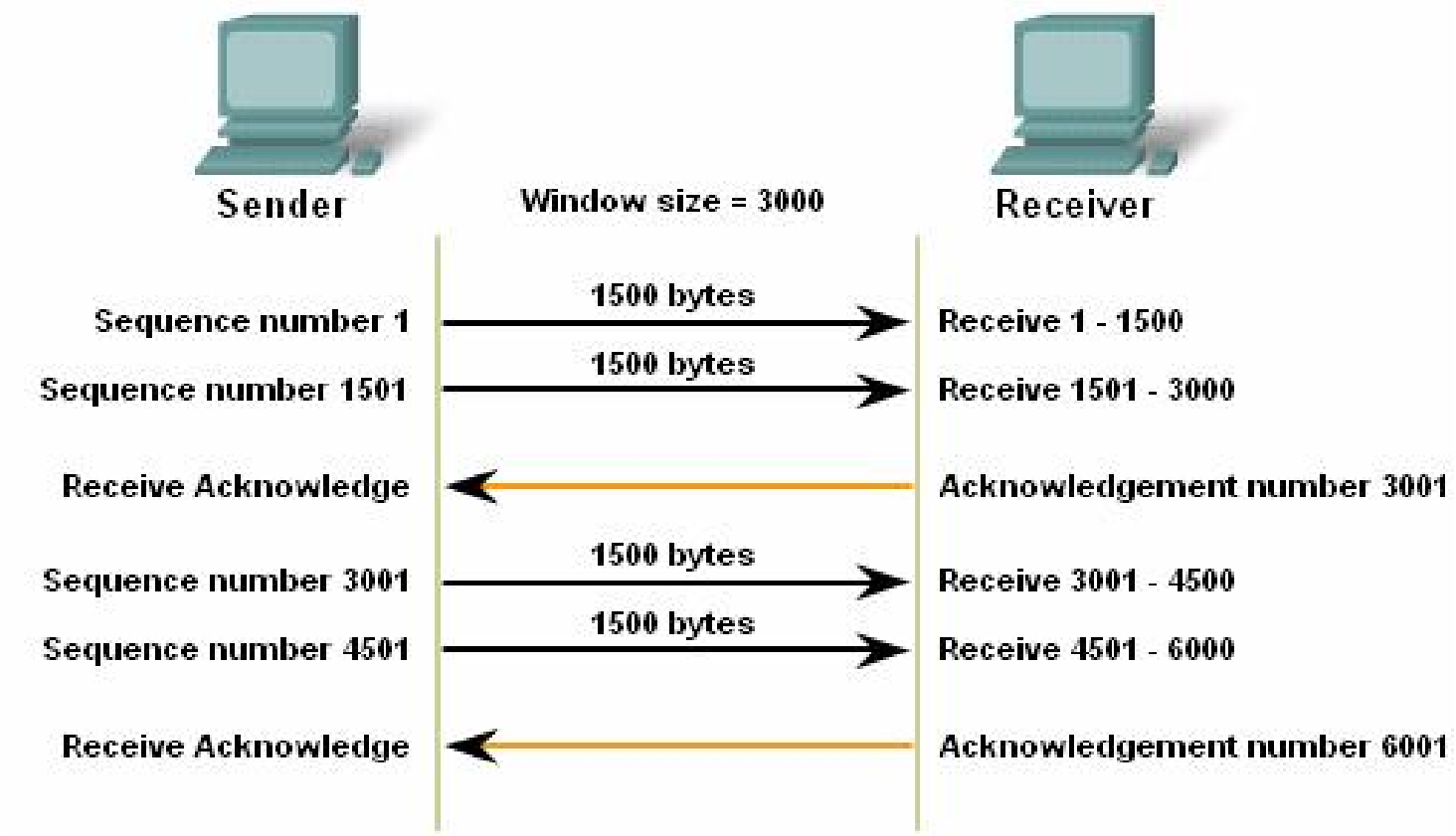
A	Poslati i potvrđeni bajtovi
B	Poslati, ali još nepotvrđeni bajtovi
C	Bajtovi koji se odmah mogu poslati
D	Bajtovi koje se ne mogu odmah poslati

TCP kontrola toka

- "Prozor" određene veličine se pomera po nizu aplikativnih podataka koji se šalju
  - Zato se mehanizam prozora često naziva i *Sliding Window*
- Manji prozor
  - više čekanja, sporije slanje
- Veći prozor
  - manje čeka, brže slanje



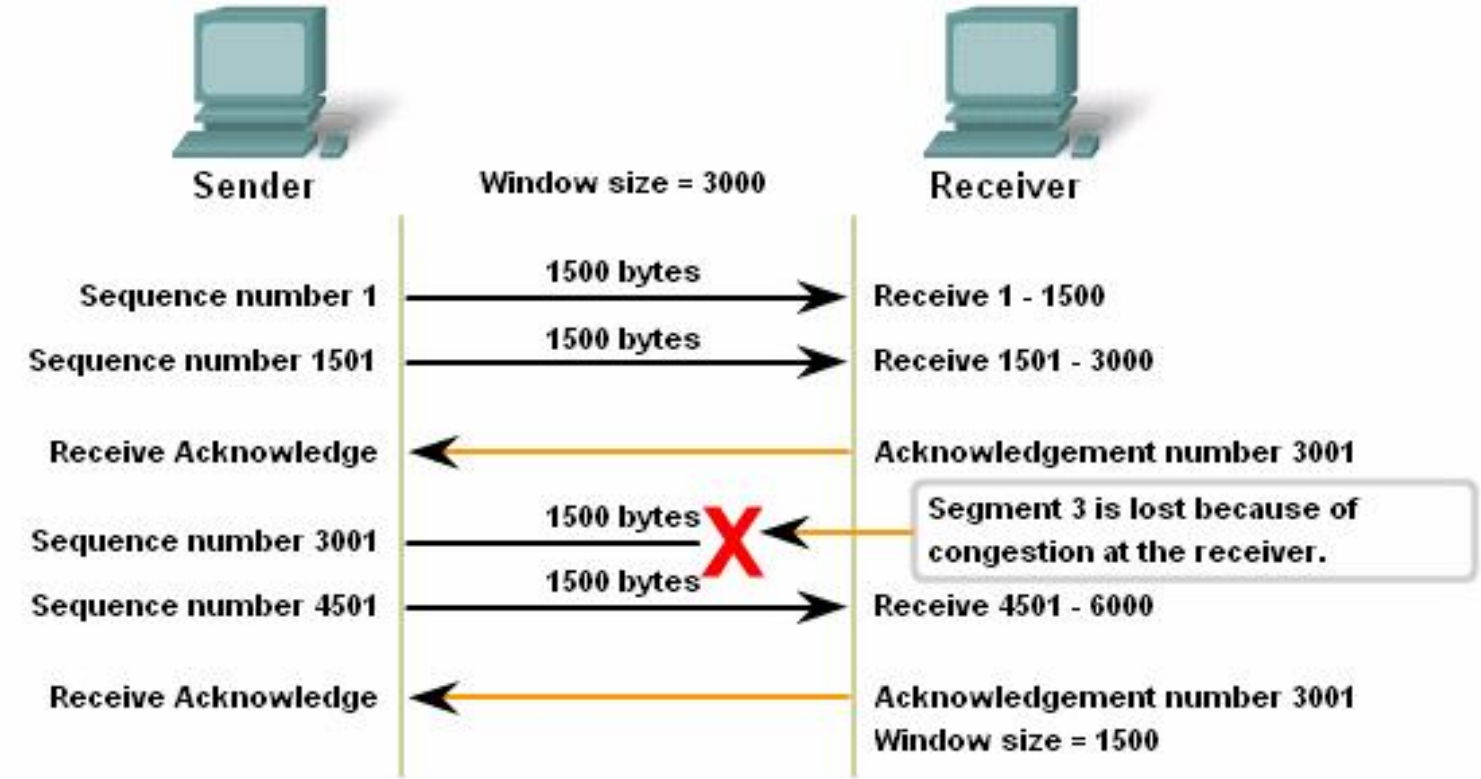
TCP kontrola toka



TCP kontrola toka

- TCP dinamički uspostavlja veličinu prozora
  - Zato se mehanizam prozora često naziva i *Dynamic Window*
- Inicijalno se obe strane dogovore o veličini prozora
- U slučaju gubitka paketa ili greške
  - vrši se oporavak od greške – retransmisija
  - ako je prijemna strana zagušena, može da zahteva smanjenje trenutne veličine prozora zajedno sa ACK
  - znak pošiljaocu da uspori slanje segmenata
- U slučaju da nema grešaka – pošiljalac postepene povećava veličinu prozora

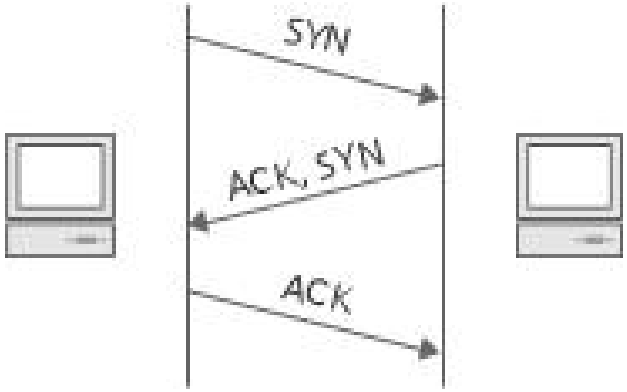
TCP kontrola toka



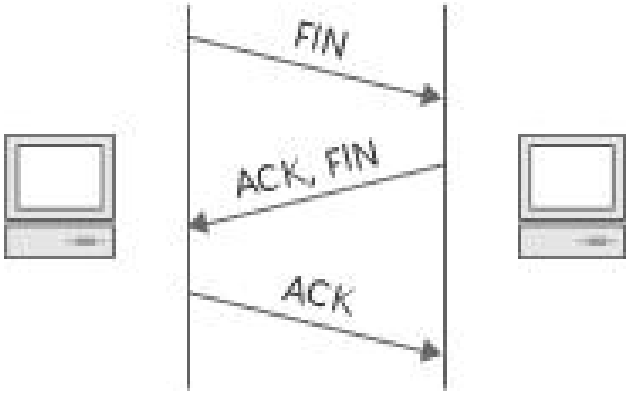
TCP kontrola zagušenja

- Kontrola zagušenja (*congestion control*)
- TCP može da se dinamički prilagodi trenutnom opterećenju u mreži
  - pošiljalac prilagođava brzinu slanja u zavisnosti od propusnog opsega i potencijalnog zagušenja na putu
- Moderne implementacije TCP protokola obuhvataju sledeće algoritme kontrole zagušenja:
  - *Slow start*
  - *Congestion avoidance*
  - *Fast retransmit*
  - *Fast recovery*

Uspostavljanje i raskid veze

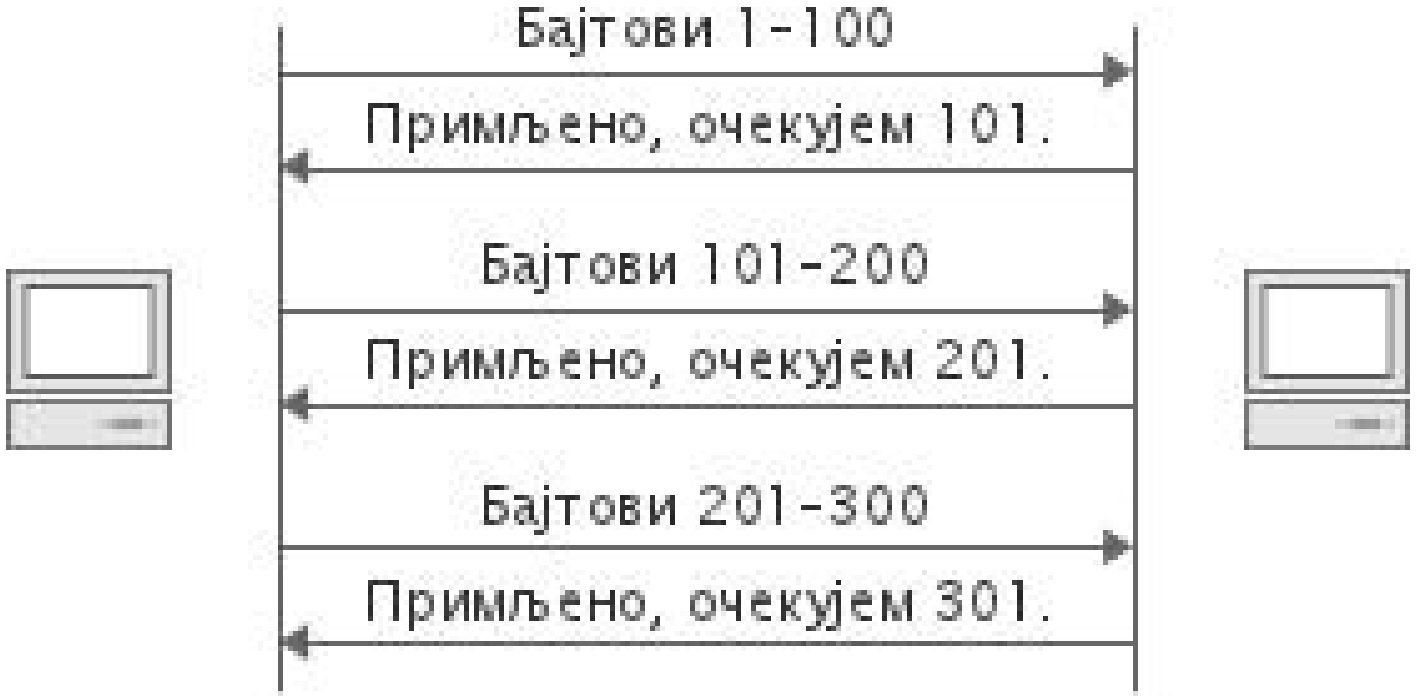


Uspostava veze

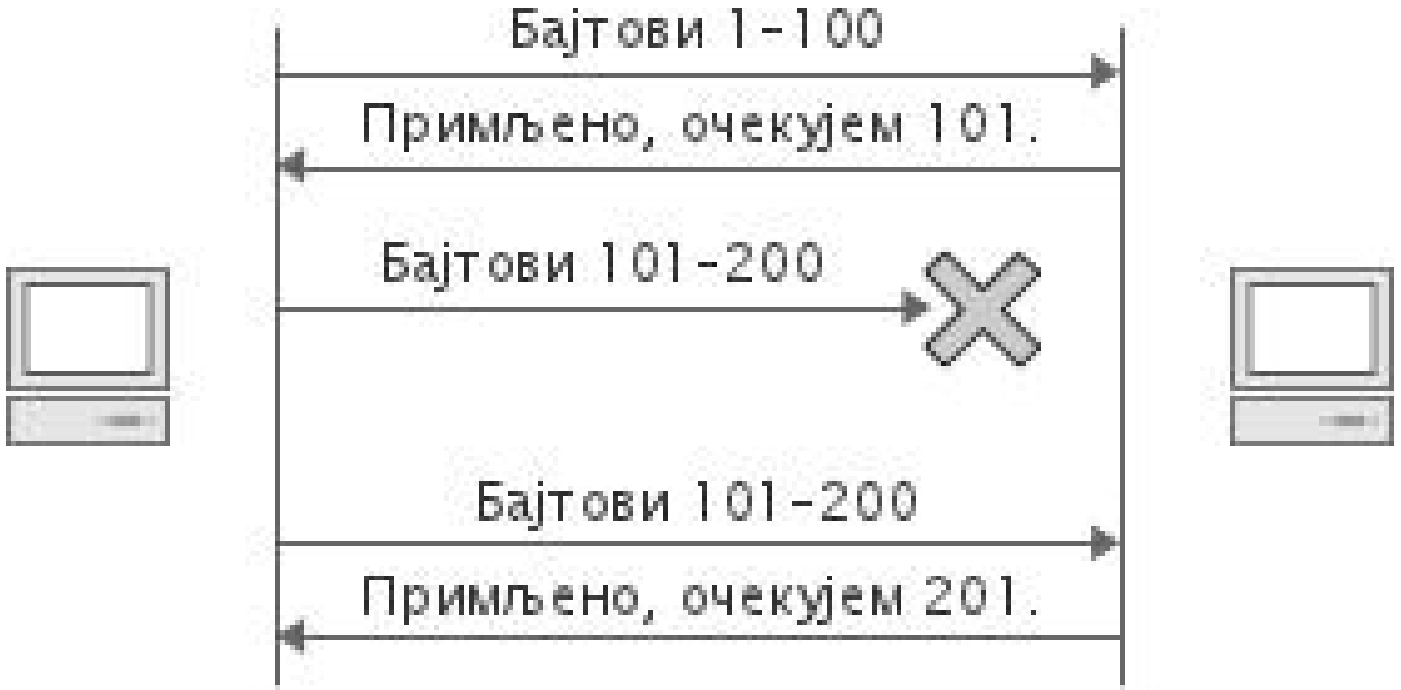


Raskid veze

Potvrda prijema



Potvrda prijema i greška



Zbirna potvrda prijema

Zbirna potvrda prijema i greška

Zatrpavanje primaoca

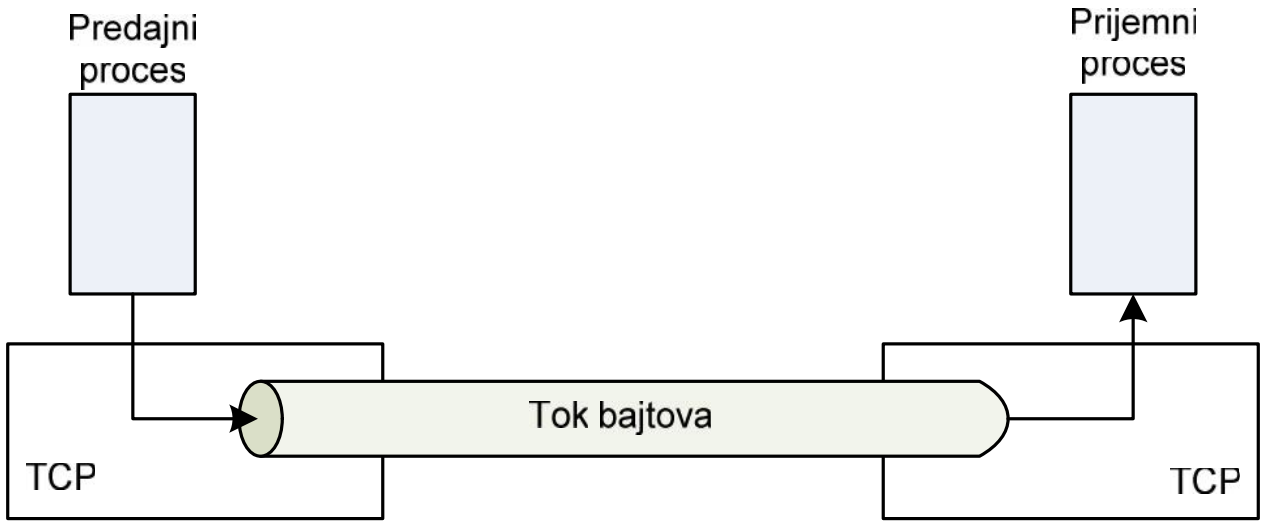
TCP kontrola zagušenja

TCP protokol

- TCP vidi nestruktuirane podatke, kao niz bajtova
- Redni broj je u bajtovima, ne u segmentima
- Inicijalni redni broj se bira slučajno
- TCP je u punom dupleksu – redni brojevi podataka su nezavisni za svaki smer
- Broj potvrde je broj sledećeg bajta koji se očekuje od pošiljaoca

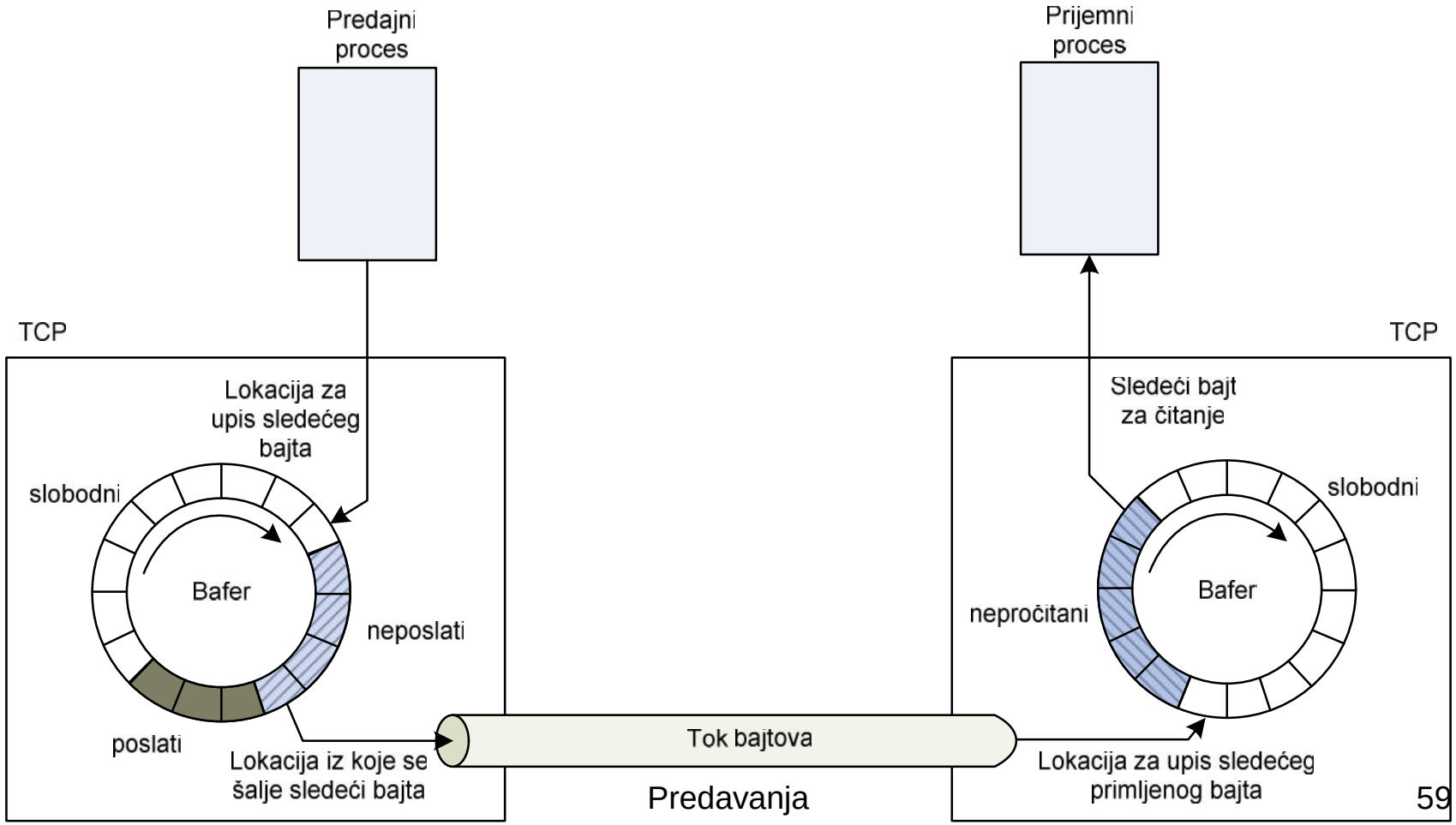
TCP protokol

- Orijentacija na tok



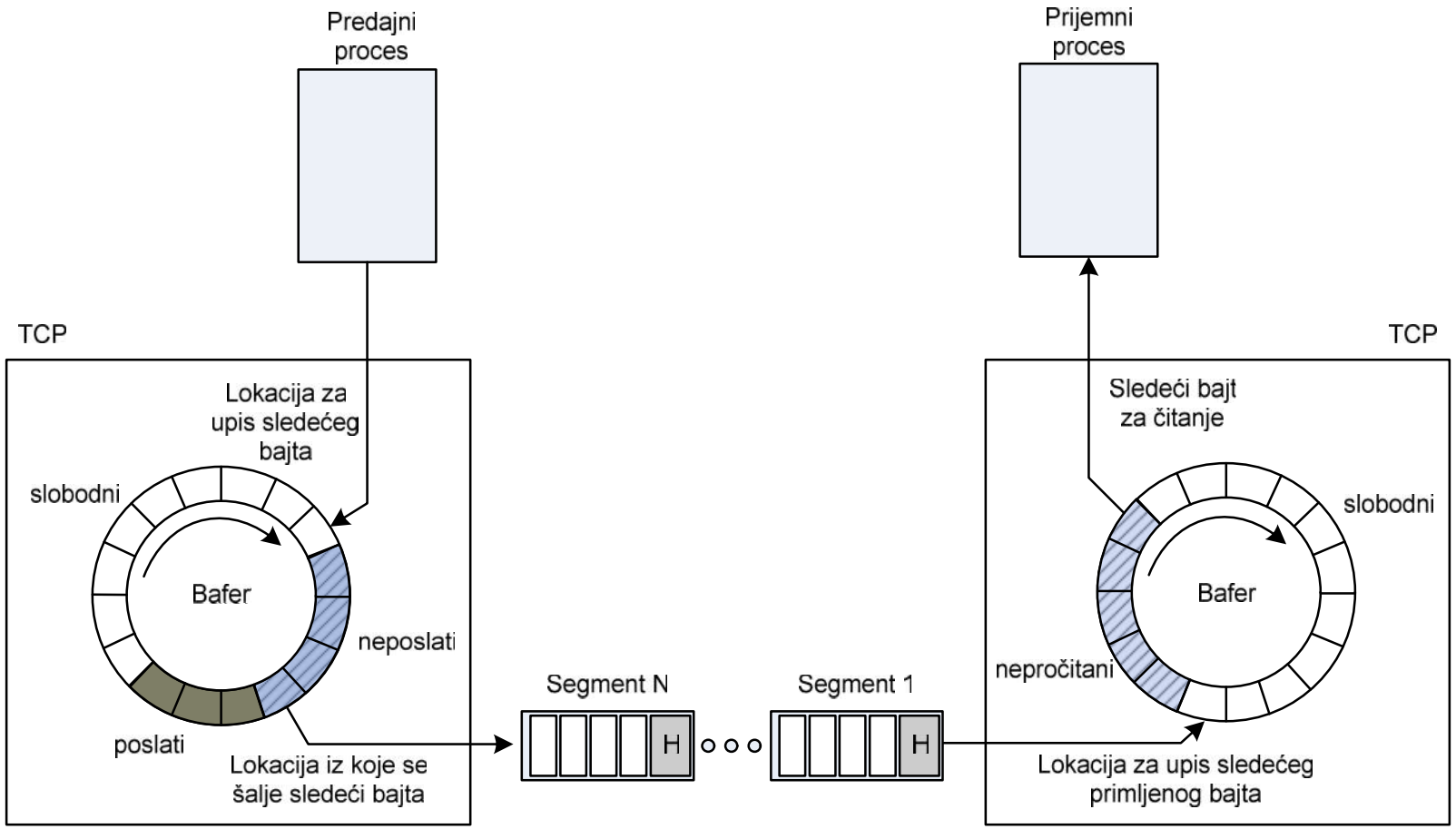
TCP protokol

- Prijemni i predajni baferi



Predavanja

TCP protokol



Dijagram stanja

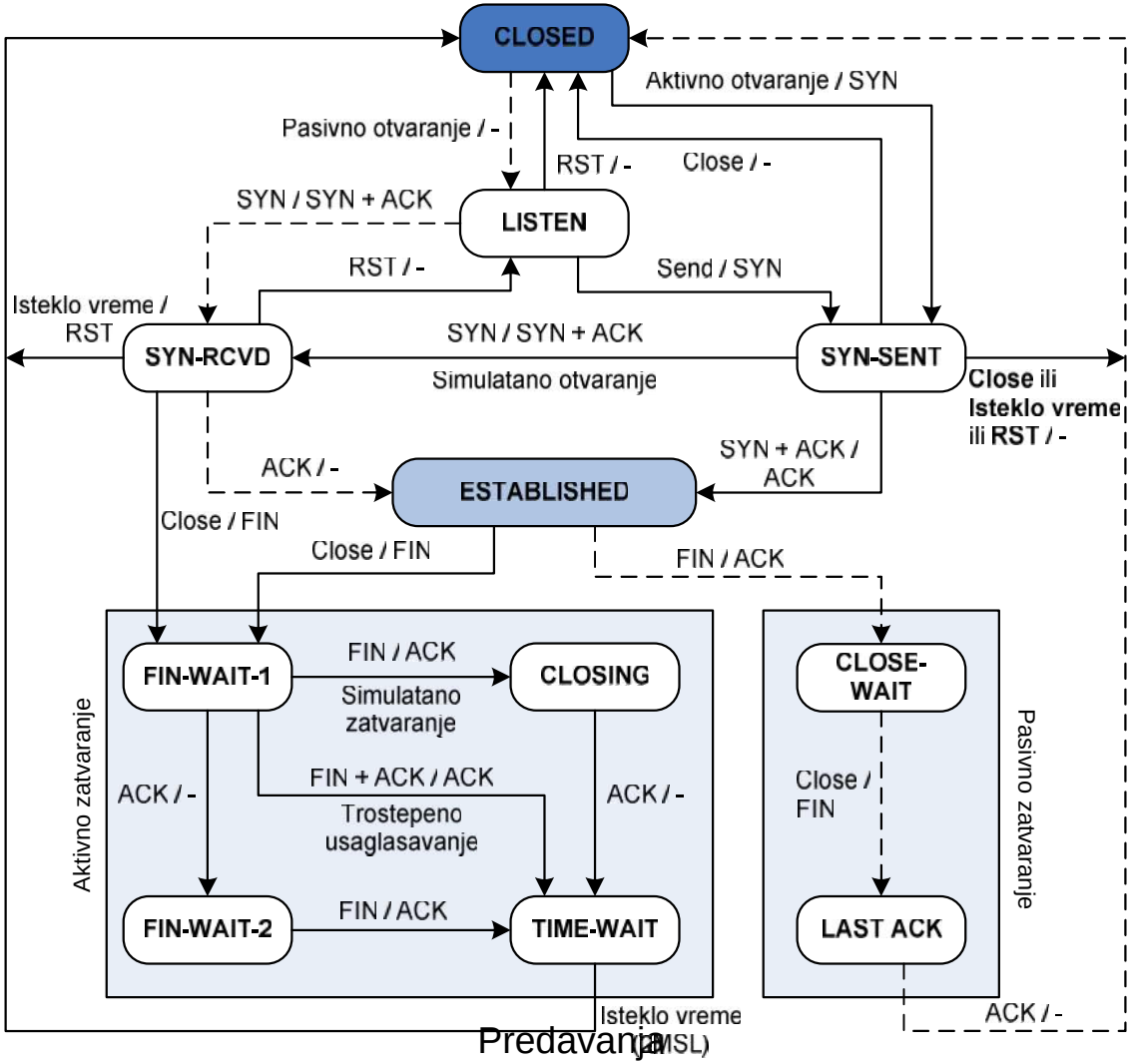
- Da bi se olakšalo praćenje različitih događaja i brojnih izuzetnih sitacija koje se mogu desiti u toku uspostavljanja konekcije, prenosa podataka i zatvaranja konekcije, TCP softver je realizovan u vidu konačnog automata (FSM – *Finite State Machine*).

- Ovaj konačni automat ima 11 stanja i može se predstaviti u vidu dijagrama stanja

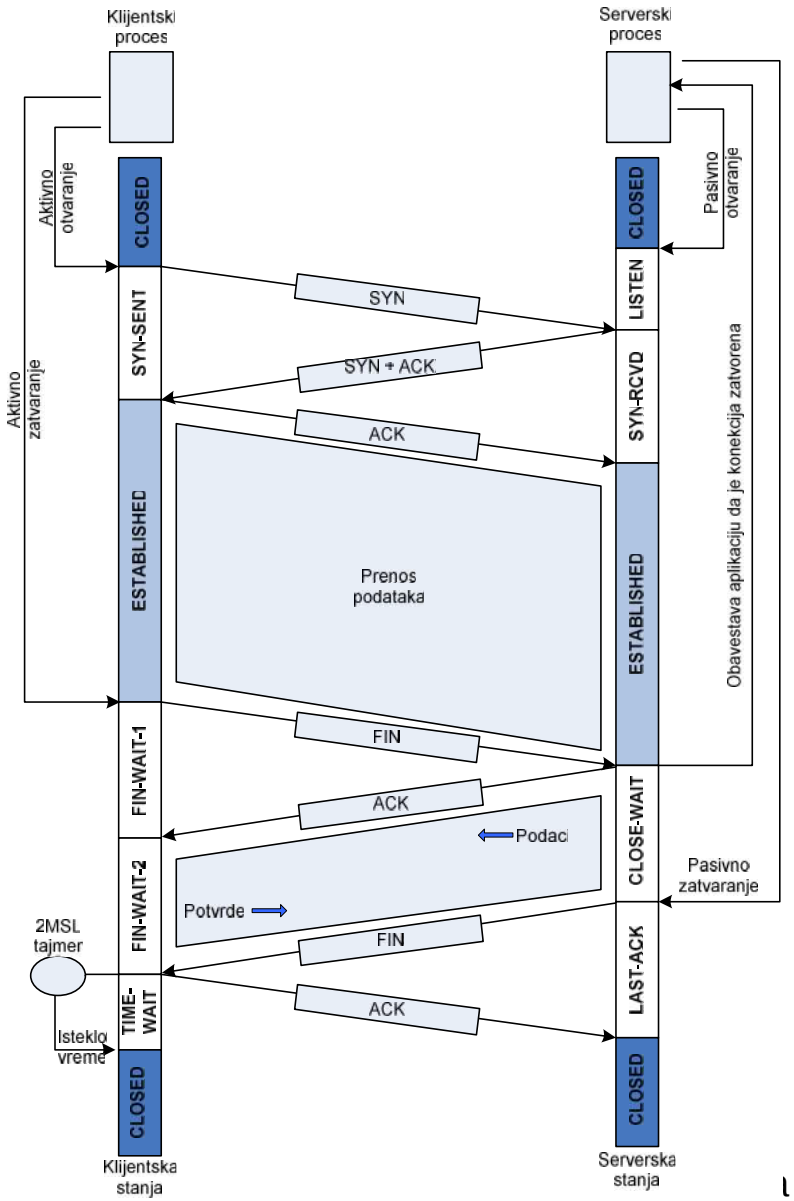
Dijagram stanja

Stanje	Opis	
CLOSED	zatvorena konekcija	Konekcija ne postoji
LISTEN	server čeka na poziv (apl. izvršila LISTEN)	
SYN RCVD	primljen zahtev za otvaranje konekcije	Otvaranje
SYN SENT	aplikacija izvršila CONNECT	
ESTABLISHED	stanje za normalni prenos podataka	Konekcija je otvorene
FIN WAIT 1	aplikacija izvršila CLOSE	Zatvarane konekcije
FIN WAIT 2	druga strana potvrdila FIN	
TIMED WAIT	čekanje da paketi nestanu iz mreže	
CLOSING	Istovremeni pokušaj zatvaranja	
CLOSE WAIT	druga strana je inicirala zatvaranje konekcije	
LAST ACK	čekanje da paketi nestanu iz mreže	

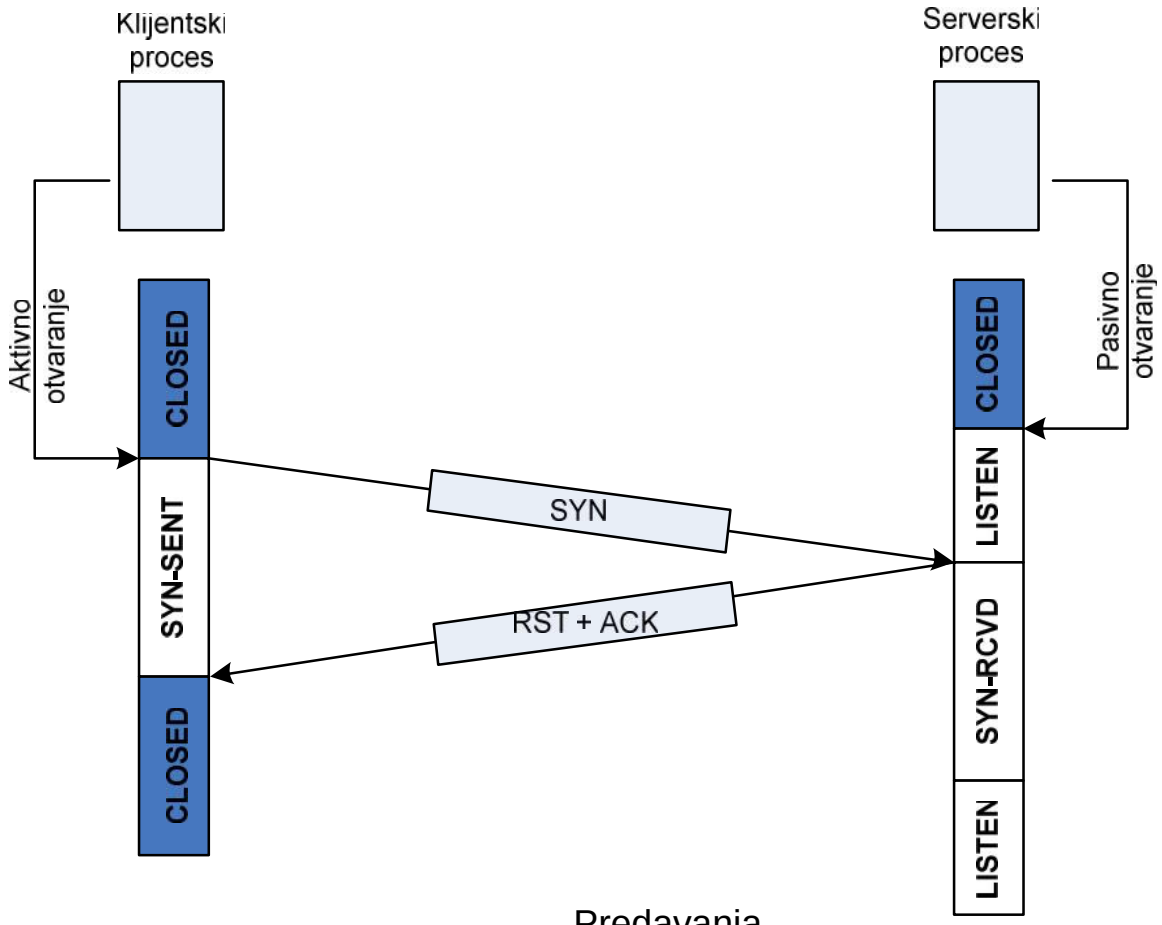
Dijagram stanja



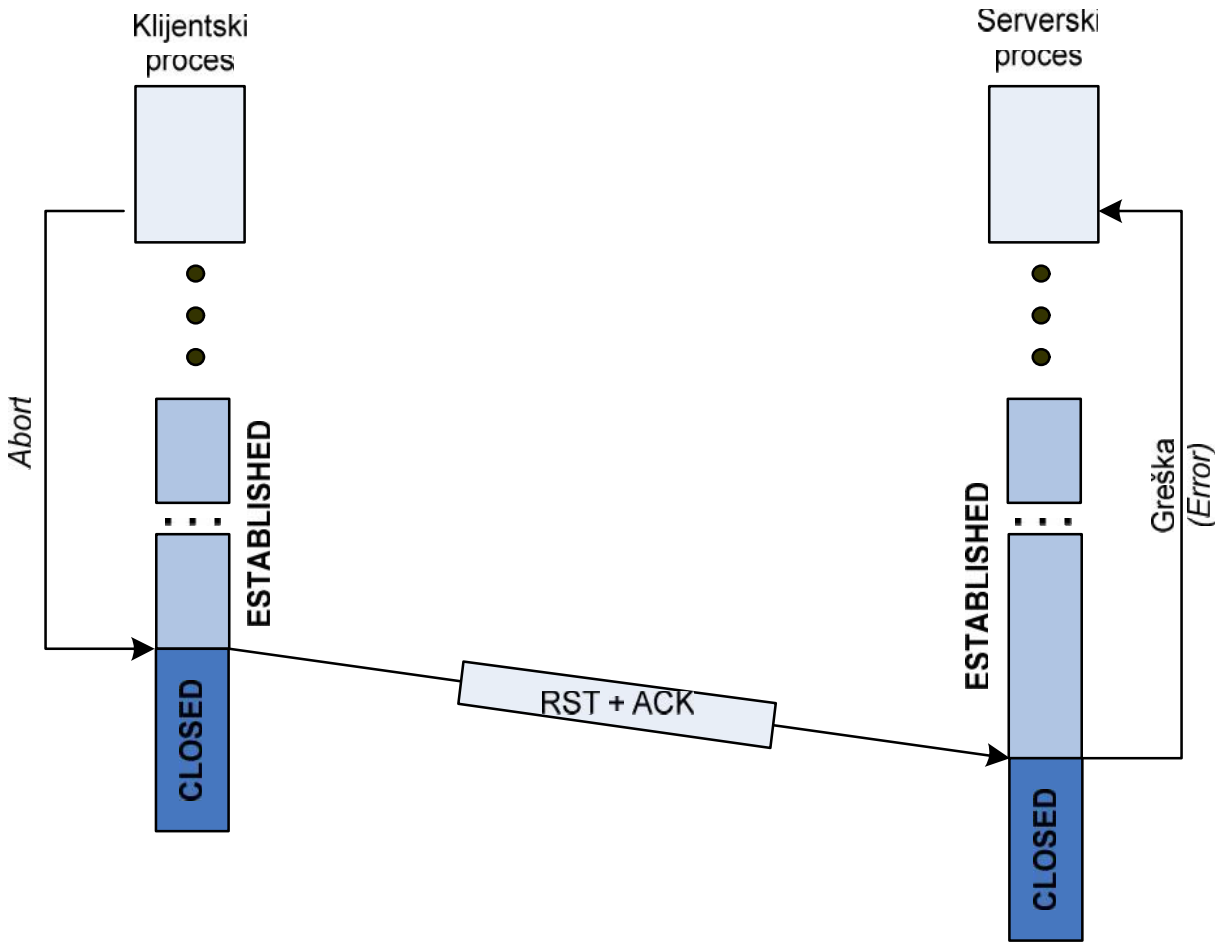
Scenario rada



Odbijanje konekcije



Prekidanje konekcije



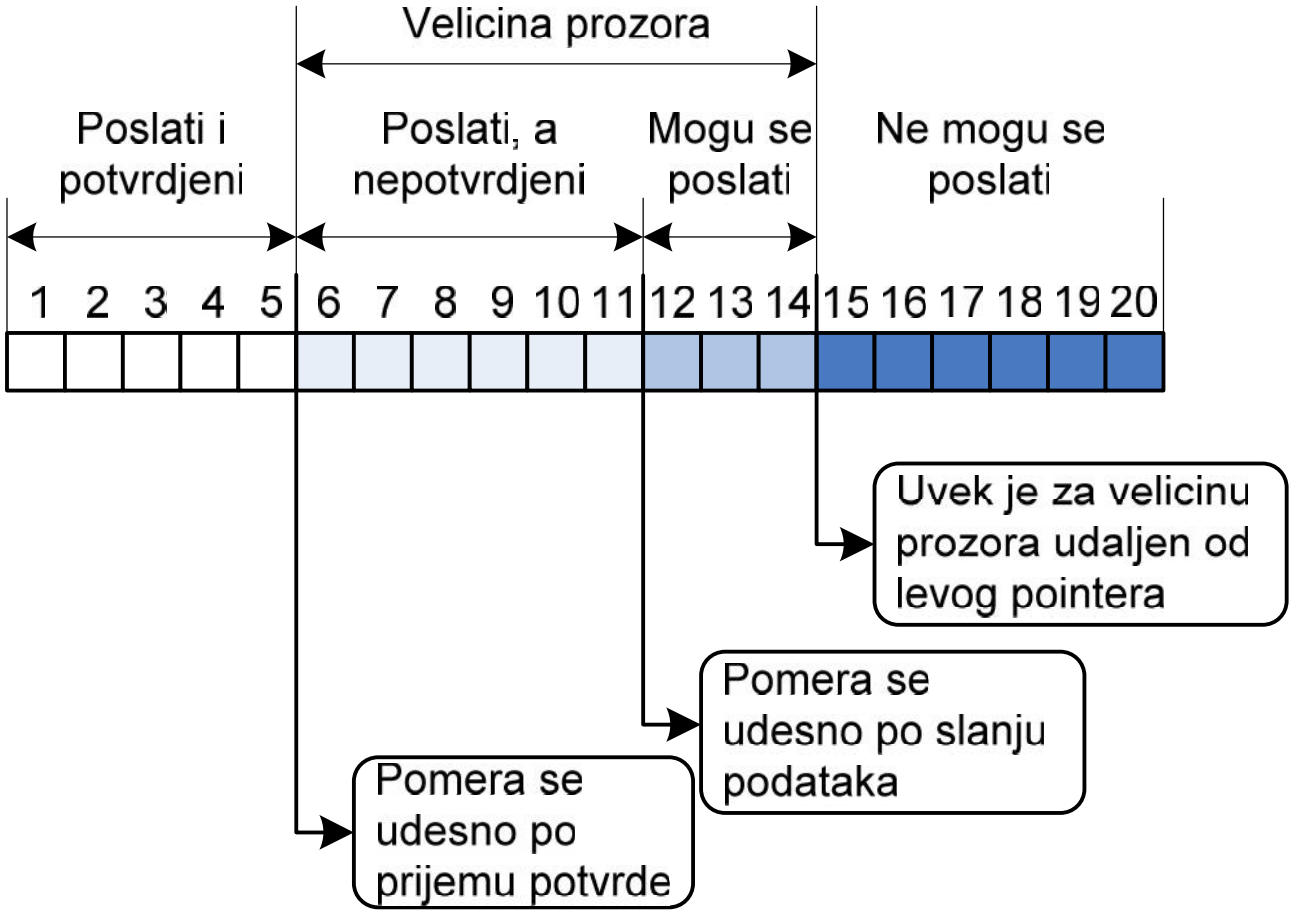
TCP kontrola zagušenja

- Zagušenjem upravlja mrežni sloj, ali to nije dovoljno
- Zagušenje se može rešiti samo smanjenjem brzine prenosa podataka
- Kada se uspostavlja veza
  - Bira se podesna veličina prozora
  - Primalac predlaže veličinu prozora na osnovu svoga bafera
  - Pošiljalac može taj predlog da prihvati

TCP kontrola zagušenja

- Posledica zagušenja – tajmeri padaju na nulu – paketi se odbacuju
- Zagušenje može da nastane iz dva razloga
  - kapacitet mreže
  - kapacitet primaoca
- Svaki pošiljalac održava dva prozora koji prate prethodna dva problema
  - prozor koji je odobrio primalac
  - prozor zagušenja
- Prozor zagušenja se u koracima povećava – klizeći prozor

Klizeći prozor



Upravljanje tajmerima

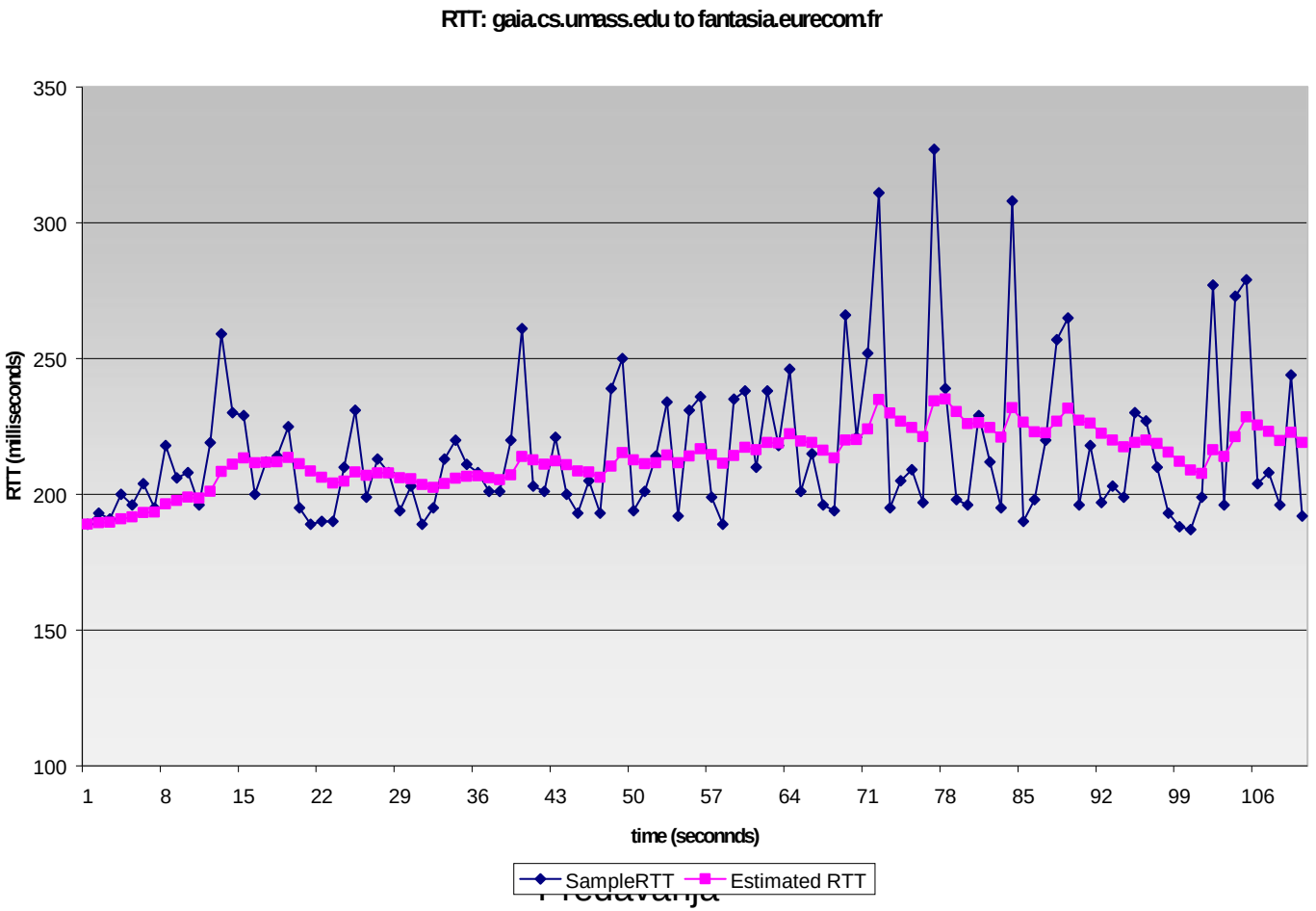
- Postoji više tajmera, a najvažniji je tajmer za ponovno slanje (*retransmission timer*)
  - Aktivira se sa slanjem segmenta
  - Deaktivira se kada stigne potvrda
  - Ako istekne, segment se ponovo šalje
- Problem dodeljivanja roka tajmeru
  - Nije jednostavno kao na sloju veze podataka
  - Mali rok – Internet se zagušuje nepotrebnim segmentima
  - Veliki rok – neefikasnost Interneta
- Algoritam za dinamičko podešavanje tajmera

$$RTT=\alpha RTT+(1-\alpha)M$$

M je vreme korektnog stizanja potvrde

α - koeficijent izglađivanja (npr $\alpha=7/8$)

Upravljanje tajmerima



Upravljanje tajmerima

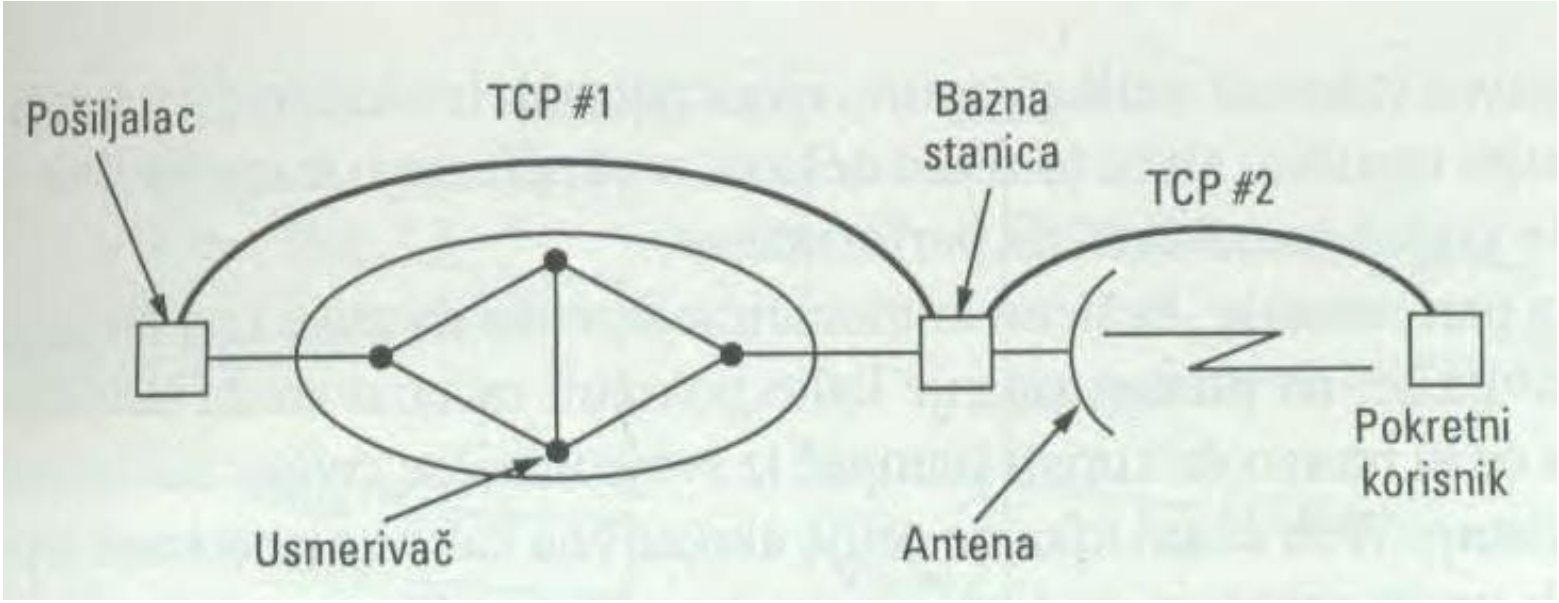
- Tajmer za ograničenje čekanja
  - Sprečava kružno blokiranje (uzajamno čekanje)
- Tajmer za proveru stanja veze
  - Kada se veza neko vreme potpuno utiša, treba proveriti da li je druga strana još uvek na vezi

Bežični TCP i UDP protokoli

- Transportni protokoli ne bi trebalo da zavise od tehnologije mrežnog sloja
- U praksi, TCP protokoli su optimizovani za žične veze
- Žične veze:
  - Paketi se gube zbog zagušenja
  - Treba usporiti slanje
- Bežični prenos
  - Paketi se gube zbog uslova prenosa
  - Treba što pre ponovo slati pakete

Bežični TCP i UDP protokoli

- Heterogene mreže: prenos paketa kroz žične i bežične mreže
- Princip: TCP veza se deli na dve zasebne veze: do bazne stanice i od bazne stanice



Bežični TCP i UDP protokoli

- Bežični prenos smeta i UDP protokolu
- U principu, UDP ne nudi nikakve garancije, ali se podrazumeva da se paketi retko gube
- Kod bežičnih veza, gubljenje paketa je evidentno
  - Slabe se performanse mreže

Performanse

- Osnovni problem je zagušenje mreže
- Strukturna neravnoteža resursa – spor računar na gigabitnoj mreži ne stiže da obradi pakete
- Problem sa difuznim slanjem ako port nije odgovarajući
  - Npr. 10000 računara odgovara da nešto nije u redu
  - Problem je rešen tako što je izmenjen UDP protokol da ne odgovara na ovakve greške
- Brzi procesori sa velikim memorijama, a malo memorije je dodeljeno baferima
- Loše podešeni tajmeri

Performanse

- Iskustvena pravila:
  - Brzina procesora je važnija od brzine mreže
  - Smanjenje broja paketa skraćuje vreme obrade
    - Svaki paket nosi zaglavlje, izaziva softverski prekid itd.
  - Povećanje propusnog opsega ne menja kašnjenje
    - Kašnjenje se može smanjiti boljim protokolom, operativnim sistemom ili mrežnim interfejsom