TRANSPORTNI SLOJ

Razlika između TCP i UDP protokola je ta što TCP omogućava pouzdan, a UDP brz prenos podataka.

Na podatke koje silaze sa aplikacionog sloja se dodaje ili TCP ili UDP zaglavlje (header).
Rezultat enkapsulacije kada dodamo TCP zaglavlje na podatke se naziva segment.
Rezultat enkapsulacije kada dodamo UDP zaglavlje na podatke se naziva datagram.

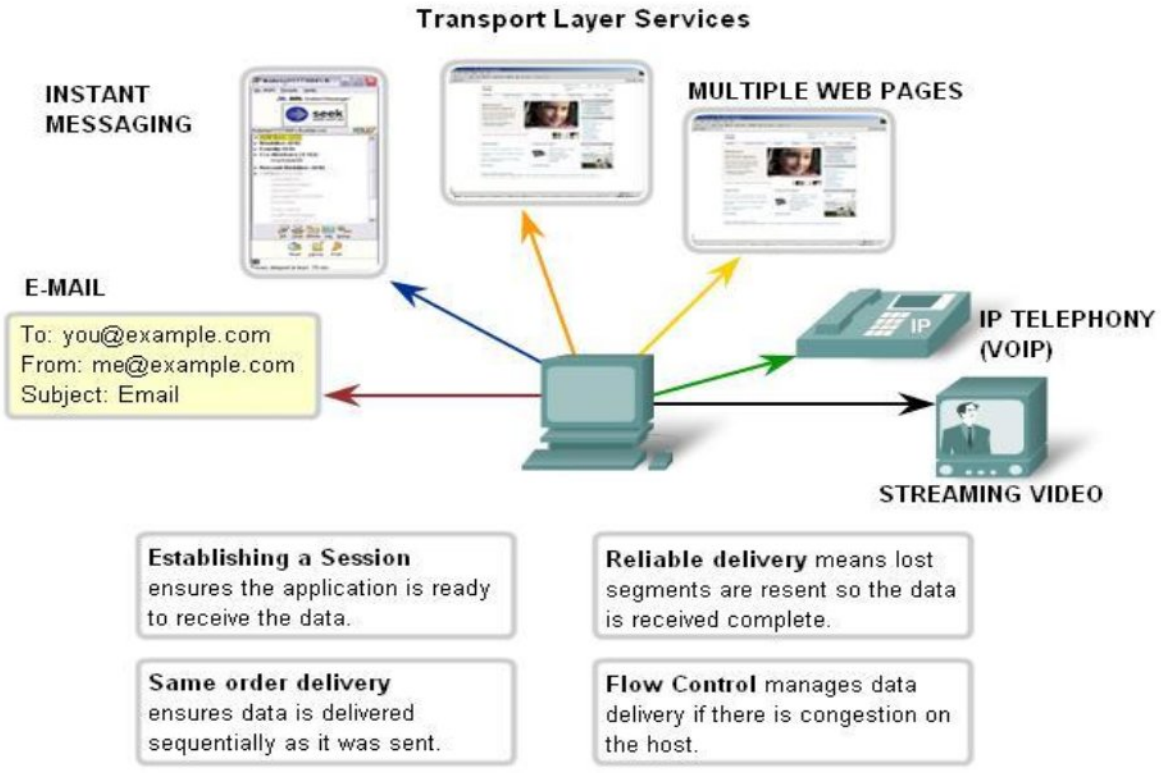
Tipična veličina TCP zaglavlja je 20 bajtova, a UDP zaglavlja 8 bajtova.

I TCP i UDP omogućavaju 3 osnovne funkcije transportnog sloja:

1) Segmentaciju podataka
2) Adresiranje procesa (određivanje šta je source, a šta destination port)
3) odluka o (ne)pouzdanom prenosu na osnovu prirode aplikacije – izbor između TCP-a i UDP-a

TCP ostvaruje četiri karasterike, koje ne ostvaruje UDP:

1) pouzdan prenos
2) uspostavljanje veze
3) reasembliranje
4) kontrola toka



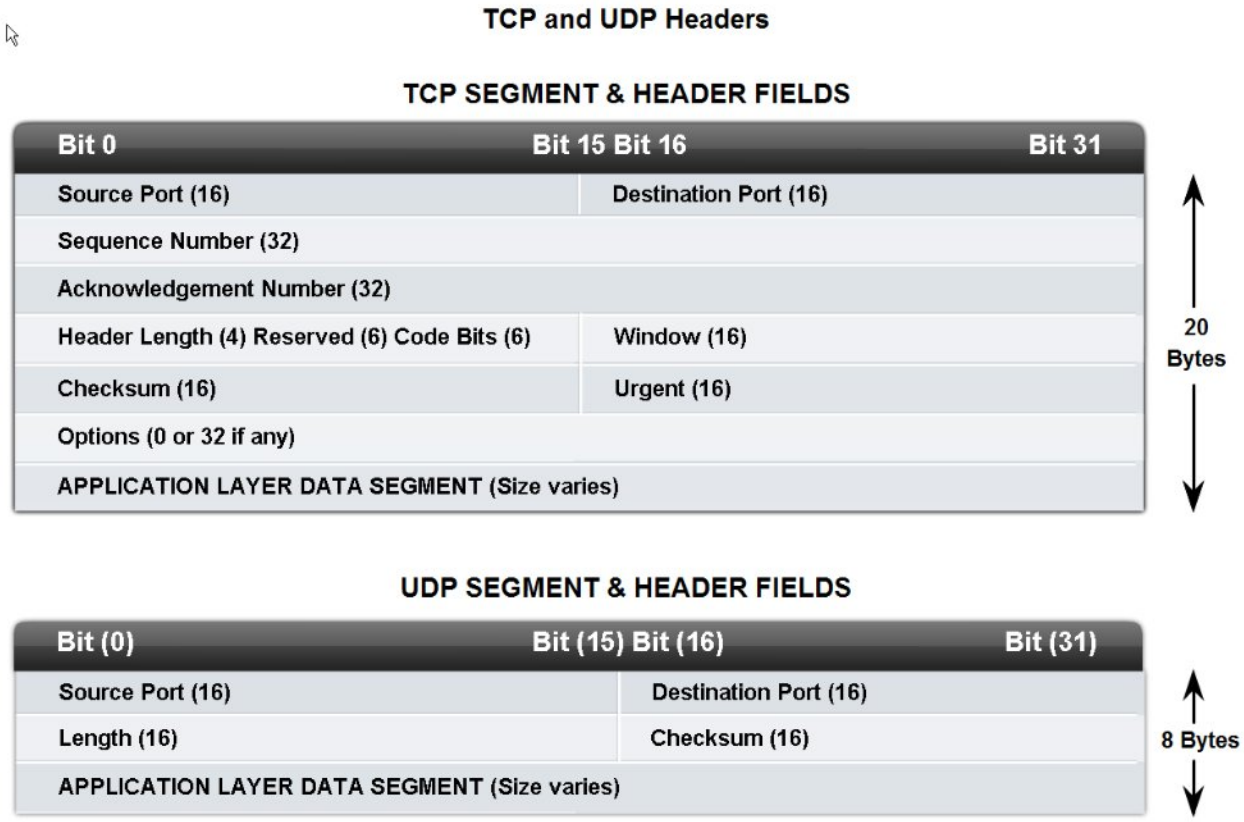
TCP postoji zato što postoji nesavršen prenos u računarskim mrežama, odnosno da bi se gubljenje podataka tokom komunikacije mogao nadoknaditi na adekvatan način.

Pouzdan prenos je prenos kod koga znamo da je nešto stiglo na željenu lokaciju.

Upostavljanje veze podrazumijeva da se pošiljalac I primalac prije bilo kakvog slanja poruka prvo "nađu", odnosno da znaju da su oba dostupna za komunikaciju.

Podaci se dijele u veliki broj malih poruka. Te poruke se mogu slati različitim putanjama nezavisno jedne od druge, I mogu doći u različitim trenucima na odredište. Te poruke se na odredištu moraju poredati u odgovarajući redoslijed, kako bismo dobili željene podatke na odredištu. To se ostvaraju tako što se svaka ta mala poruka označi određenim rednim brojem.

Kod UDP protokola, nema ponovnog slanja poruka ako neke poruke ne dođu na odredište. Ako poruke stignu u pogrešnom redoslijedu na odredište, neće se vršiti redanje tih poruka.



TCP zaglavlje tipično ima 20 bajtova.
Opcije se rijetko koriste, I ne ulaze u standardno TCP zaglavlje.

TCP zaglavlje počinje sa portovima = 4 bajta.
Port ima dva bajta, pa za portove Source I Destination trebaju po 2 bajta.
Portova može biti 65 536.
Portovi se dijele na:

1) **well-known portovi** 0-1023 : dodijeljeni nekim dobro poznatim aplikacijama
2) **dinamički portovi** 1024-65 536 : operativni sistem dinamički alocira ove portove nekim procesima

Sledeće polje je **Sequence Number** (SN) = **4 bajta.**
Služi za numerisanje segmenata, kako bi se omogućilo da oni budu pravilno složeni na odredištu.
U Sequence Numberu se šalje redni broj bajta koji se šalje. U jednom segment može da bude više korisnih bajtova. Dva uzastopna segmenta ne moraju da imaju uzastopne Sequence Number.

Ako je u prvom segmentu SN=1, I prvim segmentom se šalje 10 bajtova, onda će SN sledećeg segmenta biti 11. Sequence Number uvijek pokazuje na redni broj prvog bajta koji se šalje.

Sledeće polje je **Acknowledge Number** (AN) = **4 bajta.**
AN služi za potvrđivanje nekog prijema.

TCP veza je uvijek "jedan prema jedan". Kada se uspostavlja veza između dva hosta A I B, zapravo se uspostavljaju dvije jednosmjerne veze sa suprotnim smjerovima:

1) veza od hosta A ka hostu B
2) veza od hosta B ka hostu A

Primjer:
Ako šaljemo segment sa 10 korisnih bajtova, čiji je SN = 1 prema hostu B, host B potvrđuje prijem ovog segmenta pomoću polja AN. On radi to tako što u segment koji host B šalje host A stavi u polje AN vrijednost 11, koji pokazuje na redni broj prvog sledećeg bajta koji host B očekuje da primi od hosta A. Na ovaj način host B potvrđuje prijem svih korisnih bajtova prije 11. bajta.
TCP veza je dvije nezavisne veze, jer ono što host A šalje prema hostu B nema veze nikako sa onim što host B šalje hostu A.
SN hosta A je u sinhronizaciju sa AN hosta B, I obrnuto.
Na ovaj način se ostvaruje pouzdanost, jer ako druga strana u određenom vremenskom interval ne javi da je primila određeni segment, vrši se ponovno slanje tog segmenta.

Sledeće polje je **Header Length** (veličina zaglavlja) = HL
Zaglavlje je uglavnom veličine 20 bajtova, ali može biti I više ako se koriste opcije u zaglavlju.
HL zauzima 4 bajta I u polje HL se upisuje broj četvorobajtnih podataka koji čine zaglavlje.
Uglavnom je u HL polje upisano vrijednost 5. (5*4 bajta = 20 bajtova)

Sledeći dolaze **FLAG-ovi**.

Ukupno postoji 8 flagova, **6** je bitnih:
FIN = flag koji se koristi za raskidanje veze.
PSH = postavlja se kada se u segmentu šalju podaci I označava da te podatke treba proslijediti dalje na aplikacioni sloj
URG = postavljen, ako se koristi polje Urgent. Ove dvije stavke rade sinhronizovano.
RST = resetovanje konekcije
SYN
ACK

Veoma je važno I da se izvrši zatvaranje TCP veze, jer u suprotnom se dešava da druga strana može da nastavi sa pokušavanjem slanja segmenata, jer nema informaciju da se druga strana povukla iz komunikacije.
Za zatvaranje komunikacije se koristi FIN. Strana A šalje FIN. Druga strana šalje svoj FIN, zajedno sa ACK(potvrdom da je primila prethodnu poruku segmenta A).
Ako se ne dobija ACK sa druge strane, tada prva strana opet šalje FIN, očekujući odgovor od druge strane. Ako druga strana ne odgovori na 16 puta poslati FIN, onda strana A zatvara TCP vezu.
Prilikom zatvaranja veze se šalju čak četiri poruke:

1) FIN strane A
2) ACK strane B
3) FIN strane B
4) ACK strane A

Sledeće je polje **Window** = **2 bajta.**
Kada je TCP prvi put projektovan, zamišljeno je da prva strana šalje jedan bajt, druga strana potvrđuje primanje tog bajta, pa prva strana šalje drugi bajt, druga strana potvrđuje prijem drugog bajta, itd.
To je danas prevaziđeno jer bi trajalo čitavu vječnost.
Zbog toga se ne potvrđuje svaki bajt, nego svaki N-ti bajt. Strane u komunikaciji se dogovaraju koliko podataka mogu međusobno da prime, prije nego što potvrde prijem tog poslednjeg bajta. To je zapravo brzina kod TCP-a = veličina upisana u Window.

Window postoji I sa jedne I sa druge strane. Obe strane imaju opciju da jave koliko bajtova mogu da prime prije nego što pošalju AN drugoj strani.
Ta vrijednost upisana u polje Window se dinamički mijenja.

Primjer:
Prva strana šalje 1 bajt, dobije AN od druge strane. Onda šalje 2 bajta, dobije AN od druge strane. Pošalje 4 bajta, dobije AN od druge strane. Pošalje 8, ne dobije AN od druge strane. Onda smanji broj bajtova koji šalje, pa proba sa 7, I čeka AN od druge strane, itd..

Sledeće je polje **Checksum = 2 bajta**.
Služi za verifikaciju teksta.

U polje Checksum se spušta vrijednost koja se dobije sabiranjem svih bajtova u segmentu, računajući header I korisne bajtove.
Druga strana može da izračuna tu vrijednost na isti način. Ako se te dvije vrijednost poklapaju, znači da je to taj segment koji se šalje, ako se ne poklapaju, onda se odbacuje taj segment.

Sledeće je polje **Urgent** (hitno) = **2 bajta.**

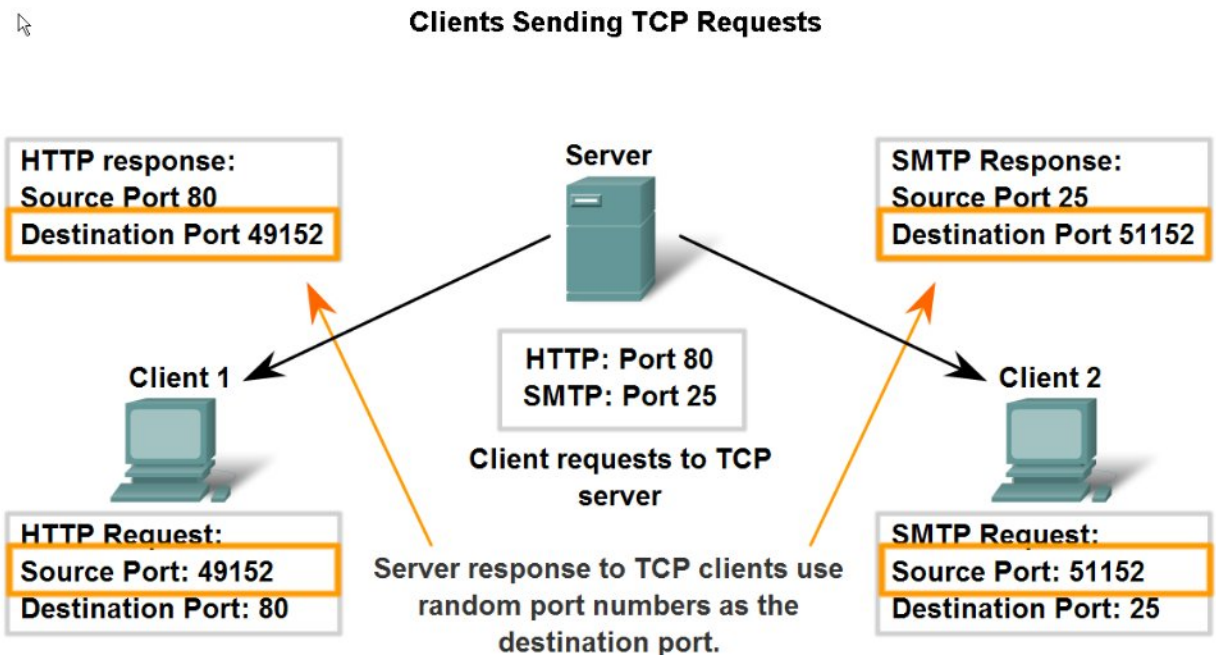
Ovo polje se usko povezuje sa QoS (quality of service).
Ovim poljem se naglašava da se neki segment treba procesirati prije drugog.

UDP ima portove, checksum I veličinu zaglavlja, pošto I veličina UDP headera može biti promjenljiva, iako je uglavnom 8 bajtova.
Ako se UDP Datagram izgubi, on je nepovratno izgubljen,, neće se ponovo slati.

TCP Serveri

Jedan server ne može imati isti broj porta za dvije svoje različite usluge!

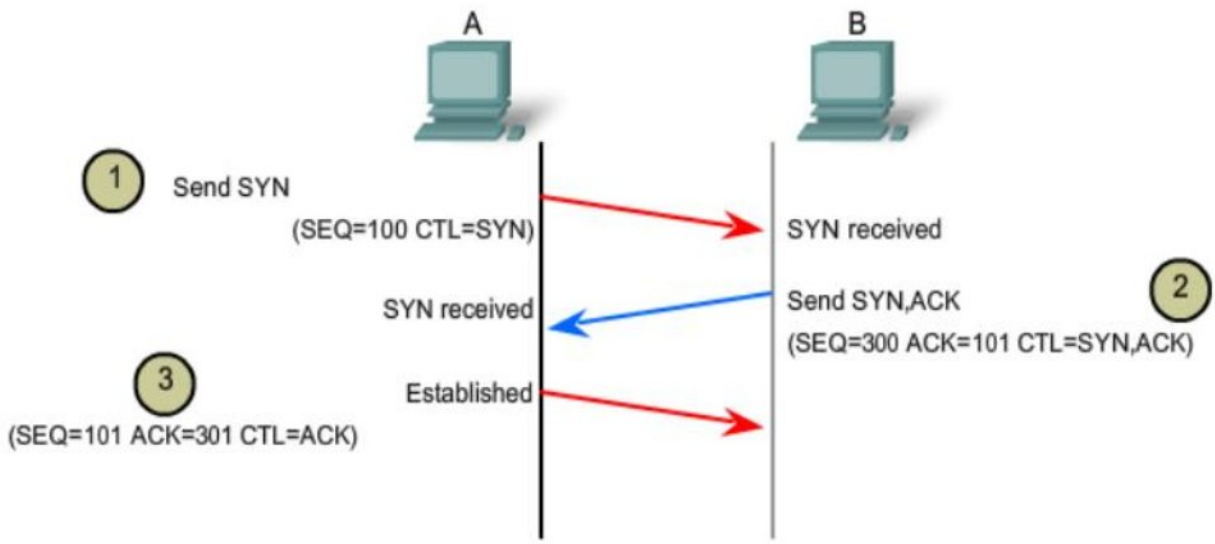
Sigurnost servera se povećava ako se odobri pristup samo autorizovanim korisnicima tih usluga



Uspostavljanje konekcije:

Mehanizam koji se koristi za uspostavljanje konekcije je three-way-handshake.

Ovo podrazumjeva da se korisni bajtovi nikad ne odbacuju dok se ne dobije potvrda od druge strane da je primila te bajtove.



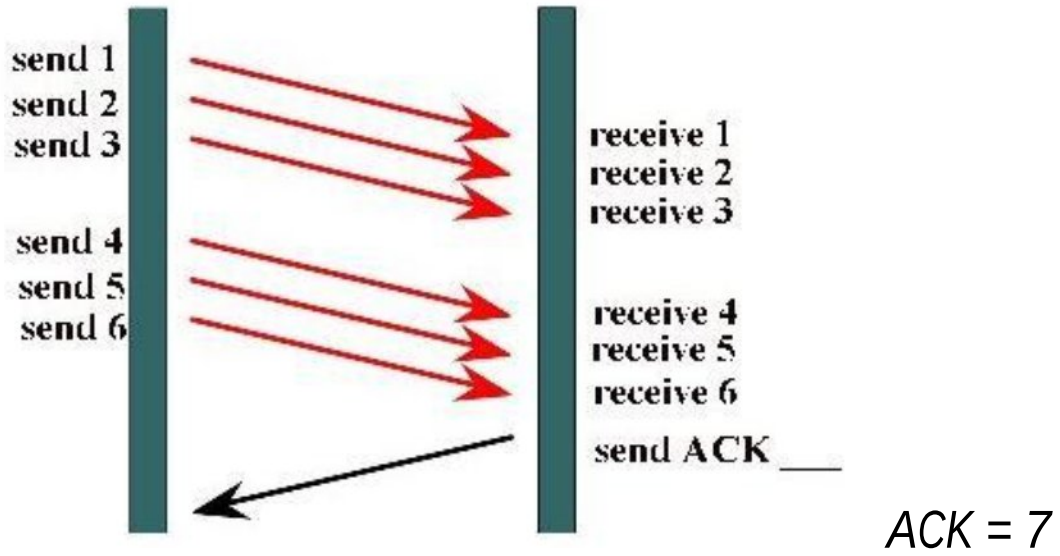
Strana A obavjestava stranu B da želi da upostavi TCP vezu. TWH omogućava da se strane u komunikaciji dogovore o početnim rednim vrijednostima bajtova.
Strana A prvo šalje prazan segment (samo TCP header), u kojem je postavljen jedan od flagova, a to je SYN. SYN se upotrebljava samo kad se želi uspostaviti veza sa drugom stranom.
Druga strana ako je spremna da uspostavi TCP vezu odgovara slanjem praznog segmenta, kod kojeg je takodje SYN flag setovan.
SYN se upotrijebi samo jednom sa obe strane komunikacije.
Flag ACK (acknowledgement) (NE MIJESATI SA Acknowledgment number, on ima 4 bajta, dok flag ACK ima vrijednost ili 0 ili 1).
ACK flag se postavlja na 1 uvijek kada se potvrđuje prijem nekih poruka.
Strana B kada šalje prazan segment prema strani A u kojem potvrđuje želju za uspostvljanjem TCP veze, osim postavljenog SYN bita, ima setovan I ACK flag.
Ako strana B nema postavljen SYN bit, on ipak ima postavljen ACK flag, jer potvrđuje da je dobio prazan segment od strane A.

Sve se ovo radi zbog uspostavljanje početnih brojeva.
Host A postavi svoj SN(npr. 100). Host B kad odgovara, u svoje polje SN može staviti bilo koju vrijednost(npr. 300), ali u polju AN stavlja vrijednost 101 (ako je segment došao neoštećen do hosta B).
Tada host A u SN stavlja 101, a u polje AN stavlja 301.

Reasembliranje segmenta je lako uraditi, jer postoje Sequnce Numberi I znamo koliko je u svakom segmentu poslato bajtova, pa ih je lako poredati na odredištu.

- U originalnoj TCP implementaciji host šalje bajt, stavlja njegovu kopiju u red za ponovno slanje i starta tajmer. Ako se primi potvrda, bajt se briše iz reda. U suprotnom, ponovo se šalje.

- Potvrđivanje svakog bajta vremenski zahtjevno => ideja da se primi sekvenca bajtova pa potvrdi samo zadnji (windowing)
- Odredišni host koji koristi TCP potvrđuje samo podatke u neprekidnom nizu bajtova (potvrđuju se samo bajtovi koji kompletiraju niz)
- Na primjer, ako se prime bajtovi sa sequence brojevima od 1500 do 3000 i od 3400 do 3500, ACK broj bi bio 3001.



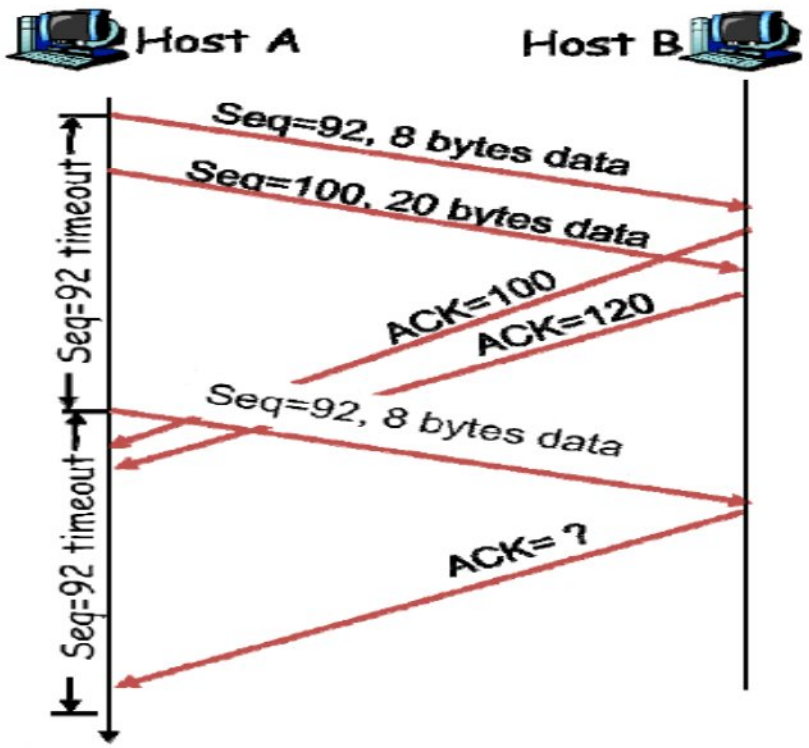
window size – broj bajtova nakon kojih se očekuje potvrda

ACK *number* – redni broj sljedećeg očekivanog bajta

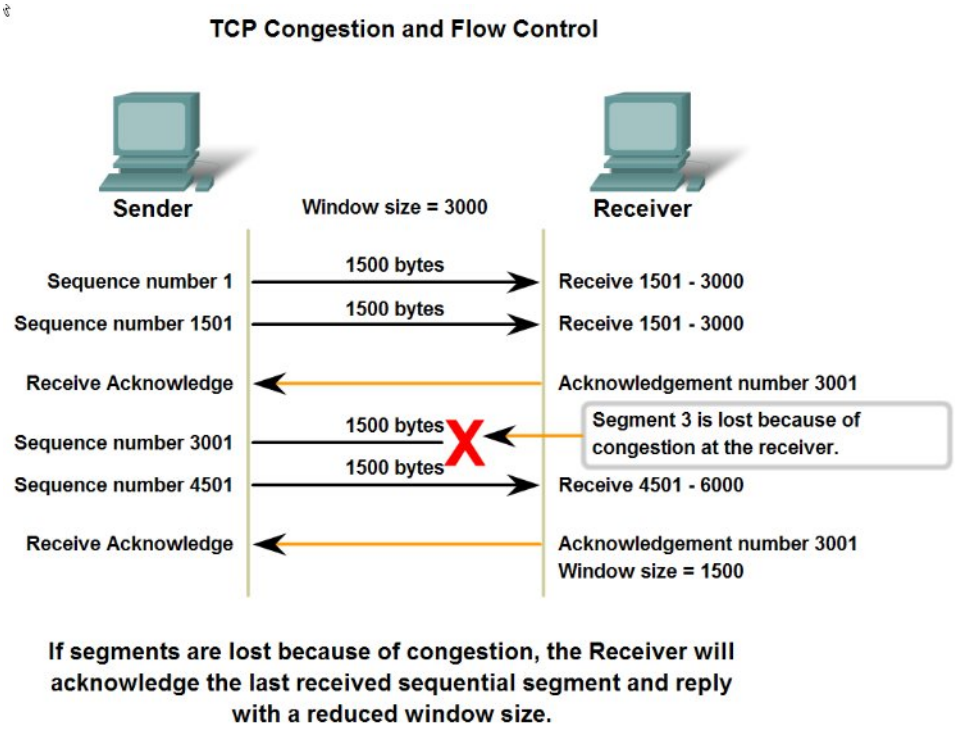
Ne potvrđuje se svaki bajt
već samo šesti (window size = 6)!

Pošiljalac odgovara sa
rednim brojem bajta
kojeg sljedećeg očekuje!

Ne miješati flag ACK (0-1) sa
poljem ACKnowledgment
Number ($0 – 2^{32}-1$)!



Host A šalje dva segmenta, prvi sa 8 bajtova korisne informacije, a drugi sa 20 bajtova korisne informacije. Nakon slanja svakog ovog segmenta, Host A aktivira tajmere u okviru kojeg vremena očekuje ogovor od Hosta B. Ako Host A u okviru tog vremena ne dobije potvrdu o prijemu tog segmenta od strane hosta B, onda on automatski šalje ponovo taj segment.



Kontrola toka se odnosi na brzinu kojom hostovi komuniciraju, a brzina se definiše vrijednosti prozora.
Kontrola zagušenja se odnosi na kanal, odnosno na kontrolisanje kanala preko kojeg hostovi komuniciraju.