Računalniške komunikacije

2020/21

transportna plast

sprotno potrjevanje, TCP

Pridobljeno znanje s prejšnjih predavanj

- **usmerjanje**
  - decentralizirani (podazdeljeni) algoritmi (usmerjanje z vektorji razdalj)
    - posredovalne tabele se računajo iterativno z minimizacijo vsote cene do soseda in ocenjenje cene sosedove razdalje do cilja
    - principa: good news travel fast, bad news travel slow
- **transportna plast - splošno**
  - uporaba vrat in vtičev
  - izvorna in ciljna vrata (16-bitni podatek v datagramih)
  - napad portscan
- **protokol UDP**
  - nabor storitev, prednosti in slabosti
  - primernost uporabe
- **konstrukcija protokola TCP s končnimi avtomati**
  - osnovna funkcija (pošiljanje in prejemanje)
  - reševanje iz napak pri prenosu (ACK in NAK, ponovno pošiljanje)
  - izguba paketa, uvedba časovne kontrole
  - izguba potrditve, reševanje podvojenih paketov
  - prekratek časovni interval, reševanje podvojenih paketov in potrditev

4. Izboljšava: posredno potrjevanje (samo ACK)

- enak učinek potrjevanja dosežemo, če uporabimo samo **ACK, v katerega vključimo številko segmenta**, ki ga potrjujemo,

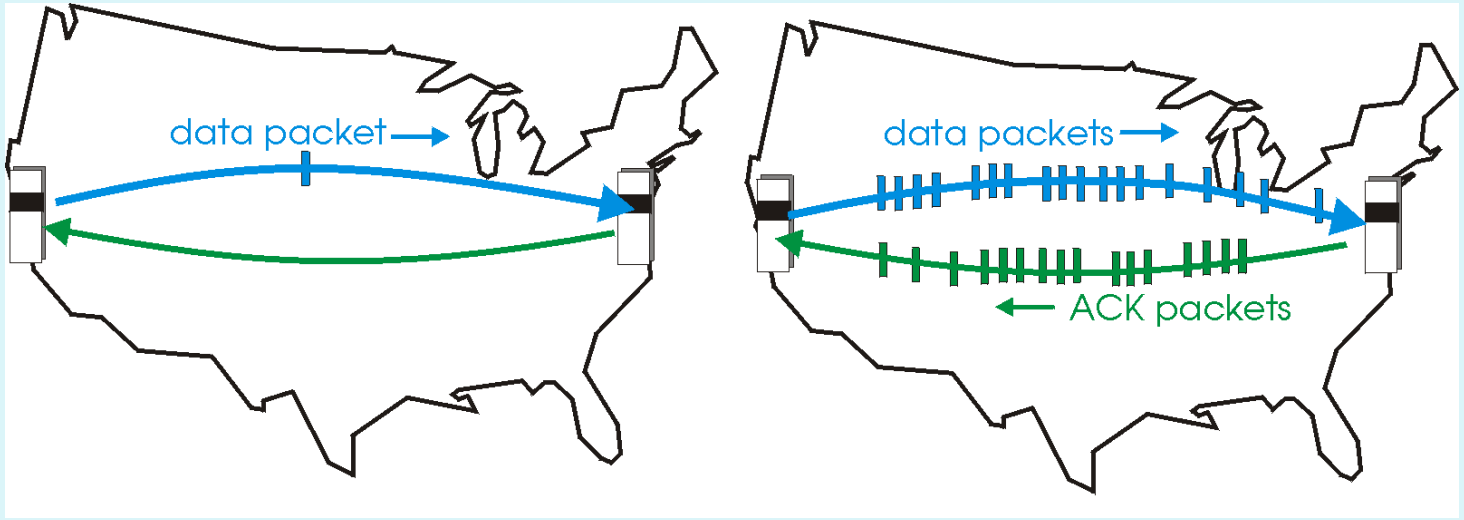
NEPOSREDNO POTRJEVANJE: uporaba ACK in NAK

POSREDNO POTRJEVANJE: uporaba samo ACK

- pri vsakem prejetem segmentu (pravilno sprejetem ali okvarjenem), prejemnik odgovori z ACK za **zadnji še uspešno prejeti segment**,
- če pošiljatelj prejme ACK za isti segment dvakrat, to pomeni, da segment, ki je sledil, ni bil sprejet pravilno (torej enako kot NAK)

5. Reševanje neučinkovitosti sprotnega potrjevanja

- problem: pošiljatelj, ki uporablja sprotno potrjevanje (*stop&wait protocol*), mora pred oddajo naslednjega paketa čakati na potrditev prejšnjega.
- ideja: namesto zaporednega pošiljanja, implementirajmo **vzporedno** oziroma **tekoče (cevovodno, *pipelined*) pošiljanje**



- ker lahko istočasno potuje več paketov, brez sprotnega čakanja na njihovo potrditev, je izkoriščenost kanala večja

5. Tekoče pošiljanje

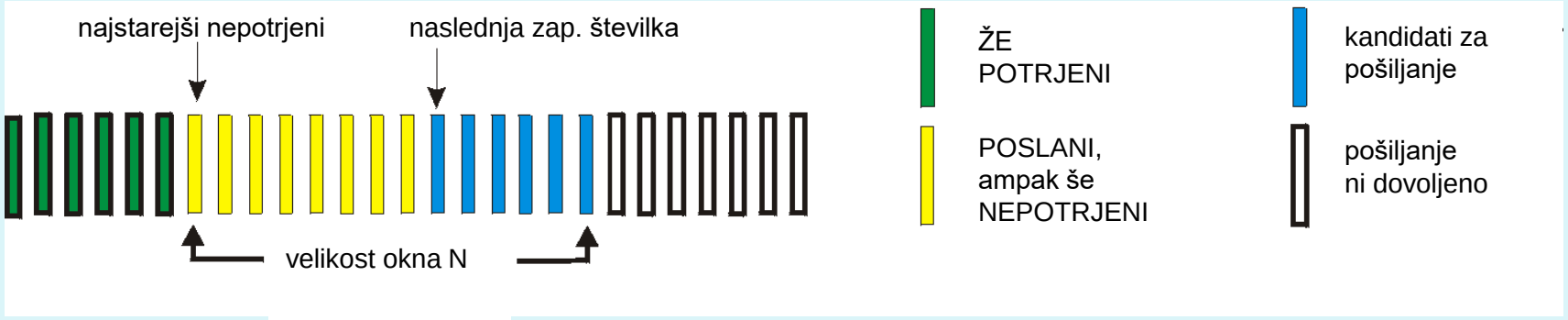
- potrebujemo:
  - večji razpon števil za številčenje paketov
  - shranjevanje paketov (pomnilnik) na strani pošiljatelja in prejemnika

- poznamo dve obliki protokolov za tekoče pošiljanje
  - ponavljanje **N nepotrjenih** (*go-back-N*)
  - ponavljanje **izbranih** (*selective repeat*)

5. Tekoče pošiljanje: **ponavljanje N nepotrjenih**

- pošiljatelj hrani "okno" največ dovoljenih nepotrjenih paketov
  - tak protokol imenujemo tudi **protokol z drsečim oknom**
- ko prejemnik pošlje ACK(n), potrdi s tem vse pakete do vključno n
- časovna kontrola za najstarejši paket
- ko časovna kontrola poteče, pošlji vse nepotrjene pakete v oknu ponovno

- DEMO: applet ← zelo dobra ponazoritev

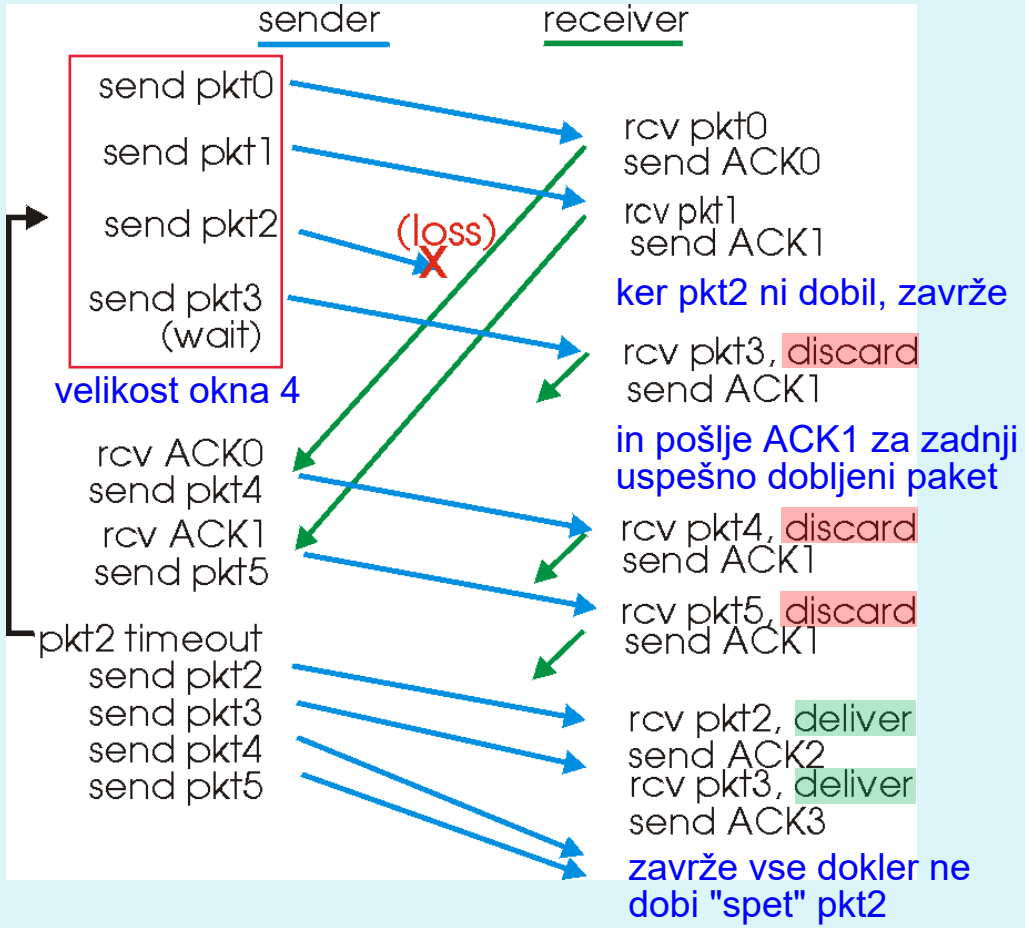


5. Tekoče pošiljanje: **ponavljanje N nepotrjenih**

velikost okna

- prejemnik lahko prejme **podvojene pakete**, ko pošiljatelj pošlje vse nepotrjene -> razpozna jih po zaporednih številkah in zavrže
- prejemnik lahko prejme pakete v **napačnem vrstnem redu** -> te zavrže, saj bodo poslani ponovno, ko poteče štoparica za najstarejšega

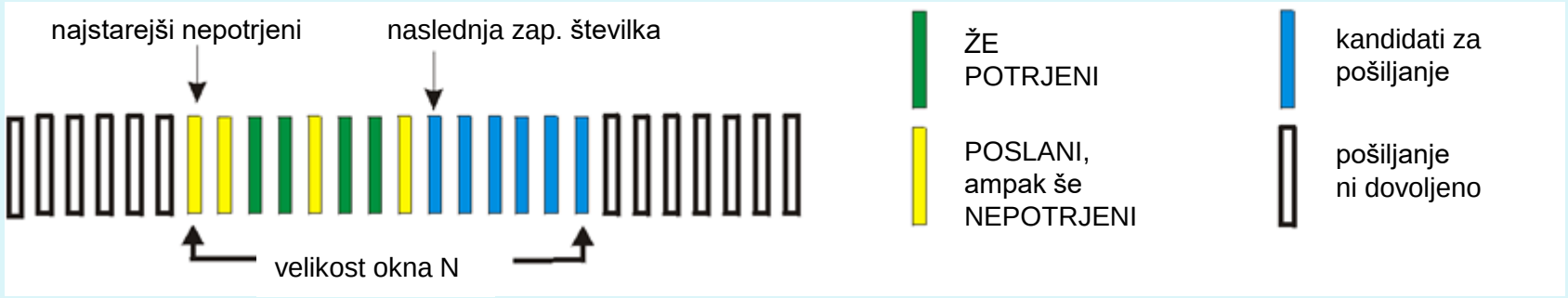
izguba potrditve ni kritična,
poglej si animacijo z prejšnjega slajda



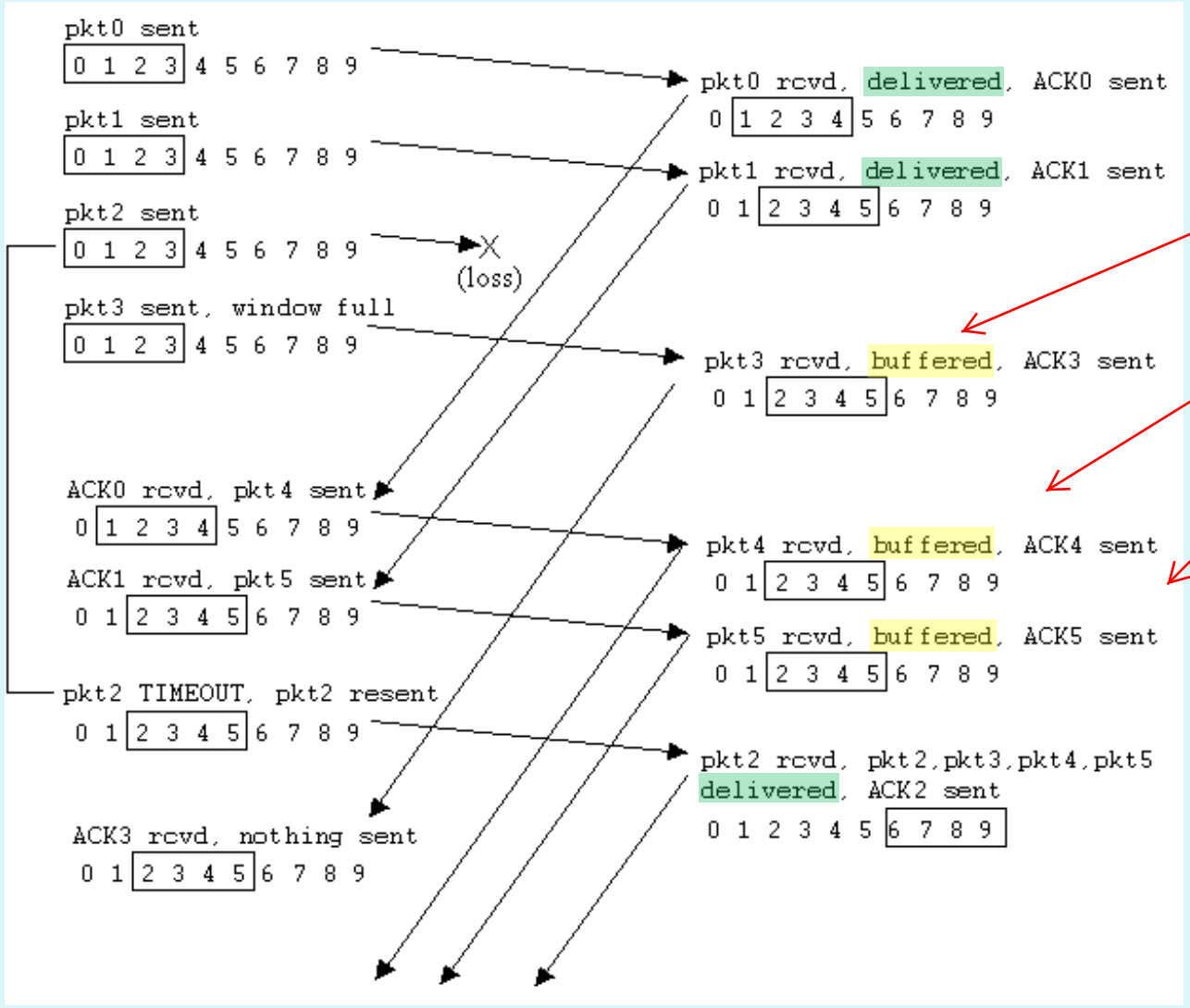
5. Tekoče pošiljanje: **ponavljanje izbranih**

- prejemnik potrjuje vsak prejeti paket posamezno
- prejemnik shranjuje pakete, prejete v napačnem vrstnem redu, in jih **sortira** pred dostavo aplikaciji
- pošljatelj **ponovno pošlje samo tiste** pakete, za katere ni dobil ACK
- pošiljatelj hrani **štoparico za vsak posamezni** nepotrjeni paket ← dodatna komplikacija protokola

- DEMO: applet



5. Tekoče pošiljanje: **ponavljanje izbranih**



pkt0, pkt1 ima, pkt2 nima, zato pkt3, pkt4, pkt5 shrani v buffer in čaka na pkt2

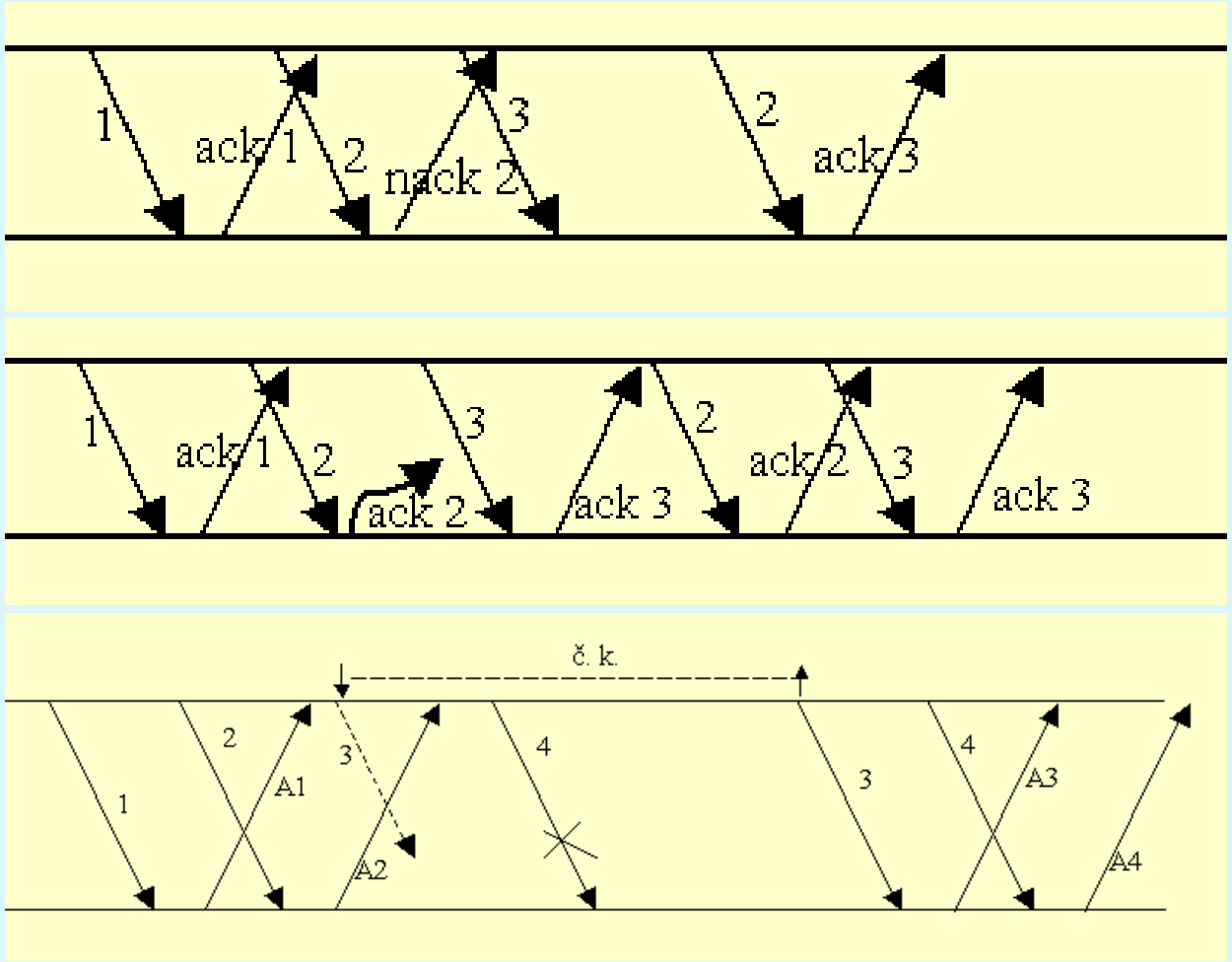
Potrjevanje

- možne so vse kombinacije potrjevanja
  - neposredno: ACK in NAK, posredno: samo ACK
  - sprotno: za vsak paket sproti, tekoče: z uporabo drsečega okna za več paketov

	SPROTNO	TEKOČE pošiljanje
NEPOSREDNO	✓	✓
POSREDNO	✓	✓

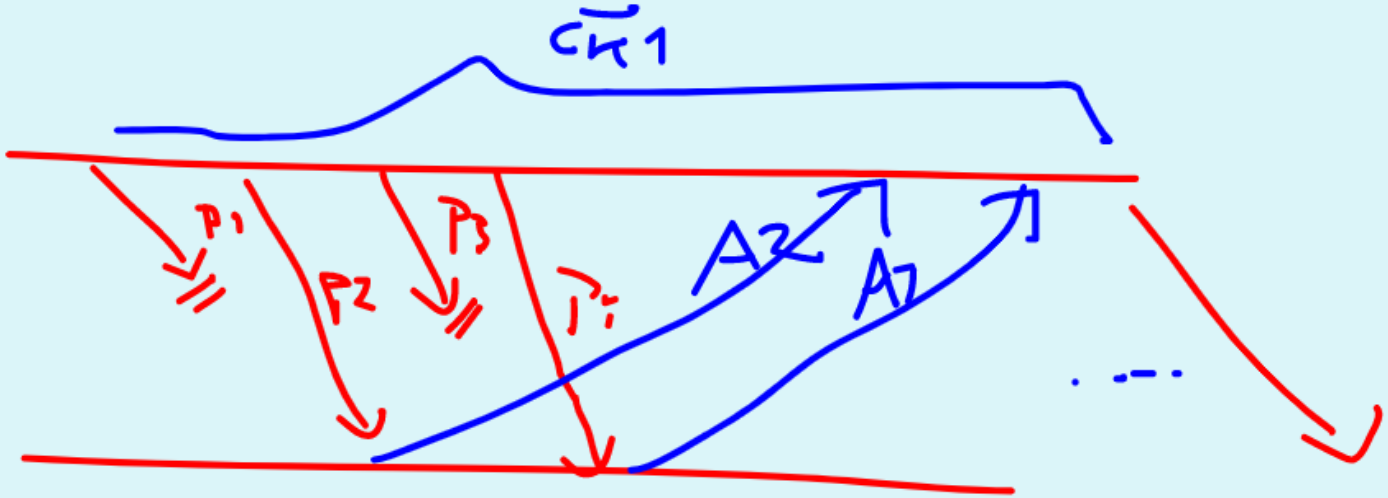
Vaja 1

- Za katere tipe potrjevanj gre v spodnjih primerih?

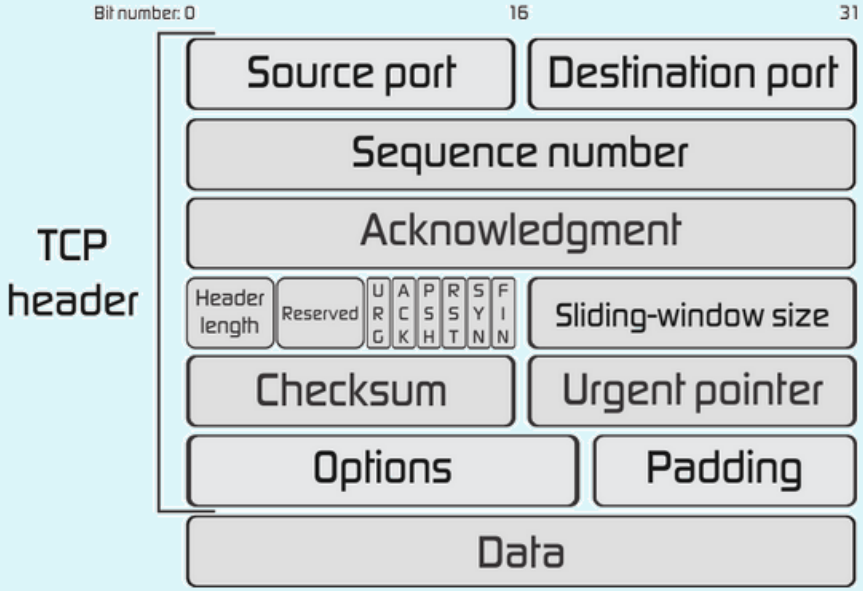


Vaja 2

- Uporabljamo tekoče pošiljanje 6 paketov s protokolom za ponavljanje le izbranih paketov. Širina okna je 4.
- Nariši shemo prenosov, če se izgubita 1. in 3. paket, po ponovitvi pa še potrditev 1. paketa?

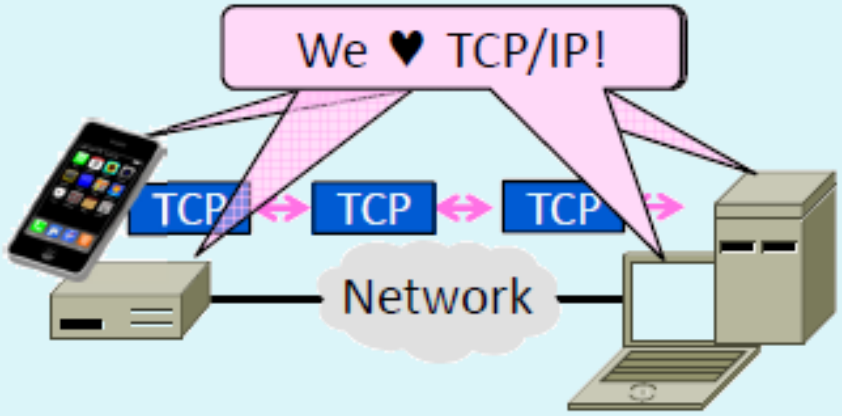


Protokol TCP

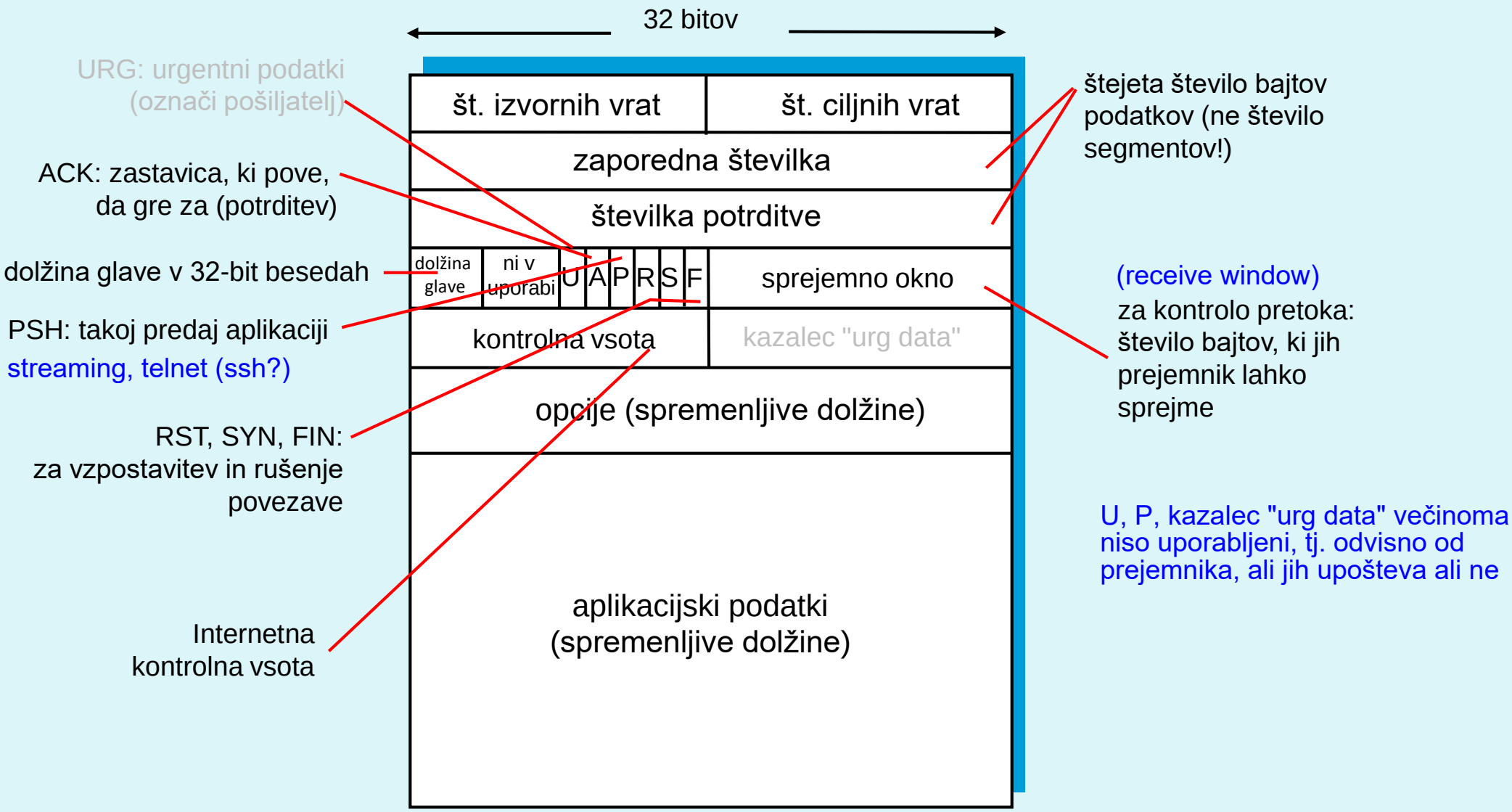


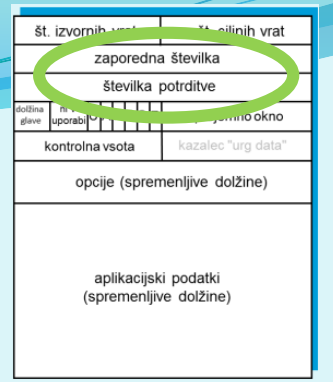
Lastnosti protokola TCP

- izvaja se med dvema točkama (point-to-point): **en pošiljatelj, en sprejemnik**
- je **povezavni protokol** (uporablja vzpostavitev/rušenje zveze)
- izvaja **dvosmerni promet** pri povezavi (full duplex, MSS)
- nudi **zanesljiv**, urejen tok podatkov
- ima **kontrolo pretoka** (angl. *flow control,* pošiljatelj ne preobremeni prejemnika)
- ima **kontrolo zasičenja** (angl. *congestion control,* pošiljatelj ne preobremeni omrežja)
- uporablja **tekoče pošiljanje**, velikost okna se avtomatsko določa glede na kontrolo pretoka in kontrolo zasičenja



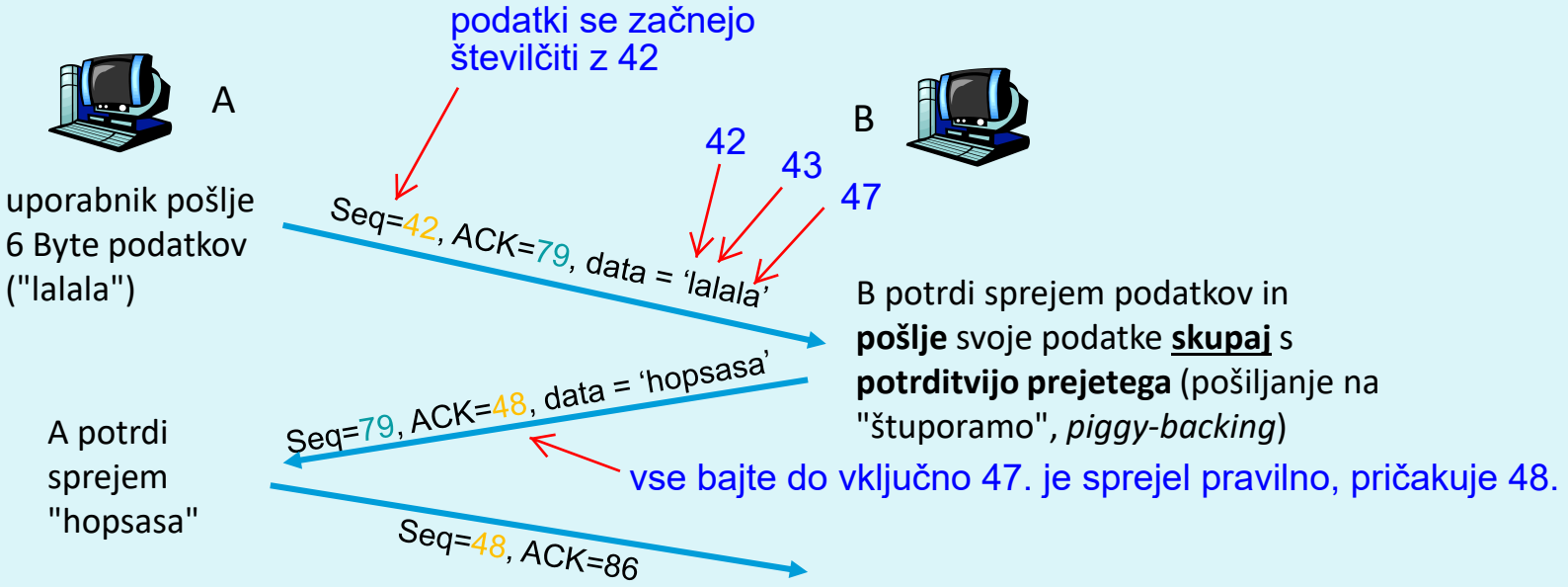
TCP segment

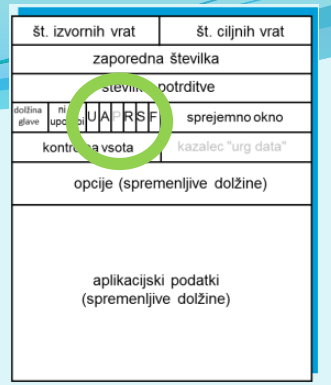

32 bitov
URG: urgentni podatki (označi pošiljatelj)
ACK: zastavica, ki pove, da gre za (potrditev)
dolžina glave v 32-bit besedah
PSH: takoj predaj aplikaciji
streaming, telnet (ssh?)
RST, SYN, FIN: za vzpostavitev in rušenje povezave
Internetna kontrolna vsota
št. izvornih vrat
št. ciljnih vrat
zaporedna številka
številka potrditve
dolžina glave
ni v uporabi
U
A
P
R
S
F
sprejemno okno
kontrolna vsota
kazalec "urg data"
opcije (spremenljive dolžine)
aplikacijski podatki (spremenljive dolžine)
štejeta število bajtov podatkov (ne število segmentov!)
(receive window)
za kontrolo pretoka: število bajtov, ki jih prejemnik lahko sprejme
U, P, kazalec "urg data" večinoma niso uporabljeni, tj. odvisno od prejemnika, ali jih upošteva ali ne



Številčenje segmentov in potrditev

- pošiljatelj in prejemnik najprej VZPOSTAVITA ZVEZO. Povezava je nato **dvosmerna** (vsak lahko pošilja drugemu)
- pošiljatelj lahko v enem segmentu **istočasno** pošlje **nove podatke in potrditev** (ACK) prejšnjega segmenta
- **številke pomenijo:**
  - **SEQ (zaporedna številka):** številka prvega Byte-a v segmentu
  - **ACK (potrditev):** številka naslednjega pričakovanega Byte-a



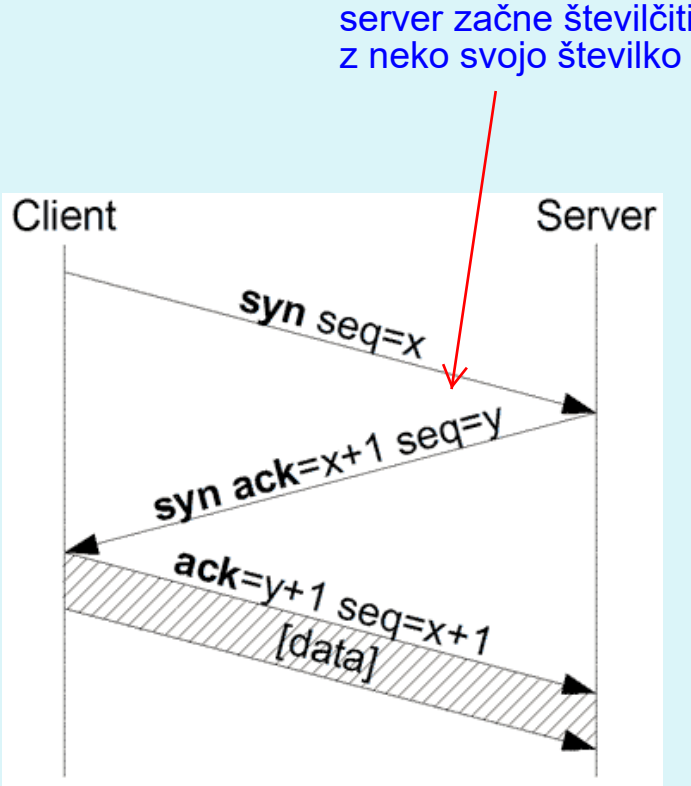


TCP: vzpostavljanje povezave

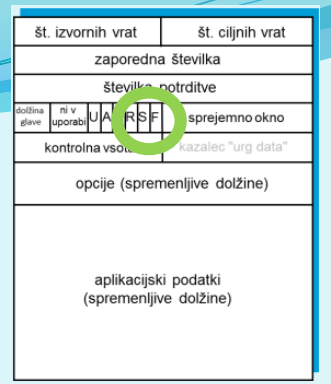
- pošiljatelj in prejemnik pred pošiljanjem izvedeta rokovanje (handshake), v katerem izmenjata parametre:
  - **začetne pričakovane zaporedne številke** (naključno določene)
  - **velikosti medpomnilnikov** (za kontrolo pretoka)

- trojno rokovanje (*three-way handshake*)

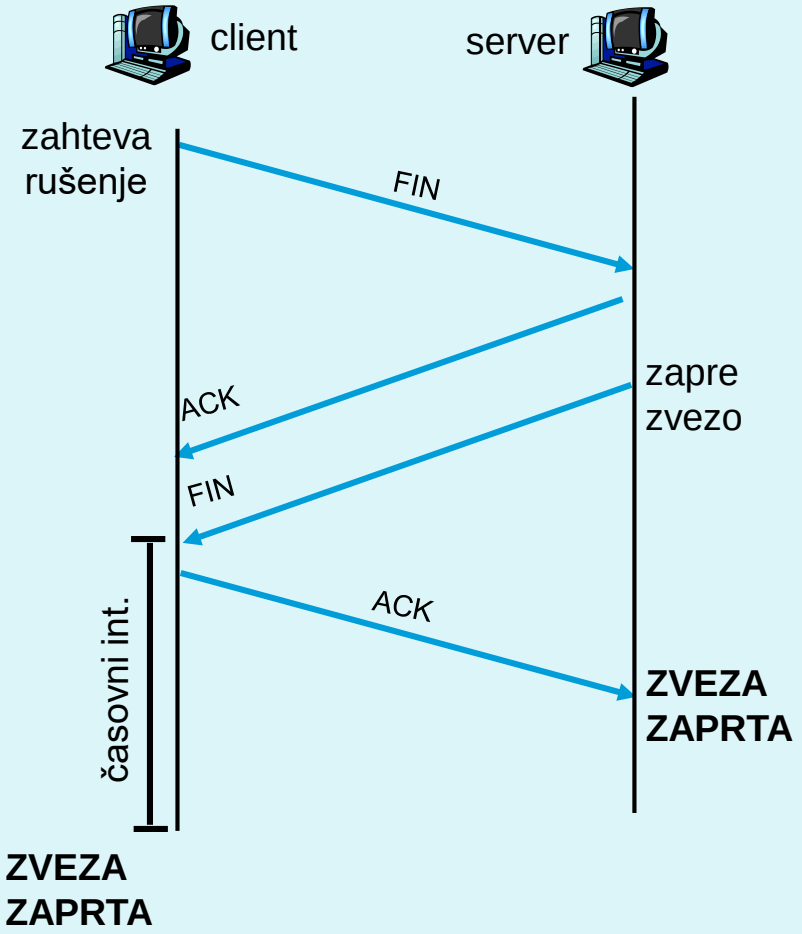
  1. Odjemalec pošlje segment z zastavico **SYN** (sporoči začetno številko segmenta, ni podatkov)
  2. Strežnik vrne segment **SYN ACK** (rezervira medpomnilnik, odgovori z začetno številko svojega segmenta)
  3. Odjemalec vrne **ACK**, lahko že s podatki ("štuporama")



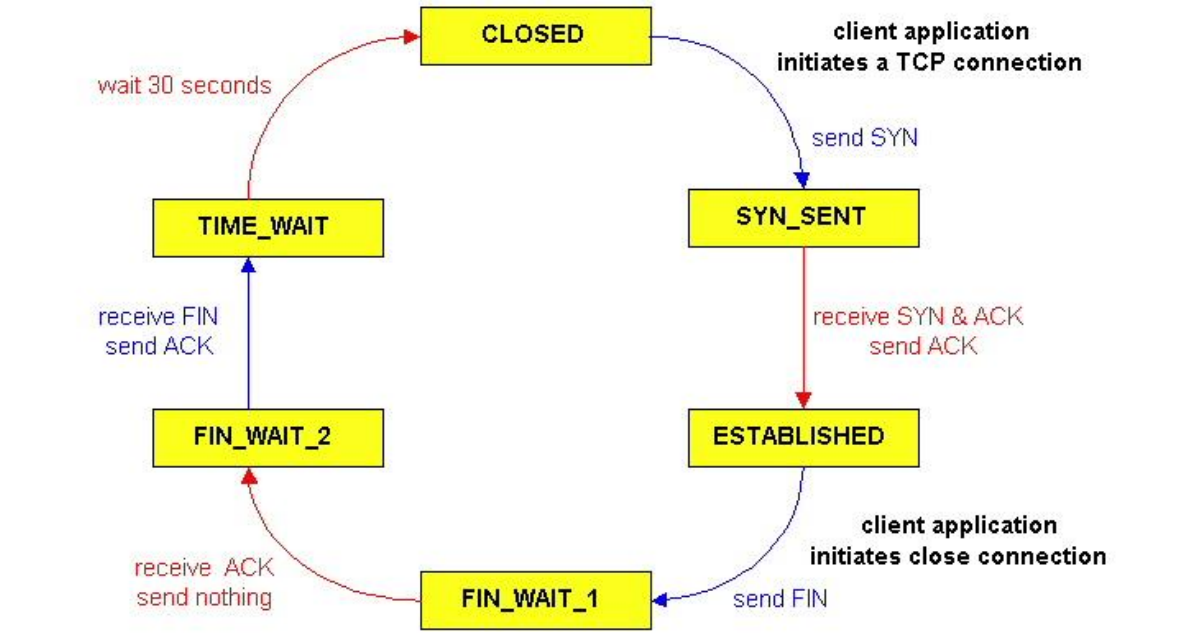
TCP: rušenje povezave



1. odjemalec pošlje segment TCP FIN strežniku
2. strežnik potrdi z ACK, zapre povezavo, pošlje FIN
3. odjemalec prejme strežnikov FIN, potrdi ga z ACK
   - počaka časovni interval, da po potrebi ponovno pošlje ACK, če se ta izgubi
4. strežnik sprejme ACK, končano

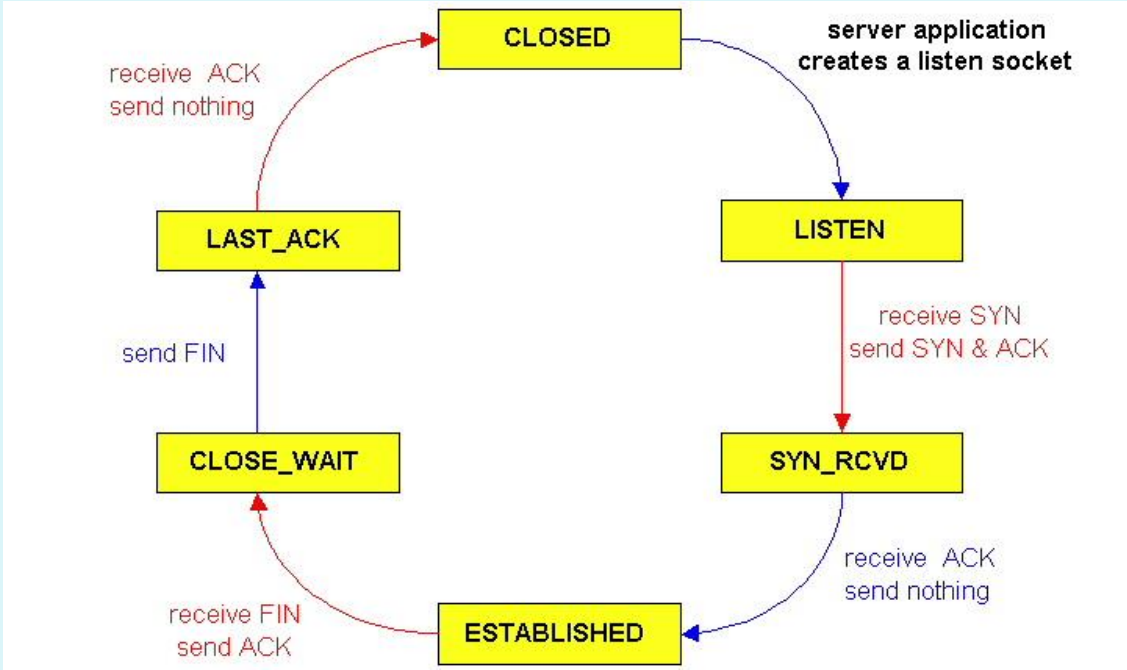


Življenjska cikla odjemalca in strežnika



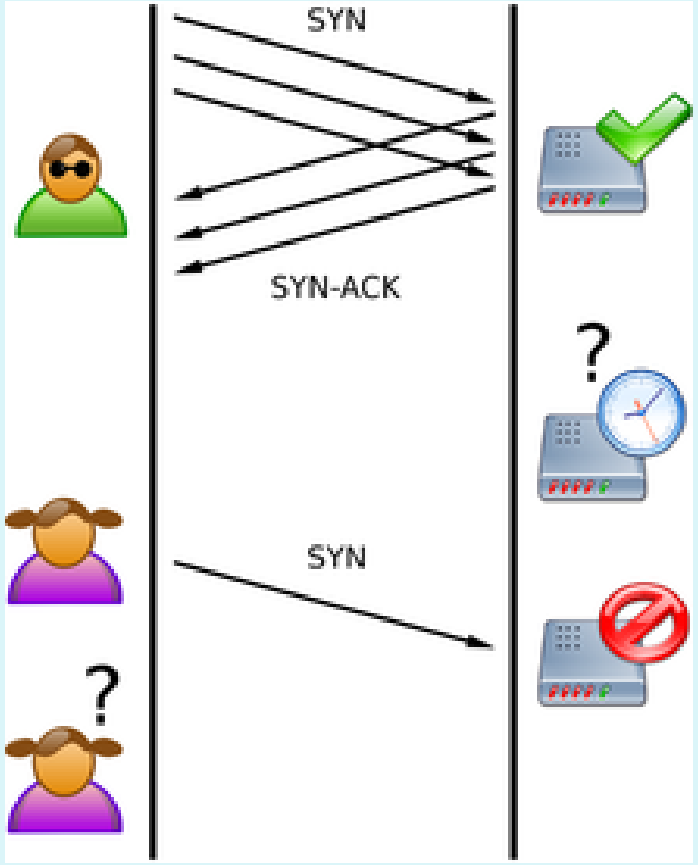
TCP odjemalec

TCP strežnik



Varnost: napad SYN FLOOD

- napad, v katerem napadalec pošlje strežniku veliko število paketov za vzpostavitev zveze (TCP SYN), pri čemer strežnik vsakič rezervira del svojega medpomnilnika
- pomnilnik ostane zaseden zaradi napol odprtih zvez (napadalec ne zaključi tretjega koraka rokovanja z ACK). Zaradi velikega števila odprtih povezav strežniku zmanjka prostora in pride do odpovedi sistema (angl. *denial of service*)
  - porazdeljeni DoS napad: pošiljanje TCP SYN iz več virov



Naslednjič gremo naprej!

- protokol TCP