TCP (Transmission Control Protocol)

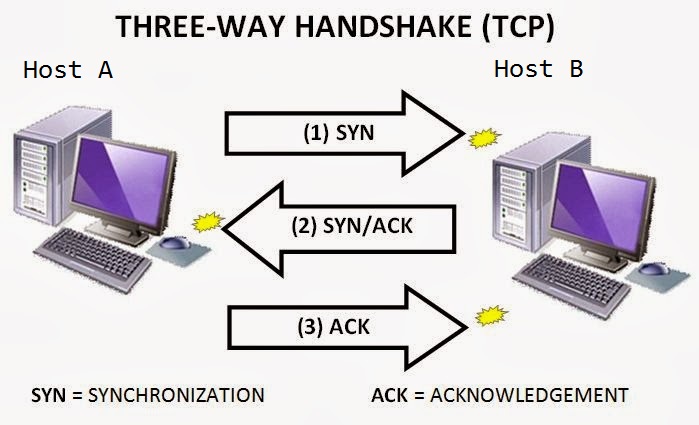
TCP je konekcijski orijentiran(connection-based) protokol koji nudi pouzdan(reliable) prijenos podataka između umreženih računala.

Definiran je od strane IETF-a (Internet Engineering Task Force) u RFC-u (Request For Comments) 793.

U OSI(Open Systems Interconnection) slojevitom modelu kao i u TCP/IP modelu TCP protokol djeluje na transportnom sloju (Transport Layer).

Konekcijski orijentiran znači da se između povezanih računala uspostavlja virtualna veza preko koje se onda šalju podatci.

Uspostavljanje konekcije u TCP-u vrši se tzv. „Three-Way Handshake“ procedurom.



Slika 1.0 Prikaz Three-Way Handshake procedure

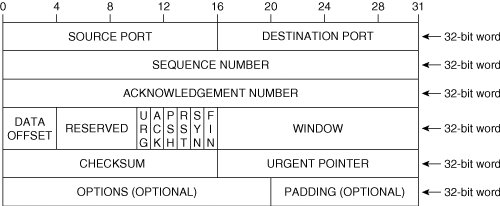
Three-Way Handshake procedura sastoji se iz 3 koraka:

1. Host A šalje hostu B zahtjev za povezivanjem(sinkronizacijom, SYN zastavica)
2. Zatim host B šalje hostu A potvrdu za zahtjev za povezivanjem (ACK zastavica), te i on šalje zahtjev za sinkronizacijom(SYN zastavica)
3. U zadnjem koraku host A šalje potvrdu hostu B za zahtjev za sinkronizacijom (ACK zastavica) i tada je veza uspostavljena

Pouzdan prijenos znači da TCP garantira da su poslani podatci ispravno preneseni, a ako se otkrije greška pri prijenosu vrši se retransmisija podataka.

Podatci se šalju u obliku datagrama.

TCP datagram sastoji se od korisničkih podataka na koje se dodaje zaglavlje TCP datagrama.



Slika 2.0 Header (zaglavlje) TCP datagrama

TCP koristi zastavice(flags) za kontrolu stanja konekcije.

TCP zastavice su 1-bitne boolean vrijednosti(1 – zastavica postavljena, 0 – zastavica nije postavljena).

Postoji 6 različitih zastavica:

1. ACK (Acknowledgment) – služi kao potvrda za uspješno primljene podatke
2. SYN (Synchronization) – služi kao zahtjev za povezivanjem (sinkronizacijom)
3. PSH(Push) – forsira slanje podataka iz buffer-a čak i ako buffer nije skroz pun
4. RST(Reset) – Zatvara konekciju zbog neke pogreške
5. FIN(Finish) – Zatvara konekciju
6. URG(Urgent) – Oznacava hitne podatke, ovi podatci se obrađuju prvi jer imaju najveći prioritet

Zastavice će nam biti izrazito korisne u slijedećem dijelu kada budemo hvatali pakete pomoću Wireshark-a.

IP (Internet Protocol)

Internet protokol je protokol koji služi za komunikaciju između hostova preko Internet mreže.

Podatci se preko IP-a šalju u obliku paketa između routera, ili u obliku frameova između switcheva.

Internet Protokol je nepouzdan što znači da ne garantira da će poslani podatci biti ispravno prenešeni, ili da će uopće biti dostavljeni.

U OSI(Open Systems Interconnection) slojevitom modelu IP protokol djeluje na mrežnom sloju (Network Layer), a u TCP/IP modelu na internet sloju (Internet Layer).

Svi uređaji unutar mreže identificiraju se pomoću ip adrese.

Postoje dvije vrste IP adresa: IPv4 i IPv6 adrese

IPv4 adrese sastoje se od 32 bita, ali često se označavaju sa 4 dekadska broja koja mogu imati vrijednosti od 0-255, i odvojeni su točkom. (npr. 192.168.0.1)

Koristeći 32 bita moguće je kreirati 2^32 = 4.294.967.296 različitih ip adresa.

IP adresa ima dva dijela: mrežni(net id) i host (host id) dio

Mrežni dio (Net Id) služi za identificiranje mreže i kreiranje podmreža.

Host dio (Host Id) služi za identificiranje pojedinih računala u mreži.

Postoji 5 klasa ip adresa: A, B, C, D, E

|  |  |  |  |
| --- | --- | --- | --- |
| Klasa | Raspon adresa | Broj podmreža | Broj računala |
| A | 0.0.0.0 – 127.255.255.255 | 126 | 16.277.214 |
| B | 128.0.0.0 – 191.255.255.255 | 16.384 | 65.534 |
| C | 192.0.0.0 – 223.255.255.255 | 2.097.152 | 254 |
| D | 224.0.0.0 – 239.255.255.255 |  |  |
| E | 240.0.0.0 – 255.255.255.255 |  |  |

Klase A, B i C koriste se u svrhe normalnog (unicast) adresiranja, klase D koriste se za multicast, a klase E su rezervirane za eksperimentalne svrhe.

Zbog masovnog povećanja broja korisnika (i još većeg broja različitih uređaja) na internetu

broj različitih IPv4 adresa (4.294.967.296) nije dovoljan za adresiranje svih uređaja, stoga su kreirane IPv6 adrese.

IPv6 adrese sastoje se od 128 bita, tako da nude mnogo veći broj različitih adresa koje možemo kreirati. (2^128 različitih ip adresa)

Opis TCP paketa uhvaćenih pomoću Wireshark alata

Server aplikacija pokreće TCP server na adresi 127.0.0.1 i portu 6666.

Chat aplikacija trenutno radi samo unutar lokalne mreže (localhost, 127.0.0.1).

Pomoću Wireshark alata uhvatio sam pakete koju su poslani pri jednostavnoj komunikaciji sa 2 klijenta.

Uhvaćene pakete spremio sam u wireshark\_logs/wireshark\_log.pcapng file.

Možete ih učitati unutar vašega Wireshark-a.

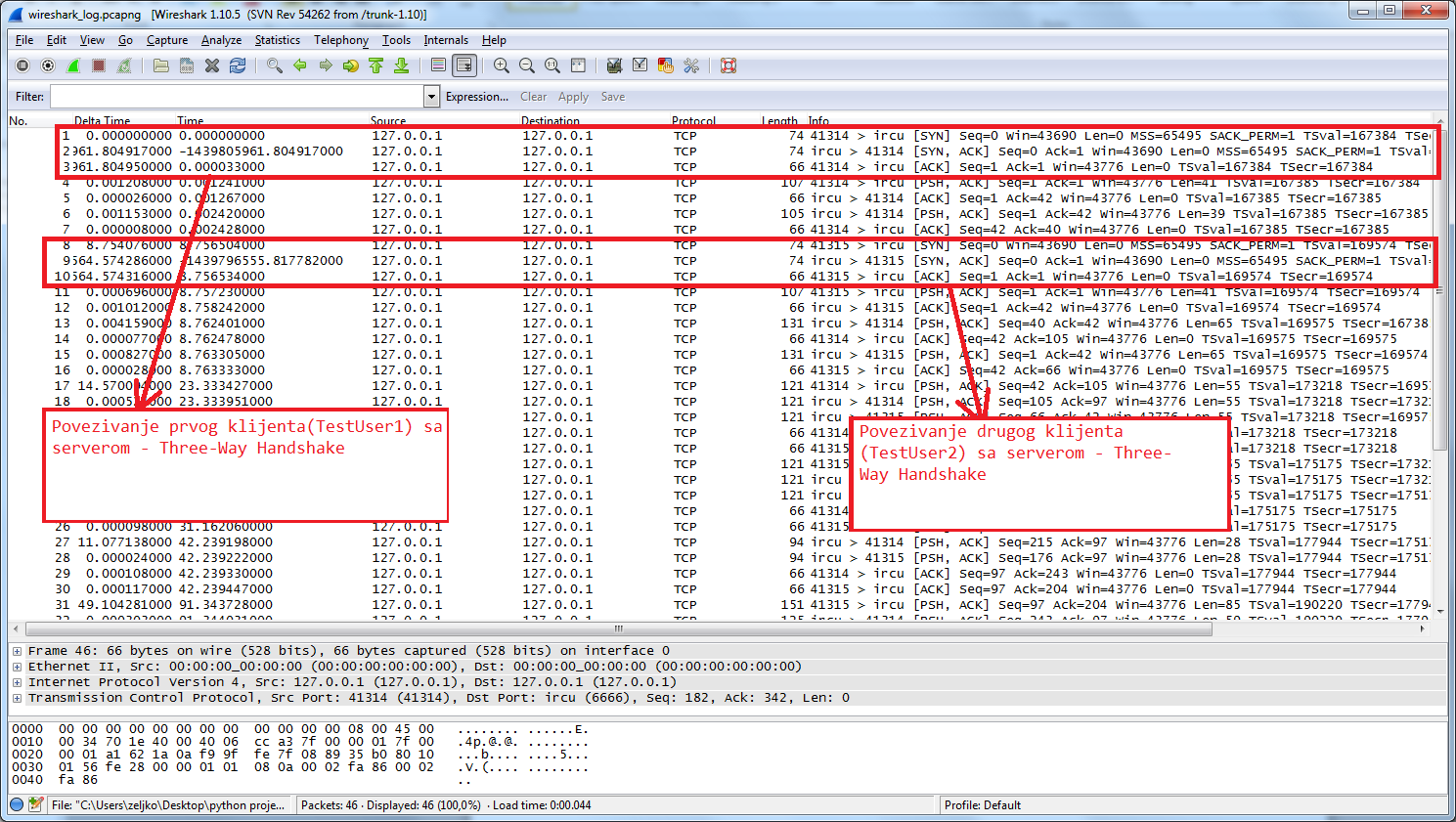
Kako bismo mogli hvatati samo pakete unutar localhost-a, u Wiresharku u Windows okruženju potrebno je instalirati Microsoft Loopback Adapter interface.

Ako ga nemate, Microsoft Loopback Adapter unutar Windowsa možete jednostavno instalirati prateći korake sa navedenog linka: <https://wiki.wireshark.org/CaptureSetup/Loopback>

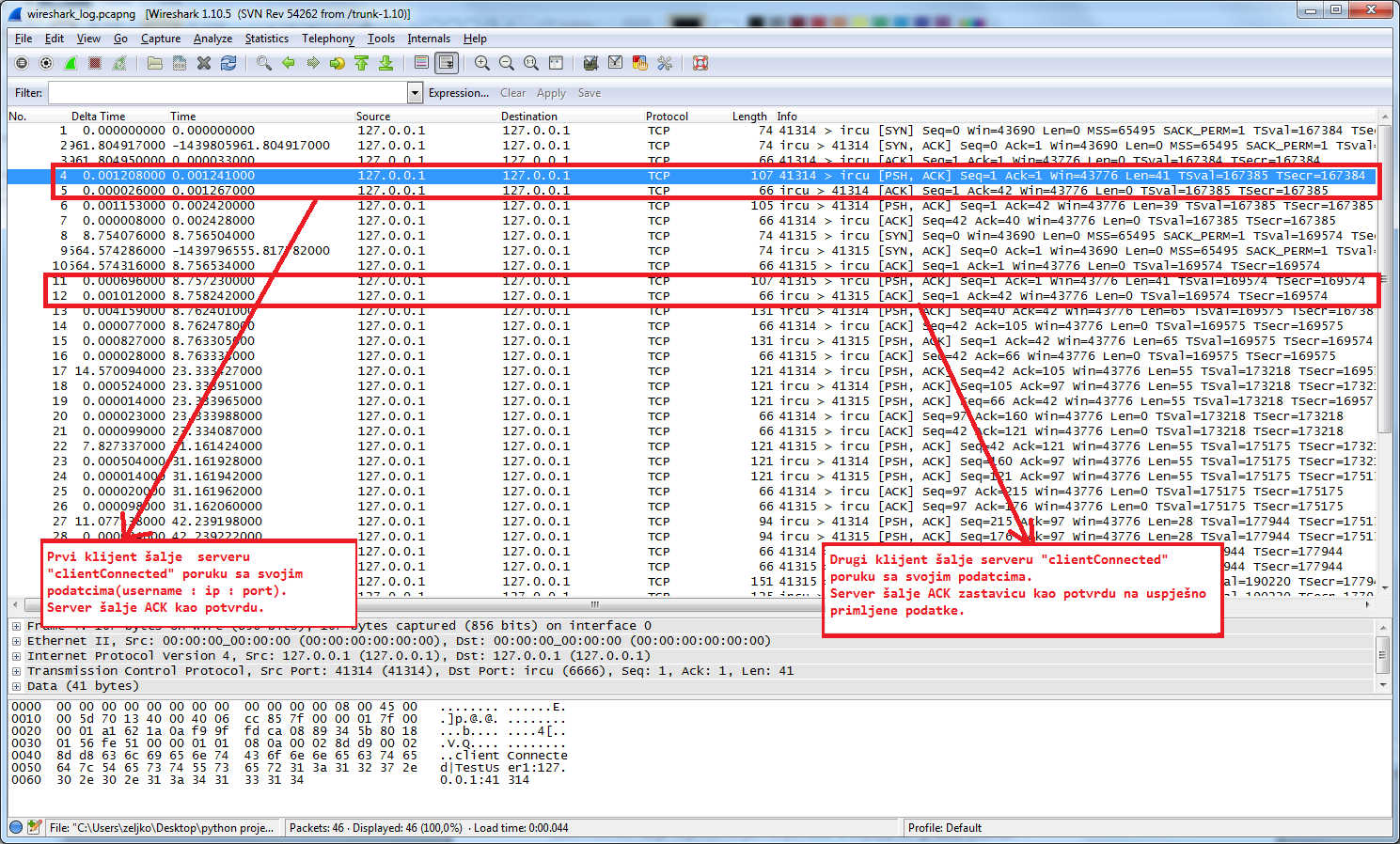
Nakon instalacije Loopback Adapter-a pokrenuo sam Wireshark i sa liste interface-a odabrao Microsoft Loopback Adapter koji sam prethodno instalirao i zapoceo hvatanje paketa.

Zatim sam pokrenuo Server aplikaciju, te nakon toga pokrenuo sam Client aplikaciju dva puta.

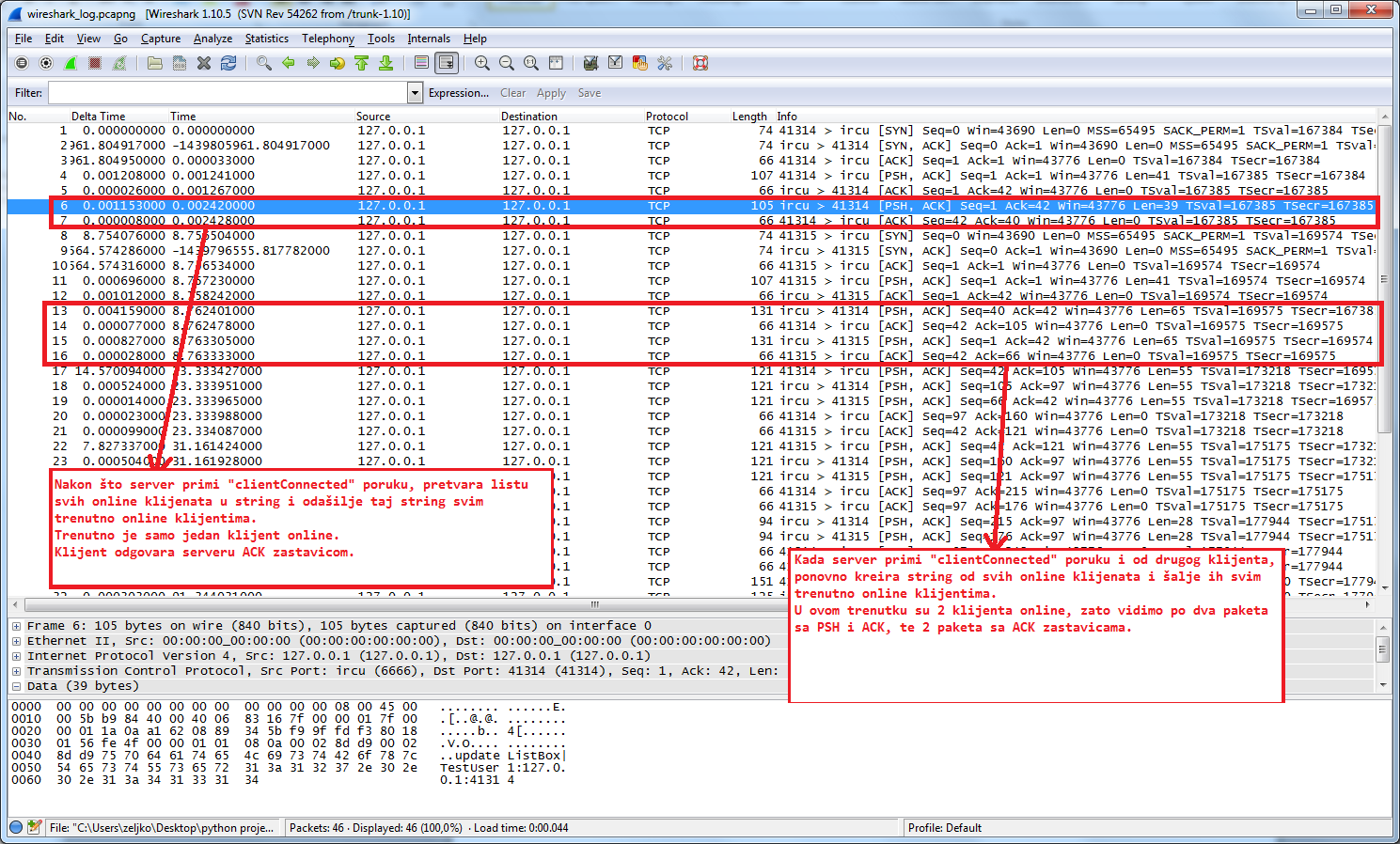
Prvom klijentu dao sam korisničko ime **TestUser1,** a drugome **TestUser2**, te klikom na „Connect“ gumb unutar klijentskih aplikacija vrši se povezivanje klijenta sa serverom Three-Way Handshake procedurom.



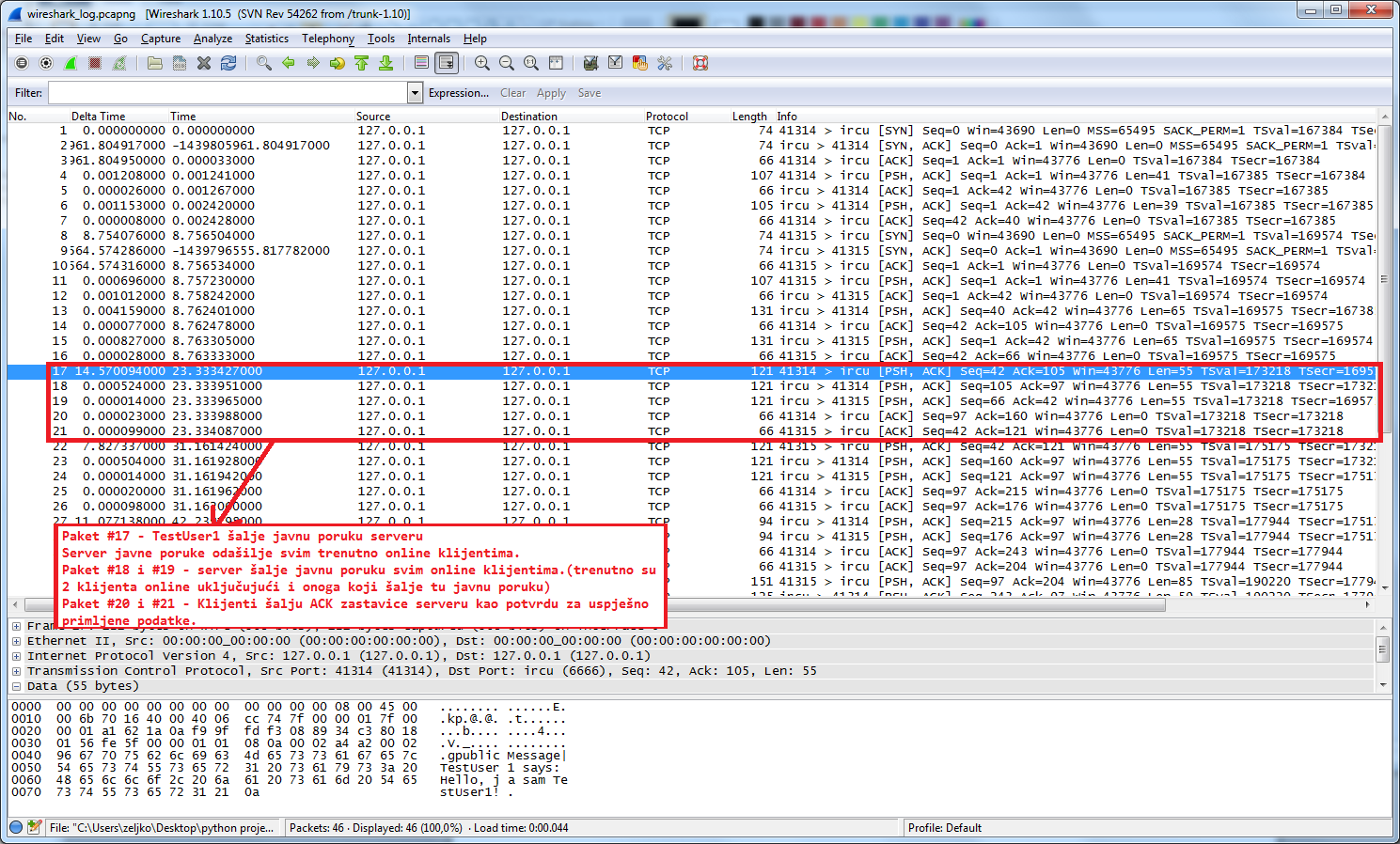
Ako se klijent uspješno poveže na server, klijentova aplikacija u pozadini šalje serveru „clientConnected“ poruku u kojoj se nalaze korisnički podatci kako bi server mogao dodati klijenta u listu trenutno povezanih klijenata.



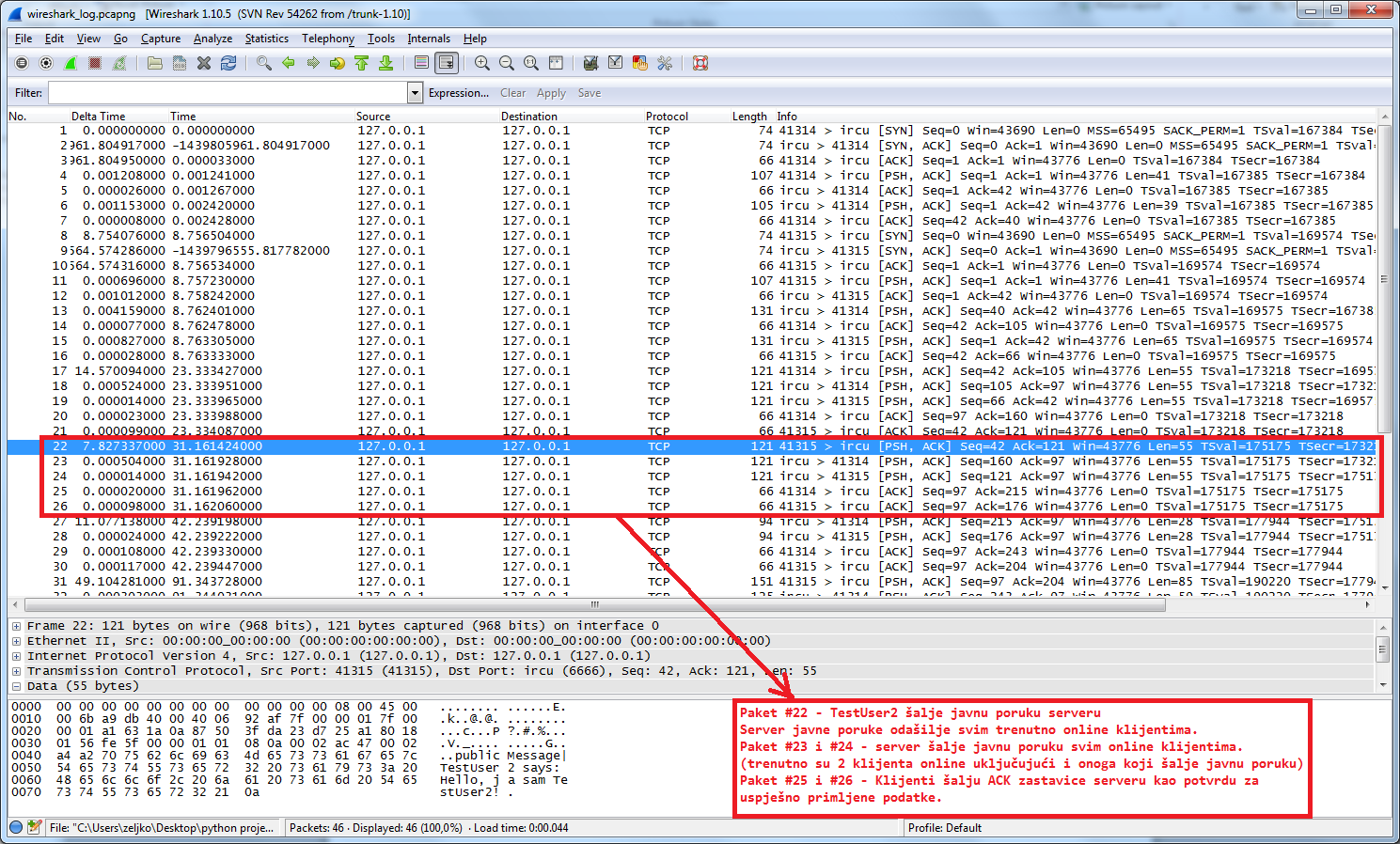
Kada server primi „clientConnected“ poruku od nekog klijenta, on pretvara listu povezanih klijenata u string i odašilje ju svim online klijentima kako bi mogli ažurirati svoje liste koje prikazuju online klijente(„updateListbox“ poruka).



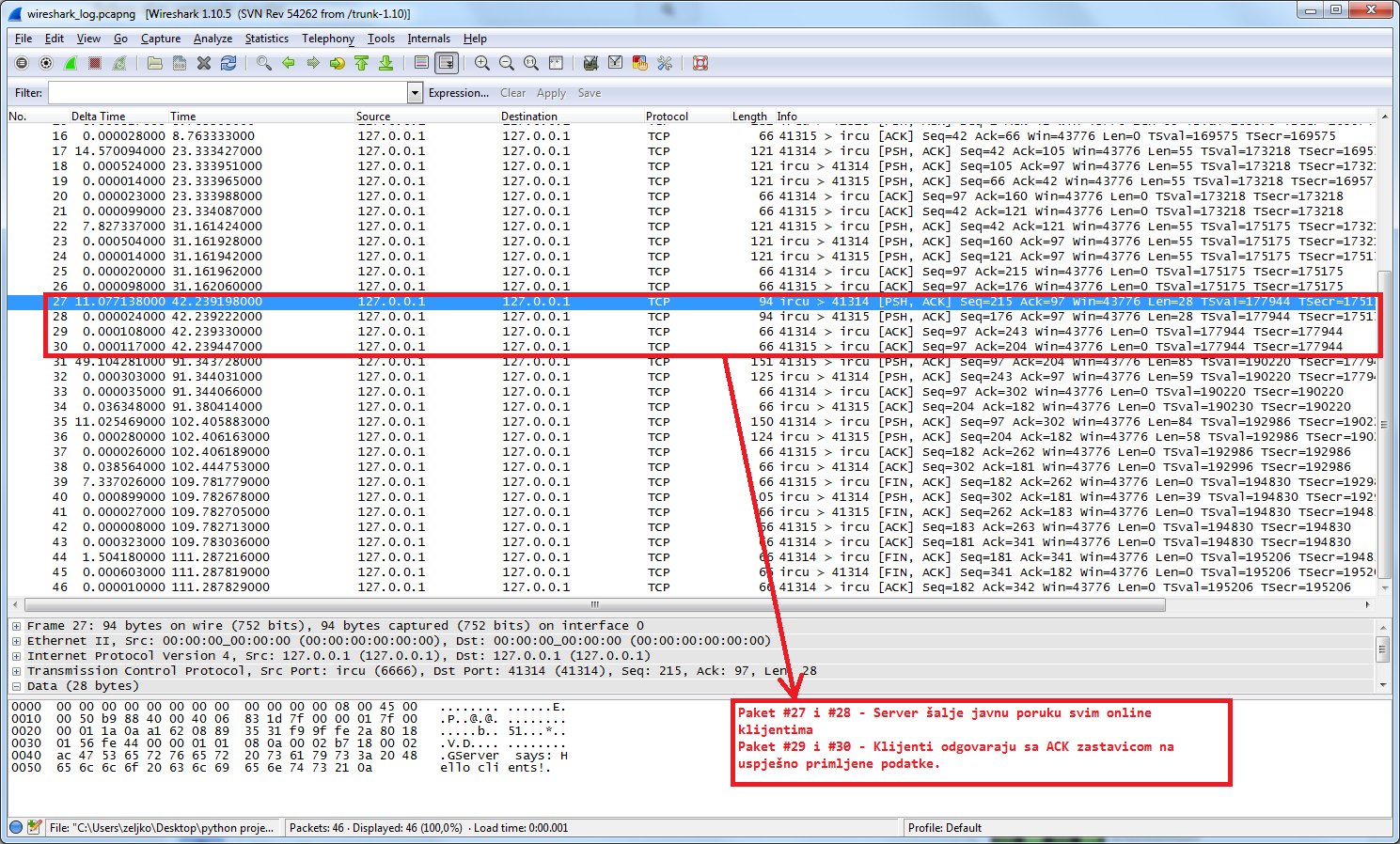
Kada se oba klijenta uspješno povežu sa serverom, **TestUser1** šalje javnu poruku u kojoj piše: „***Hello, ja sam TestUser1!“***.



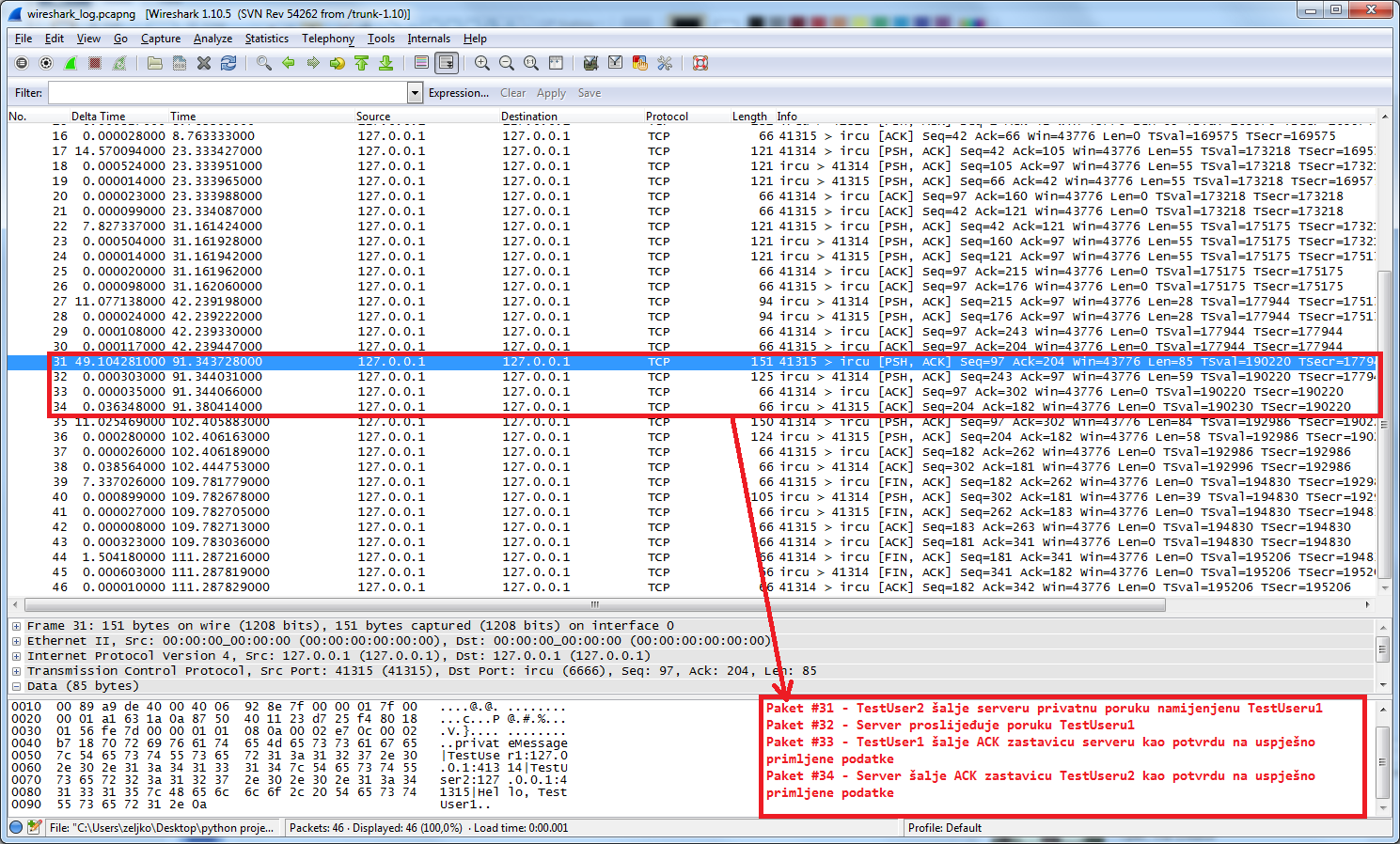
Nakon toga i drugi klijent, **TestUser2**  šalje javnu poruku u kojoj piše: ***„Hello, ja sam TestUser2!“***.



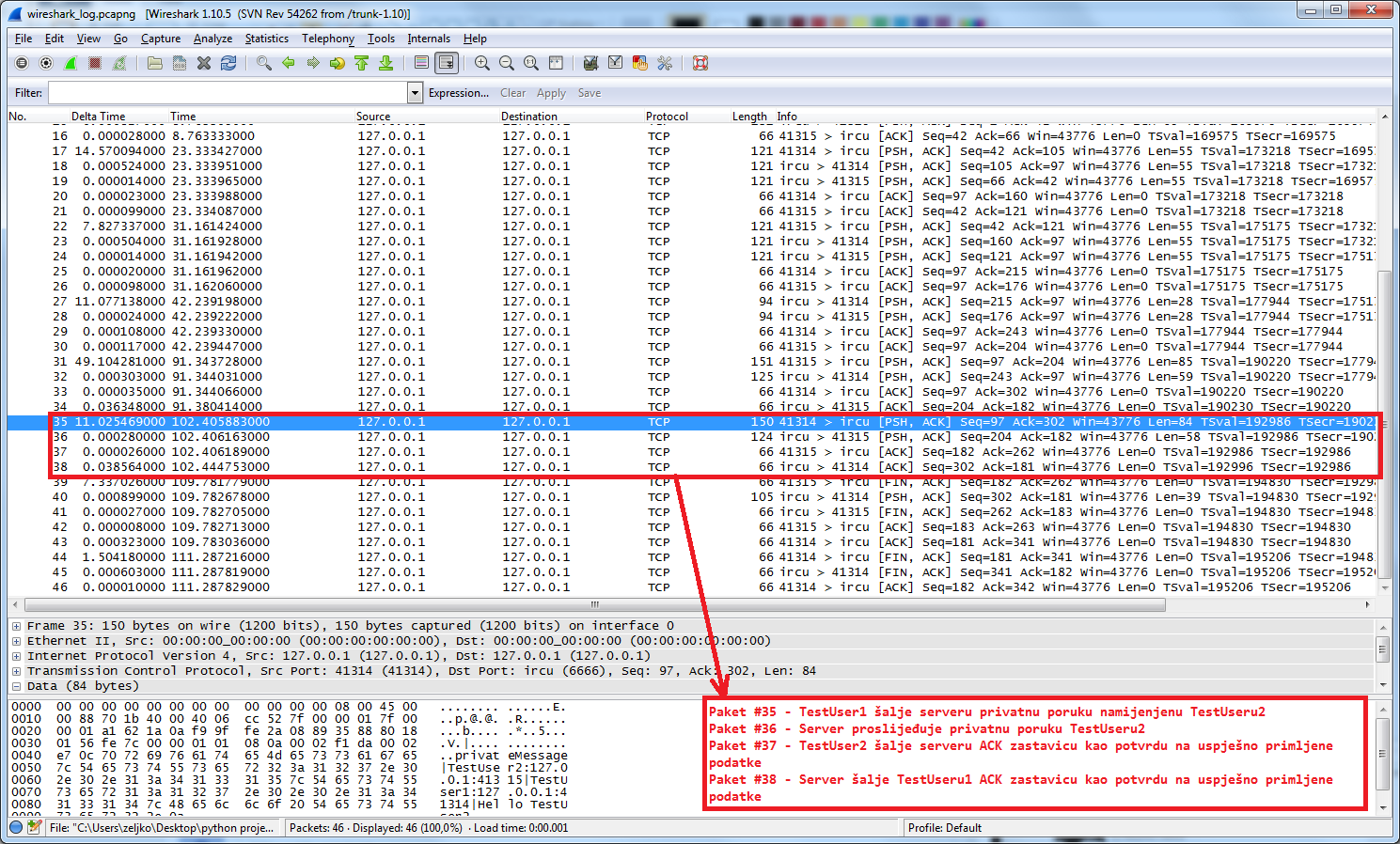
Zatim server šalje klijentima javnu poruku u kojoj piše: ***„Hello clients!“***.



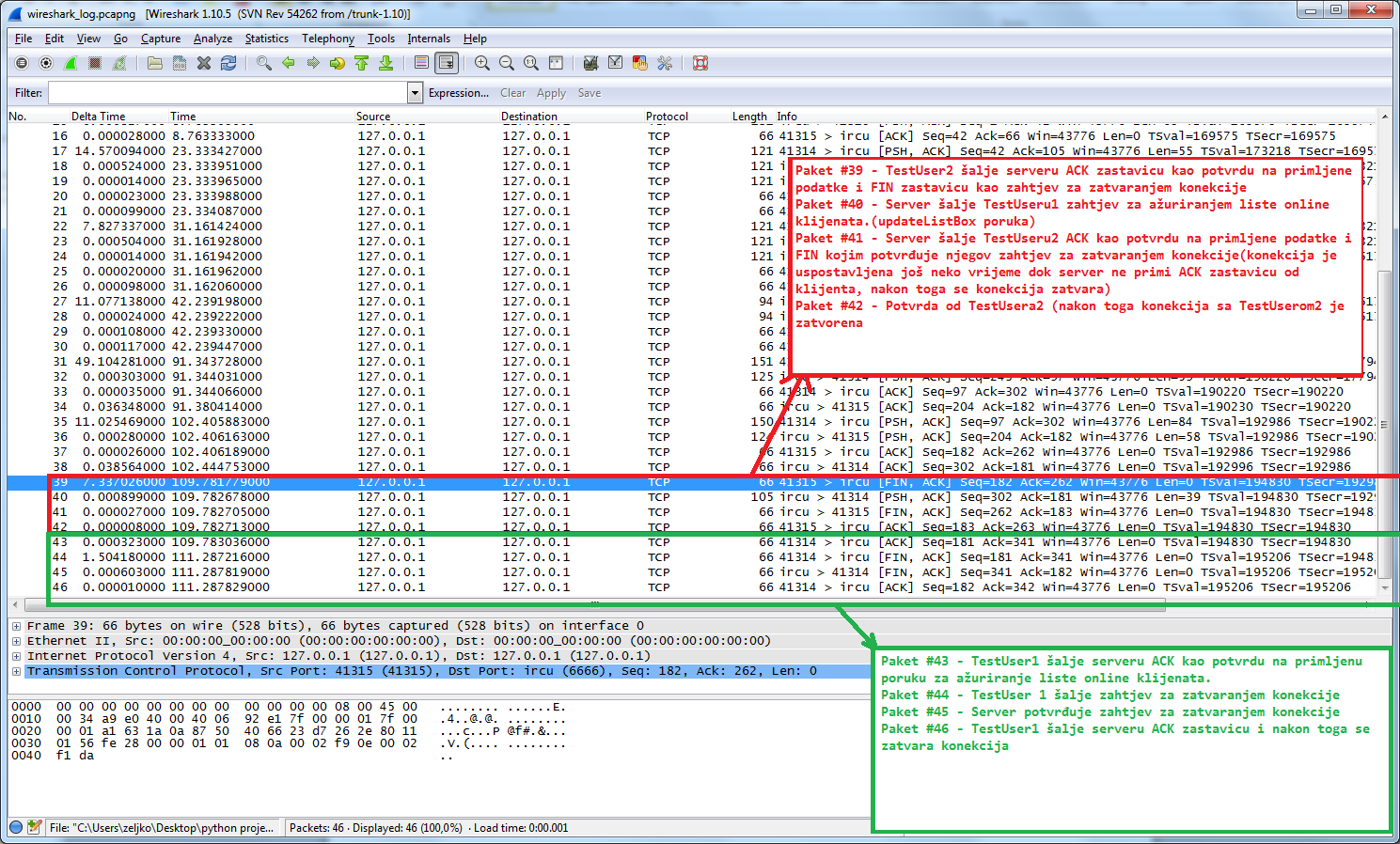
Poslije toga **TestUser2** šalje **TestUser-u 1** privatnu poruku u kojoj piše: ***„Hello TestUser1.“****.*



**TestUser1** odgovara privatnom porukom **TestUser2** u kojoj piše: ***„Hello TestUser2.“***



Nakon toga oba klijenta napuštaju chat i zatvara se konekcija sa serverom.

****