

Vežba 8 – Višeznačno upućivanje (engl. *Multicast*)

Cilj vežbe

Na prethodnim vežbama komunikacija je uvek bila tipa jedan-na-jedan (**unicast**): klijent i server su se međusobno adresirali IP adresom i brojem porta. Na ovoj vežbi uvodi se **multicast**, način komunikacije u kojem pošiljalac jednom porukom istovremeno dostavlja podatke čitavoj **grupi** prijemnika.

Po završetku vežbe student treba da:

- objasni razliku između *unicast*, *broadcast* i *multicast* komunikacije,
- prepozna *multicast* adresu (klasa D) i zna koji opseg adresa se koristi u lokalnoj mreži,
- objasni ulogu IGMP protokola pri učlanjenju u grupu i napuštanju grupe,
- implementira *multicast* pošiljaoca i prijemnika u programskom jeziku C (na Linux ili Windows OS-u),
- u Wireshark-u prepozna IGMP poruke i *multicast* pakete.

Teorijske osnove

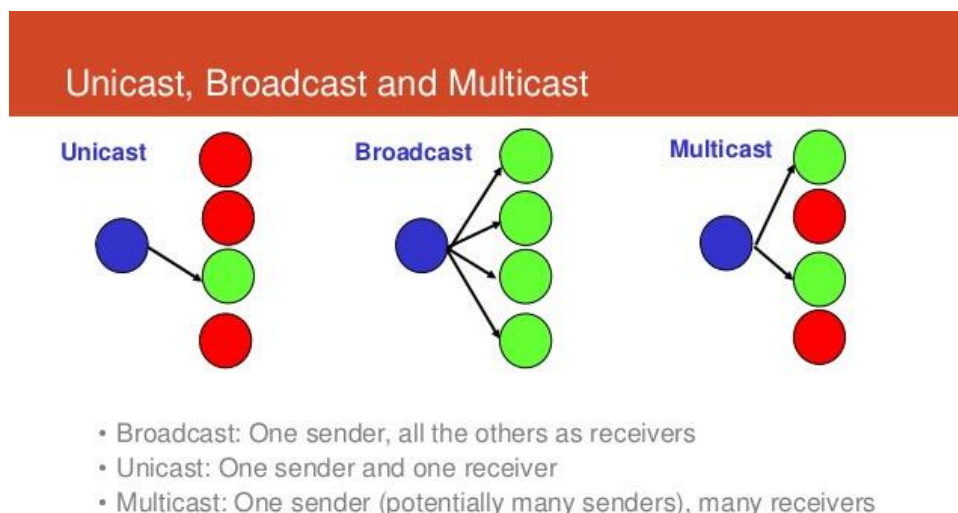
1. Načini isporuke podataka: *unicast*, *broadcast* i *multicast*

Kod *unicast* komunikacije svaki paket ima tačno jednog pošiljaoca i tačno jednog primaoca. U praksi, međutim, postoje aplikacije u kojima isti podaci treba istovremeno da stignu do većeg broja primalaca:

- **IPTV kod internet provajdera:** svaki TV kanal je jedna *multicast* grupa. Kanal se kroz mrežu provajdera prenosi jednom, bez obzira na broj gledalaca, a promena kanala na TV prijemniku (engl. *set-top box*) svodi se na napuštanje jedne grupe i učlanjenje u drugu.
- **Video nadzor:** IP kamera šalje jedan video tok, koji istovremeno primaju uređaj za snimanje, monitori u kontrolnoj sobi i softver za analizu slike.
- **Istovremena instalacija operativnog sistema na više računara** (npr. u računarskoj laboratoriji): slika diska se kroz mrežu šalje jednom, a primaju je svi računari odjednom.
- **Prenos predavanja, konferencija i obraćanja** unutar kompanije ili univerziteta, koje istovremeno prati veliki broj korisnika.

To se može uraditi na tri načina (Slika 1):

- **Unicast** (jedan-ka-jednom): pošiljalac šalje posebnu kopiju poruke svakom primaocu. Za N primalaca šalje se N kopija, pa opterećenje pošiljaoca i mreže raste sa brojem primalaca.
- **Broadcast** (jedan-ka-svima): poruka se šalje na *broadcast* adresu mreže (npr. 192.168.1.255) i stiže do **svih** računara u lokalnoj mreži, i do onih kojima nije potrebna. Svaki računar mora da je primi i obradi. *Broadcast* ne prolazi kroz rutere.
- **Multicast** (jedan-ka-grupi): pošiljalac šalje **jednu** kopiju poruke na adresu **grupe**, a mreža je isporučuje samo računarima koji su se prijavili u tu grupu.



Slika 1. Unicast, broadcast i multicast

Multicast grupa je skup prijemnika koje zanimaju isti podaci. Grupa se identifikuje jednom IP adresom (adresa grupe). Bitne osobine grupe:

- Pošiljalac šalje na adresu grupe i **ne zna** ko su članovi, niti koliko ih ima.
- Pošiljalac **ne mora da bude član** grupe da bi slao na nju.
- Prijemnici se sami **učlanjuju** u grupu i **napuštaju** je, u bilo kom trenutku, bez obaveštavanja pošiljaoca.
- Jedan računar može istovremeno biti član više grupa.

Multicast se može uporediti sa radio stanicom. Stanica emituje program na jednoj frekvenciji (adresa grupe) i ne zna ko je sluša. Slušalac bira stanicu podešavanjem prijemnika (učlanjenje u grupu) i može da je promeni kad god želi.

2. Multicast adrese (klasa D)

Za *multicast* grupe rezervisan je opseg IPv4 adresa **klase D**: od **224.0.0.0** do **239.255.255.255**. Prva četiri bita ovih adresa su uvek 1110. *Multicast* adresa se koristi **samo kao odredišna** adresa, dok je izvorna adresa paketa uvek *unicast* adresa pošiljaoca.

Opseg	Namena	Primeri
224.0.0.0 - 224.0.0.255	Kontrolni saobraćaj u lokalnoj mreži. Ruteri ove pakete nikad ne prosleđuju dalje.	224.0.0.1 (svi računari), 224.0.0.2 (svi ruteri), 224.0.0.22 (IGMPv3)

Opseg	Namena	Primeri
239.0.0.0 - 239.255.255.255	Administrativno ograničen opseg, za upotrebu unutar jedne organizacije (slično privatnim adresama 192.168.x.x).	239.1.7.1 (ova vežba)
ostale adrese klase D	Globalne grupe koje dodeljuje IANA i posebne namene.	224.0.1.1 (NTP)

Na vežbi se koristi isključivo opseg **239.x.x.x**. Adrese iz opsega 224.0.0.x ne treba koristiti za sopstvene aplikacije.

3. Multicast i transportni sloj

Multicast se realizuje preko **UDP** protokola. Kao što je navedeno u Vežbama 4 i 5, TCP zahteva uspostavu veze između tačno dve strane, pa ne podržava *multicast*. Zato *multicast* nasleđuje sve osobine UDP-a: nema uspostave veze, nema potvrde prijema, a poruke se mogu izgubiti ili stići drugim redosledom.

Odredište *multicast* poruke određuje par **adresa grupe + port**. Računar prima poruku samo ako je:

1. učlanjen u grupu, i
2. neka aplikacija na njemu sluša na odredišnom portu.

4. Multicast na nivou veze (Ethernet)

Kod *multicast* okvira odredišna MAC adresa se ne dobija ARP-om, već se računa direktno iz adrese grupe: prva tri bajta su uvek **01:00:5E**, a u poslednja 23 bita upisuju se poslednja 23 bita IP adrese grupe.

Primer: grupa **239.1.7.1** → MAC adresa **01:00:5E:01:07:01**.

Ovo je dovoljno da se *multicast* okviri prepoznaju u Wireshark-u.

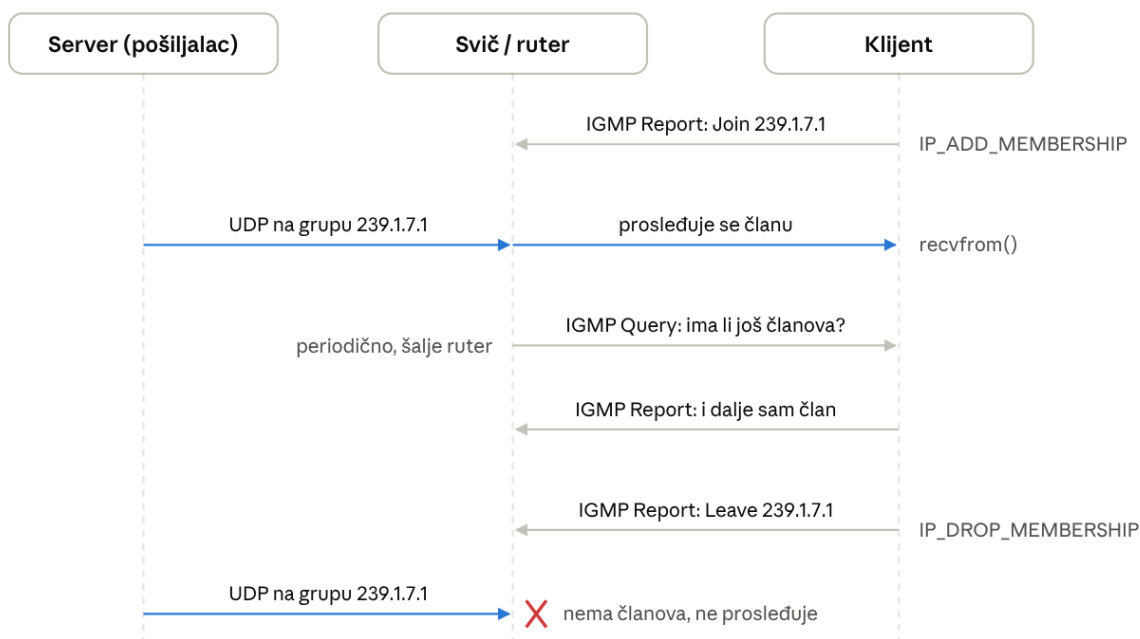
5. IGMP protokol

Da bi mreža znala kome da isporuči *multicast* poruke, računari moraju da prijave svoje članstvo u grupama. Za to služi **IGMP** (engl. *Internet Group Management Protocol*), protokol mrežnog nivoa, kao i ICMP iz Vežbe 2. IGMP poruke se prenose direktno u IP paketu (polje *Protocol* = 2) sa TTL = 1, tako da nikad ne napuštaju lokalnu mrežu.

Aplikacija nikada sama ne šalje IGMP poruke. Kada program zatraži učlanjenje u grupu ili njeno napuštanje, IGMP poruku šalje **operativni sistem** (Slika 2).

Osnovne IGMP poruke su:

- **Membership Report** (prijava članstva): računar javlja da želi da prima saobraćaj grupe. Šalje se odmah pri učlanjenju (engl. *join*) i kao odgovor na upit rutera.
- **Membership Query** (upit): lokalni ruter periodično proverava da li u mreži još ima članova grupe.
- **Leave** (napuštanje): računar javlja da mu saobraćaj grupe više nije potreban.



Slika 2. IGMP poruke pri ućlanjenju u grupu i napućtanju grupe

Postoje tri verzije protokola (IGMPv1, v2 i v3). Savremeni operativni sistemi (Linux, Windows) podrazumevano koriste **IGMPv3**, u kome se i ućlanjenje i napućtanje šalju kao poruka **Membership Report** na adresu **224.0.0.22**. Wireshark ih u koloni *Info* prikazuje kao *Join group* odnosno *Leave group*. Svaka poruka se obićno pošalje dva puta, za slućaj da se prva izgubi.

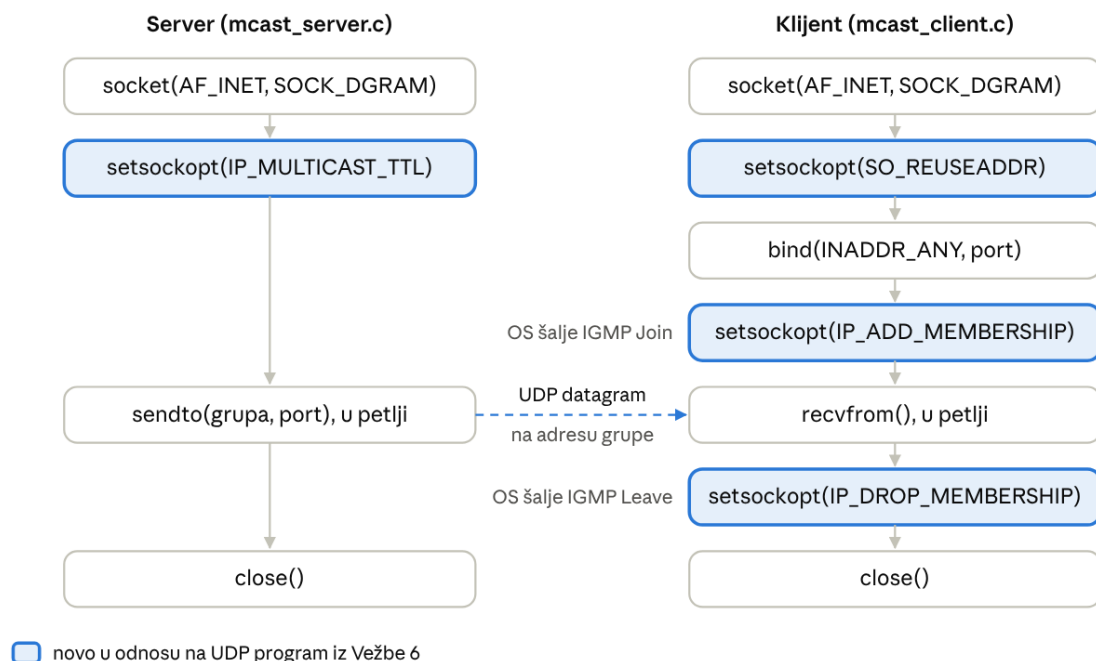
6. Doseg *multicast* poruka (TTL)

Polje TTL u IP zaglavlju (Vežba 2) kod *multicasta* odrećuje koliko daleko poruka moēe da stigne, a podećava ga pošiljalac:

- **TTL = 1**: poruka ostaje u lokalnoj mreēi. Ovo je podrazumevana vrednost i koristi se na veēbi.
- **TTL > 1**: poruka moēe proći kroz rutere, ali samo ako oni podrēavaju *multicast* rutiranje (npr. protokol PIM). *Multicast* rutiranje prevazilazi okvire ovih veēbi, dovoljno je znati da postoji.

Programski interfejs za *multicast*

Multicast program se piēe **istim funkcijama** kao UDP program iz Veēbe 6 (socket, bind, sendto, recvfrom, close). Novo je samo nekoliko poziva funkcije **setsockopt**, kojom se utićnici zadaju *multicast* opcije (Slika 3).



Slika 3. Tok poziva funkcija: *multicast* server (levo) i klijent (desno)

Server (pošiljalac):

1. Kreira UDP utičnicu.
2. Podešava TTL *multicast* paketa (opcija `IP_MULTICAST_TTL`).
3. Šalje poruke funkcijom `sendto` na adresu grupe i port, isto kao što je u Vežbi 6 slao poruke jednom klijentu. Serveru nisu potrebni ni `bind` ni ućlanjenje u grupu.

Klijent (prijemnik):

1. Kreira UDP utičnicu.
2. Dozvoljava deljenje porta (opcija `SO_REUSEADDR`), da bi više klijenata na istom računaru moglo da sluša isti port.
3. Povezuje utičnicu sa portom i adresom `INADDR_ANY` (`bind`).
4. Ućlanjuje se u grupu (opcija `IP_ADD_MEMBERSHIP`). U tom trenutku operativni sistem šalje *IGMP Membership Report*.
5. Prima poruke funkcijom `recvfrom`.
6. Napušta grupu (opcija `IP_DROP_MEMBERSHIP`) i zatvara utičnicu.

Setsockopt funkcija

- **Linux (Ubuntu)**

Funkcija:	Opis:
<code>setsockopt</code>	Postavlja vrednost opcije utičnice.
Parametri:	Opis:
<code>int socket</code>	Utičnica.
<code>int level</code>	Nivo protokola na koji se opcija odnosi: <code>IPPROTO_IP</code> za IP opcije, <code>SOL_SOCKET</code> za opšte opcije utičnice.
<code>int option_name</code>	Opcija koja se postavlja (npr. <code>IP_ADD_MEMBERSHIP</code>).

Funkcija:	Opis:
<code>const void *option_value</code>	Pokazivač na vrednost opcije (int ili struktura, zavisno od opcije).
<code>socklen_t option_len</code>	Veličina vrednosti <code>option_value</code> u bajtima.
Povratna vrednost:	Opis:
<code>int</code>	U slučaju uspešnog izvršenja funkcija vraća vrednost 0. U slučaju greške funkcija vraća vrednost -1 i errno fleg postavlja na odgovarajući kod greške.

- **Windows**

Funkcija:	Opis:
<code>setsockopt</code>	Postavlja vrednost opcije utičnice.
Parametri:	Opis:
<code>SOCKET s</code>	Utičnica.
<code>int level</code>	Nivo protokola na koji se opcija odnosi: <code>IPPROTO_IP</code> za IP opcije, <code>SOL_SOCKET</code> za opšte opcije utičnice.
<code>int optname</code>	Opcija koja se postavlja (npr. <code>IP_ADD_MEMBERSHIP</code>).
<code>const char FAR* optval</code>	Pokazivač na vrednost opcije (int ili struktura, zavisno od opcije).
<code>int optlen</code>	Veličina vrednosti <code>optval</code> u bajtima.
Povratna vrednost:	Opis:
<code>int</code>	U slučaju uspešnog izvršenja funkcija vraća vrednost 0. U slučaju greške funkcija vraća vrednost <code>SOCKET_ERROR</code> . Pozivom funkcije <code>WSAGetLastError</code> moguće je saznati kod nastale greške.

Na Windows-u je parametar `optval` tipa `const char*`, a na Linux-u `const void*`. Zato se adresa vrednosti uvek prosleđuje sa eksplicitnom konverzijom (**`const char *`**), što se ispravno prevodi na oba sistema.

Multicast opcije

Opcija	Nivo (level)	Tip vrednosti	Koristi	Opis
<code>IP_MULTICAST_TTL</code>	<code>IPPROTO_IP</code>	<code>int</code>	server	TTL <i>multicast</i> paketa. Podrazumevano 1.
<code>IP_ADD_MEMBERSHIP</code>	<code>IPPROTO_IP</code>	<code>struct ip_mreq</code>	klijent	Učlanjenje utičnice u grupu.
<code>IP_DROP_MEMBERSHIP</code>	<code>IPPROTO_IP</code>	<code>struct ip_mreq</code>	klijent	Napuštanje grupe.
<code>SO_REUSEADDR</code>	<code>SOL_SOCKET</code>	<code>int</code>	klijent	Dozvoljava da više utičnica na istom računaru koristi isti port.
<code>IP_MULTICAST_LOOP</code>	<code>IPPROTO_IP</code>	<code>int</code>	server	Da li se poslate poruke isporučuju i članovima grupe na istom računaru. Podrazumevano uključeno.
<code>IP_MULTICAST_IF</code>	<code>IPPROTO_IP</code>	<code>struct in_addr</code>	server	Izbor mrežne kartice za slanje, kada računar ima više kartica.

Struktura ip_mreq

Opcije IP_ADD_MEMBERSHIP i IP_DROP_MEMBERSHIP kao vrednost primaju strukturu **ip_mreq** (engl. *multicast request*), definisanu u zaglavlju netinet/in.h (Linux), odnosno ws2tcpip.h (Windows):

```
struct ip_mreq {
    struct in_addr imr_multiaddr;    // adresa grupe, npr. 239.1.7.1
    struct in_addr imr_interface;    // adresa mrezne kartice; INADDR_ANY = bira OS
};
```

Primer uclanjenja uticnice sock u grupu 239.1.7.1:

```
struct ip_mreq mreq;

inet_pton(AF_INET, "239.1.7.1", &mreq.imr_multiaddr);
mreq.imr_interface.s_addr = htonl(INADDR_ANY);

if (setsockopt(sock, IPPROTO_IP, IP_ADD_MEMBERSHIP,
               (const char *)&mreq, sizeof(mreq)) < 0) {
    // greska
}
```

Napuštanje grupe se radi na isti način, samo sa opcijom IP_DROP_MEMBERSHIP. Utičnica se može učlaniti u više grupa, jednim pozivom setsockopt za svaku grupu.

Ostale razlike između Linux i Windows platforme (WSAStartup, closesocket, zaglavlja) su iste kao u Vežbi 4 i u zadatku su već rešene u fajlu **mcast_common.h**.

ZADACI VEŽBE

Scenario: multicast info kanali. Svaki student na svom računaru pravi mali *multicast* info servis. Server emituje poruke na tri kanala (VREME, VESTI, KURS), pri čemu je svaki kanal posebna *multicast* grupa. Klijent bira kanale u koje se učlanjuje, prima samo njihove poruke i posle 10 primljenih poruka napušta prvi izabrani kanal.

U ovom zadatku server je samo pošiljalac *multicast* poruka, a klijenti su prijemnici. Između njih se **ne uspostavlja** veza kao kod TCP-a.

Svaki računar u laboratoriji dobija **broj računara X**. Iz njega se računaju adrese grupa i port, tako da serveri na različitim računarima ne ometaju jedan drugog:

Kanal	Naziv	Adresa grupe	Port	Primer za X = 7
1	VREME	239.1.X.1	5000 + X	239.1.7.1 : 5007
2	VESTI	239.1.X.2	5000 + X	239.1.7.2 : 5007
3	KURS	239.1.X.3	5000 + X	239.1.7.3 : 5007

Sva tri kanala koriste **isti port**. Klijent bira kanale isključivo učlanjenjem u odgovarajuće grupe.

Fajlovi zadatka:

- **mcast_common.h** - sve razlike između Linux i Windows platforme, parametri vežbe i funkcije make_group_addr i make_port. Ovaj fajl se ne menja.
- **mcast_server.c** - kostur servera, delovi označeni sa TODO 1, TODO 2 i TODO 3.
- **mcast_client.c** - kostur klijenta, delovi označeni sa TODO 4, TODO 5 i TODO 6.

- **CMakeLists.txt** - opis prevođenja oba programa za alat CMake. Ovaj fajl se ne menja.

Prevođenje i pokretanje:

Programi se prevode alatom **CMake**. Fajl CMakeLists.txt opisuje oba programa, pa jedan poziv komande make prevodi i server i klijent. Folder build se pravi samo prvi put, a posle svake izmene koda dovoljno je u njemu ponovo pozvati make (na Windows-u mingw32-make).

```
# Linux (Ubuntu)
mkdir build
cd build
cmake ..
make
./mcast_server 7
./mcast_client 7 1 3

# Windows (MinGW)
mkdir build
cd build
cmake -G "MinGW Makefiles" ..
mingw32-make
mcast_server.exe 7
mcast_client.exe 7 1 3
```

Zadatak 1: Multicast server

- Dopuniti mcast_server.c:
 - **TODO 1:** podesiti TTL *multicast* paketa na vrednost MCAST_TTL (opcija IP_MULTICAST_TTL).
 - **TODO 2:** za svaki kanal popuniti odredišnu adresu (sin_family, sin_port, sin_addr). Adresa grupe se dobija funkcijom make_group_addr, a port funkcijom make_port. Obratiti pažnju na mrežni redosled okteta (Vežba 4).
 - **TODO 3:** poslati poruku na adresu grupe funkcijom sendto.
- Prevesti i pokrenuti server sa svojim brojem X. Server na svake 2 sekunde ispisuje po jednu poslatu poruku za svaki kanal.

Uočiti da server radi i kada ne postoji nijedan klijent: pošiljalac ne zna da li neko sluša grupu.

Zadatak 2: Multicast klijent

- Dopuniti mcast_client.c:
 - **TODO 4:** povezati utičnicu (bind) sa adresom INADDR_ANY i portom make_port(X).
 - **TODO 5:** u funkciji set_membership (na početku fajla) popuniti strukturu ip_mreq i pozvati setsockopt. Ista funkcija služi i za učlanjenje i za napuštanje grupe, u zavisnosti od parametra option.
 - **TODO 6:** posle LEAVE_AFTER primljenih poruka napustiti prvi izabrani kanal pozivom set_membership sa opcijom IP_DROP_MEMBERSHIP.

Zadatak 3: Provera na jednom računaru

- U jednom terminalu pokrenuti server, a u drugom klijent koji prati kanale VREME i KURS: mcast_client X 1 3.
- Očekivani ispisi klijenta (adrese, vremena i redni brojevi će se razlikovati):

```
>>> Uclanjen u grupu 239.1.7.1 (kanal VREME)
>>> Uclanjen u grupu 239.1.7.3 (kanal KURS)
Cekam poruke na portu 5007...
1. [od 192.168.1.107:44741] [VREME] #2 Tacno vreme: 11:40:59
2. [od 192.168.1.107:44741] [KURS] #2 1 EUR = 117.24 RSD
...
10. [od 192.168.1.107:44741] [KURS] #6 1 EUR = 117.12 RSD
>>> Napustena grupa 239.1.7.1 (kanal VREME)
11. [od 192.168.1.107:44741] [KURS] #7 1 EUR = 117.19 RSD
...
20. [od 192.168.1.107:44741] [KURS] #16 1 EUR = 117.22 RSD
>>> Napustena grupa 239.1.7.3 (kanal KURS)
Klijent završava rad.
```

- Proveriti:
 - poruke kanala VESTI ne stižu, iako ih server šalje na isti port;
 - posle 10. poruke stižu samo poruke kanala KURS;
 - prva poruka (#1) obično nedostaje jer je poslata pre nego što se klijent učlanio. *Multicast* ne čuva poruke za kasnije članove.
- Pokrenuti istovremeno dva klijenta, mcast_client X 1 i mcast_client X 1 2. Oba primaju poruke kanala VREME: server šalje jednu poruku, a primaju je svi članovi grupe.

Zadatak 4: Rad u lokalnoj mreži

- U dogovoru sa kolegom pored vas pokrenuti klijent koji prati njegov kanal VESTI: mcast_client Y 2, gde je Y broj njegovog računara.
- U ispisu se kao pošiljalac vidi IP adresa računara vašeg kolege.
- Svi serveri u laboratoriji rade istovremeno u istoj mreži, a klijent prima samo poruke grupa u koje se učlanio.

Zadatak 5: IGMP poruke u Wireshark-u

- Pokrenuti Wireshark i započeti hvatanje paketa na mrežnoj kartici računara. U polje za filtriranje uneti **igmp**.
- Sa pokrenutim serverom pokrenuti klijent mcast_client X 1 3.
- Pronaći poruku **Membership Report** sa *Join group 239.1.X.1* i *Join group 239.1.X.3*. Proveriti odredišnu IP adresu (224.0.0.22) i TTL (1).
- Posle 10 primljenih poruka pronaći poruku sa *Leave group 239.1.X.1*, a na kraju rada klijenta i *Leave group 239.1.X.3*.
- Promeniti filter u **ip.dst == 239.1.X.3** i u jednom paketu proveriti:
 - protokol UDP i odredišni port 5000 + X;
 - TTL = 1;

- određi MAC adresu, koja počinje sa 01:00:5e. Poslednja tri bajta su donja 23 bita adrese grupe u heksadecimalnom zapisu, npr. za X = 12 MAC adresa je 01:00:5e:01:0c:03.

Dodatni zadatak

- Proširiti klijent tako da za svaki kanal pamti redni broj poslednje primljene poruke i ispisuje upozorenje kada neka poruka nedostaje. Redni broj se iz poruke može pročitati funkcijom `sscanf(buf, "[%*[^]]] #d", &seq)`.
- Dok klijent radi, zaustaviti server i ponovo ga pokrenuti. Objasniti šta klijent primeti i zašto *multicast* ne može sam da nadoknadi izgubljene poruke.

Česti problemi

Problem	Uzrok i rešenje
Klijent prima poruke lokalnog servera, ali ne i sa drugog računara	Windows zaštitni zid (engl. <i>firewall</i>) blokira dolazne UDP pakete. Pri prvom pokretanju dozvoliti pristup programu, ili proveriti ping-om (Vežba 2) da su računari u istoj mreži.
Poruke ne stižu ni na istom računaru, a računar ima više mrežnih kartica (VirtualBox, VPN, Wi-Fi)	Operativni sistem je izabrao pogrešnu karticu. U <code>mcast_common.h</code> u <code>IFACE_ADDR</code> upisati IP adresu računara u laboratorijskoj mreži (ipconfig na Windows-u, ip addr na Linux-u).
bind vraća grešku " <i>Address already in use</i> "	Opcija <code>SO_REUSEADDR</code> mora biti postavljena pre poziva <code>bind</code> (u kosturu je već postavljena).
<code>setsockopt(IP_ADD_MEMBERSHIP)</code> vraća grešku	Proveriti da li je adresa grupe ispravno upisana u <code>imr_multiaddr</code> i da li je iz opsega klase D.
Klijent ne prima ništa, a server radi	Proveriti da li klijent i server koriste isti broj X, i da li je port u <code>bind</code> pozivu pretvoren funkcijom <code>htons</code> .
Wireshark ne prikazuje mrežne kartice	Na Linux-u pokrenuti Wireshark sa <code>sudo</code> , a na Windows-u proveriti da je instaliran Npcap.