IZGRADNJA APLIKACIJE ZA STRUJANJE MULTIMEDIJE

Jurica Migač
Robert Langus
Bruno Mataušić
Antonio Hip

LogoType

Sadržaj

RTP

Real-Time Transport Protocol

- Mrežni protokol
- IETF Audio-Video Transporting Group, 1996.
- Prijenos multimedije end-to-end
- UDP protokol
- Sadrži sequence number, payload identification, frame indication, source, intermedia synchronization
- RTCP, RTSP

RTP

- Koristi se za:
  - Stream
  - Telefoniju
  - Video sastanke
  - Aplikacije koje prenose audio/video

Live streaming

- Strujanje medija u real-timeu
- Mobitel/računalo
- Problemi kod spore internetske veze

Live streaming

- Prilike:
  - Usredotočeniji slušatelj/gledatelj
  - Stvaranje identiteta, razvijanje komunikacijskih vje
    ština, pokazivanje talenta...
- Rizici:
  - Nema cenzure
  - Poticanje na loše ponašanje
  - Odavanje tajnih ili privatnih podataka

Strujanje pjesme preko udaljenog računala

Moguće implementacije?

*vlc-mp3_stream_from_website.py - C:\Users\Jurica\Desktop\MR2_project\python-simple\vl...

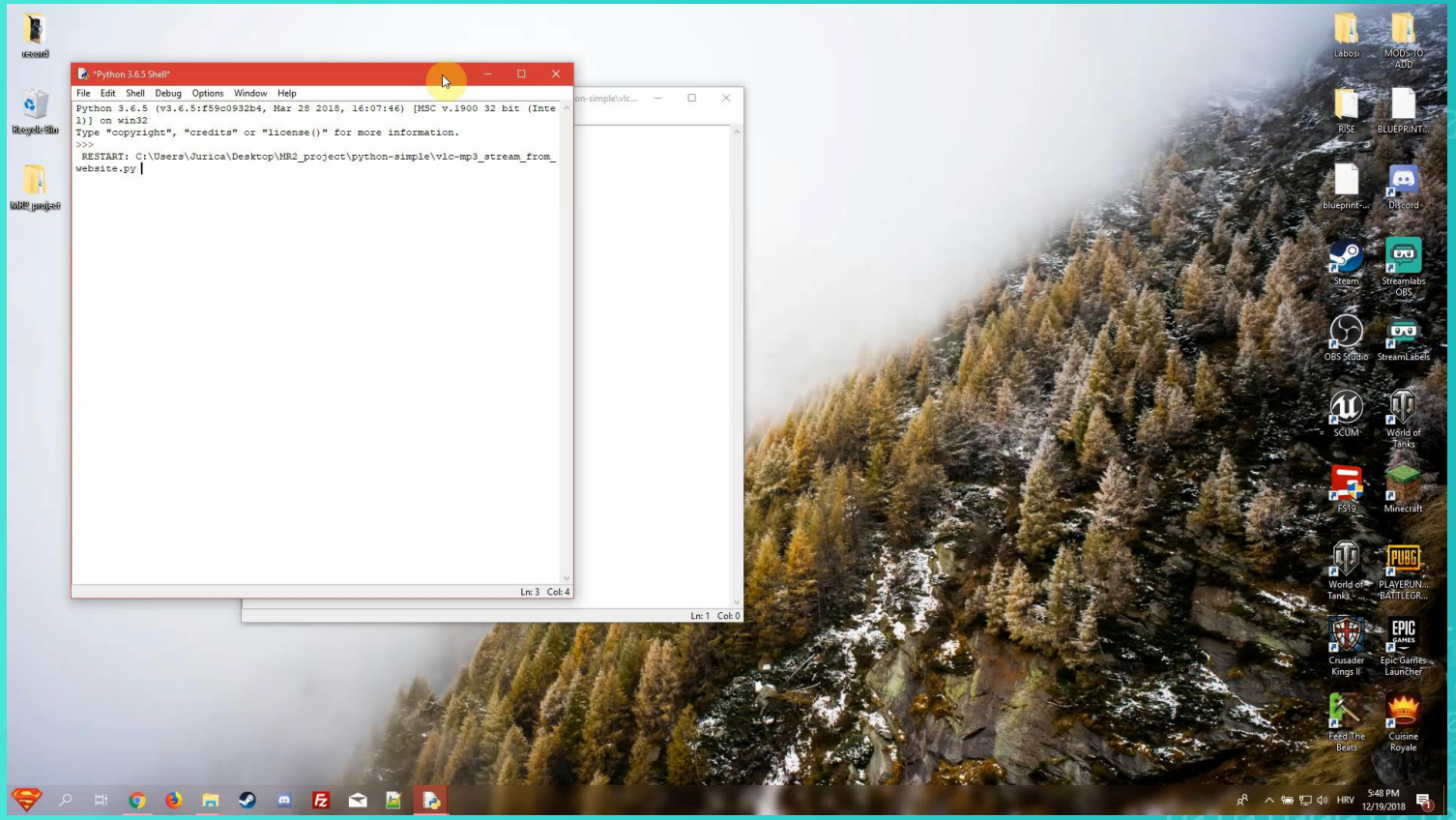
File Edit Format Run Options Window Help

```python
import vlc
url = 'http://mr2.juricamigac.com/windows.mp3'
instance = vlc.Instance()
player=instance.media_player_new()
media=instance.media_new(url)
player.set_media(media)
player.play()
```

Ln: 10 Col: 0

Strujanje pjesme preko udaljenog računala

Moguće implementacije?



Wireshark snimka

Wireshark snimka strujanja pjesme preko udaljenog računala

*Ethernet

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/> Expression...

No.	Time	Source	Destination	Protocol	Length	Info
16	2.193683	13.107.21.200	192.168.1.6	TCP	60	443 → 54363 [ACK] Seq=1 Ack=1 Win=1024 Len=0
17	2.193729	192.168.1.6	13.107.21.200	TCP	54	[TCP ACKed unseen segment] 54363 → 443 [ACK] Seq=2 Ack=2 Win=1021 Len=0
18	2.194561	13.107.21.200	192.168.1.6	TCP	60	[TCP Previous segment not captured] 443 → 54363 [ACK] Seq=2 Ack=2 Win=1027 Len=0
19	2.513344	104.16.59.37	192.168.1.6	TLSv1.2	210	Application Data
20	2.553382	192.168.1.6	104.16.59.37	TCP	54	54039 → 443 [ACK] Seq=1 Ack=157 Win=257 Len=0
21	3.000249	185.169.199.130	192.168.1.6	RTCP	94	Receiver Report
22	3.419539	192.168.1.6	185.169.199.130	RTCP	102	Sender Report
23	3.841736	192.168.1.6	192.168.1.1	DNS	79	Standard query 0x090d A mr2.juricamigac.com
24	3.907815	192.168.1.1	192.168.1.6	DNS	176	Standard query response 0x090d A mr2.juricamigac.com A 31.22.4.37 NS ns1.byethost8.org NS ns2.byethost8.org A 31.22.4.20 A 31.22.4.37
25	3.910439	192.168.1.6	31.22.4.37	TCP	66	54376 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
26	3.976792	31.22.4.37	192.168.1.6	TCP	66	80 → 54376 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1422 SACK_PERM=1 WS=512
27	3.976848	192.168.1.6	31.22.4.37	TCP	54	54376 → 80 [ACK] Seq=1 Ack=1 Win=66816 Len=0
28	3.977035	192.168.1.6	31.22.4.37	TCP	81	54376 → 80 [PSH, ACK] Seq=1 Ack=1 Win=66816 Len=27 [TCP segment of a reassembled PDU]
29	3.999961	185.169.199.130	192.168.1.6	RTCP	94	Receiver Report
30	4.036087	31.22.4.37	192.168.1.6	TCP	60	80 → 54376 [ACK] Seq=1 Ack=28 Win=29696 Len=0
31	4.036121	192.168.1.6	31.22.4.37	HTTP	172	GET /windows.mp3 HTTP/1.1
32	4.100382	31.22.4.37	192.168.1.6	TCP	60	80 → 54376 [ACK] Seq=1 Ack=146 Win=29696 Len=0
33	4.116755	31.22.4.37	192.168.1.6	TCP	1476	80 → 54376 [ACK] Seq=1 Ack=146 Win=29696 Len=1422 [TCP segment of a reassembled PDU]
34	4.117822	31.22.4.37	192.168.1.6	TCP	1476	80 → 54376 [ACK] Seq=1423 Ack=146 Win=29696 Len=1422 [TCP segment of a reassembled PDU]
35	4.117844	192.168.1.6	31.22.4.37	TCP	54	54376 → 80 [ACK] Seq=146 Ack=2845 Win=66816 Len=0
36	4.119405	31.22.4.37	192.168.1.6	TCP	1476	80 → 54376 [ACK] Seq=2845 Ack=146 Win=29696 Len=1422 [TCP segment of a reassembled PDU]
37	4.120517	31.22.4.37	192.168.1.6	TCP	1476	80 → 54376 [ACK] Seq=4267 Ack=146 Win=29696 Len=1422 [TCP segment of a reassembled PDU]
38	4.120549	192.168.1.6	31.22.4.37	TCP	54	54376 → 80 [ACK] Seq=146 Ack=5689 Win=66816 Len=0
39	4.122079	31.22.4.37	192.168.1.6	TCP	1476	80 → 54376 [ACK] Seq=5689 Ack=146 Win=29696 Len=1422 [TCP segment of a reassembled PDU]
40	4.123445	31.22.4.37	192.168.1.6	TCP	1476	80 → 54376 [ACK] Seq=7111 Ack=146 Win=29696 Len=1422 [TCP segment of a reassembled PDU]
41	4.123447	31.22.4.37	192.168.1.6	TCP	1476	80 → 54376 [ACK] Seq=8533 Ack=146 Win=29696 Len=1422 [TCP segment of a reassembled PDU]
42	4.123481	192.168.1.6	31.22.4.37	TCP	54	54376 → 80 [ACK] Seq=146 Ack=9955 Win=66816 Len=0
43	4.125780	31.22.4.37	192.168.1.6	TCP	1476	80 → 54376 [ACK] Seq=9955 Ack=146 Win=29696 Len=1422 [TCP segment of a reassembled PDU]
44	4.125806	192.168.1.6	31.22.4.37	TCP	54	54376 → 80 [ACK] Seq=146 Ack=11377 Win=66816 Len=0
45	4.126567	31.22.4.37	192.168.1.6	TCP	1476	80 → 54376 [ACK] Seq=11377 Ack=146 Win=29696 Len=1422 [TCP segment of a reassembled PDU]
46	4.127355	31.22.4.37	192.168.1.6	TCP	1476	80 → 54376 [ACK] Seq=12799 Ack=146 Win=29696 Len=1422 [TCP segment of a reassembled PDU]
47	4.127385	192.168.1.6	31.22.4.37	TCP	54	54376 → 80 [ACK] Seq=146 Ack=14221 Win=66816 Len=0
48	4.176744	31.22.4.37	192.168.1.6	TCP	1476	80 → 54376 [ACK] Seq=14221 Ack=146 Win=29696 Len=1422 [TCP segment of a reassembled PDU]
49	4.177887	31.22.4.37	192.168.1.6	TCP	1476	80 → 54376 [ACK] Seq=15643 Ack=146 Win=29696 Len=1422 [TCP segment of a reassembled PDU]
50	4.177908	192.168.1.6	31.22.4.37	TCP	54	54376 → 80 [ACK] Seq=146 Ack=17065 Win=66816 Len=0
51	4.178720	31.22.4.37	192.168.1.6	TCP	1476	80 → 54376 [ACK] Seq=17065 Ack=146 Win=29696 Len=1422 [TCP segment of a reassembled PDU]
52	4.179536	31.22.4.37	192.168.1.6	TCP	1476	80 → 54376 [ACK] Seq=18487 Ack=146 Win=29696 Len=1422 [TCP segment of a reassembled PDU]
53	4.179551	192.168.1.6	31.22.4.37	TCP	54	54376 → 80 [ACK] Seq=146 Ack=19909 Win=66816 Len=0
54	4.183421	31.22.4.37	192.168.1.6	TCP	1476	80 → 54376 [ACK] Seq=19909 Ack=146 Win=29696 Len=1422 [TCP segment of a reassembled PDU]
55	4.184452	31.22.4.37	192.168.1.6	TCP	1476	80 → 54376 [ACK] Seq=21331 Ack=146 Win=29696 Len=1422 [TCP segment of a reassembled PDU]
56	4.184468	192.168.1.6	31.22.4.37	TCP	54	54376 → 80 [ACK] Seq=146 Ack=22753 Win=66816 Len=0
57	4.186121	31.22.4.37	192.168.1.6	TCP	1476	80 → 54376 [ACK] Seq=22753 Ack=146 Win=29696 Len=1422 [TCP segment of a reassembled PDU]
58	4.186919	31.22.4.37	192.168.1.6	TCP	1476	80 → 54376 [ACK] Seq=24175 Ack=146 Win=29696 Len=1422 [TCP segment of a reassembled PDU]
59	4.186950	192.168.1.6	31.22.4.37	TCP	54	54376 → 80 [ACK] Seq=146 Ack=25597 Win=66816 Len=0
60	4.190296	31.22.4.37	192.168.1.6	TCP	1476	80 → 54376 [ACK] Seq=25597 Ack=146 Win=29696 Len=1422 [TCP segment of a reassembled PDU]
61	4.190299	31.22.4.37	192.168.1.6	TCP	1476	80 → 54376 [ACK] Seq=27019 Ack=146 Win=29696 Len=1422 [TCP segment of a reassembled PDU]

```
> Frame 16: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
> Ethernet II, Src: Sagemcom_e5:23:b9 (c0:d0:44:e5:23:b9), Dst: HewlettP_69:c5:00 (98:e7:f4:69:c5:00)
> Internet Protocol Version 4, Src: 13.107.21.200, Dst: 192.168.1.6
> Transmission Control Protocol, Src Port: 443, Dst Port: 54363, Seq: 1, Ack: 1, Len: 0
```

wireshark_6FA20AEE-2369-411F-960B-60BBB3572D72_20181219190618_a13308 | Packets: 620 · Displayed: 620 (100.0%) · Dropped: 0 (0.0%) | Profile: Default

Strujanje glasa preko UDP

Moguće implementacije?

klijent.py - C:\Users\Jurica\Desktop\python-pitch2\klijent.py (3.6.5)

File Edit Format Run Options Window Help

```python
import socket
import pyaudio
import wave

# Pyaudio Initialization
CHUNK = 1024
CHANNELS = 2
RATE = 44100
INPUT = True
FORMAT = pyaudio.paInt16

# Socket Initialization
HOST = '127.0.0.1'
PORT = 953
SIZE = 1024

if __name__ == '__main__':
    p = pyaudio.PyAudio()
    stream = p.open(format=FORMAT,
                    channels=CHANNELS,
                    rate=RATE,
                    input=True,
                    frames_per_buffer=CHUNK)
    connection = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    while 1:
        data = stream.read(CHUNK)
        connection.sendto(data,(HOST,PORT))

    connection.close()
    stream.stop_stream()
    stream.close()
    p.terminate()
```

Ln: 1 Col: 0

server.py - C:\Users\Jurica\Desktop\python-pitch2\server.py (3.6.5)

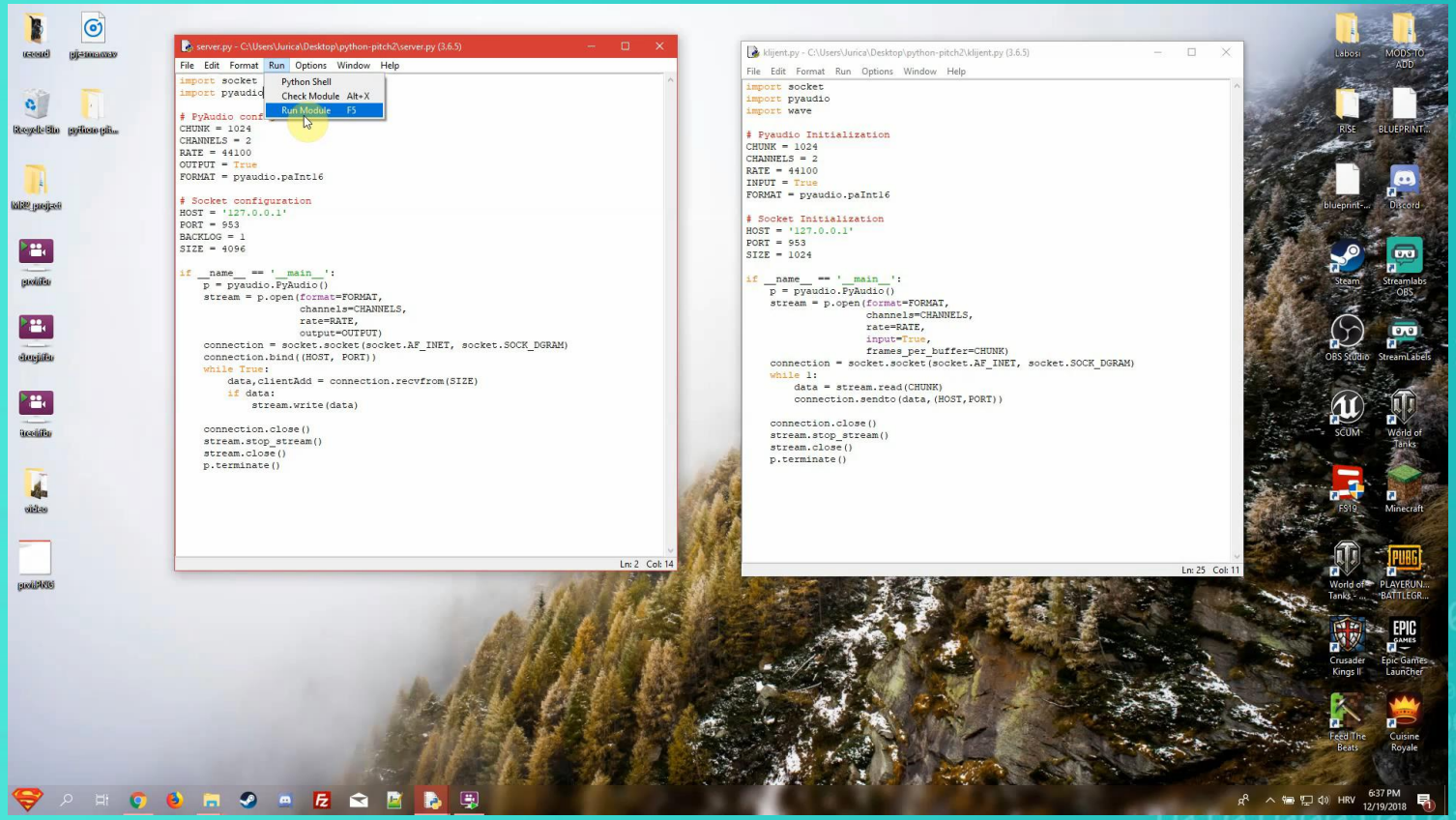
File Edit Format Run Options Window Help

```python
import socket
import pyaudio

# PyAudio configuration
CHUNK = 1024
CHANNELS = 2
RATE = 44100
OUTPUT = True
FORMAT = pyaudio.paInt16

# Socket configuration
HOST = '127.0.0.1'
PORT = 953
BACKLOG = 1
SIZE = 4096

if __name__ == '__main__':
    p = pyaudio.PyAudio()
    stream = p.open(format=FORMAT,
                    channels=CHANNELS,
                    rate=RATE,
                    output=OUTPUT)
    connection = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    connection.bind((HOST, PORT))
    while True:
        data,clientAdd = connection.recvfrom(SIZE)
        if data:
            stream.write(data)

    connection.close()
    stream.stop_stream()
    stream.close()
    p.terminate()
```

Ln: 8 Col: 13

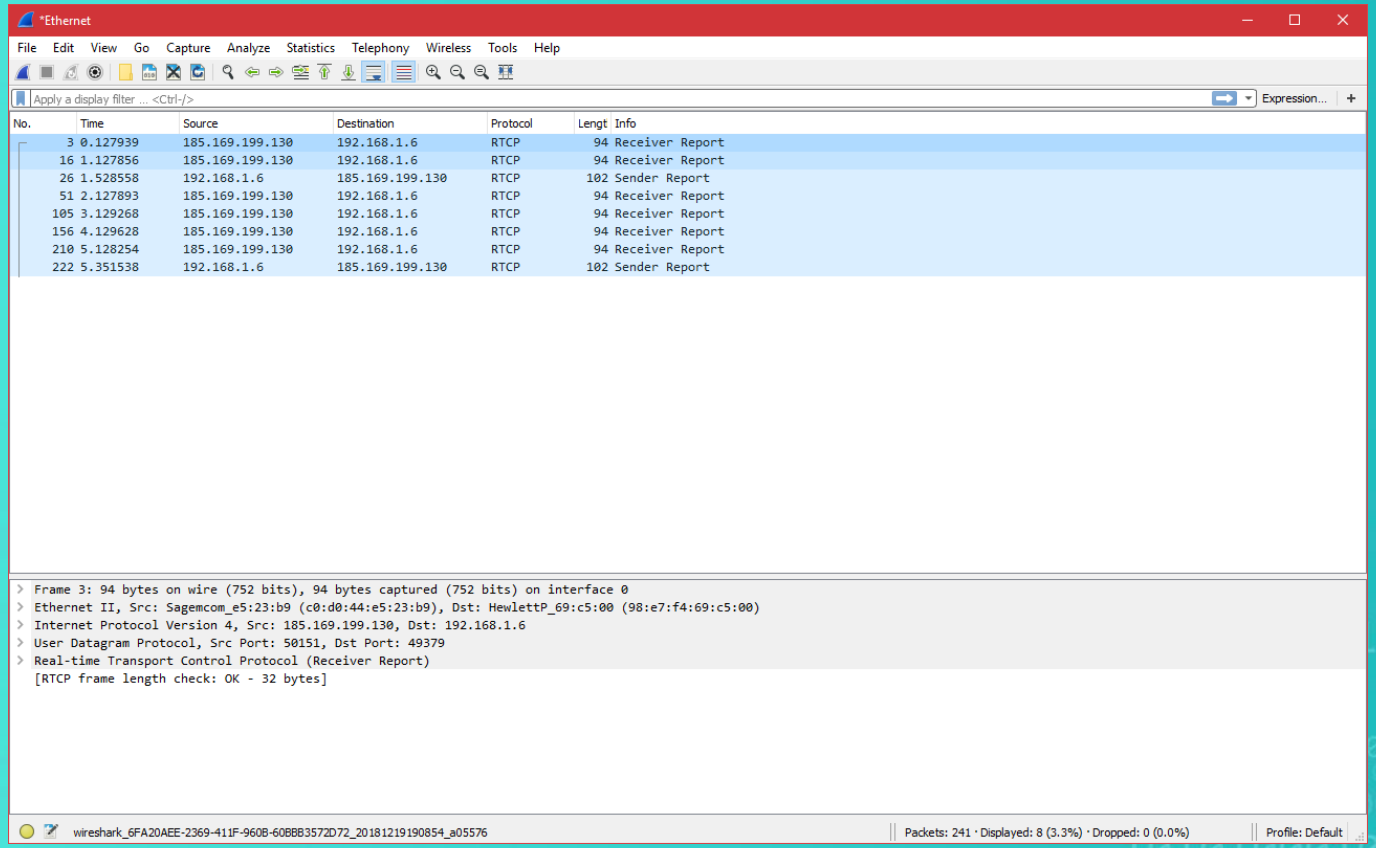
Strujanje glasa preko UDP

Moguće implementacije?



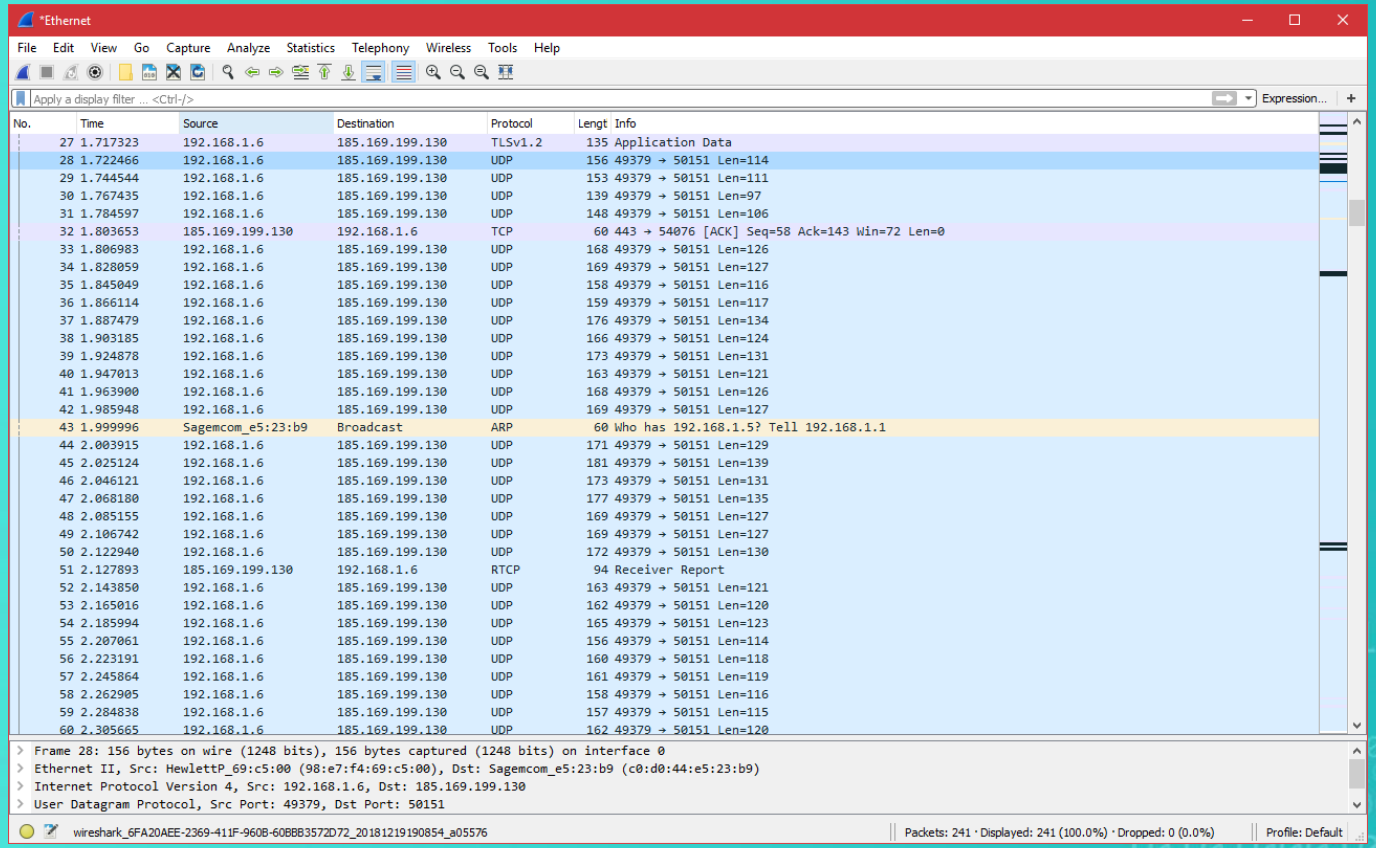
Wireshark snimka

Wireshark snimka strujanja glasa preko UDP protokola



Wireshark snimka

Wireshark snimka strujanja glasa preko UDP protokola



No.	Time	Source	Destination	Protocol	Lengt	Info
27	1.717323	192.168.1.6	185.169.199.130	TLSv1.2	135	Application Data
28	1.722466	192.168.1.6	185.169.199.130	UDP	156	49379 → 50151 Len=114
29	1.744544	192.168.1.6	185.169.199.130	UDP	153	49379 → 50151 Len=111
30	1.767435	192.168.1.6	185.169.199.130	UDP	139	49379 → 50151 Len=97
31	1.784597	192.168.1.6	185.169.199.130	UDP	148	49379 → 50151 Len=106
32	1.803653	185.169.199.130	192.168.1.6	TCP	60	443 → 54076 [ACK] Seq=58 Ack=143 Win=72 Len=0
33	1.806983	192.168.1.6	185.169.199.130	UDP	168	49379 → 50151 Len=126
34	1.828059	192.168.1.6	185.169.199.130	UDP	169	49379 → 50151 Len=127
35	1.845049	192.168.1.6	185.169.199.130	UDP	158	49379 → 50151 Len=116
36	1.866114	192.168.1.6	185.169.199.130	UDP	159	49379 → 50151 Len=117
37	1.887479	192.168.1.6	185.169.199.130	UDP	176	49379 → 50151 Len=134
38	1.903185	192.168.1.6	185.169.199.130	UDP	166	49379 → 50151 Len=124
39	1.924878	192.168.1.6	185.169.199.130	UDP	173	49379 → 50151 Len=131
40	1.947013	192.168.1.6	185.169.199.130	UDP	163	49379 → 50151 Len=121
41	1.963900	192.168.1.6	185.169.199.130	UDP	168	49379 → 50151 Len=126
42	1.985948	192.168.1.6	185.169.199.130	UDP	169	49379 → 50151 Len=127
43	1.999996	Sagemcom_e5:23:b9	Broadcast	ARP	60	Who has 192.168.1.5? Tell 192.168.1.1
44	2.003915	192.168.1.6	185.169.199.130	UDP	171	49379 → 50151 Len=129
45	2.025124	192.168.1.6	185.169.199.130	UDP	181	49379 → 50151 Len=139
46	2.046121	192.168.1.6	185.169.199.130	UDP	173	49379 → 50151 Len=131
47	2.068180	192.168.1.6	185.169.199.130	UDP	177	49379 → 50151 Len=135
48	2.085155	192.168.1.6	185.169.199.130	UDP	169	49379 → 50151 Len=127
49	2.106742	192.168.1.6	185.169.199.130	UDP	169	49379 → 50151 Len=127
50	2.122940	192.168.1.6	185.169.199.130	UDP	172	49379 → 50151 Len=130
51	2.127893	185.169.199.130	192.168.1.6	RTCP	94	Receiver Report
52	2.143850	192.168.1.6	185.169.199.130	UDP	163	49379 → 50151 Len=121
53	2.165016	192.168.1.6	185.169.199.130	UDP	162	49379 → 50151 Len=120
54	2.185994	192.168.1.6	185.169.199.130	UDP	165	49379 → 50151 Len=123
55	2.207061	192.168.1.6	185.169.199.130	UDP	156	49379 → 50151 Len=114
56	2.223191	192.168.1.6	185.169.199.130	UDP	160	49379 → 50151 Len=118
57	2.245864	192.168.1.6	185.169.199.130	UDP	161	49379 → 50151 Len=119
58	2.262905	192.168.1.6	185.169.199.130	UDP	158	49379 → 50151 Len=116
59	2.284838	192.168.1.6	185.169.199.130	UDP	157	49379 → 50151 Len=115
60	2.305665	192.168.1.6	185.169.199.130	UDP	162	49379 → 50151 Len=120

Strujanje glasa preko TCP

Moguće implementacije?

klijent.py - C:\Users\Jurica\Desktop\MR2_project\voice\klijent.py (3.6.5)

File Edit Format Run Options Window Help

```python
import socket
import pyaudio
import wave

#PyAudio config
CHUNK = 1024
FORMAT = pyaudio.paInt16
CHANNELS = 1
RATE = 44100
RECORD_SECONDS = 5
#Server config
HOST = 'localhost'
PORT = 841

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((HOST, PORT))
p = pyaudio.PyAudio()

stream = p.open(format=FORMAT,
                channels=CHANNELS,
                rate=RATE,
                input=True,
                frames_per_buffer=CHUNK)

print("#snimam#")
frames = []
for i in range(0, int(RATE/CHUNK*RECORD_SECONDS)):
 data  = stream.read(CHUNK)
 frames.append(data)
 s.sendall(data)

print("#zavrseno snimanje#")

stream.stop_stream()
stream.close()
p.terminate()
s.close()

print("#zatvoren socket#")
```

Ln: 1 Col: 0

server.py - C:\Users\Jurica\Desktop\MR2_project\voice\server.py (3.6.5)

File Edit Format Run Options Window Help

```python
import socket
import pyaudio
import wave
import time

#PyAudio config
CHUNK = 1024
FORMAT = pyaudio.paInt16
CHANNELS = 1
RATE = 44100
RECORD_SECONDS = 5
WAVE_OUTPUT_FILENAME = "server_output.wav"
WIDTH = 2
frames = []
#Server config
HOST = 'localhost'
PORT = 841


p = pyaudio.PyAudio()
stream = p.open(format=p.get_format_from_width(WIDTH),
                channels=CHANNELS,
                rate=RATE,
                output=True,
                frames_per_buffer=CHUNK)

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((HOST, PORT))
s.listen(1)
conn, addr = s.accept()
data = conn.recv(1024)
i=1

while data != '':
    stream.write(data)
    data = conn.recv(1024)
    i=i+1
    frames.append(data)
stream.stop_stream()
stream.close()
p.terminate()
conn.close()
```

Ln: 1 Col: 0

Strujanje glasa preko TCP

Moguće implementacije?

klijent.py - C:\Users\Jurica\Desktop\MR2_project\voice\klijent.py (3.6.5)

File Edit Format Run Options Window Help

```python
import socket
import pyaudio
import wave

#PyAudio config
CHUNK = 1024
FORMAT = pyaudio.paInt16
CHANNELS = 1
RATE = 44100
RECORD_SECONDS = 5
#Server config
HOST = 'localhost'
PORT = 841

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((HOST, PORT))
p = pyaudio.PyAudio()

stream = p.open(format=FORMAT,
                channels=CHANNELS,
                rate=RATE,
                input=True,
                frames_per_buffer=CHUNK)

print("#snimam#")
frames = []
for i in range(0, int(RATE/CHUNK*RECORD_SECONDS)):
 data  = stream.read(CHUNK)
 frames.append(data)
 s.sendall(data)

print("#zavrseno snimanje#")

stream.stop_stream()
stream.close()
p.terminate()
s.close()

print("#zatvoren socket#")
```

Ln: 1 Col: 0

server.py - C:\Users\Jurica\Desktop\MR2_project\voice\server.py (3.6.5)

File Edit Format Run Options Window Help

Python Shell
Check Module Alt+X
Run Module F5

```python
import socket
import pyaudio
import wave
import time

#PyAudio config
CHUNK = 1024
FORMAT = pyaudio.paInt16
CHANNELS = 1
RATE = 44100
RECORD_SECONDS = 5
WAVE_OUTPUT_FILENAME = "server_output.wav"
WIDTH = 2
frames = []
#Server config
HOST = 'localhost'
PORT = 841


p = pyaudio.PyAudio()
stream = p.open(format=p.get_format_from_width(WIDTH),
                channels=CHANNELS,
                rate=RATE,
                output=True,
                frames_per_buffer=CHUNK)

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((HOST, PORT))
s.listen(1)
conn, addr = s.accept()
data = conn.recv(1024)
i=1

while data != '':
    stream.write(data)
    data = conn.recv(1024)
    i=i+1
    frames.append(data)
print (i)
wf = wave.open(WAVE_OUTPUT_FILENAME, 'wb')
```

Ln: 1 Col: 0

Hvala vam na pažnji