IHLP- **NW 3**

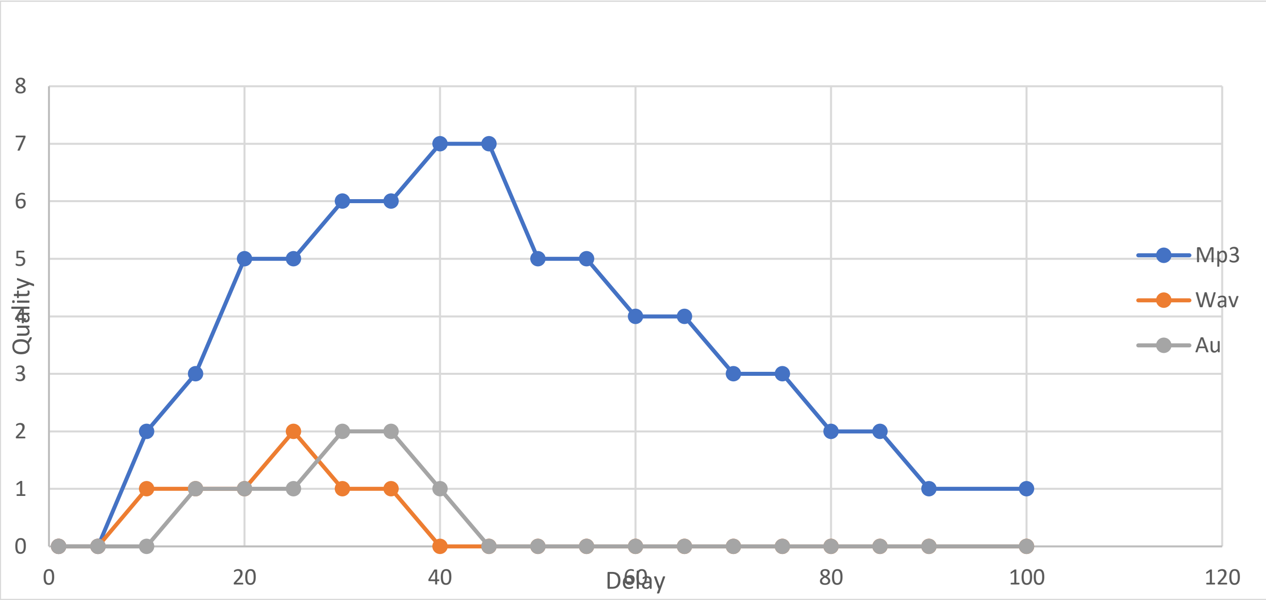
UDP Packet Radio Transmitter

Group W2

Charanpreet Kaur Dhir - ckd22@njit.edu (Group Leader)
Soumyadeep Basu - sb2356@njit.edu
Sayantan Maity - sm2734@njit.edu
Akhil Mangalamadaka - am2825@njit.edu
Pruthvi Chintakayala - pc422@njit.edu
Rohini Puvvula - rp787@njit.edu

Part 1:

X-axis: Delay
Y-axis: Quality



Graph 1: Audio File b0

Part 2:

The graph represents the delay duration from 10 ms to 100 ms. Initially the quality is very low as the packets are transmitted at a higher rate of transmission and further the packet loss is observed to increase as the graph goes down as the delay duration is increased. The highest quality of the audio was at a delay of 40 ms for mp3 format file. The quality for .wav and .au files the quality is very bad. The buffer size is maintained constantly i.e. 1024 bytes.

The delays used for mp3 file are 10,20,30..100. The delays used for .wav and .au files are reduced to very less duration compared to the mp3 delays; for instance 1,5,10,15..30.

The parameters considered in the program is delay; which also implies that the audio quality is affected by packet loss and jitter.

The three factors affecting the quality of the audio file are

- Packet loss:

Ideally, a network would never lose a single packet. Of course, in the real world they do, and for two reasons. Every Transmission medium will flip a bit once in a while, and then the whole packet is lost.

If such an error occurs, the lost packet needs to to be retransmitted. But if packet loss is too high, UDP will run out of buffer space and the transfer has to stop until the retransmitted lost packet has been received.

The second reason packets get lost is because UDP is sending so fast that switch buffers fill up faster than packets can be transmitted. When a buffer is full and another packet comes in, the switch can do only one thing: "drop" the packet.

- Jitter:

Jitter is the variation in the delay of received packets. High jitter results in choppy voice or temporary glitches and packets can even get dropped when they arrive with excessive delay. Jitter on UDP voice stream results noisy output.

- Transmission delay

Transmission delay is the amount of time required to push all the packet's bits into the wire. Here we noticed that increasing the delay from '20' to '100'. The quality of songs improves steadily until the point '50' for .mp3 file. After this the quality decreases. And the song was in fast forward till 10 msec. But for the .wav and .au file there was long delay and song was played for few seconds and by decreasing delay no voice can be heard and by increasing the same there was audio played for 2 sec. The reason for this is the file size of the format. As these large files size makes WAVs impractical streaming.

<u>Commands</u> :

The transmitter is invoked by using the following command.

Command: ./udp destination_IP port_number audio_file delay_duration

For instance: ./udp 10.192.254.133 2025 b0.mp3 50

The ffmpeg file was invoked as we copied the ffmpeg file provided to our home directory and used the following commands to convert the files.

./ffmpeg -i b0.mp3 b0.wav
./ffmpeg -i b0.mp3 b0.au

Hardcoding the buffer size: 1024 bytes

<u>CODE</u>:

```
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<netdb.h>
#include<sys/types.h>
#include<netinet/in.h>
#include<sys/socket.h>
#include<arpa/inet.h>
#include<time.h>
#define _POSIX_C_SOURCE 200809L
int main(int argc,char* argv[])
{
    if (argc!=5)
        {
          printf("Arguments Missing: Please enter all the arguments and execute the program again\n");
          return 0;
```

```
    }
 int portnumber, delay, buffersize=1024, n=0;
 char buffer[buffersize];
 long ms;
 struct timespec t;
 portnumber = atoi(argv[2]);
 delay=atoi(argv[4]);
 ms = (1000000*delay);
 t.tv_nsec = ms;
 FILE *filename = NULL;
 filename = fopen(argv[3], "rb");
 struct sockaddr_in socpart;
 socpart.sin_port = htons(portnumber);
 socpart.sin_family = AF_INET;
 inet_pton(AF_INET,argv[1],&(socpart.sin_addr.s_addr));
 int dnserverid = socket(AF_INET, SOCK_DGRAM, 0);
  if (dnserverid<0)
   {
     printf("Socket error \n ");
      return 0;
   }
 if (filename != NULL)
     {
        for(;;)
               {
                    do
                      {
                           n = fread(buffer, 1, buffersize, filename);
                           sendto(dnserverid, buffer, n, 0, (struct sockaddr *) &socpart, sizeof(socpart));
                            nanosleep(&t,&t);
                       }
               while(n>0);
               if(n<=0)
                      {
                             printf("No more data to send. Closing program.");
                              break;
                      }
        fclose(filename);
        close(dnserverid);
                  }
           }
   return 0;
}
```

udp.c