**P2: UDP Radio Transmitter**

**Group M2 (**[bew7@njit.edu](mailto:bew7@njit.edu)**)**

**Regarding our radio transmitter**

* Called from the command line using the following parameters:
  + ./udp <audio\_file> <destination\_ip> <ms\_delay>
  + Example: ./udp quartet.mp3 10.200.48.206 50
* We hard-code two important parameters within the program:
  + The **destination port: port 44444**
  + The **buffer size** (and size of data we try to send at once): **1500 bytes**
    - Though we change this size once for testing purposes (see Figure 1.2)
* We used ffmpeg to convert sample .mp3 files to .wav and .au. An example ffmpeg call used follows:
  + ./ffmpeg -i quartet.mp3 quartet.au

**Initial Tests: with a constant buffer size, we observed delay’s impact on sound quality**

**Figure 1.1: Sound quality “peaks” at a certain delay, then gradually drops as distance from peak increases**

When delay is **too high**, the sound experiences **pauses**; when it’s **too low**, the sound **skips forward**

* Quality drops roughly in a **bell curve** from the peak for all three audio formats
* The delays used for .mp3 are (30,40,50,60,70,80,90,100,110); delays used for .wav and .au are 1/10th these (3,4,5… etc.). This is because .mp3 is extremely low-quality at very low delays (it constantly stutters and skips, finishing a 30-second sample in less than 3-4), while .wav/.au experience massive pauses between tiny blips of sound given .mp3’s higher delays
* Interestingly, .wav and .au peak around 7, which is 1/10th of the latency peak seen for .mp3. Likewise, when the difference in latency “step size” is accounted for (10 for .mp3 vs. 1 for .wav/.au), the graphs have remarkably similar curves

**(Note: The exact latency peak differs between testing locations and conditions, but the bell curve is the same)**

How does bitrate factor in?

* The bitrate of one of our samples in .mp3 format was 160 kb/s. The bitrate of the same sample in .wav and .au is 1536 kb/s; thus, there is **an ~10x increase in bitrate moving from .mp3 to .wav/.au**
  + File size experiences the same ~10x increase, from 536KB to 5.6MB
    - This is expected: the duration of both audio clips is 30 seconds, so the files with the 10x bitrate (.wav and .au) would need 10x more data to have the same duration as .mp3
* We’ve seen this before – **10x is the same difference between .mp3’s delay peak and .wav’s/.au’s!**
  + Likewise, .wav and .au are sending 10x more data to produce roughly the same sound

**Theory: if we increase our buffer size when sending .wav/.au by 10x, will the resulting data points look similar to our .mp3 graph above?**

**Figure 1.2: Despite sending the same amount of sound at roughly the same rate, sound quality takes a hit with larger buffers**

Our theory was incorrect: .mp3 peaks at 7.5 quality at 1500 bytes, while .wav/.au peak lower at 5/5.5 with 15000 bytes

* This means that sound quality is impacted by more than just the relationship between bitrate, buffer size, and delay. The most likely culprit is **connection inconsistency/loss (jitter) and fragmentation**
  + - Jitter is inevitable on any connection, and is expected to affect every packet equally. The reason why jitter has a larger impact on larger buffer sizes is because we overlooked something important – **our buffer sizes haven’t been the same as the sizes of our packets!**
      * All of our tests so far have been getting fragmented into multiple packets – our 1500-byte tests were fragmented into 3 packets per buffer, and our 15000-byte tests were fragmented into *21!*
      * Because of how IP fragmentation works, **if one fragment is missing, the whole datagram is lost –** so as the number of fragments increases (so **as** **buffer size increases**), the **rate of datagram loss increases**
      * When we hard-code a buffer size too small to activate fragmentation (1300), sound quality increases even further

If .wav and .au have ~10x higher bitrate, why don’t they sound better than .mp3?

* This probably has to do with how we made our .wav/.au files. **We converted from the *lower* bitrate into the *higher* one**; since ffmpeg can’t produce new samples from thin air, it had to have just duplicated the .mp3 samples ~10 times and played each for ~1/10th the duration, resulting in no real difference in sound quality