Audio Streaming and Chatting Application for Mobile Devices using Wi-Fi Direct and Performance Evaluation[1].

Hema Bhatia
hema.bhatia@cs.utah.edu

Naveen DS
naveends@cs.utah.edu

1. INTRODUCTION

In this project, we used the new Wi-Fi-direct [1] technology to build applications for audio chatting and audio streaming for Android phones and analyze performance of the streaming application. WiFi Direct is a new standard developed by WiFi alliance that allows peer-to-peer connection using WiFi technology. This is done by making one device act as a soft Access Point (soft-AP), and other peers connect to it. The device acting as a soft-AP is called the Group Owner and other peers are called the Group Members. Group Owner is decided either automatically based on capabilities of the device or manually when a device decides to be one. WiFi-Direct eliminates the need of an access point for collaboration between peers. This technology is very useful in many scenarios like sharing data, streaming audio and video, collaboration amongst peers in field-operations. In this project, we leverage benefits of WiFi-Direct and create applications for Android mobiles for peer-to-peer audio chatting and streaming. In order to find performance of our application and also see the performance of WiFi Direct, we have done experimental analysis of the streaming application we have built.

2. RELATED WORK

Since Wi-Fi Direct is a new feature, there are not many studies and research done on this. But whatever has been done, it has been done on the performance of physical and the data link layer. [2] is about mac layer where they discuss RTS, CTS mechanism and how they can be modified. Paper [3] talks about making the scheme energy efficient with respect to group formation and connection establishment between devices. But there is no work related to applications in WiFi direct and evaluating them in terms of range supported, and burst and loss pattern analysis.

3. METHODOLOGY

Our project consists of three major parts.

3.1 Audio Streaming Application

When two WiFi Direct devices are connected to each other, one peer can request to select an audio file from the other connected peer, and listen to the streaming of selected audio file directly. Packets are streaming over the UDP connection, and buffered on the play-out buffer at the receiver peer. Packet size to stream is based on sample rate, encoding, and configuration channels of the audio file being streamed.

3.2 Audio Chatting Application

Here, a two-way communication channel is established between the peers using TCP, and packets are recorded on microphone of one end and played on the other side. Packets are passed using UDP similar to streaming feature. For this feature, datagram packet size is measured using fixed parameters of sample rate (44100), configuration channel (STEREO) and PCM encoding (8 BIT). We decided these parameters after some research online of how audio frames get played out and from other audio chatting applications.

3.3 Performance Evaluation of Audio Streaming Application

In order to find performance of our application and also see the performance of WiFi Direct in terms of how much distance apart the mobiles can be and to understand the loss and burst patterns, we have done experimental analysis of the streaming application. We considered various environments for the experimentation like indoors, outdoors, and non-line of sight. We evaluated how the loss varies with the distance between peers in various scenarios, how burst pattern changes with the environment, and how the streaming works when peers have no direct line-of-sight.

4. IMPLEMENTATION AND EXPERIMENTATION[1]

4.1 Application Development

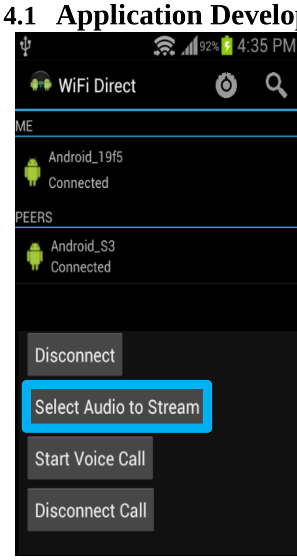

Figure 1: Audio Streaming selection

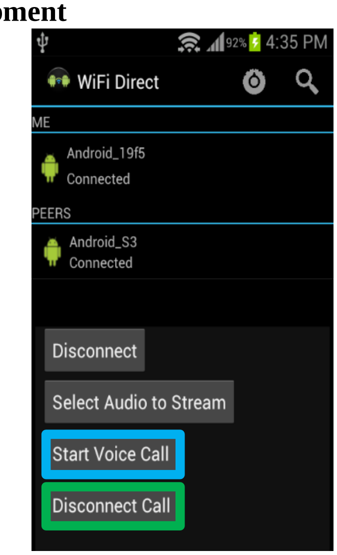

Figure 1: Audio Calling Selection

[1] Code link : https://github.com/hema09/wifidirect

Figures:https://github.com/hema09/AllFigures_WiFiDirect

We built an Android application, which uses WiFi-Direct ability of the phones to build the connection between peers. There are three important classes over which both the applications are built. First is activity class which handles discovering peers and choosing the peer to connect to. Second class handles connection between specific peers and displays the screen featuring our calling and streaming applications. Third class which is a background service, has all the functionality and decides how to send, receive and play audio packets between the peers. We got a stub for streaming from this blog [4]. *We use TCP for handling the control packets and UDP for transferring the data packets in our application.*

4.2 WiFi Direct Audio Streaming Application:

Once the peers are connected, they can select the audio file to stream from the peer they are connected to, and listen to the chosen file. Once the file is chosen (see Fig 1a), we decide the packet size to stream based on sample rate, encoding, and configuration channels of the audio file chosen. At the receiver peer's side, we make use of Android's AudioTrack class. The AudioTrack class manages and plays an audio resource and allows streaming PCM audio buffers to the audio hardware for playback by "pushing" (writing) the data to the AudioTrack object.

4.3 WiFi Direct Audio Chat Application:

Once the connected peers choose to start audio calling (see Fig 1b), TCP connection is setup between the peers, and AudioTrack object is setup to play-out the received datagram packets. We use AudioRecord class provided by Android to capture packets from the microphone and write to a stream. AudioRecord class manages the audio resources to record audio from the audio input hardware of the platform. This is achieved by fetching (reading) the data from the AudioRecord object. Datagram packet size we are using is measured using fixed parameters of sample rate (44100), configuration channel (STEREO) and PCM encoding (8 BIT) as explained earlier. At the received side we play out the received packets as we do in streaming.

4.4 Performance Analysis of WiFi Direct using Applications developed:

For measuring the performance, we use our audio streaming application. We are not evaluating audio calling performance as it cannot be repeated. We could have measured the performance of audio calling using Mean Opinion Score, but that involves lot of other factors which we did not experiment due to time constraints. We did experimentation under various environments like indoors and outdoors with line of sight, and indoors without line of sight. Our all experiments and analysis are discussed in the next section.

4.5 Experimentation and Results:

a. **Experiments with line of sight**: Here we measured packet loss changes with changes in distance. We took 5 readings at each distance ranging from 5m to 80m and took the average packet loss at each distance for each file. We measured packet loss statistics for four different file sizes, 2 Mb, 4Mb, 6Mb and 8 Mb.

   1. **Indoors**: First we took measurements in MEB second floor (IndoorMEB.png) but found that the results are not much satisfactory and did not make much sense. We realized that it may be due to the interference from WiFi accesses in the research rooms nearby (although we performed in the evening). Later we did the same experiments in WEB which has wider passages, and there are only classrooms and no labs. Readings here were good (Fig 2 and Fig 3). In both of these experiments, we found that we were able to establish peer connections even at distance of 60m.
   2. **Outdoors**: We did the same experiment as above in the MEB parking lot for 2 different file sizes, 4Mb and 6Mb. (MEBParkingLot.png)

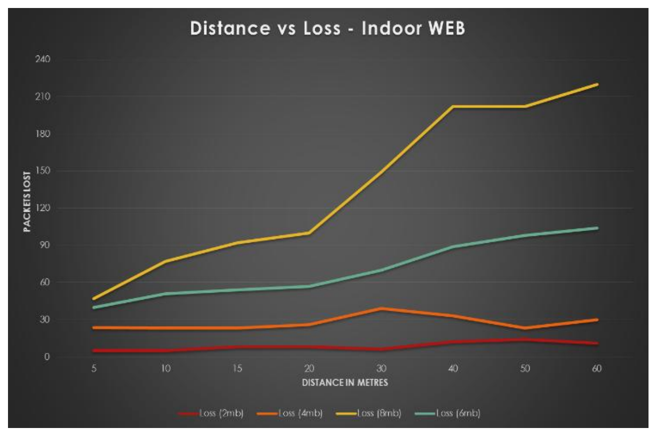

Figure 2 : Distance vs Packet Loss, Indoor WEB

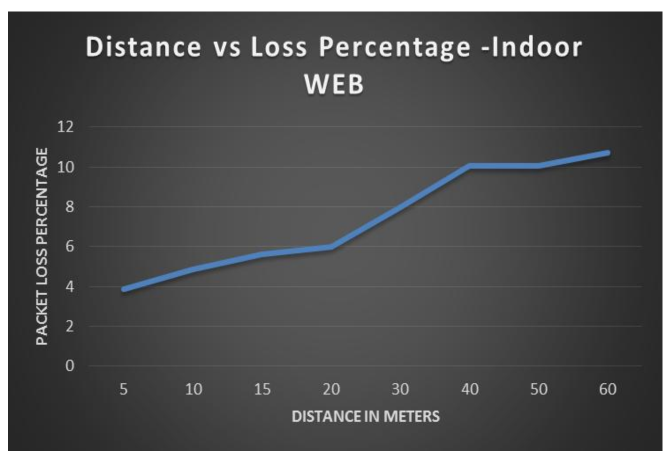

Figure 3 : Distance vs Packet Loss Percentage - Indoor

b. **Experiments with no line of sight** :
We conducted this experiment in four different scenarios. *First* experiment was done in WEB with peers on opposite side of the concrete wall. We had clear connection with almost no losses but losses increased drastically when we moved by even 5m further. The good connection may be attributed to the fact that there was a glass window at the top of the wall. *Second* was in MEB with one peer inside the TA room and other outside. There is a thin wooden wall in between and we found clear communications with almost no losses.
Our *third* experiment was with peers in 2 different rooms with a room in between (1 peer in TA room, other in our classroom MEB 3105). Here we had problems with streaming, and there were big bursts of packet loss, but audio calling was very smooth. *Fourth*, we did the experiment in MEB-WEB tunnel, and we found that the results were very bad when there was no line of sight. There were lot of losses, and we think it might be due to the concrete walls.

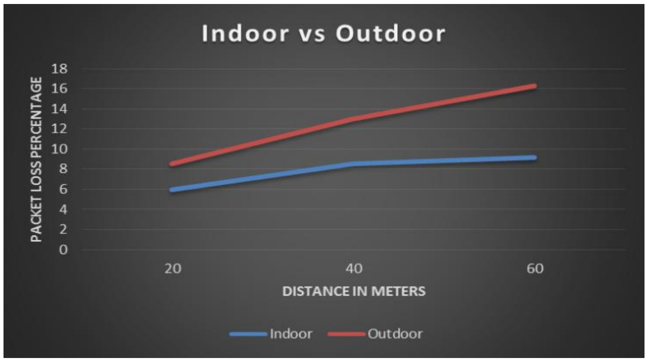

Figure 4: Distance vs Packet Loss Percentage, Indoor VS Outdoor

c. **Analysis**:

1. **Distance vs Packet Loss**
We found that packet loss increased with increase in distance as we had expected in both indoor and outdoor settings. Losses were more in outdoor settings compared to indoor as expected again (Fig 4), but the performance here was average and streaming was smooth, till 40m and then the quality worsened. At 80m we had frequent disruptions in connection and hence we were not able to perform any measurements. As we see from the graphs (Fig 3), it was not completely linear or exponential and hence we were not able to come up with a perfect model suiting the values.

2. **Burst Analysis**
In indoor settings, our audio file streamed smoothly with only a few small bursts. BurstPatterns.png shows the burst pattern we observed in almost all cases. We also observed few outliers like BurstPatterns.png (Fig1) in outdoor settings, bursts were higher compared to indoor, as expected. You can see the diagrams of burst analysis here BurstPatterns.png. We observed that the burst pattern was like if a burst occurs, the losses persist with a peak, which we take as max burst and then smoothens where there are no losses and then another burst occurs.

4.6 Challenges Faced:

There were many challenges faced by us during the development of our project.

a. Available information for studying Wi-Fi Direct was very less and we could not find any papers detailing the lower level details of WiFi Direct.
b. Learning about audio file formats
All files had different formats, and we had to learn about the details of them for streaming the audio packets.
c. Performance measurements
Like in MEB indoor, we did not get correct results in one attempt and we had to analyze what went wrong and re-conduct the experiments.

5. CONCLUSION

Based on the application we designed and evaluated, we feel that WiFi Direct has an excellent future with respect to Peer – Peer communication. We conclude that the applications we have developed will be really useful in the places where there is no internet connection and people want to communicate. Our tests indicate that the maximum distance between peers, with line of sight (indoor) can be till 70 meters and outdoors till 60 meters. And also we were easily able to talk from one room to a room that was two rooms apart (~40 meters), so our streaming application can be used in a situation where there is an ongoing talk and we want to stream the talk to others in different room. Also we observed concrete walls are hindrance for communication and also when there is moisture in the air, we saw more losses because of reflection and diffraction due to moisture particles.

5.1 Future Work

Based on our tests and evaluations we saw that, in audio calling the voice is not clear enough, so we can improve the voice clarity by some of the known methods like equalization. Developing a video chat application and evaluating it will also be a good work to do in future.

REFERENCES

1. Wi-Fi Direct information from Wi-Fi Alliance website: http://www.wi-fi.org/discover-and-learn/wi-fi-direct

2. Wi-Fi in Ad Hoc Mode: A Measurement study, by Giuseppe et al.

3. Designing Energy Efficient Access Points with Wi-Fi Direct, by Daniel Camps et al, NEC labs.

4. Streaming Audio in Android - Stub: https://code.google.com/p/rxwen-blog-stuff/source/browse/trunk/android/streaming_audio/#streaming_audio