Sindh Univ. Res. Jour. (Sci. Ser.) Vol. 48 (2) 285-288 (2016)

SINDH UNIVERSITY RESEARCH JOURNAL(SCIENCE SERIES)

Development of Mobile Ad-hoc Network android application using WIFI direct

M. WAQAS, K. MAHMOOD[++], J. KHAN

Department of Electrical Engineering,Iqra National UniversityPeshawar, Pakistan

Received 18thOctober2015 and Revised 11th March 2016

Abstract: During the transmission of voice over wireless broadcast ad-hoc network, a packet loss is commonly seen which may result in disruptions or error. Packet loss is encountered when one or more than one packet is lost during transmission. To keep this loss to a minimum level, a network codec is required at the receiver side. In this paper an android based application is developed to see whether the dropped packets are recoverable using network codec (Reed Solomon code) in real-time environment. A music wav file was encoded and sent wirelessly. Packet loss is intentionally introduced at transmitting side to see if the original file could be recovered at receiving side by using the proposed application. Practical results show that drop packets were recovered successfully within reasonable time span. It was also observed that those files in which packets were lost were received more quickly than the non-dropped packets.

Keywords: ad-hoc network, Reed Solomon, network codec, WIFI-Direct, Android, API, WIFI

1. INTRODUCTION

During the transmission of voice over wireless broadcast ad-hoc network, certain packet loss is commonly seen which may results in disruptions or errors. Packets are lost due to several reasons such as delay, routing etc. This causes loss of sound at the receiving side. More packet loss can cause extensive communication failure and decline in the transmission range of the nodes. To retrieve the lost packets in the network, a network codec is vital. The network codec uses Forward Error Correction (FEC) technique to encode, produce and transfer redundant data, the data is then used in the receiver to decode and recover the lost packets. We have used Reed Solomon code as the FEC technique and based on this technique an android based application is developed to see whether the dropped packets are recoverable.

Android delivers an abundant amount of Application Program Interface(API) to simplify communication between the software and the operating system. The WIFI-Direct is an API that provides an ad-hoc connection in android devices. Since WIFI-Direct technology is relatively new, most developers are therefore unaware of this specific API. This API includes a variety of possible uses for communication systems. It has been developed upon the well-known WIFI requirement, meaning that it is a part of the IEEE 802.11 group. WIFI-Direct supports data rates of all the IEEE 802.11 protocols except for the IEEE 802.11b. An important advantage of WIFI-Direct is the efficient use of wireless bandwidth.

WIFI-Direct devices also have connectivity for legacy WIFI devices. This means that a normal non Smartphone WIFI device can also discover and connects to the WIFI-Direct groups. A WIFI-Direct group is a group of devices which are connected via WIFI-Direct to create an ad-hoc network. A group owner will appear as a simpleAccess Point(AP) for legacy equipment.

This paper is arranged as the following, Section II gives us an overview of the WIFI-Direct API and networkcodec, Section III shows us the system model and SectionIV shows us the results. Concluding remarks are and conclusion from given in Section V.

2. OVERVIEW

2.1. OVERVIEW OF WIFI-DIRECT API

Since WIFI-Direct is still rather new technology, not a lot of work has been done in it. A full overview of the unique functionalities defined in WIFI-Direct, and an experimental assessment that shows the performance to be estimated in real circumstances has been performed in this paper to show the delays estimated in practice when WIFI-Direct devices discover each other and establish a connection, and the performance of its new power saving protocols (Conti *et al.,* 2013). The WIFI-Direct API has been used to send text data only on experimental basis such as finding the position of a human being using WIFI using IEEE1588 Precision Timing Protocol (PTP) (Liu *et al,* 2011).

WIFI-Direct is a new technology that has made its entry into the smartphones world recently. Android 4.0 was the first version with built-in WIFI-Direct abilities (Google Inc. and Open Handset Alliance. Android 4.0 Platform Highlights). The first announcement of this platform was debut in October 2011, one year after the WIFI alliance (WFA)'s launch of WIFI-Direct. On May 1, 2012 roughly 5% of the users who accessed Google

[++]Corresponding author email:drkhalidmahmood@inu.edu.pk

Play operated on a device with an Android 4.0 - 4.0.3 platform (Google Inc. and Open Handset Alliance. Platform Versions). The reason for this small number of users is the low growth of the platform, and the small number of Android users with WIFI-Direct friendly smartphones. The Android API level 14 and above are competible with the applications to discover, connect and communicate with each other by using WIFI-Direct (Google Inc. and Open Handset Alliance. WIFI-Direct).

2.2 Overview Of Reed Solomon Code Network Codec

A network codec is needed to encode and decode the music wav files, while recovering the file to its original state if the packets were dropped. Unlike Automatic Repeat Request (ARQ), Forward Error Correct (FEC) transmits extra encoded bits or data called redundant data to remove errors. It is appropriate for real-time transmission but at the cost of increased data rate. For voice transmission in real-time FEC is a decent choice. The concept of FEC is functional on packet level. Reed Solomon codes are non-binary maximum distantce seprable (MDS) codes created by Irving Reed and Gustave Solomon. By adding redundant symbols.

$$Redundant\ Symbols = n - k \quad (1)$$

Where n is the block length and k is the message length.

It can detect and correct symbols given as

$$Detect\ \&\ Correct\ Symbols = \ (n-k)/2 \ (2)$$

In case of erasures, it can recover symbols equivalent to equation 1 (Filippo Casu et al., 2011). Reed Solomon coding is applied on packet level instead of bits to recover lost packets. The size of the n is fixed while p is a prime number and m represents number of bits in a symbol and is calculated as:

$$n = p^m - 1 (3)$$

The amount of redundancy for data packets can be increased or decreased by the value of k as it is variable. The values of n and k is taken as 31 and 15 while p and m as 2 and 5. A symbol is represented by five bits. Symbols from each of the data packets are encoded to generate 5 redundant symbols each in separate packets. So for every 15 data packets, 5 redundant packets are generated. One of the main advantages of Reed Solomon encoding is that it can recover packets in case of burst of packet lost regardless of their position in the group. The amount of redundancy can be increased and decreased by changing the value of the n and k. The value of n depends on the parameter p and m. As already mentioned, for reconstructing the data of one packet regardless of its position in the group only one redundancy packet is required. The value of n and k was also changed to check on the quality, n was changed to 32 and k was changed to 16 which improved the positioning of the packets and improved from previous test. Since real time transport protocol (RTP) packets are used by the WIFI-Direct API and Reed Solomon is applied on packets.

3. SYSTEM MODEL

The main class called the WIFI_P2PManager, has the following 3 main parts:

- Intents
- Request functions
- Listeners

The messages that go through WIFI-Direct works asynchronously. This API requires a listener return or callback functions that are responsible for replying to requests from the application. Following are the 5 listener types that are used:

- Peer List Listener
- Channel Listener
- Action Listener
- Connection Info Listener
- Group Info Listener

Each of these listeners has a return function which is activated when a response is sent. The action listener's return function notifies whether the action was successful or not. The channel listener's return function will be active if the channel gets disconnected from the network (Google Inc. and Open Handset Alliance. WIFIP2pManager.ChannelListener). The remaining listeners are called when an exact request is sent and the information is available.

The WIFI-Direct API has nine different request functions. While some of these requests are required to be implemented, others are just optional. In order to implement WIFI-Direct function in an android application, a registration is required to the WIFI framework (Google Inc. and Open Handset Alliance. WIFIP2pManager). This is achieved by running the initialize function, while all the other request functions in the WIFI-Direct API rely on this registration. This must always be the first WIFI-Direct step to be executed. When the application is in the initialized mode, it will be able to find other nearby peer devices. To find the nearby peer devices, the application must perform the discover Peers function. This function initiates a peer discovery, which means sending a request to the device to scan for available peer devices. When the peer devices are discovered, it can then request for all the current devices from the network by calling the request Peers function. When the available list of nearby peer devices has been discovered and received, a connection request can be started with one of the devices in the list. This is done by executing the connect function. If the requested device is not a part of the P2P group, this request will start a group negotiation with the peer device (Google Inc. and Open Handset

Alliance. WIFIP2pManager). A group negotiation is a must in order to decide which device is going to be the group owner. If a group negotiation is cancelled, the cancel connect function is then executed. When there is a successful group negotiation and the application knows that the connection has been established, it can find out if any network connectivity exists. If a connection has been established, a request for connection info can also be requested by executing the request connection info function. If a group is created, a request for the group info can be requested by running the request group info function. The create group function makes the current running device create an empty P2P group with its own self while acting as the group owner. This function is only meant to be used where the peer devices are legacy devices, and will not be used in ordinary WIFI-Direct operations. This request method is specifically useful since the WIFI-Direct technology is pretty new to most consumer's devices, but this does not mean that other device can't be a group owner. In order to disconnect from the network a disconnection request can be sent to the connected group, the remove Group function must be executed.

For a WIFI-Direct device to be able to know when a certain WIFI-Direct specific event has occurred, it needs to listen for broadcasted intents. Broadcast receivers with intent filters need to be listed at the application. The following intent actions are applicable for WIFI-Direct (Google Inc. and Open Handset Alliance. WIFIP2pManager):

- WIFI_P2P_PEERS_CHANGED_ACTION
- WIFI_P2P_STATE_CHANGED_ACTION
- WIFI_P2P_THIS DEVICE_CHANGED_ACTION
- WIFI_P2P_CONNECTION_CHANGED_ACTION

To start a discovery process and connect with the peer devices, WIFI-Direct must be enabled on the current device, this will be broadcasted by the Android system. By registration of a broadcast receiver with an intent filter including the WIFI_P2P_STATE_CHANGED_ACTIONintent action, the device can be informed if WIFI-Direct is enabled on the current device or not. Each time the user turns the WIFI-Direct on or off, an intent will be broadcasted causing the device to constantly be updated on the WIFI-Direct mode's status. If the WIFI_P2P_PEERS_CHANGED_ACTION intent action is also added to the filter then the device can be informed when peers are found. In order for the device to identify any changes in the WIFI connection, the WIFI_P2P_CONNECTION_CHANGED_ACTIONmus talso be added into the filter. By adding the WIFI_P2P_THIS_DEVICE_CHANGED_ACTION intent to the intent filter, the device will be informed when an alteration in the device's P2P properties has occurred.



Fig. 1. System Model

4. EXPERIMENTAL RESULTS

Different tests were conducted on this application to test the durability and the reliability of the network coder and the connection. A group of 3 different sized files giving a total of 15 tests were done to check the time taken for each file by changing the distance in feet. Each group was tested against two conditions, namely without data loss and with data loss.

i. No Loss data transmission

At first wav files of the size 2, 5 and 10 MB were sent without adding any loss to it. Each file was sent and received at different distances which can be seen in **(Tables 1, 2 and 3)** and is plotted in **(Fig.1).**

Table 1. No loss transmission for 2MBfile size

2MB File	
Distance	**Time Taken**
1 Foot	2435 ms
10 Feet	2565 ms
20 Feet	2879 ms
30 Feet	4210 ms
40 Feet	5252 ms

Table 2. No loss transmission for 5MBfile size

5MB File	
Distance	**Time Taken**
1 Foot	4886 ms
10 Feet	4998 ms
20 Feet	5241 ms
30 Feet	6845 ms
40 Feet	7526 ms

Table 3. No loss transmission for 10MBfile size

10MB File	
Distance	**Time Taken**
1 Foot	8983 ms
10 Feet	8779 ms
20 Feet	9685 ms
30 Feet	11253 ms
40 Feet	12985 ms

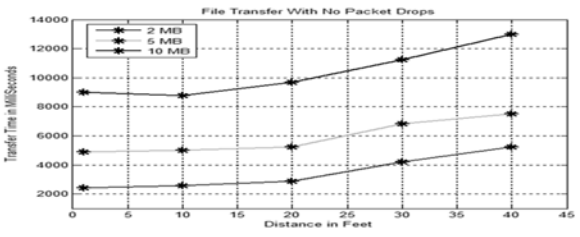


Fig.2.No packet loss data

ii. With Loss data transmission

The wav files were then sent with loss to it. Loss is added by reading the file as binary and then changing the bits with null values on each even value. Each file bytes are nulled according to their file size at almost half. Each file was sent and received at different distances which can be seen in **(Table 4,5 and 6)** and plotted in **(Fig. 2).**

From the graph in Fig. 2 it's shown that varying the distance and file size affects the time taken for each file to be sent from point 'A' to point 'B'. For 2MB file we can see that the values and the graph is almost the same and does not changes significantly. The main difference is seen, when the file size and the distance is increased. A clear rise in the time is seen in **(Fig. 1)** when the file size is 5MB and the distance is 30 feet, while in **(Fig. 2)** the same file has a somewhat gradual rise.

Table 4.loss transmission for 2MBfile size

2MB File	
Distance	**Time Taken**
1 Foot	2133 ms
10 Feet	1939 ms
20 Feet	2664 ms
30 Feet	3884 ms
40 Feet	4465 ms

Table 5. loss transmission for 5MBfile size

5MB File	
Distance	**Time Taken**
1 Foot	4101 ms
10 Feet	3885 ms
20 Feet	3784 ms
30 Feet	4462 ms
40 Feet	6445 ms

Table 6.loss transmission for 10MBfile size

10MB File	
Distance	**Time Taken**
1 Foot	5668 ms
10 Feet	6685 ms
20 Feet	7985 ms
30 Feet	10232 ms
40 Feet	10989 ms

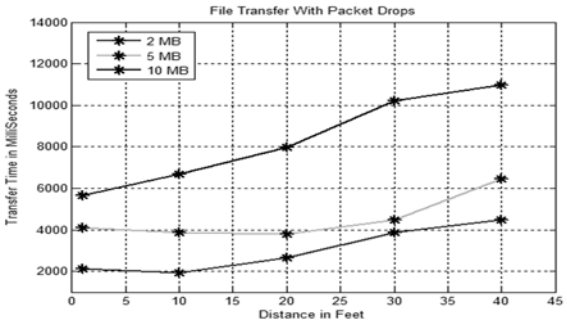


Fig.2. With Packet Loss

The main difference can be seen when the file size is changed to 10MB, we can clearly see from Figure 1 that the time rises with increasing the distance while in Figure 2 we see that the graph changes gradually and the time taken is much **less** with the packet loss than the time taken without packet loss.

5. CONCLUSION

Packet loss in a real-time voice communication in Ad-hocnetwork creates disruptions in sound.Continuous increase in packet loss can cause complete failure of voice communication. In order to remove the disruptions, a network codec can be used. In this paper an android based application is developed to see whether the dropped packets are recoverable using the network codec (Reed Solomon code) in real-time environment. A music wav file was encoded and sent wirelessly. Packet loss is intentionally introduced at the transmitting side to see if the original file could be recovered at the receiving side by using our newly developed application. Practical results show that drop packets were recovered successfully within reasonable time span. It was also observed that those files in which packets were lost were received more quickly than the non-dropped packets.

REFERENCES:

Casu, F., F. J. Narciso, G. J. Cabrera, C. J. Cabrera (2011) Inter-Packet Symbol Approach To Reed-Solomon FEC Codes For RTP-Multimedia Stream Protection. In: IEEE,

Conti, M., F. Delmastro, G. Minutiello and R. Paris, (2013) "Experimenting opportunistic networks with WiFi Direct," in Wireless Days (WD), IFIP, Valencia.

Google Inc. and Open Handset Alliance. Android 4.0 Platform Highlights. http://developer.android.com/sdk/android-4.0-highlights.html

Google Inc. and Open Handset Alliance. Platform Versions. http://developer.android.com/resources/dashboard/platform-versions.html

Google Inc. and Open Handset Alliance. WIFI-Direct. http://developer.android.com/guide/topics/wireless/WIFIp2p.htmI

Google Inc. and Open Handset Alliance. WIFIP2pManager. http://developer.android.com/reference/android/net/WIFI/p2p/WIFIP2p Manager.html

Google Inc. and Open Handset Alliance. WIFIP2p Manager. Channel Listener. http://developer.android.com/reference/android/net/WIFI/p2p/WIFIP2pManager.ChannelListener.html

Liu, X., J. Huang and Z. Chen, (2011) "The human positioning system based on the WiFi Direct and Precision Time Protocol," in Transportation, Mechanical, and Electrical Engineering (TMEE), International Conference on , Changchun.