AssistDirect: A Framework for Multi-Hop Mobile Ad-Hoc Networking

Adam Engelhart, Yoram Haddad, Yinon Mishali
Jerusalem College of Technology,
Jerusalem, Israel
haddad@jct.ac.il

***Abstract*—As modern technology continues to develop, cultures become more reliant on the constant need to be connected. This connectivity is usually dependent on local access points, such as Wi-Fi routers, modems and cellular service providers. When there is no access point available, users are left without a practical means to remain connected. Ad-Hoc networking is a viable solution to this issue, however implementing it is very challenging and complex. In this paper we provide a framework for Mobile Ad-Hoc network (MANET) based on Android applications. This framework uses Wi-Fi Direct technology, which would enable connectivity in cases where there is no reachable and accessible wireless access point. In addition, it supports Multi-Hop routing and IP addressing among other requirements for proper networking together with network management which can be incorporated into any Android application to provide consistent and dependable connectivity.**

***Index Terms*—Device to Device, Ad-Hoc communication, Android, Mobile application, Emergency communication**

I. Introduction

Wireless connectivity is a means to transfer information and data between two or more devices without a physical medium connecting those devices. Wi-Fi (802.11) is an IEEE specification that regulates and defines wireless connectivity. There are different standards for Wi-Fi. Each standard is defined by its range for transmitting and receiving signals, range of frequencies, and rate of transmitting and receiving. There are two main ways in which this specification is implemented in real world settings. The first and more common usage are networks that have a static access point, which acts as the facilitator of connectivity by means of defining the parameters for which each endpoint communicates (Includes defining IP address, choosing which channel the communication will occur on, and next hop routing). This is also known as infrastructure mode. The second usage is known as Ad-Hoc Mode. Ad-Hoc Mode is defined as connectivity between two or more endpoints, with no man in the middle to facilitate the communication. This type of connection is also called Machine-to-Machine (M2M) or Device-to-Device (D2D) communication. Ad-Hoc mode relies on the endpoints to define the parameters of the connection, and requires unique solutions for reaching outof- range devices. For Ad-Hoc networks where the devices and their availability is infrequently changed, the solution is trivial. However when we deal with a dynamic Ad Hoc network where the devices and their reachable are constantly changing, defining the parameters of the connection, such as discovery, topology building and address distribution becomes more challenging. Another name for dynamic Ad-Hoc networking is Mobile Ad-Hoc Networking(MANET). This kind of communication mode is considered as a key aspect of the next generation cellular networks [1]. One of the most common use case for these networks is for disaster relief [2]. In this paper, we will discuss the issues pertaining to connectivity between android phones using the Wi-Fi Ad-Hoc Mode. Android supports a library called Wi-Fi Direct, which provides programmers with easy use of the Wi-Fi Ad Hoc mode. Wi-Fi Direct uses ISM (Industrial, Scientific and Medical) Radio Bands as the medium of communication between the devices network cards. When a programmer is interested in applying Wi-Fi Direct to an Android application, the application must properly interface with the Wi-Fi Direct API. Wi-Fi Direct enables connection between two or more devices in the same range, however, in its current form, it does not support multi-hop networks requiring out of range communication. As a result, Wi-Fi Direct is limited in its ability to build multi-hop networks. Therefore, there is a need to develop a platform for Android applications that can be used to create multi-hop networks based on Ad-Hoc networking for mobile devices. Within the guidelines of this paper, we outlined and surveyed different routing protocols, researched standards and unique techniques for passing data, and reviewed the basic parameters for network setup, in an effort to create a framework for Android applications to run on Wi-Fi Direct and support multi-hop networks. The framework was developed using Android Studio.

II. Background

Setting up a Mobile Ad-Hoc network comes with many challenges that need to be addressed. On the one hand, the medium for sending data is very limited, while on the other, the network is required to handle heavy traffic from the plethora of messages needed to sustain a dynamic network and its frequent adaptation. The adaptation referenced consists of accepting new members into the network, building and rebuilding routes to each device on the network, and noticing changes in connectivity (no connectivity, half duplex, full duplex). These changes require that the devices in the network react quickly and accurately to maintain the integrity of the network. However, this needs to be done with minimal overhead so as not to congest the network with too many messages. In this section,

we will address the main issues pertaining to the setup of a MANET using Wi-Fi Direct and different possibilities of how to address them. The section is divided into three sections: Routing, Addressing, and Message transfer

A. Routing

There are dozens of routing protocols that were designed to provide accurate topologies for Ad-Hoc Networks. Here, we will review a few of the more popular algorithms.

1) Flooding: Flooding [3] is the simplest of the routing protocols that we will address. Flooding works by having each member of the network broadcast a message. When another member receives the message, the message is then broadcasted a following time to let other members (who are possibly out of range of the originator) know that the originator of the message is in the network. The result of the algorithm is that each member knows all the other members in the network. There is no need for direct routing when using flooding, because all communication is done by broadcasting. Flooding is effective in transmitting and receiving messages, however, this protocol is not sustainable as the network grows. Studies [4] show that when there are more than five members in a network, dramatic packet loss occurs.

2) OLSR: OLSR [5] is a link state routing protocol, which means that the protocol builds first the topology for the network and then builds routes to each node in the network. OLSR generates two types of messages: Hello messages and TC messages. Hello messages contain a list of neighbors of the sending node, and are sent out by broadcast. TC messages contain a list of nodes that chose the sending member as an MPR (Multi-Point Relay), and are sent only to the sending nodes MPRs to be further proliferated. MPRs are selected nodes in the network that act as relays for other nodes. As opposed to flooding, OLSR allows only nodes that are MPRs to send out broadcast. MPRs are chosen based on calculations to determine the minimum amount MPRs required in a network to cover all of the nodes in the network. Finding an absolute minimum of MPRs to satisfy a network is considered an NP Hard problem, and therefore, the MPRs are calculated by mathematic approximations. In Figure 1, we can see the MPRs selected by Node A:

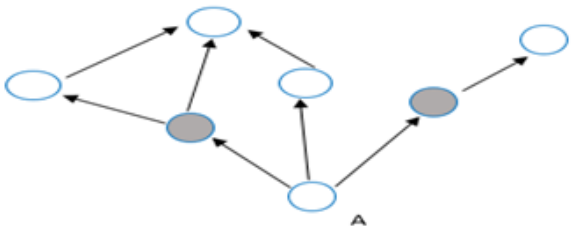

Fig. 1. MPRs of Node A

Each member maintains four tables that are used to generate Hello and TC messages and to calculate routes. The four tables are namely the neighbor table, the MPR Selector Table, the Topology Table and the Routing Table. The data stored in each table can be seen in Figure 2

Neighbor Table	IP Address	MAC Address	Link Status	2-Hop Neighbors	Sequence Number
Node <x>					t_i
Node <x>					
Node <x>					

MPR Selector Table	IP Address	Sequence Number of Neighbor Table	Sequence Number
Node <x>			t_i
Node <x>			

Topology Table	IP Address of Destination	IP Address of Sender (Last Hop to Destination)	MPR Sequence Number
1			
2			
3			
4			

Routing Table	Destination	Next Hop	Distance to Destination
1			
2			
3			
4			

Fig. 2. OLSR Tables

The Neighbor Table consists of all nodes of the network who are direct neighbors of the node that manages the table. This information is added to the table upon handling a received Hello Message. If the receiving node sees the IP address belonging to it in the received Hello Message, then the Link Status is set to bi-direction, otherwise, the Link Status is set to uni-directional. In addition, upon completing the computation to determine the MPRs for the receiving node, the state MPR can be assigned as Link status if one of the nodes was chosen as its MPR. The MPR Selector Table is also determined by receiving a Hello Message, and it records all nodes that chose the receiving node as an MPR. The Topology Table is determined from received TC Messages. For each node found in the TC Message, a record is created in the Topology table which contains the address of the node and its MPR. The MPR is always the last stop before a message arrives at its destination. The Routing Table is determined by the values of all the tables. Each record is the result of the calculation of the most efficient route to each known node based on the information stored in the tables.

3) BATMAN: BATMAN [6] is an algorithm that attempts to route messages based on the probability of the message succeeding to reach its destination. Each node periodically broadcasts an Originator Message (OGM). Each OGM is then rebroadcasted by the receiving node. Upon receiving an OGM, the receiving node takes note about the node it received the OGM from. When a node wants to send a message to a specified destination, BATMAN reviews which nodes the destinations OGMs came from. The node that relayed the most OGMs to the sending node becomes the next hop from the sending node. This process continues until the message reaches its destination. Because no routing tables are built

and maintained, BATMAN is a very lightweight protocol, with very little traffic for upkeep. In the case depicted in Figure 3, node 1 sends a message intended for node 6. Node 1 received more OGMs that originated from node 6 which passed through node 2 than from the other nodes neighboring node 1. As a result, node 1 sends the message to node 2, as node 2 has the highest probability of transferring the message successfully to node 6. Node 2 then repeats the process, and relays the message to the neighbor (excluding the previous sender) with the highest probability of reaching node 6 successfully. This process ends when the message successfully reaches node 6.

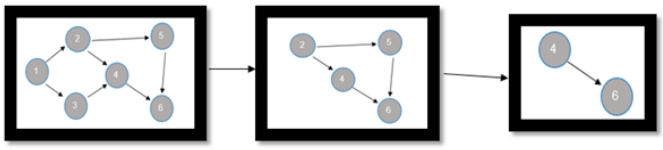

Fig. 3. From left to right: a) Node 1 sending message to Node 6 through Node 2; b) Node 2 relaying Node 1 message to Node 4; c) Node 4 delivering the message from Node 1 to Node 6

B. Addressing

Addressing schemes are techniques in which to assign each node in a network with a unique address. For static networks, two main schemes are widely used and provide fast and accurate addressing. The first technique is to hard code each node with an address that is changed only by manual manipulation by the IT manager of the network. The second technique is dynamic memory allocation, where one node on the network is responsible for assigning the other nodes with a unique address for a short time. In this case, the address is returned to the allocator when the borrowing node is offline or not in use. In most modern networks, DHCP (Dynamic Host Configuration Protocol) is used where the access point acts as the distributor of the addresses. In a mobile network, where nodes are constantly moving and entering and exiting networks, neither of these techniques are viable in their current forms. If each node has a hard-coded address, there is no system to ensure that two nodes have unique addresses. If a dynamic allocation approach is used, then issues can arise when the designated allocator leaves the network. Therefore, it is necessary to develop addressing schemes designed to adapt for use in MANETs. The two schemes designed for MANETs are namely neighbor based schemes and calculation based schemes. A neighbor based scheme is a means to provide addresses where each node knows all the other addresses already in use in the network, and therefore, when a new node joins the network, a neighbor can assign the new node a unique address while letting the other nodes know that the address assigned is no longer available. This scheme can cause congestion in the network, and if two or more new nodes join a network at the same time, there is a chance that they will be assigned the same address. Calculation based schemes have each device calculate its own address using mathematical manipulations. The advantages of this scheme are its efficiency and lack of overhead; however, this scheme does not provide a serious response to the issue of duplicate addresses in a network.

C. Data Transfer

Wi-Fi Direct provides an API for Android developers to use when building an application designed for P2P connectivity. The API allows for connectivity to take place in two modes: P2P mode and Group Mode. P2P mode is connectivity between two nodes, while group mode is connectivity between many nodes in the range of a specific GO (Group Owner), who then acts as the middle man between all nodes in the group. Each device, not including the GO, can only be connected with a single end point at any given time. Wi-Fi Direct in its current form does not support for multi-hop networks. It is important to note that in these two modes, the Wi-Fi Direct library is responsible for all aspects of connection setup, such as choosing the communication channel, distributing addresses, and managing timers. In addition, for two endpoints to connect to one another using Wi-Fi Direct, the user is required to physically press a button to confirm the pairing between the two devices. This poses a significant challenge. Requiring confirmation of pairing every time a node send out a message, such as a routing message, would significantly inhibit the effectiveness of the network setup, and would result in constant disruption and delays in the network. As a result, an alternative means of connection is necessary to ensure that data transfer can occur seamlessly and without disruption. In [7] a framework called MUMBLE is proposed, that uses the Discovery broadcast of Wi-Fi Direct to pass information, thus bypassing the pairing process between endpoints. The framework takes data from an external application, parses the data into 85-bit packets, and inserts the packets into the Discovery broadcast. Upon receiving a message, the framework extracts the packets, reconstructs the original data, and then relays the data to the application on the receiving endpoint. This process is done with no need for user intervention, and allows for communication between many endpoints simultaneously, bypassing the limitation of Wi-Fi Direct to allow a device to connect to only one other device at a time.

III. Our platform

In this section we present our platform that can be used to develop Android applications based on multi-hop, Wi-Fi Direct networks. The platform is responsible for network setup and maintenance, including proper addressing, constant routing calculations, and all other aspects to provide the successful transfer of data. In parallel to the development of this platform, an application was developed in order to test the platform and verify that it functions as required. This application is called AssistDirect. AssistDirect is an Android application where users can request service from others in the network, and receive responses according to what the other members of the network agree to. All requests are sent in broadcast and relayed further until every node received the request. A response to a request is sent in a unicast message.

The request message serves to test the effectiveness of the data transfer mechanism, while the response acts to gauge the accuracy of the routing protocol. Figure 4 displays a request message being sent out (green arrows, left part of the figure), and a response being sent (red arrows, right part of the figure):

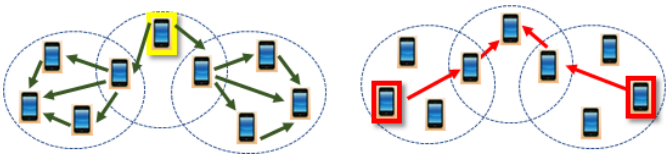

Fig. 4. Request and response sending simulation

The application runs on the platform developed. All messages that are sent or received run through MUMBLE which allows for the seamless data transfer to occur with no need for user intervention. The Messages are placed in a queue to preserve the order in which they were sent and ensure that each message is sent. Each data section of the message, is converted from the respective object and passed as a JSON string. Upon reception, the JSON string is converted back into the original object, and dealt with appropriately.

In addition to the data sent by the application, data is also sent by the routing protocol, providing the topology information for all nodes on the network. In designing the platform, we used OLSR, however, the platform is designed modularly, which allows OLSR to be easily replaced by other routing protocols. The routing protocol is managed by a service that starts running at device startup. The platform also uses a calculation based addressing scheme which is based on mathematical manipulations of the serial number of the device. Since the serial number of a device is unique, the chance of having two or more nodes in a network with the same address is very low.

The modules in the application run on different threads and communicate with each other by means of broadcasts and API interfacing. The routing service is constantly generating and sending messages, while simultaneously appending records to its tables and building routes. This service runs in the background, and has its own thread to ensure that it can function as it needs to, without delaying the rest of the application.

In each activity in the application, there exists a timer which is responsible for handling the various data that was received. The method constantly polls the MUMBLE service to get new data that may have arrived. For each message type (application based, routing based, or system based), the method sends it to the proper handler. In our application, we expect three different types of messages namely: 1) request, 2) response and 3) routing. The request and response messages were placed in the database designated for the application, while the Routing messages were passed to their respective handlers for analysis and to compute the various stages of the routing process.

In Figure 5, we show the entire platforms modules interaction diagram. The Application and Routing protocol send messages to a JSON parser, which then relays the newly created JSON to the MUMBLE Module to be sent out. MUMBLE also is responsible for receiving messages from other devices. The messages received are passed to the Application. The Application then handles the messages that are relevant to it, while relaying other messages to the appropriate modules. A developer can include different modules as needed, with the only requirement being that the application properly passes the data to the appropriate module. This modularity makes our platform easy to work with and adaptable, allowing for easy and intuitive developing experience.

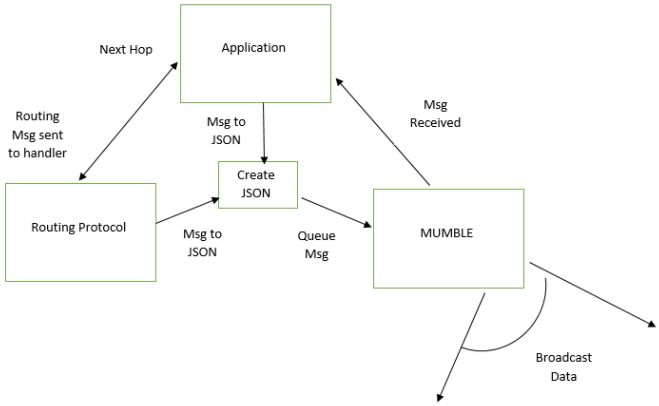

Fig. 5. AssistDirect Platform

IV. Conclusion

In this paper we presented a solution to Android developers by developing a platform which provides the basic modules required for communicating in multi-hop networks using Wi-Fi Direct technology. This platform includes routing mechanisms, and addressing scheme and enables a seamless data transfer without user intervention, which is not available in the current form of the Wi-Fi Direct library. By using this platform as a base, any Android application can be transformed into an application based solely on P2P connections.

References

[1] R. Chávez-Santiago, M. Szydełko, A. Kliks, F. Foukalas, Y. Haddad, K. E. Nolan, M. Y. Kelly, M. T. Masonta, and I. Balasingham, "5g: The convergence of wireless communications," *Wireless Personal Communications*, vol. 83, no. 3, pp. 1617–1642, Aug 2015. [Online]. Available: https://doi.org/10.1007/s11277-015-2467-2

[2] M. Friedman, Y. Haddad, and A. Blekhman, "Acoufind: Acoustic ad-hoc network system for trapped person detection," in *2015 IEEE International Conference on Microwaves, Communications, Antennas and Electronic Systems (COMCAS)*, Nov 2015, pp. 1–4.

[3] R. P. Marinho, U. B. Menegato, and R. A. Oliveira, "Imsn routing on wi-fi direct enabled devices," in *Proceedings of the 13th ACM International Symposium on Mobility Management and Wireless Access*, ser. MobiWac '15. New York, NY, USA: ACM, 2015, pp. 31–38. [Online]. Available: http://doi.acm.org/10.1145/2810362.2810375

[4] N. Ramakrishnaiah and P. Chenna, "A review of addressing protocols in mobile ad-hoc networks," *International Journal of Computer Applications*, vol. 132, no. 9, pp. 31–36, 2015.

[5] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot, "Optimized link state routing protocol for ad hoc networks," in *Proceedings. IEEE International Multi Topic Conference, 2001. IEEE INMIC 2001. Technology for the 21st Century.*, 2001, pp. 62–68.

[6] D. Johnson, N. Ntlatlapa, and C. Aichele, "A simple pragmatic approach to mesh routing using batman," in *In 2nd IFIP International Symposium on Wireless Communications and Information Technology in Developing Countries, Pretoria, South Africa*, 2008.

[7] A. Bhojan and G. W. Tan, "Mumble: Framework for seamless message transfer on smartphones," in *Proceedings of the 1st International Workshop on Experiences with the Design and Implementation of Smart Objects*, ser. SmartObjects '15. New York, NY, USA: ACM, 2015, pp. 43–48. [Online]. Available: http://doi.acm.org/10.1145/2797044.2797048