Wi-Fi Ad-Hoc and Cellular Hybrid Network

by

R.M.D.D.C. Ranasinghe

Registration No: 2015cs109

This dissertation is submitted to the University of Colombo School of Computing
In partial fulfilment of the requirements for the
Degree of Bachelor of Science Honours in Computer Science

University of Colombo School of Computing
35, Reid Avenue, Colombo 07,
Sri Lanka
July 2020

Declaration

I, Ranasinghe Mudalige Don Damitha Chathuranga Ranasinghe(2015CS109) hereby certify that this dissertation entitled Wi-Fi Ad-Hoc and Cellular Hybrid Network is entirely my own work and it has never been submitted nor is currently been submitted for any other degree.

Candidate: R.M.D.D.C. Ranasinghe

July 24, 2020

....

Date Candidate's Signature

I, C. I. Keppetiyagama, supervised this dissertation entitled Wi-Fi Ad-Hoc and Cellular Hybrid Network conducted by R.M.D.D.C. Ranasinghe in partial fulfilment of the requirements for the degree of Bachelor of Science Honours in Computer Science.

....

Date Signature of Supervisor

I, Kenneth Thilakarathna, Co-supervised this dissertation entitled Wi-Fi Ad-Hoc and Cellular Hybrid Network conducted by R.M.D.D.C. Ranasinghe in partial fulfilment of the requirements for the degree of Bachelor of Science Honours in Computer Science.

....

Date Signature of Co-supervisor

Abstract

In the past, only a handful of research was done in the area of Wifi Ad-Hoc and Cellular Hybrid Networks. The main problem with these networks is their unreliability. In the current time, the cellular networks have become more stable and much cheaper so this gives us an opportunity to use the cellular network as the control plane and backup network alongside with the Wifi Ad-Hoc network in order to provide a better solution.

The proposed architecture consists of two network interfaces cellular and Wifi Ad-Hoc in a multiplexing manner to provide a hybrid network connection. Every device consists of a demon(a simple program that monitors the state of the mobile devices connected to the network) that calls a server giving metadata about the node of the network. The server is responsible for managing the routing information of the active nodes in the network using the cellular control plane. The actual data transmission happens in the wifi ad-hoc network. The network returns to its initial state if one of the networks get disconnected.

A proof of concept network was implemented to explore how emerging technologies can be used to implement this mobile Ad-Hoc networking solution and to find out the implications of creating such a system in the end-users perspective.

When comparing the test results gathered it was evident that there were some desirable qualities like no-cost peer to peer connections, low latency single-hop connection and above-average power efficiency. But the downside to this is that latency increase and reaction time increase when using multihop Wifi Ad-Hoc connections for devices to device communication.

Keywords: Hybrid Network, Ad-hoc Network, Mobile Ad-hoc network, Android

Acknowledgement

I would like to express my sincere gratitude to my research supervisor, Dr. C. I. Keppetiyagama, senior lecturer of University of Colombo School of Computing for providing me continuous guidance and supervision throughout the research.

I would also like to extend my sincere gratitude to Mr Kenneth Thilakarathna, Lecturer of University of Colombo School of Computing and Dr. Primal Wijesekera, researcher of University of California, Berkeley for providing feedback on my research proposal and interim evaluation to improve my study by providing guidance. I also take the opportunity to acknowledge the assistance provided by Dr. H.E.M.H.B. Ekanayake as the final year computer science project coordinator.

Many thanks to my beloved mother and my dear father for always being my strength, showing me the correct direction, and making me who I am today. This thesis is dedicated to all my family members, to primary and secondary school teachers, to lecturers of University of Colombo School of Computing and to anyone who supported even by words to make my dreams come true.

Contents

List of Figures

List of Tables

Acronyms

AOSP - Android Open Source Project
VPN - Virtual Private network
4G - Fourth cellular generation
5G - Fifth cellular generation
LTE - Long-Term Evolution
OSI - Open Systems Interconnection model
OS - Operating system
TCP - Transmission Control Protocol
UDP - User Datagram Protocol
MPTCP - Multi-path TCP
MANET - Mobile Ad-hoc network
SSH - Secure Socket Layer
P2P - Peer to peer
SPAN - Smartphone Ad-hoc networks
LAN - Local Area Network
WAN - Wide Area Network
WLAN - Wireless Local Area Network
NAT - Network Address Translation”
PPTP - Point-to-Point Tunneling Protocol
KVM - Kernel-based Virtual Machine
DNS - Domain Name System
D2D - Device to Device
FTP - File Transfer Protocol
TTL - Time To Live
AP - Access Point
WPA - Wi-Fi Protected Access
ISP - Internet Service Provider

Chapter 1

Introduction

The popularity of mobile broadband has grown rapidly in recent years. Report from the International Telecommunication Union [3] has shown that in 2015 alone there are more than 7 billion mobile cellular subscriptions, corresponding to a penetration rate of 97%, up from 738 million in 2000. And it also states that globally, mobile broadband penetration reaches 47% in 2015, a value that increased 12 times since 2007. And the competitive nature of the mobile phone market has pushed manufacturers to manufacture and innovate in a rapid phase. This means consumers can buy highly sophisticated devices for very cheaper prices.

Nowadays mobile devices have multiple interfaces with different characteristics for their communication. These interfaces can be infrastructure networks like cellular or infrastructure-less networks like WIFI -Direct or Bluetooth. The availability of these vastly different network interfaces and communication protocols in mobile devices has open new research opportunities in providing reliable, low-cost, high bandwidth connections without any additional cost.

With mobile devices, Internet traffic surpassing that of computers the need for a reliable, low-cost high bandwidth mobile connection is apparent more than ever. Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2017–2022[4]. In the past, this problem has been addressed by implementing ad-hoc networks. This is a good solution as we have a high density of mobile devices in most of the crowded place. But the thing about the ad-hoc network is that they can be very unpredictable. As ad-hoc infrastructure-less in nature WIFI and Bluetooth connections are better suited than their cellular counterpart. The use of Bluetooth connection is not ignored due to the low bandwidth connections they make.

In the past researchers have tried to solve this problem by implementing an ad-hoc network using only the WIFI interface. To my knowledge, the most important part of an ad-hoc network is to keep track of information about the nodes in real-time. In the past researchers have used a separate channel in the ad-hoc network for this.(Liu, et al.2016)[5]. Due to the ad-hoc network being unreliable this is not a good solution.

These days the mobile devices have become very sophisticated. They possess multiple network interfaces with considerable processing power. Technologies like GSM and WIFI had become matured technologies. Nowadays GSM networks can provide good coverage in a very reliable manner. GSMA - Network Coverage statistics [6]. With the introduction of WIFI p2p technologies like WIFI-Direct, it has become very easy to form a peer to peer networks using WIFI interfaces. So this shift of paradigm provides a unique research opportunity in developing a hybrid network that uses WIFI peer to peer networks as its data plane and cellular network as its control plane and the backup data plane.

The proposed architecture has two network interfaces working in a multiplexing manner to provide network connections. Every device consists of a demon(a simple program that keeps track of the node state in real-time) that calls to a server giving metadata about the node. It processes those data and comes up with routing paths for the ad-hoc network. Then the server transfers back the relevant data for the available nodes to provide an optimized routing path. So when we need to transfer data to a device in the ad-hoc network it uses WIFI p2p rather than cellular saving the valuable bandwidth. Traffic that is gong to the Internet is routed to the GSM network using the cellular interface without any change.

Generally, WIFI networks have greater bandwidth than that of a GSM network. So it is evident that we can improve the speed if we use WIFI connection rather than a cellular connection. As we are using a much reliable GSM network for our control plane can provide a stable connection than a pure ad-hoc network. The usage of local routing in the ad-hoc network can provide a cheap reliable high bandwidth low latency connection in some scenarios. We are going to investigate the efficiency of this with the previous Android MANET implementation and current GSM connections to further justify the significance of this solution.

1.1 Motivation

According to the Telecommunication Development Bureau of ITU statistics [3] by the end of 2015, there are more than 7 billion active mobile cellular subscriptions worldwide. According to them globally, mobile broadband penetration reaches 47% in 2015, a value that increased 12 times since 2007. So it is obvious that mobile broadband is a dynamic market segment.

Telecommunication Development Bureau of ITU estimates that the 3G coverage has increased from 45% to 69% from 2011 to 2015. This corresponds to 29% present 3G coverage in rural areas and 89% in urban areas. Some developed countries like the USA have higher mobile broadband penetration at around 78% while some developing countries like Africa have a mobile broadband penetration around 17.4% So with the above statistical evidence, we can say that mobile broadband penetration needs to improve in most parts of the world.

With the emergence of web 2.0, most of the popular application has become web-based ones. The majority of current applications nowadays need a working Internet connection to function properly. According to the Cisco VNI Mobile[4], there is a significant increase in mobile traffic for the past few years and it is expected to grow at a rapid phase. They further point out that the share of smart devices and connections as a percentage of the total will increase from 53 per cent in 2017 to nearly three-fourths, at 73 per cent, by 2022. So with the above evidence, it is safe to say there is a shift of paradigm for Internet traffic from personal computers to mobile devices

According to the above source[4], the majority of users are shifting from static web content to dynamic web content in recent years. These dynamic content can vary from video streaming applications to multiplayer virtual reality games. As they are dynamic they need very high bandwidth, low latency connection to function properly.

As they point out in Facts and Figures 2019 by the Telecommunication Development Bureau of ITU[3] the cost of data for mobile devices has not become sufficiently cheap when comparing to that of the developing countries. According to them only developed countries and some of the developing countries can achieve the relevant pricing for broadband data in the world.

In recent years mobile devices have become sophisticate in various aspects. Mobile processors have gained in processing power while increasing power efficiency. This is evident when comparing the benchmark of those old ones with new ones. The charge capacity of current mobile devices batteries has gained many improvements in recent years. The majority of mobile devices have at least one-day battery life. They

Currently, almost all of the smartphones possess multiple network interfaces for communication. They can be of Cellular interfaces in which devices can support different protocols and multiple network bands or WIFI interfaces with dual-band support built. These networks have a rich diversity of properties unique to them. For example, cellular networks are reliable, high-cost, long-range with limited bandwidth connections while Ad-hoc networks have volatile, short-range and high-bandwidth connections. If somehow we can make a hybrid network using the best qualities of each interface then we can improve the quality of the network for a vast number of mobile broadband users.

In the past approach for this kind of problem is to make a mobile ad-hoc network using the WIFI interface of the devices. COMONet(Wijesekera and Keppitiyagama 2007)[1]. In the past, cellular technology has not matured to use it side by side with WIFI ad-hoc network. As we are approaching the problem in the present time a lot of scenarios have changed for the better. With the emergence of the Android open source project, it has become easy to work with the inner working of the mobile devices. In the past major mobile OS source codes were not publicly available. So to get an idea of the inner working of the device OS researcher needs to read the OS binary files using special kind of software and reverse engineer a solution [1]. modification Due to Android OS, using Linux as its kernel network modification to Android devices has become much more feasible than before.

So with these changes, most of the old limitations in solving such a problem are no more. So as a computer science student, it is evident that we need to revisit the problem. We need to use these newfound opportunities to provide a better solution in solving this apparent communication need. If we can provide such a solution it will become a widely impactful one.

1.2 Research significance

Most of the mobile devices in the present era needs Internet access to properly function. But the current offering from the cellular broadband providers is not on par with this demand. In the past, this kind of problem needs to be addressed using peer to peer Ad hoc networks. To create the ad hoc network researchers use a single network interface to implement the peer to peer network. Even though that was the best solution at that time it is not the case in the present. What this researcher is proposing is a hybrid between the cellular and WIFI peer to peer networks that uses the cellular network as its control plane and the backup connection while using the WIFI p2p connection as the main data channel.

With the emergence of new paradigms like AR and VR applications, online games, people need high-quality network connections for a cheap price to get most out of these applications. For example, let's consider the new AR multiplayer games. Those need low latency, high bandwidth connection between the local devices to function properly. Those applications need the Internet connection side by side to function. For this kind of application, this is an ideal solution. This will be a great addition to creating a local connection that needs more than the bandwidth provided by the general Bluetooth connections. Examples for this kind of apps can be varying from Video editing studios that can multiplex different video streams run on Android devices, Classroom teaching applications, AR, VR applications, Multiplayer games to Local peer to peer chat applications.

As this network can be implemented using already available hardware in the mobile systems there is no additional cost involved. Networking Protocols we use are already implemented in the Android OS and Linux kernel. The only drastic modification we are doing is to use root access to change the routing tables in the devices. If this can be implemented using the OS level we can mitigate this limitation as well.

As for the contribution to the Computer Science community, there is no implementation of this kind of network in the current time. So this research opens the door to a new frontier of technology. As this research is implementing a working prototype of the proposed network the researcher can see what kind of problems are there. Most of the previous implementations did not use cellular side by side due to its being less matured nature. But now it is a fully matured system in most of the areas of the world. So by revisiting the old problem, the researcher can understand the current problems that need to address in such a network.

In the past, node discovery has not been done in an automated manner. This research addresses that particular problem. This itself is a massive problem as the control plane needs to keep track of the node metadata in real-time. Control planes operating on cellular connections efficiently address this problem. This is a great milestone in the hybrid mesh networking paradigm.

The gains from the current hybrid network for the current smartphone users is a huge one. They will get a more reliable network with higher local bandwidth and lower latency for a cheaper price. This will create the opportunity for different categories of applications that rely on high local bandwidth connection as this connection is free compared to cellular ones.

1.3 Related Technologies

1.3.1 What are wireless Networks

A wireless network is a computer network that uses wireless data connections between network nodes. They are very useful in scenarios where it is either costly or impractical to use wires to connect the nodes in the networks. They are particularly useful for connecting mobile nodes to a network. They use different ranges of the electromagnetic spectrum to make wireless links between network nodes.

In general, when we increase the frequency of the electromagnetic spectrum in the data link we will have higher bandwidth. At the same time, this will decrease the rage of the datalink needing more nodes to cover the same area. On the other side if we use low frequency in the electromagnetic spectrum for the data link it will increase the range of the network while decreasing the bandwidth of the connection.

There are various kinds of wireless networks that are currently in use. Those can be currently categorized as fallows.

Wireless personal area Network(PAN)

These are used in connecting devices within a relatively small area, that is generally within a person's reach. Generally, they are low power, low bandwidth connections which used to connect personal devices. Bluetooth, ZigBee networks are examples of this kind of network.

wireless local area Network (WLAN)

wireless local area network (WLAN) links two or more devices over a short distance using a wireless distribution method. It usually provides a connection through an access point for Internet access.

The use of spread-spectrum or orthogonal frequency-division multiplexing(OFDM) technologies may allow users to move around within a local coverage area and remain connected to the network. There can be fixed wireless technology that implements point-to-point links between computers or networks at two distant locations, often using the dedicated microwave or modulated laser light beams over the line of sight.

WIFI network connections belong to this category of networks. Any device that uses the IEEE 802.11 WLAN standards are marketed under the Wi-Fi brand name.

Wireless Ad-Hoc Network

A wireless ad hoc network, also known as a wireless mesh network or mobile ad hoc network (MANET), is a wireless network made up of radio nodes organized in a mesh topology. Each node forwards messages on behalf of the other nodes and each node perform routing. Ad hoc networks can "self-heal", automatically re-routing around a node that has lost power

Wireless wide area Networks(WAN)

These networks can be used to connect distance nodes using directional antennas on the 2.4 GHz and 5.8Ghz frequency bands of the electromagnetic spectrum. we are not using this kind of networks in our research currently.

1.3.2 Cellular Networks

The major different of this network is that the network is distributed over land areas called ”cells”, each served by at least one fixed-location transceiver, but more normally, three cell sites or base transceiver stations. These base stations provide the cell with the network coverage which can be used for transmission of voice, data, and other types of content. A cell typically uses a different set of frequencies from neighbouring cells, to avoid interference and provide guaranteed service quality within each cell.[7]

When joined together, these cells provide radio coverage over a wide geographic area. This enables numerous portable transceivers (e.g., mobile phones, tablets, and laptops equipped with mobile broadband modems, pagers, etc.) to communicate with each other and with fixed transceivers and telephones anywhere in the network, via base stations, even if some of the transceivers are moving through more than one cell during transmission.

In cities, each cell site may have a range of up to approximately ½ mile (0.80 km), while in rural areas, the range could be as much as 5 miles (8.0 km). It is possible that in clear open areas, a user may receive signals from a cell site 25 miles (40 km) away.

There are several different digital cellular technologies, including Global System for Mobile Communications (GSM), General Packet Radio Service (GPRS), CDMA One, CDMA2000, Enhanced Data Rates for GSM Evolution (EDGE), Universal Mobile Telecommunications System (UMTS). Further details on those technologies are out of the scope of this research so it will be pointless to describe those here.

Cellular networks offer several desirable features:

- More capacity than a single large transmitter, since the same frequency can be used for multiple links as long as they are in different cells.
- Mobile devices use less power than with a single transmitter or satellite since the cell towers are closer
- Larger coverage area than a single terrestrial transmitter, since additional cell towers can be added indefinitely and are not limited by the horizon.
- Reliability of the network as it uses multiple node connections at the same time.

Currently, there are different generations of networks varying from the first generation networks to the fifth generation of networks for mobile phones. Currently, 3G and 4G networks are widely used worldwide for mobile broadband connections. 5G networks are currently at its early adoption stage As we are using those technologies lets consider those technologies in detail.

What are 3G Networks

3G is the third generation of wireless mobile telecommunications technology. It is the upgrade for 2G and 2.5G GPRS networks, for faster data transfer speed. This is based on a set of standards used for mobile devices and mobile telecommunications use services and networks that comply with the International Mobile Telecommunications -2000 (IMT-2000) specifications by the International Telecommunication Union.

3G telecommunication networks support services that provide an information transfer rate of at least 144 kbit/s.[8] Later 3G releases often denoted 3.5G and 3.75G, also provide mobile broadband access of several Mbit/s to smartphones and mobile modems in laptop computers. This ensures it can be applied to wireless voice telephony, mobile Internet access, fixed wireless Internet access, video calls, and mobile TV technologies.

it is expected that IMT-2000 will provide transmission rates: a minimum data rate of 2 Mbit/s for stationary or walking users, and 348 kbit/s in a moving vehicle," and this can be varied from a different carrier to carrier. [9]

What are 4G Networks

This as a communication standard set by the International Telecommunications Union-Radio communications sector (ITU-R). They specified a set of requirements for 4G standards, named the International Mobile Telecommunications Advanced (IMT-Advanced) specification, setting peak speed requirements for 4G service at 100 megabits per second (Mbit/s)(=12.5 megabytes per second) for high mobility communication (such as from trains and cars) and 1 gigabit per second (Gbit/s) for low mobility communication (such as pedestrians and stationary users)[10]

This standard is implemented using different IMT-2000 compliant 4G standards such as LTE Advanced, 3GPP Long Term Evolution (LTE), Mobile WiMAX (IEEE 802.16e). Currently, this is being replaced by a 5G standard.

What are 5G Networks

5G is the fifth-generation wireless technology for digital cellular networks that have begun wide deployment in 2019. Like previous standards, the covered areas are divided into regions called cells, serviced by individual antennas. The frequency spectrum of 5G is divided into millimetre waves, mid-band, and low-band. Low-band uses a similar frequency range as the predecessor.

Initially, the term was associated with the International Telecommunication Union's IMT-2020 standard, which required a theoretical peak download speed of 20 gigabits per second and 10 gigabits per second upload speed, along with other requirements. But the actual speeds can be ranged from 50 Mbit/s to over a gigabit. Mid-band 5G, by far the most common, will usually deliver between 100 to 400 Mbit/s.

Frequencies are above 24 GHz reaching up to 72 GHz which is above Extremely high frequency's lower boundary. The reach of these antennas is short, so more cells are required. Millimetre waves have difficulty traversing many walls and windows, so indoor coverage is limited.

Currently, there is no 5G coverage in Sri Lanka when conducting this research. SO we cannot practically say anything about this technology. But due to the limited coverage of these 5G networks need for a hybrid network to mitigate the downsides will become apparent more than ever.

1.3.3 What are WIFI Networks

Wi-Fi is a family of wireless networking technologies, based on the IEEE 802.11 family of standards, which are commonly used for local area networking of devices and Internet access.

Wi-Fi uses multiple parts of the IEEE 802 protocol family and is designed to seamlessly work with Ethernet. The data is organized into 802.11 frames that are very similar to Ethernet frames at the data link layer, but with extra address fields. MAC addresses are used as network addresses for routing over the LAN. Compatible devices can network through a wireless access point to each other as well as to wired devices and the Internet. The different versions of Wi-Fi are specified by various IEEE 802.11 protocol standards, with the different radio technologies determining radio bands, and the maximum ranges, and speeds that may be achieved.

Wi-Fi most commonly uses the 2.4 gigahertz (120 mm) UHF and 5 gigahertz (60 mm) SHF ISM radio bands. Most of the current devices can use both of these bands simultaneously. These bands are subdivided into multiple channels. Channels can be shared between networks but only one transmitter can locally transmit on a channel at any moment in time.

Wi-Fi's wavebands have relatively high absorption and work best for line-of-sight use. Many common obstructions such as walls, pillars, home appliances, etc may greatly reduce range, but this also minimizes interference between different networks. An access point (or hotspot) often has a range of about 20 meters (66 feet) indoors while some modern access points claim up to a 150-meter (490-foot) range outdoors.

Hotspot coverage can be as small as a single room with walls that block radio waves, or as large as many square kilometres using many overlapping access points with roaming permitted between them. Over time the speed and spectral efficiency of Wi-Fi have increased. As of 2019, at close range, some versions of Wi-Fi running on suitable hardware can achieve speeds of over 1 Gbit/s (gigabit per second).

Equipment frequently supports multiple versions of Wi-Fi but to communicate two devices must use a common Wi-Fi version. The versions differ between the radio wavebands they operate on, the radio bandwidth they occupy, the maximum data rates they can support and other details. Some versions permit the use of multiple antennas, which permits greater speeds as well as reduced interference. Fallow is a table explaining those different standards.

IEEE Standard	Maximum link rate	Adopted	Frequency
802.11b	1 to 11 Mbit/s	1999	2.4 GHz
802.11a	1.5 to 54 Mbit/s	1999	5 GHz
802.11g	3–54 Mbit/s	2003	2.4 GHz
Wi-Fi 4	72–600 Mbit/s	2009	2.4/5 GHz
Wi-Fi 5	433–6933 Mbit/s	2014	5 GHz
Wi-Fi 6	600–9608 Mbit/s	2019	2.4/5 GHz

Table 1.1: Theoretical Bandwidths of different WIFI standards

WIFI networks can function in different network modes. They are Infrastructure mode, Ad hoc mode and WIFI -direct mode.

Infrastructure mode

In infrastructure mode, which is the most common mode used, all communications go through a base station. For communications within the network, this introduces an extra use of the airwaves but has the advantage that any two stations that can communicate with the base station can also communicate through the base station, which enormously simplifies the protocols.

Ad-Hoc mode

According to Toh, C. (2002). Ad hoc wireless networks: Protocols and systems. [11] an ad hoc network is a network that is comprised of individual devices interacting with each other directly. The term(ad-hoc) implies spontaneous or impromptu construction because these networks often bypass the central access point such as a router.

Many ad hoc networks are local area networks where devices are enabled to send data directly to one another rather than going through a centralized access point. In more complex protocols nodes may forward packets, and nodes keep track of how to reach other nodes, even if they move around.

Ad-hoc mode was first invented and realized by Chai Keong Toh in his 1996 invention[12] of Wi-Fi ad-hoc routing, implemented on Lucent WaveLAN 802.11a wireless on IBM ThinkPads over a size nodes scenario spanning a region of over a mile. The success was recorded in Mobile Computing magazine (1999)[12] and later published formally in IEEE Transactions on Wireless Communications, 2002[12] and ACM SIGMETRICS Performance Evaluation Review, 2001.[12].

This wireless ad hoc network mode has proven popular with multiplayer handheld game consoles, such as the Nintendo DS, PlayStation Portable, digital cameras, and other consumer electronics devices.

WIFI Direct mode

According to Wi-Fi Alliance(2019, Aug) www.wi-fi.org/discover-wi-fi/wi-fi-direct [13], Wi-Fi Direct is a Wi-Fi communication standard that facilitates device connections without requiring a wireless access point (WAP). The devices are connected using Wi-Fi, thus achieving Wi-Fi level connection and transfer speeds from the connectivity.

Wi-Fi Direct is single radio hop communication, not multihop wireless communication, unlike wireless ad hoc networks and mobile ad hoc networks. Wi-Fi ad hoc mode, however, supports multi-hop radio communications, with intermediate Wi-Fi nodes as packet relays.

The basic function of the Wi-Fi Direct is to enable the connection between devices and facilitate data transfer through the use of built-in wireless modules without the aid of a dedicated access point.

Wi-Fi Direct has become a standard feature in smartphones, portable media players, and feature phones as well. The process of adding Wi-Fi to smaller devices has accelerated, and it is now possible to find printers, cameras, scanners and many other common devices with Wi-Fi in addition to other connections, like USB.

Multiple access points

An Extended Service Set may be formed by deploying multiple access points that are configured with the same SSID and security settings. Wi-Fi client devices typically connect to the access point that can provide the strongest signal within that service set. increasing the number of Wi-Fi access points for a network provides redundancy, better range, support for fast roaming and increased overall network-capacity by using more channels or by defining smaller cells.

1.3.4 A Peer-to-Peer (P2P) Network

According to (Schollmeier,2001)[14], peer-to-peer network is a network of interconnected nodes ("peers") serving resources amongst each other without the use of a centralized administrative system. Due to not having a centralized system, the network is resilient for node failures. Peers make a portion of their resources, such as processing power, disk storage or network bandwidth, directly available to other network participants, without the need for central coordination by servers or stable hosts. Peers are both suppliers and consumers of resources, in contrast to the traditional client-server model in which the consumption and supply of resources are divided.

Architecture

A peer-to-peer network is designed around the notion of equal peer nodes simultaneously functioning as both "clients" and "servers" to the other nodes on the network. This model of network arrangement differs from the client-server model where communication is usually to and from a central server.

Peer-to-peer networks generally implement some form of virtual overlay network on top of the physical network topology. The nodes in the overlay form a subset of the nodes in the physical network. Data is still exchanged directly over the underlying TCP/IP network, but at the application, layer peers can communicate with each other directly, via the logical overlay links (each of which corresponds to a path through the underlying physical network). Overlays are used for indexing and peer discovery and make the P2P system independent from the physical network topology.

Based on how the nodes are linked to each other within the overlay network, and how resources are indexed and located, we can classify networks as unstructured, structured or as a hybrid between the two.

- Unstructured networks
- Structured networks
- Hybrid networks

Unstructured Networks

Unstructured peer-to-peer networks do not impose a particular structure on the overlay network by design, but rather are formed by nodes that randomly form connections to each other.

Because there is no structure globally imposed upon them, unstructured networks are easy to build and allow for localized optimizations to different regions of the overlay. Also, because the role of all peers in the network is the same, unstructured networks are highly robust in the face of high rates of "churn. The primary

limitations of unstructured networks also arise from this lack of structure. In particular, when a peer wants to find the desired node in the network, the search query must be flooded through the network to find as many peers as possible that knows that particular node. Flooding causes a very high amount of signalling traffic in the network which can cause network failure.

Structured Networks

In structured peer-to-peer networks, the overlay is organized into a specific topology, and the protocol ensures that any node can efficiently search the network for a file/resource. The most common type of structured P2P network implements a

distributed hash table (DHT), in which a variant of consistent hashing is used to assign ownership of each file to a particular peer. This enables peers to search for resources on the network using a hash table: that is, (key, value) pairs are stored in the DHT, and any participating node can efficiently retrieve the value associated with a given key. However, to route traffic efficiently through the network, nodes in

a structured overlay must maintain lists of neighbours that satisfy specific criteria. This makes them less robust in volatile networks.

Hybrid Networks

Hybrid models are a combination of peer-to-peer and client-server models. A common hybrid model is to have a central server that helps peers find each other. Spotify was an example of a hybrid model[**?**]. There are a variety of hybrid models, all of which make trade-offs between the centralized functionality provided by a structured client-server network and the node equality afforded by the pure peer-to-peer unstructured networks. Currently, hybrid models have better performance than either pure unstructured networks or pure structured networks because certain functions, such as searching, do require a centralized functionality but benefit from the decentralized aggregation of nodes provided by unstructured networks.

The decentralized nature of P2P networks increases robustness because it removes the single point of failure that can be inherent in a client-server based system. As nodes connect the demand to the system increases, at the same time total capacity of the network also increases, and the likelihood of failure decreases. If one peer on the network fails to function properly, the whole network is not compromised or damaged.

1.3.5 Mobile Ad-Hoc Network(MANET)

MANET stands for Mobile ad-hoc Network also called a wireless ad-hoc network that usually has a routable networking environment on top of a Link Layer ad hoc network. They consist of a set of mobile nodes connected wirelessly in a self-configured, self-healing network without having a fixed infrastructure. Nodes of this network can move freely so they can make and break the connection on the fly.[15] Characteristics of MANET

- Dynamic Topologies: Network topology which is typically multihop, may change randomly and rapidly with time, it can form unidirectional or bi-directional links.
- Bandwidth constrained variable capacity links: As nodes have different bandwidth the capacity of the link depending on the capacity of the lowest node.
- Autonomous Behavior: Each node can act as a host and router, which shows its autonomous behaviour.

- Energy Constrained Operation: As some or all the nodes rely on batteries or other exhaustible means for their energy. Mobile nodes are characterized with less memory, power and lightweight features.
- Limited Security: Wireless network is more prone to security threats. A centralized firewall is absent due to its distributed nature of the operation for security, routing and host configuration.
- Less Human Intervention: They require minimum human intervention to configure the network, therefore they are dynamically autonomous.

1.3.6 Smartphone Ad-Hoc Networks(SPANs)

This is a mobile ad hoc network implementation that is done using commercially available smartphones to create peer-to-peer networks without relying on cellular carrier networks, wireless access points, or traditional network infrastructure. Apple's iPhone with version 8.4 ios and higher have been enabled with multi-peer ad hoc mesh networking capability[16]

1.3.7 Wi-Fi Peer-to-Peer (P2P) mode in Android

The Wi-Fi peer-to-peer (P2P) APIs allow applications to connect to nearby devices without needing to connect to a network or hotspot. To do this first both parties must agree to join a common network group. The connection process can be automated after that point. This function is available only for Android devices on or above API level 14.Google (2019, Aug) Android API documentation[17] This

API provides a means to discover and connect to other Android devices when each device supports Wi-Fi P2P. With this API WIFI interface, the IP of the group owner can be found with a little effort. Even though the new standard by

WIFI -alliance includes a WIFI ad-hoc mode Android open-source project hasn't implemented it. So to use the ad hoc functionality implementation of these services needs to be done. This is already done in B.A.T.M.A.N. project[18] by adding ad-hoc supported kernel to Android OS. Furthermore, this research is more focused on making a hybrid cellular ad hoc network rather than a pure ad hoc network using commercial Android devices. So implementing this kind of network is out of scope so. Furthermore, the testbed of this research consists of three Nexus 5X devices.

So WIFI direct implementation with GO configuration is more than sufficient to simulate ad hoc property of this research.

1.3.8 Virtual Private Network(VPN)

A virtual private network (VPN) is the extension of a private network that encompasses links across shared or public networks like the Internet. VPN enables us to send data between two computers across a shared or public Internet work in a manner that emulates the properties of a point-to-point private link. (Microsoft,2019)[19]

1.3.9 VPN Services of Android OS

In the early days, Android API did not have any option to perform the VPN configuration. This has to be done manually using the settings menu in the system interface. With the introduction of API Level 14, there was an option to configure the VPN using the programming API. Android VPN service is an application interface that gives application developers to create VPN services. Google (2019, Aug) Android API documentation https://developer.Android .com/[20]. As shown

in the diagram the connection to the VPN gateway is created by the protected socket in the Android operating system. Then it creates a local tunnel interface in the Linux kernel and dumps that traffic to it. Then the traffic is channelled to and from the Android device using virtual interfaces. Applications can access the network using those virtual interfaces.

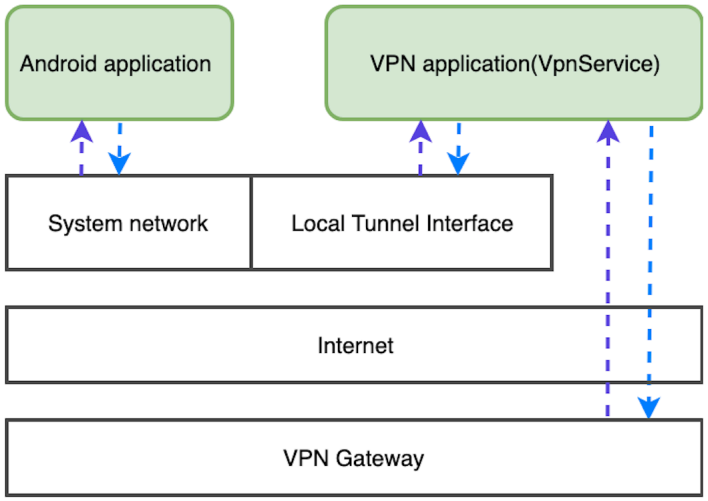

Figure 1.1: Inner workings off the VPN implementation of Android

As mentioned previously the multihop ad-hoc functionality of the devices has been disabled at the OS level by restricting the creation of the interfaces by the user. But with this new VPN API, we can bypass this limitation by tunnelling the wifp2p interfaces to the tunnel interfaces. This can be illustrated as below.

1.3.10 Linux kernel

The Linux kernel is the main component of a Linux operating system and is the core interface between a computer's hardware and its processes. It communicates between the two, managing resources as efficiently as possible. The kernel is so

named because like a seed inside a hard shell it exists within the OS and controls all the major functions of the hardware, whether it's a phone, laptop, server, or any other kind of computer. The kernel has 4 jobs:

- Memory management: Keep track of how much memory is used to store what, and where.
- Process management: Determine which processes can use the CPU, when, and for how long.
- Device drivers: Act as mediator/interpreter between the hardware and processes.

- System calls and security: Receive requests for service from the processes.

As Linux is a monolithic kernel it includes the networking implementation already built into it. OS access the functionality of the kernel using the system call interfaces.

1.3.11 Ad-Hoc mode on Linux kernel

Ad hoc mode has been implemented in the Linux kernel as a loadable module present in it called Kernel AODV. It implements AODV routing between computers equipped with WLAN devices.[21]

1.3.12 Linux kernel and Android OS

According to [22] Android uses Linux kernels that are downstream of Long Term Supported (LTS) kernels and include patches of interest to the Android community that has not been merged into LTS. This means Android Common Kernels contain additional features that are not been merged to the mainstream kernel. This can be new features tailored for Android needs (e.g. interactive cpufreq governor) to Vendor/OEM features that are useful for others (e.g. sdcardfs). Features like MPTCP, Ad-hoc routing support are not merged into Android common kernel due to not specified reasons. But as Android is open source. So anyone can add those features back and compile to get the kernel binaries.

1.3.13 OSI layer

The Open Systems Interconnection model (OSI model) is a conceptual model that characterizes and standardizes the communication functions of a telecommunication or computing system without regard to its underlying internal structure and technology. Its goal is the interoperability of diverse communication systems with standard communication protocols. The model consists of the following layers.

Layer 1: Physical Layer

The physical layer is responsible for the transmission and reception of unstructured raw data between a device and a physical transmission medium. It converts the

digital bits into electrical, radio, or optical signals.

Layer 2: Data Link Layer:

The data link layer provides a node-to-node data transfer links between two directly connected nodes. It detects and possibly corrects errors that may occur in the physical layer. IEEE 802 divides the data link layer into two sublayers: [23]

- Medium access control (MAC) layer – responsible for controlling how devices in a network gain access to a medium and permission to transmit data.
- Logical link control (LLC) layer – responsible for identifying and encapsulating network layer protocols, and controls error checking and frame synchronization.

Layer 3: Network Layer

The network layer provides the functional and procedural means of transferring variable length data sequences (called packets) from one node to another connected in different networks. Message delivery at the network layer is not necessarily guaranteed to be reliable; a network layer protocol may provide reliable message delivery, but it need not do so.

Layer 4: Transport Layer

The transport layer provides the functional and procedural means of transferring variable-length data sequences from a source to a destination host while maintaining the quality of service functions.

Layer 5: Session Layer

The session layer controls the dialogues (connections) between computers. It establishes, manages and terminates the connections between the local and remote applications.

Layer 6: Presentation Layer

The presentation layer establishes context between application-layer entities, in which the application-layer entities may use different syntax and semantics if the presentation service provides a mapping between them.

Layer 7: Application Layer

The application layer is the OSI layer closest to the end-user, which means both the OSI application layer and the user interact directly with the software application. This layer interacts with software applications that implement a communicating component.

1.4 Internet Protocol(IP)

The Internet Protocol (IP) is the principal communications protocol in the Internet protocol suite for relaying datagrams across network boundaries. Its routing function enables the Internet. The Internet Protocol is responsible for addressing host

interfaces, encapsulating data into datagrams and routing datagrams from a source host interface to a destination host interface across one or more IP networks. For these purposes, the Internet Protocol defines the format of packets and provides an addressing system. Each datagram has two components: a header and a payload.

The IP header includes the source IP address, destination IP address, and other metadata needed to route and deliver the datagram. The payload is the data that is transported. IP routing is performed by all hosts, as well as routers, whose main

function is to transport packets across network boundaries. Routers communicate with one another via specially designed routing protocols, either interior gateway protocols or exterior gateway protocols, as needed for the topology of the network.

Router

When multiple routers are used in interconnected networks, the routers can exchange information about destination addresses using a routing protocol. Each router builds up a routing table listing the preferred routes between any two computer systems on the interconnected networks. It consists of two planes of operations called the control plane and forwarding plane. Control pane is responsible for maintaining routing tables while the forwarding plane does the packet forwarding under the routing table.

Routing tables

It is a data table stored in a router or a network host that lists the routes to particular network destinations, and in some cases, metrics (distances) associated with those routes. The routing table contains information about the topology of the network immediately around it. Linux kernel controls the routing path of IP packets using this routing table.

1.4.1 What is SSH?

Secure Shell (SSH) is a cryptographic network protocol for operating network services securely over an unsecured network. Typical applications include remote command-line, login, and remote command execution, but any network service can be secured with SSH. SSH provides a secure channel over an unsecured network in a client-server architecture, connecting an SSH client application with an SSH server.[24]

1.4.2 Reverse SSH

Reverse SSH Port Forwarding specifies that the given port on the remote server host is to be forwarded to the given host and port on the local side. To do this we need to connect to the remote server using ssh connection in reverse mode. This will create an encrypted tunnel from a remote device to the local device. With port forwarding, we can access the local machine to form the remote one. This is very useful if we have dynamic IP s behind firewalls or nat devices.

1.4.3 Multipath TCP

Multipath TCP (MPTCP) is an ongoing effort of the Internet Engineering Task Force's (IETF) Multipath TCP working group, that aims at allowing a Transmission Control Protocol (TCP) connection to use multiple paths(interfaces) to maximize resource usage and increase redundancy[25] Multipath TCP is particularly useful in the context of wireless networks using both Wi-Fi and a mobile network. In addition to the gains in throughput from inverse multiplexing, links may be added or dropped as the user moves in or out of coverage without disrupting the end-to-end TCP connection.

1.4.4 Web 2.0

Web 2.0 (also known as Participative (or Participatory) and Social Web) refers to websites that emphasize user-generated content, ease of use, participatory culture and interoperability (i.e., compatible with other products, systems, and devices) for end-users. A Web 2.0 website allows users to interact and collaborate through social media dialogue as creators of user-generated content in a virtual community. This contrasts with the first generation of Web 1.0-era websites where people were limited to passively viewing the content.

1.5 Research Questions

1. How to deploy an ad-hoc cellular hybrid network using off the shelf Android devices by taking WIFI network as data plane and cellular network as its control plane and backup network.
2. What newly emerged technologies can be used to create such a network?
3. What are the practical implications are there in creating such a network.
4. Is there any apparent gains in creating such a network.

1.6 Aim of the Reserch

This research revisits the implementation details of ad hoc networks using the off the shelf Android devices to determine the feasibility of such a network in the present time.

This is mainly to determine the feasibility of implementing and utilizing a wireless ad-hoc cellular hybrid network configuration to communicate between Android smartphones to improve the wireless connectivity of these devices.

To do this we need to first implement ad-hoc network using the off the shelf Android devices. Then we need to develop the hybrid network by using WIFI ad-hoc network as the data plain and cellular network as its control plane and backup connection.

After implementing a prototype of this hybrid system it needs to be compared with the previous ones to see if there is a visual gain in the performance.

1.7 Methodology

As from the aim of this research, we have concluded that the question addressed in this research are

1. How to deploy an ad-hoc cellular hybrid network or off the shelf Android devices by using WIFI network as data plane and cellular network as its control plane and backup network.
2. What kind of newly emerged technologies can be used to create such a network?
3. What are the practical implications are there in creating such a network.
4. Is there any apparent gains in creating such a network.

This section describes how the objectives mentioned earlier are achieved.

Objective1: Review existing ad-hoc network implementations

A thorough literature review will be done on Android ad-hoc mesh networking and ad-hock-routing fields to get a good understanding of the ad-hoc networking and to get the necessary knowledge on how to do the research.

Objective2: Develop a methodology to do local routing using the WIFI interface of the Android devices.

Currently, there are 3 prominent techniques on how to do this

- Create a custom Android OS with ad-hoc routing functionality built into it. (Thomas, et al.2012)[2]
- Root the phone and use an application to modify routing tables to route packets from one node to another. (Funai, et al.2017)[26]
- Use WIFI -direct in infrastructure mode to make connections. (Liu, et al.2016)[5] and (Oide, et al.2017)[27]
- Use the newly introduced VPN of Android alongside with the WIFI -direct to create a network.

The method that uses the Android VPN interface does not require any additional OS modifications or the root permissions. If we use the GSM interface to run the control plane of the network it will give us a more stable system than using WIFI interface alone.

Objective 3: Implementation of the hybrid ad-hoc network.

After achieving Objective 1 & 2 methods developed in objective 2 will be implemented as the hybrid ad-hoc network. This can be done by modifying the relevant routing tables to allow the local routing to flow through the WIFI interfaces when the cellular connection is in the active state.

Objective 4: Benchmarking this implementation with the currently available alternatives solutions.

After Completion, it should be tested with the currently available networks on matrixes like bandwidth, latency, network configuration time ping time to find if there is a noticeable gain in performance.

1.8 Delimitations and justifications

This is a proof of concept showing that it is feasible to create a hybrid network using cellular and WIFI interfaces side by side. So it suffices to implement this using the Android devices. So there will be no implementation of the IOS version due to that.

There will be a connection drop when we are transition from one interface to another because of IP layer routing. Even though we can solve this using the Multi-path TCP connections it is out of scope with the research question.

Due to the availability of large no of parameter data in current smartphones, we can do much better optimization in the routing algorithms. But the research question is not on route optimization so we are not touching that.

Due to the scaling of the devices, there can be congestion in the ad hoc networks and to address them we need to implement some kind of congestion control to this as well. But due to this being a production level problem we are not answering that part as well.

To simulate the Ad-hoc functionality of the network we need at least seven devices testbed. Due to the lack of resources for this research were are using only three Android devices. So the researcher cannot properly test the ad-hoc property of the network.

Even though the researcher found a way to create ad-hoc functionality without rooting the device it is not feasible due to the current timeframe of the research.

It is not possible to test the current results of this hybrid network with the old implementation of the ad-hoc system due to the non-reproducibility of the ad-hoc networks.

1.9 Summary

This chapter laid the foundations for the dissertation. It introduced the research problem and research questions and hypotheses. Then the research was justified, definitions were presented, the methodology was briefly described and justified, the dissertation was outlined, and the limitations were given. On these foundations, the dissertation can proceed with a detailed description of the research.

Chapter 2

Literature Review

To better understand the current progress in mobile ad-hoc network technology, let's look into the current findings and ongoing research in the fields of networking. There are some interesting researches done in these fields which can be used to find an answer to the above research problem.

2.0.1 COMONet: Community Mobile Network

One of the focuses of research being conducted in the field of ad-hoc networks is in making networking more flexible, both on the physical network side and the application side, as with most applications nowadays being developed with infrastructure networks and always-on connectivity in mind.

When smartphones were introduced, there were no open software devices like Android available to the public. Smartphone industry was dominated by closed echo system devices. back then Symbian Ltd(2019, Aug)What is Symbian OS https://www.symbianos.org/ [28] was the go-to OS choice for mobile devices. originated from EPOC32, an operating system created by Psion in the 1990s. In June 1998, Psion Software became Symbian Ltd. It was a major joint venture between Psion and phone manufacturers Ericsson, Motorola, and Nokia. So due to the close nature in software and hardware of those devices, modifying them to create an ad-hoc network was not an easy task back then.

Project COMONet(Wijesekera and Keppitiyagama 2007)[1] was undertaken in this kind of environment back then. COMONet was a Community based Mobile Network which utilizes Wi-Fi and Bluetooth interfaces to build an Ad-Hoc network among mobile phone users to bypass GSM base stations whenever possible. It provided the functionality to make voice calls without the help of a carrier network.

To use this functionality caller and the callee does not have to be within the Wi-Fi or Bluetooth range of each other to make a call since the COMONet(Wijesekera and Keppitiyagama 2007)[1] is capable of routing calls through the other mobile nodes that are participating in the network. It can switch in between mesh connection and the cellular connection on the fly without breaking the connection.

Let us dig deeper into the implementation of the COMONet. What the researcher has done hear was to implement a mobile ad-hoc mesh network using the readily available network interfaces of the mobile devices. COMONet uses different networking stacks to implement the vice call functionality as shown in Figure **??** below.

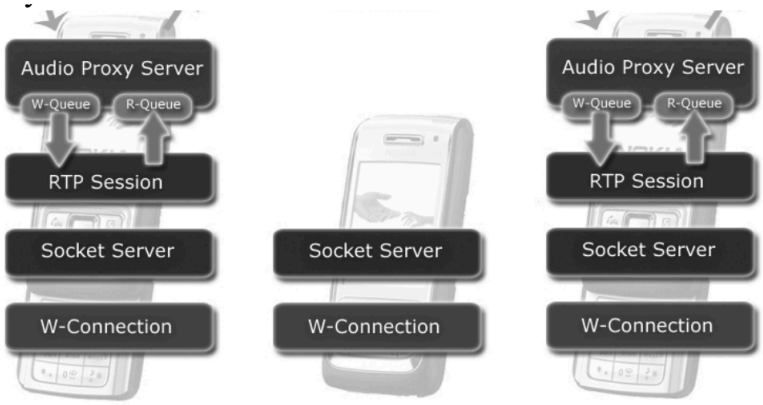

Figure 2.1: Wireless View of COMONet(Source COMONet[1])

Most of the functionalities in the above stack are not readily available to the application developer. So to get access to the networking interfaces to ring 0 privileges is a must. What the researcher had done is to use a hack to get into kernel mode and passing that data out to the application layer to use with his device. Believe me, this is no easy task to do. One way to do this is to read the machine code of the OS and modifying the relevant bits so that it gives the elevated (ring 0) privileges to the Comment app. To my knowledge, this is not a good design because if the COMONet application crashes, then the whole os crashes. But at that time this was not feasible to in the proper manner due to the closed nature of the Symbian ecosystem.

2.0.2 The SPAN project: SmartPhone Ad-Hoc Networks

The SPAN project(Thomas, et al.2012)[2] utilizes MANET (Mobile Ad-Hoc Network) technology to provide a resilient backup framework for communication between two devices when all other infrastructure is unavailable or unreliable. The MANET based solution is a headless, infrastructure-less network that allows common smartphones to link together in a dynamic way to create a mobile ad-hoc network.

They have injected the framework into the existing Android network stack between OSI layers 2 and 3 as shown in Figure 2.2. Privileges needed to create the ad-hoc network is gained by accessing the wireless chip drivers directly to set the parameters of the wireless interface. They have configured the wireless chip to work in ad-hoc mode by using the iwconfig Linux command-line utility.The Debian Project(2019,Aug) Official WIKI page https://wiki.debian.org/iwconfig [29]

They have customized the device network drivers in this implementation. By doing so they have limited the generalizability of the technology in different devices. Plus side of implementing the network in this manner is that it can change the routing protocol in a dynamic passion. To my knowledge, most of the ad-hoc implementations cannot achieve this feature.

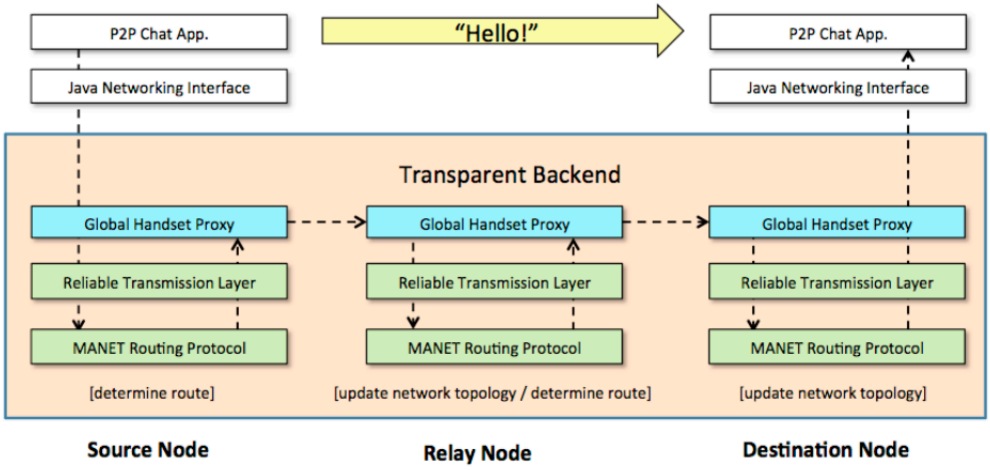

Figure 2.2: Cross-sectional view of the network(Source SPAN Project[2])

2.0.3 Better approach to mobile ad-hoc networking

BATMAN project: A better approach to mobile ad-hoc networking [18] by Open Mesh organization is a routing protocol for multi-hop ad-hoc mesh networks. People at Open-Mesh have implemented this framework to support local routing on top of the ad-hoc enabled kernel. Open Mesh (2019,Aug) B.A.T.M.A.N. project https://www.open-mesh.org/projects/batman-adv/

This system works on top of the kernel to support ad-hoc routing by modifying the messages passed inside the network environment. Initially, they flood the network with announcement messages. So every node knows who their neighbours are. Due to this if there was a missing node we only need to change the routing data of the local nodes. Besides routing, the framework provides other functionalities such as benchmarking the network and node visualization.

As they have to implement the platforms on top of an ad-hoc enabled kernel this cannot use with a normal stock Android device. To use this framework we need a custom Android OS like Lineage OS. Lineage OS Team(2019,March)[30]. They haven't automated the process of creating the network yet. So this is not a user-friendly solution as well. The efficiency of their protocol has been discussed in their documentation. To my knowledge, this is pretty efficient at routing data. We cannot use this in our research due to the limited number of the devices that they support.

2.0.4 Mobile Ad-hoc Networks over Wi-Fi Direct

In this paper, researchers try to introduce a novel method to achieve multi-hop communication among open-source, non-rooted Android devices using Wi-Fi Direct Technology. This solution tries to preserve the ad-hoc property of the network while utilizing minimum privileges. (Liu, et al.2016)[5]

Let us analyze this method. They have chosen the WIFI interface for their communication. Bluetooth interface has been discarded due to its low bandwidth rates. The protocol he had chosen here is the WIFI-Direct. It allows them to create groups inside the network. Initially, all the nodes become group owners. When it is time to transfer the message some becomes clients. After the message has been transferred node revert back to its initial state.

This method preserves the ad-hoc nature of the network. But because in each

transfer of data it needs to make and break the structure of the network transfer rates becomes low. Due to this, it is pretty hard to keep a continuous stream of data using this network.

2.0.5 Enabling multi-hop ad-hoc through WIFI direct multi-group networking

Enabling multi-hop ad hoc networks through WIFI direct multi-group networking (Heinzelman et al.)[26]

In this paper, Heinzelman et al.[26] and have proposed and analyzed different practical solutions for supporting the communications between multiple WIFI Direct groups using Android OS devices. They have analyzed the WIFI Direct standard and the limitations of the current implementation of the Android WIFI Direct framework and presented some possible solutions to interconnect different groups to create multi-hop ad hoc networks.

Let us consider their proposed method to create multi-hop ad hoc networks. In this method near devices are connected using WIFI-Direct architecture (Group owner and Client). Intergroup communication is achieved using this WIFI-Direct protocol. They have used the hotspot functionality of the Android to achieve intergroup communication. To do this they need to allocate a node to provide hotspot functionality to other groups. To enable inter-group communication they have modified the routing table of relevant devices. To do this the root access to the device is needed.

In this implementation, they have done no modifications to the operating system. They have modified the routing tables of certain devices to achieve intergroup communication. This needs root access for modification of the routing table. This limits the usability of this system in personal devices as rooting voids the warranty of most of the Android devices.

2.0.6 Infrastructure-less Communication Platform for Android

In this research, they are trying to implement an ad-hoc network without modifying the OS or rooting the Android device. They have introduced new topology for the network and relevant routing protocols to go along with that. They have used

WIFI-Direct and WIFI interfaces to implement the network.

The researchers(Takuo et al.)[27] have made some interesting trade-off between architecture, performance and the required privilege levels when implementing this network. They have used a tree architecture Figure 2.3 rather than a graph architecture when designing the network. They proposed a relay node to do the inter-cluster communication. This relay node uses the WIFI connection to transfer the data from one device to another. Other message transfers use the WIFI -direct protocol.

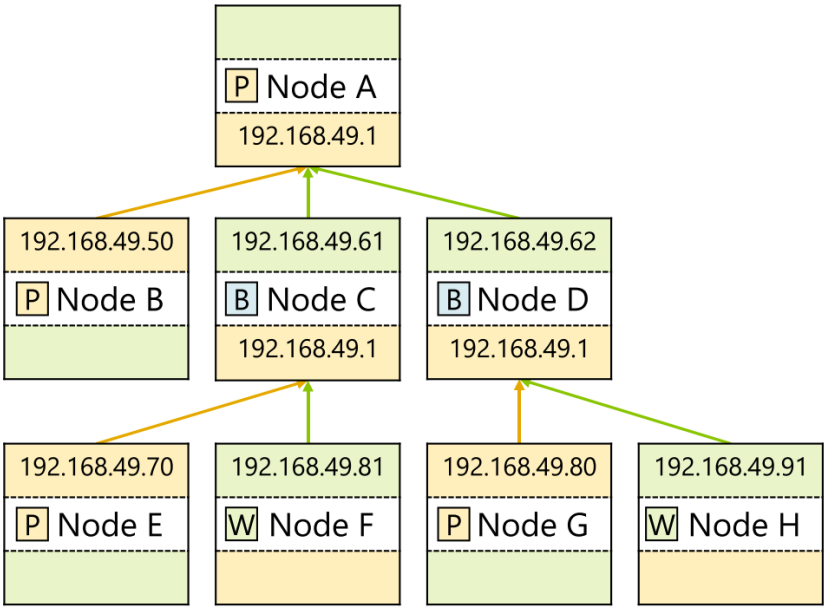

Figure 2.3: Example of topology construction

Because of the topology, they proposed the network will have a low latency connection and high bandwidth. But due to the non-distributed nature of the relay node, it takes some time to recover from a node failure. Implementing this network does not need any OS-level modifications or root privileges. Allocation of a relay node is not a good approach because that device cannot be a user in this network.

2.0.7 Integrated Cellular and Ad Hoc Relaying Systems

In this paper, they have proposed the idea of using ad-hoc and cellular connection side by side to improve the quality of the connection. In this architecture, the two

networks do not talk to each other. But in my research cellular connection is used to make the ad-hoc network. And in the above research, they are concerned only on the theoretical aspect of the network. (Tonguz at el. 2001)[31]

2.0.8 WiFi Direct and LTE D2D in Action

In this paper, they have proposed the method of using the WIFI direct and cellular connection side by side to improve the bandwidth of the connection. But here they are more interested in the implementation aspects of the network rather than the theoretical aspects. In this paper, they have analyzed the nature of the connection using different criteria as well. (Arash and Vincenzo 2013)[32]

2.0.9 A Unified Cellular and Ad-Hoc Network Architecture

In this research (Erran at el.2003)[33] have introduced an implementation of a hybrid ad-hoc cellular network. It answers the question can these two networks be synergistically combined to leverage the advantages of each other.

2.0.10 Routing Protocols in an Ad-hoc network

In the paper by (Casetti, et al.2015)[34], they have proposed a content-centric routing algorithm that can be used in a mobile ad-hoc network. In this method, the content delivery leverages the forwarding scheme through a content-centric approach.

In the paper by (Philippe, Muhlethaler, Clausen and Laouiti)[35], they have analyzed the use of OLSR(Optimized Link State Routing) protocol in an ad-hoc routing environment. This is a proactive routing protocol for mobile ad hoc networks.OLSR protocol is an optimization of a pure link-state protocol for mobile ad hoc networks. This technique significantly reduces the number of retransmissions in a flooding or broadcast procedure.

The paper A Study of Ad-Hoc Networks by (Alshaer and El-Rabaie)[36] does some higher-level analysis of the routing protocol that can be used with a mobile ad-hoc network. In this paper, researchers analyze how different aspects of the network affect these protocols.

2.0.11 Secure Key Establishment for Device Communications

In this paper, researchers(yin at el.)[37] investigate the security requirements and challenges for a device to device communications and present a secure and efficient key agreement protocol, which enables two mobile devices to establish a shared secret key for D2D communications without prior knowledge

2.1 Summary

Several efforts are in development to address many of the requirements to achieve decentralised mobile ad-hoc networking. Some researchers are based on implementation aspects of the network and others are focused on the theoretical aspects of the network. Initially, we discussed different ways we can make an ad-hoc network. This can from complete OS modification, routing table modification to application-level modification with no additional privileges(no root). Most recent implementations rely on WIFI direct protocol to implement ad-hoc networking. When designing the architecture we need to concern the modification that needs to be done on an Android system because of this affect the reach of our research.

In the later part, we discuss different kinds of cellular ad-hoc hybrid network and their performance in a theoretical and practical manner. The current implementations of ad-hoc network targets on Android devices relying on new and/or proprietary standards, ignoring the existing standards used across many existing implementations. This limits their use unless they gain mass-market adoption

Chapter 3

Research Design

Let us consider the research question first. It consists of two parts. The first part on how to deploy an ad-hoc network using the infrastructure network as a control plane. The second part of the research question is on finding what are the performance implications to the nodes of such a network.

This research was more focused on the practical aspect of deploying a cellular ad-hoc hybrid network. So we chose the constructive research approach to do this research as shown in Figure 3.1. So to find the feasibility of deploying this kind of network first need to analyze our problem thoroughly. The initial problem we face is to deploy an ad-hoc mobile network using readily available resources in mobile devices. To do this we need to do some literature survey on this field.

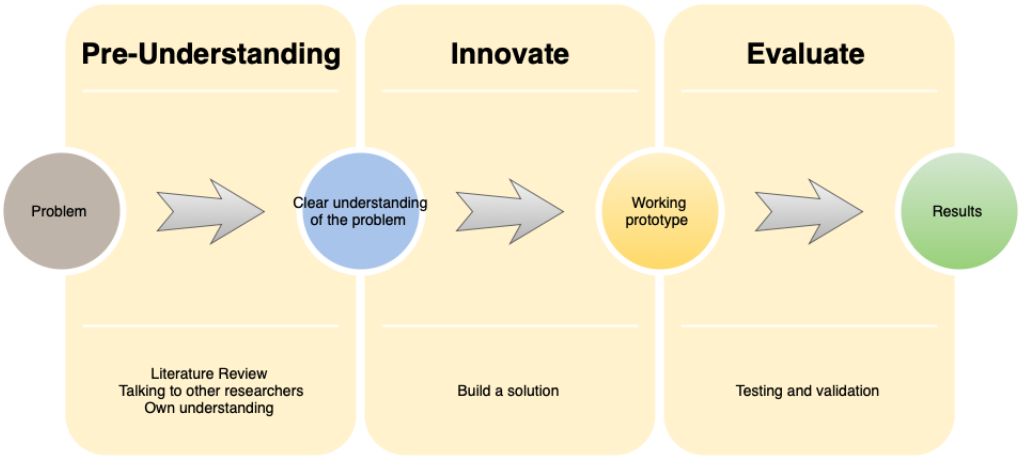

Figure 3.1: Flow of the Constructive Research Design

By doing so we have gained a good understanding regarding the mobile ad-hoc networking. It helped me to develop a well-rounded picture of how to implement an ad-hoc network. we came up with a new solution for Android Ad hoc networking using the readily available WIFI Direct protocol and VPN technology.

The next imminent problem we faced was on how to integrate the ad-hoc data plane with the cellular control plane. we were not able to find any useful resources regarding this question by doing literature reviews. So we talked with the professionals in the field to get their point of view on this problem.

By discussing with these people it occurred to me that to solve this solution we need a good networking toolkit and a thorough understanding of the architecture of the Android OS and Kernel.

So to create a good toolkit it was obvious to me that we need to port the Linux pkg manager to the Android. This was done using the Termux repository. Superuser privileges were needed to run those tools inside the Android kernel. So for the process of rooting the Android Nexus 5x device, I found the necessary instructions in the XDA developer's form. With help from those posts, we created the rooting script for my Nexus 5x devices.

By analyzing the networking implementations in the Android devices with the help of the previously mentioned tools and the Android kernel source code we were able to come up with the following architecture for this hybrid network as shown in the Figure 3.2.

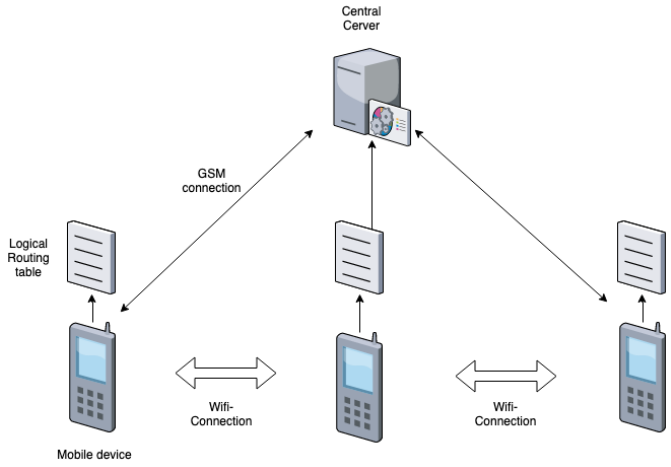

Figure 3.2: Network Architecture diagram

The first problem we faced when implementing this network architecture is that the server needs a secure way to remotely configure its end devices(Android phones). The most simple and safest way to remotely log into a Linux device is to use an SSH client. But there is a caveat, to log in to the end device first we need to know its IP address. In our case this is dynamic. So to overcome this we used reverse ssh connections. As we know the IP address of the central server every device makes a reverse ssh tunnel to that server. Then the server can use that reverse tunnel to connect to the relevant end device.

A tunnel interface is created inside the ad-hoc network to make the routing between networks to become feasible. For the network switching part, we have used routing table priories rather than a demon. This is because it is a much more simple and effective solution to implement.

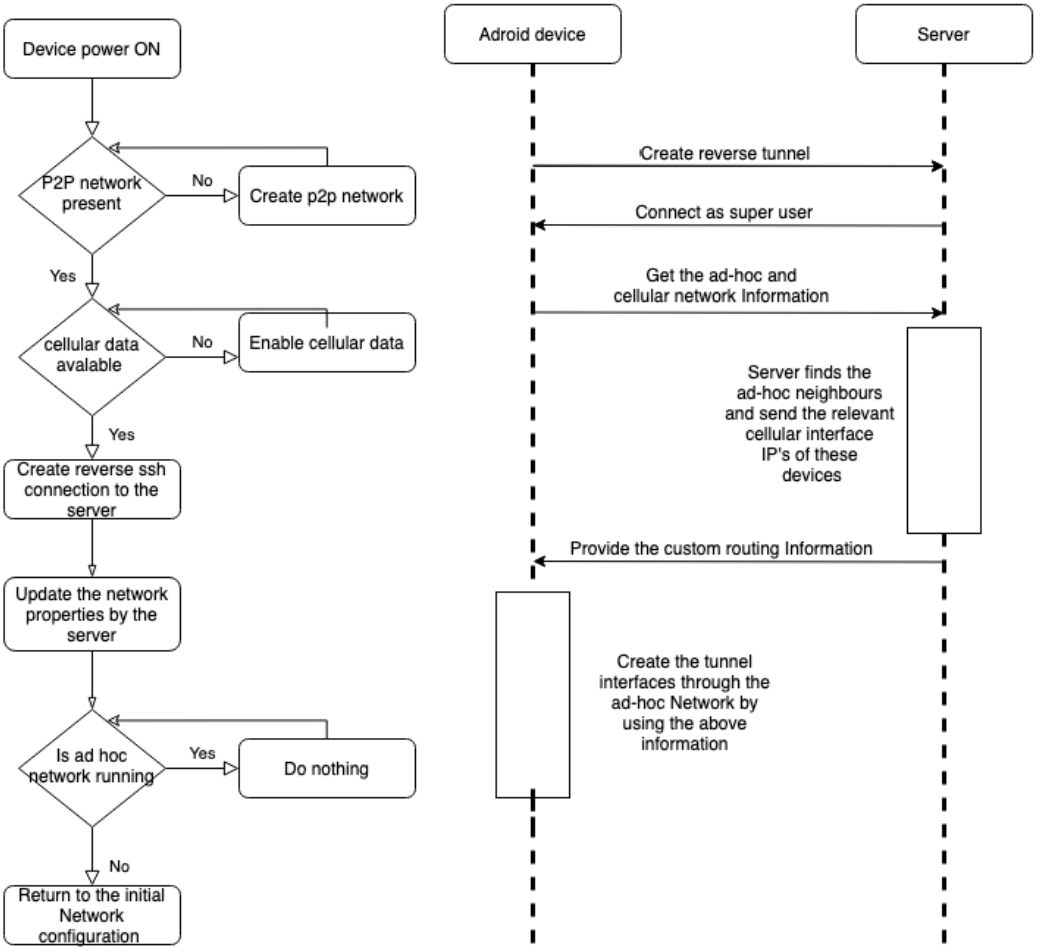

Figure 3.3: Flow diagram and activity diagram for the control plane

We need to analyze the ad-hoc network using relevant benchmarks in this field. To identify relevant benchmark metrics we have used the above literature review. we have identified Throughput, Topology construction time, Latency as valid matrices for this evaluation process. Finally, by using these data we can predict the performance of different applications on this network. To evaluate the control plane switching, we have used the switching time as an evaluation matrix.

Chapter 4

Research Implementation

Implementation of this research can be divided mainly into three parts which are Ad-hoc network implementation, Cellular plane implementation, and Control plane implementation. Let us consider those parts in detail.

4.0.1 Ad-hoc Network Implementation

Let us consider the ad-hoc network implementation. When implementing the ad-hoc network first we need to consider the Architecture of the Android OS. Android OS consists of the following components.

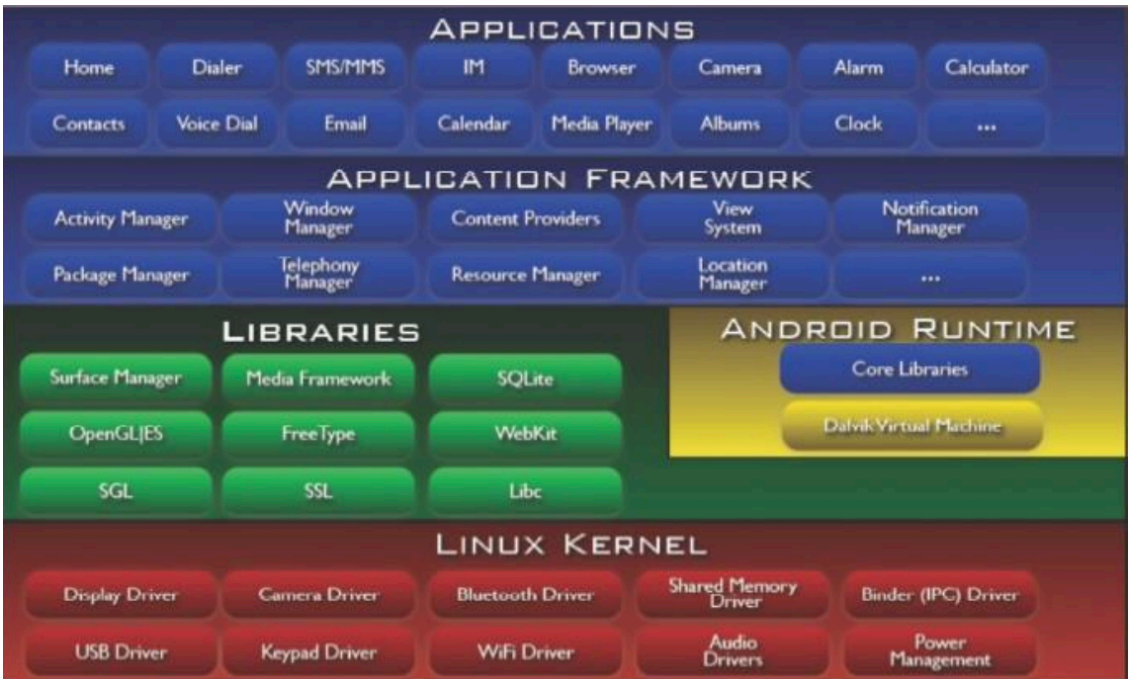

Figure 4.1: Android system architecture. Green items are written in C/C++, blue items are written in Java and run in the Dalvik VM.

When looking at the above architecture in Figure 4.1 we can see that there are three main choices to implement an ad -hoc network using an Android device. They are as follows.

1. Implement the network on the Android OS layer
2. Implement the network inside the kernel module.
3. Implement the network on the application level.

Implementing the network in the Android OS layer

First, we tried to implement the ad-hoc network using the Android OS level by modifying the Java code. For this, we need a way to implement the ad-hoc networking stack in the OS layer components.

These components include network discovery module, IP assignment module. routing module and the control plane to orchestrate all of this. After trying to implement some of the modules it was evident to us that the time was not sufficient to finish the implementation. This is because Android OS takes a huge time to compile. It takes 16 thread 60Gb ram server about one and a half hours to compile the Android Open Source project. So with the available hardware and timeframe of one year, it was not feasible to go through this method.

```
append2simg out/target/product/bullhead/obj/PACKAGING/systemimage_intermediates/system
target Unpacked: skia_nanobench (out/target/product/bullhead/obj/EXECUTABLES/skia_nano
target Symbolic: skia_nanobench (out/target/product/bullhead/symbols/data/nativetest64
target Strip: skia_nanobench (out/target/product/bullhead/obj/EXECUTABLES/skia_nanoben
append2simg out/target/product/bullhead/obj/PACKAGING/systemimage_intermediates/system
Install system fs image: out/target/product/bullhead/system.img
out/target/product/bullhead/system.img+out/target/product/bullhead/obj/PACKAGING/recov
target Unpacked: skia_dm (out/target/product/bullhead/obj/EXECUTABLES/skia_dm_intermed
target Symbolic: skia_dm (out/target/product/bullhead/symbols/data/nativetest64/skia_d
target Strip: skia_dm (out/target/product/bullhead/obj/EXECUTABLES/skia_dm_intermediat
target Unpacked: libdeqp (out/target/product/bullhead/obj/SHARED_LIBRARIES/libdeqp_int
target Symbolic: libdeqp (out/target/product/bullhead/symbols/system/lib64/libdeqp.so)
target Strip: libdeqp (out/target/product/bullhead/obj/lib/libdeqp.so)

#### make completed successfully (01:23:30 (hh:mm:ss)) ####
```

Figure 4.2: Image showing the time it takes to compile AOSP with the above-mentioned specs.

Implementing the network inside the kernel module.

The second option to enable ad-hoc networking in Android is to implement the whole system in the kernel layer. Due to the android kernel being derived from the Linux kernel it is easy to enable the batman routing protocol. To do this we need to recompile the kernel with the relevant modules enabled(B.A.T.M.A.N. module) and flash it back to the mobile device.

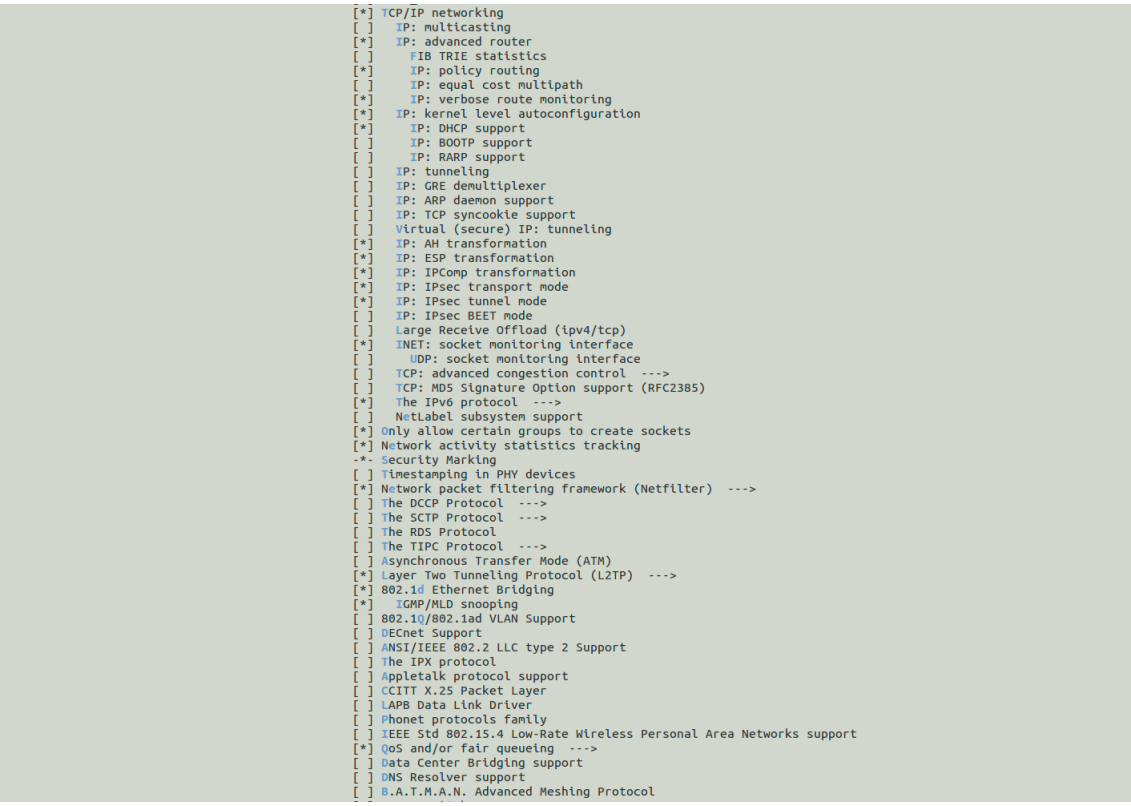

Figure 4.3: Android Kernel Default configuration

As you can see in the above Figure 4.3 default Android compilation setting does not include the most basic network drivers which are readily available to the general Linux kernel. So we need to enable those modules and recompile the Kernel to enable Ad-hoc features. Doing this is kind of risky because if you flash a faulty kernel to the device it will brick the device. This is because the kernel for the android device is contained in its boot image. So if the boot process does not happen correctly there is no way to recover from that. Recovery mode and boot-loader mode needs the boot to function properly. As we did brick a test device it is not a safe method to implement the Ad-hoc network.

Implement the network in the Application Layer

After considering all the options we choose the application layer to implement the ad-hoc network because it is the best way to implement the network. Wi-Fi Direct (P2P) allows Android 4.0 (API level 14) or later devices with the appropriate hardware to connect directly to each other via Wi-Fi without any intermediate access point[17]. Android implements the Wi-Fi Direct in accordance with the IEEE standard. So essentially this technology allows the Android developer to connect other devices using the Android built-in API.

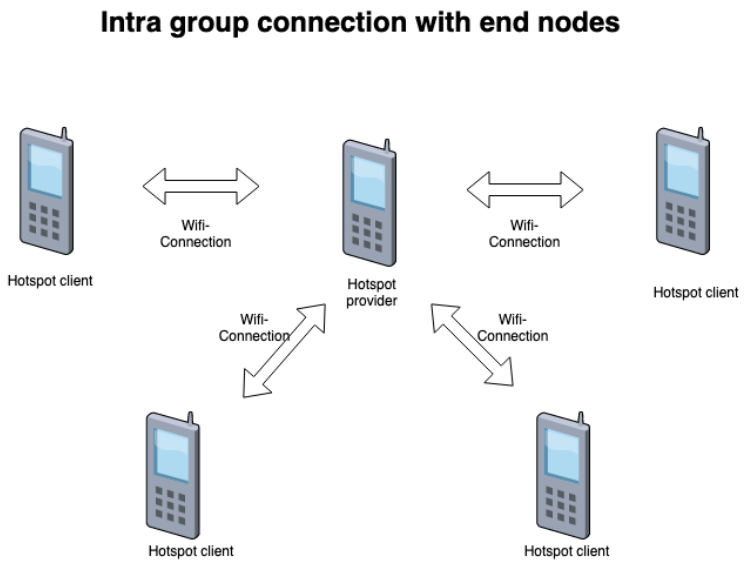

Figure 4.4: Wifi-direct group formation

We created an application to connect the devices in the Wi-Fi Direct mode using this feature. The application works in the following manner. Initial work() function starts the processes need for the application. First, it binds the GUI components like buttons tex boxes to the relevant variables.

```
private void initialWork() {
    btnOnOff=(Button) findViewById(R.id.onOff);
    btnDiscover=(Button) findViewById(R.id.discover);
    btnSend=(Button) findViewById(R.id.sendButton);
    listView=(ListView) findViewById(R.id.peerListView);
    read_msg_box=(TextView) findViewById(R.id.readMsg);
    connectionStatus=(TextView) findViewById(R.id.connectionStatus);
    writeMsg=(EditText) findViewById(R.id.writeMsg);}
```

To access the system WIFI service we need to bind the WIFI service to a wifiManager object. Then we need a channel to initialize the receiver object. This part happens in the following code segment.

```
wifiManager= (WifiManager)
    getApplicationContext().getSystemService(Context.WIFI_SERVICE);

mManager= (WifiP2pManager) getSystemService(Context.WIFI_P2P_SERVICE);
mChannel=mManager.initialize(this,getMainLooper(),null);
mReceiver=new WiFiDirectBroadcastReceiver(mManager, mChannel,this);
```

We need to create an intent filter object with the relevant permissions in order to get the privileges needed from the user..

```
    mIntentFilter=new IntentFilter();
    mIntentFilter.addAction(WifiP2pManager.WIFI_P2P_STATE_CHANGED_ACTION);
    mIntentFilter.addAction(WifiP2pManager.WIFI_P2P_PEERS_CHANGED_ACTION);
    mIntentFilter.addAction(WifiP2pManager.WIFI_P2P_CONNECTION_CHANGED_ACTION);
    mIntentFilter.addAction(WifiP2pManager.WIFI_P2P_THIS_DEVICE_CHANGED_ACTION);
```

The Lambda function below is responsible for creating available WIFI Direct devices to be connected. It first clears the list at the beginning and then adds the devices by calling peerList.getDeviceList() method. The for-loop creates a device name list which is used to create the UI device list. If there is no device to connect it gives a toast stating no device found.

```
    WifiP2pManager.PeerListListener peerListListener=new
        WifiP2pManager.PeerListListener() {
        @Override
        public void onPeersAvailable(WifiP2pDeviceList peerList) {
            if(!peerList.getDeviceList().equals(peers))
            {
                peers.clear();
                peers.addAll(peerList.getDeviceList());

                deviceNameArray=new
                    String[peerList.getDeviceList().size()];
                deviceArray=new
                    WifiP2pDevice[peerList.getDeviceList().size()];
                int index=0;
```

```
            for(WifiP2pDevice device : peerList.getDeviceList())
            {
                deviceNameArray[index]=device.deviceName;
                deviceArray[index]=device;
                index++;
            }

            ArrayAdapter<String> adapter=new
                ArrayAdapter<String>(getApplicationContext(),
            android.R.layout.simple_list_item_1,deviceNameArray);
            listView.setAdapter(adapter);
        }
        if(peers.size()==0)
        {
            Toast.makeText(getApplicationContext(),"No Device
                Found",Toast.LENGTH_SHORT).show();
            return;
        }
    }
};
```

The below anonymous inner class differentiates the group owner from the client and provides the group security key for connecting the devices that support the WIFI protocol.

```
WifiP2pManager.ConnectionInfoListener connectionInfoListener=new
    WifiP2pManager.ConnectionInfoListener() {
     @Override
     public void onConnectionInfoAvailable(WifiP2pInfo wifiP2pInfo) {
       final InetAddress groupOwnerAddress=wifiP2pInfo.groupOwnerAddress;
       final boolean isClient = !wifiP2pInfo.isGroupOwner;

       if(wifiP2pInfo.groupFormed && wifiP2pInfo.isGroupOwner){
           connectionStatus.setText(password+groupOwnerAddress.toString());
           serverClass=new ServerClass();
           serverClass.start();
       }else if(wifiP2pInfo.groupFormed){
           connectionStatus.setText("Client");
           clientClass=new ClientClass(groupOwnerAddress);
           clientClass.start();
        }}};
```

Public method onReceive implement the logic that needed in connecting the two devices and provide the relevant toast message inform the state.

```
public void onReceive(Context context, Intent intent) {
String action = intent.getAction();

if(WifiP2pManager.WIFI_P2P_STATE_CHANGED_ACTION.equals(action)){
    int state=intent.getIntExtra(WifiP2pManager.EXTRA_WIFI_STATE,-1);

    if(state==WifiP2pManager.WIFI_P2P_STATE_ENABLED){
        Toast.makeText(context,"Wifi is ON",Toast.LENGTH_SHORT).show();
    }else {
        Toast.makeText(context,"Wifi is OFF",Toast.LENGTH_SHORT).show();
    }
}else if(WifiP2pManager.WIFI_P2P_PEERS_CHANGED_ACTION.equals(action)){
    //do something
    if(mManager!=null)
    {
        mManager.requestPeers(mChannel,mActivity.peerListListener);
    }
}else
    if(WifiP2pManager.WIFI_P2P_CONNECTION_CHANGED_ACTION.equals(action)){
    //do something
    if(mManager==null)
    {
        return;
    }

    NetworkInfo
        networkInfo=intent.getParcelableExtra(WifiP2pManager.EXTRA_NETWORK_INFO);

    if(networkInfo.isConnected())
    {
        mManager.requestConnectionInfo(mChannel,mActivity.connectionInfoListener);
    }else {
        mActivity.connectionStatus.setText("Device Disconnected");
    }
}else
    if(WifiP2pManager.WIFI_P2P_THIS_DEVICE_CHANGED_ACTION.equals(action)){
   }
}
```

The GUI of the WIFI Direct application is shown in the Figure 4.5 below. Left device acts as the group owner while the other acts the client. we have implemented a simple message passing app in order to test the connectivity between the devices.

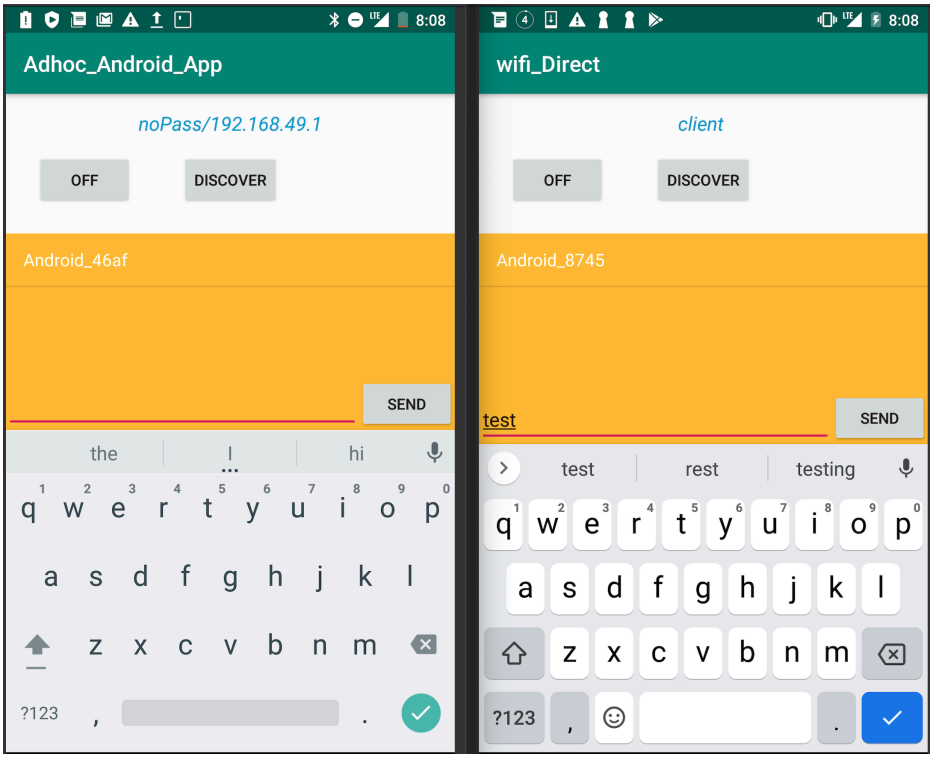

Figure 4.5: GUI of the final version of WIFI Direct client

When using the Android Wifi-direct implementation devices are provided with a hardcoded set of IP addresses. Due to this, there are limits to how many devices can be connected for a WIFI Direct group. We can mitigate this easily by modifying the AOSP code below to assign the IP address dynamically.

```
private static final String[] DHCP_RANGE = {"192.168.49.2",
    "192.168.49.254"};
private static final String SERVER_ADDRESS = "192.168.49.1";
```

Another option is to use the Android VPN connection to connect the android devices. With that, we can make the ad-hoc network using Wifi Direct protocol as follows. and this provides the least modifications to the Android OS.

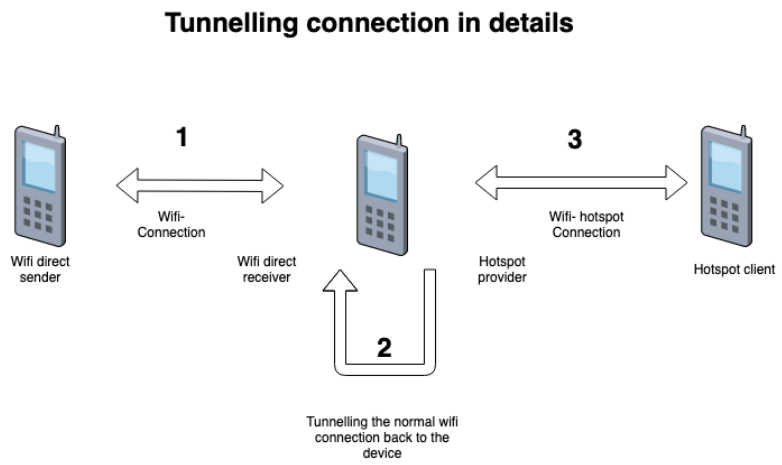

Figure 4.6: How tunnel connections are used in the network for inter-cluster connections

4.0.2 Cellular Network Layar Implementation

To work with the cellular network configuration we need good Linux networking tools like ifconfig, ip, tcpdump, and netstat with the driver support for their subcommands. The problem with working the Android OS is that it does not have any of these tools. And most of these tools need superuser execution privileges to work properly. Due to security concerns, Android OS does not provide superuser access for the users. So to do this we need to overcome these problems.

Enabling the Root access for Android OS

Android does not have superuser access by default so we need a way to implement that back to the android kernel. The easiest way is to modify the boot.img of the devices so that we can implement the root user. To do this first we need to add the su binary back to the source code. To create the su binary you need to cross-compile the source code with the correct cross compiler. In this case, it is aarch64-linux-android-4.9.

Then we need to modify the initrd so that it spawns our custom process with the root privileges. Then we can use the privilege manager application like superSU to manage the root privileges via the custom initrd process. The actual process is much more complicated than the one we mentioned previously.

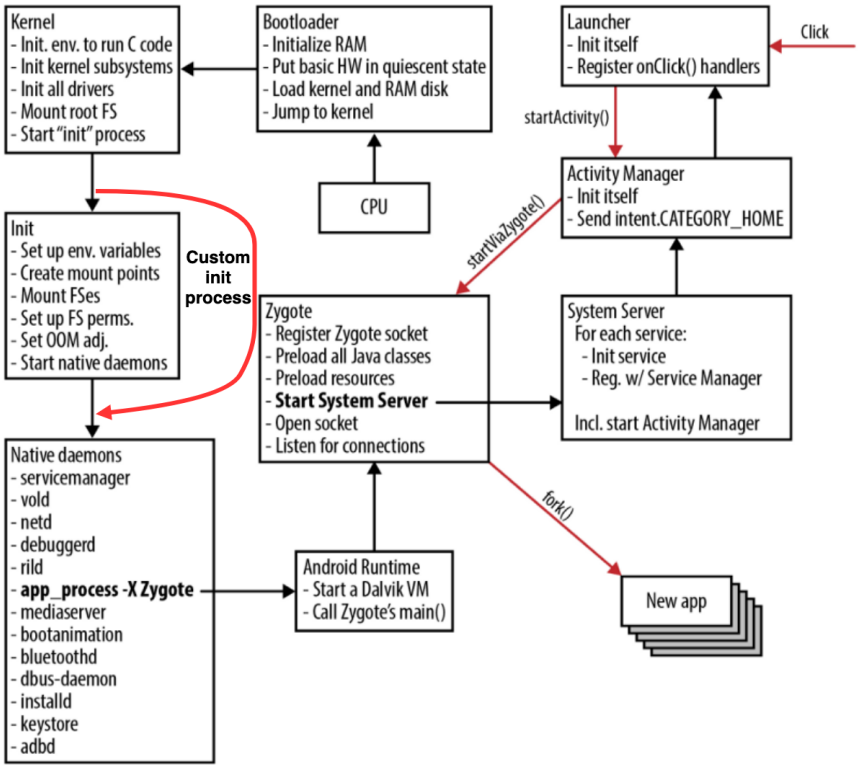

Figure 4.7: The boot process of Android

Porting the Networking tools to Android

After successfully rooting the devices next we need the networking tools to call the kernel modules. For this, we tried the already available solution called busybox. This was not sufficient because it lacks some fundamental tools which are needed for implementing the network. So we used the ported version of the Debian package manager for this research. Fortunately, there was a ported version of the Debian package manager that was available in the termux[38] open source project. So we used the termux packages and their build scripts to cross-compile the networking tools to my test device Nexus 5x.

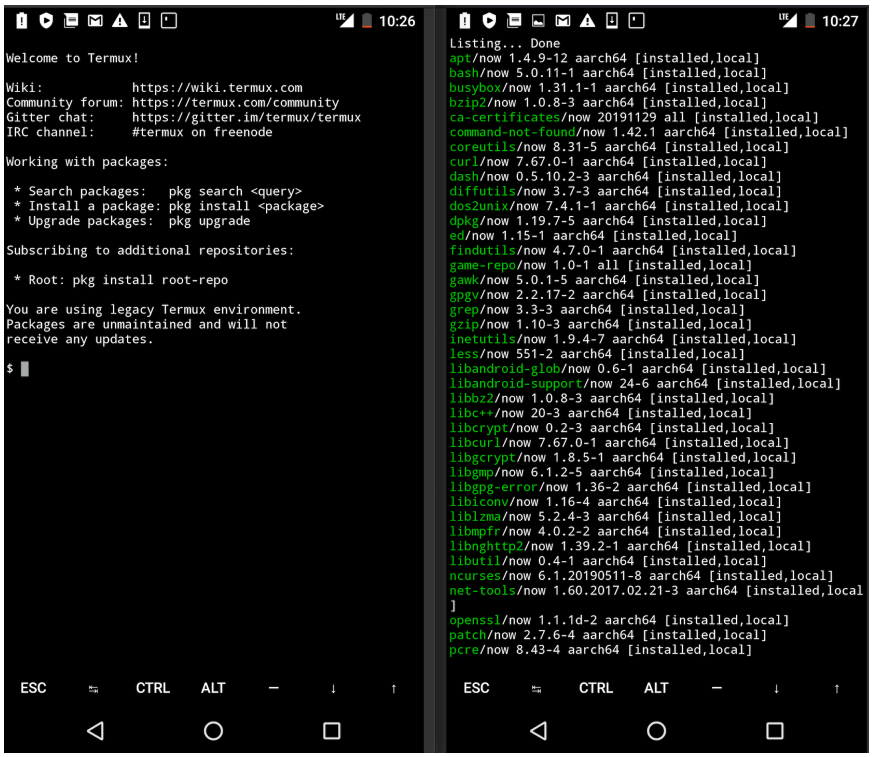

Figure 4.8: Ported Android pkg manager in Terminal Emulator

4.0.3 Implementation of the Control Plane

The control plane of this hybrid network needs to find the currently available Android devices to change the routing decisions of the network. Previously this was done by allocating a different channel inside the ad hoc network. As ad-hoc networks being are volatile this had bad side effects on the network's performance. So our proposed method uses a cellular plane as its control channel and Ad hoc network as its data plane.

When an Android device is connected to the Ad-hoc network using the WIFI Direct protocol that device contacts a predefined server creating a reverse ssh connection to that server's specific port. After that, the server connects back to the Android device using that reverse ssh tunnel bypassing the firewalls and network address translation implemented by the service provider. As the cellular connections use IP pooling the external IP changes in an uneven manner.

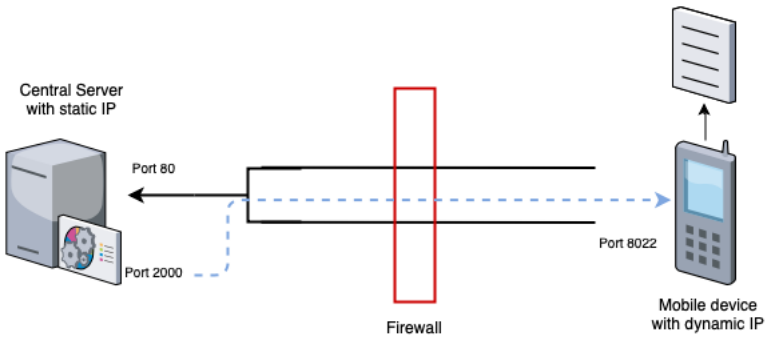

Figure 4.9: Creating a reverse SSH connection between server and the device

After connecting to the Android device it finds the IP addresses of the ad-hoc and cellular interfaces. Then it creates a tunnel through the ad-hoc network with the same IP address as the cellular interface of the devices. Then it increases the priority of the tunnel interface using the routing table such that if data is passing to the other android device using the cellular connection previously now it uses the tunnel interface. By doing so we are effectively prioritizing the ad hoc network over the cellular one. If the ad-hoc connections break then tunnel interface is automatically deleted by the bash automation script. So it returns to the normal state.

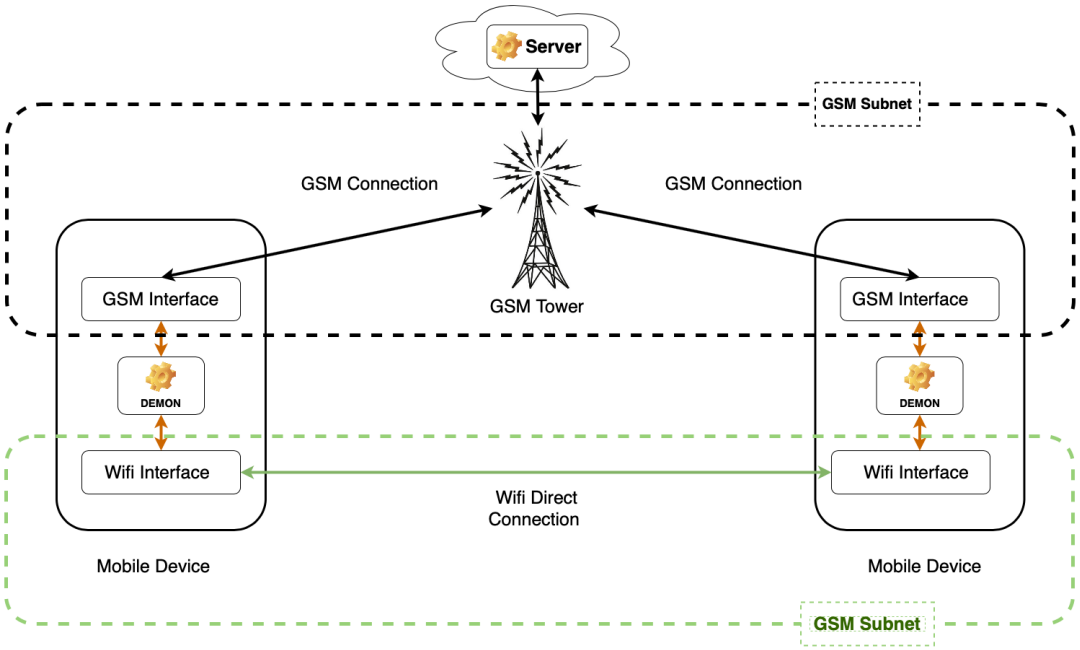

Figure 4.10: High-level Architecture diagram

The IPIP tunnel driver needed to implement this was not natively available in the android kernel. So we had to cross-compile the kerned after adding that module. After doing so we were able to configure the networks of the Android as described above using the remote server in the control plane. As you can see we were able to create a tunnel interface through the ad hoc network by using the IP point to point tunnelling protocol. We gave the external cellular network of device A to the tunnel interface of device B and vice versa. Then we modified the routing table of the Android device to give the newly created tunnel interface a higher priority. Due to this, there are two paths for routing the packets between the android device. Those are the path through the cellular interface and the Ad-hoc interface. As IP tunnel runs through the ad-hoc interface it gets the higher priority. We have implemented the demon process which checks the tunnel connectivity. If the ad-hoc connection breaks it automatically removes the tunnel interface then the Android device comes back to its initial state.

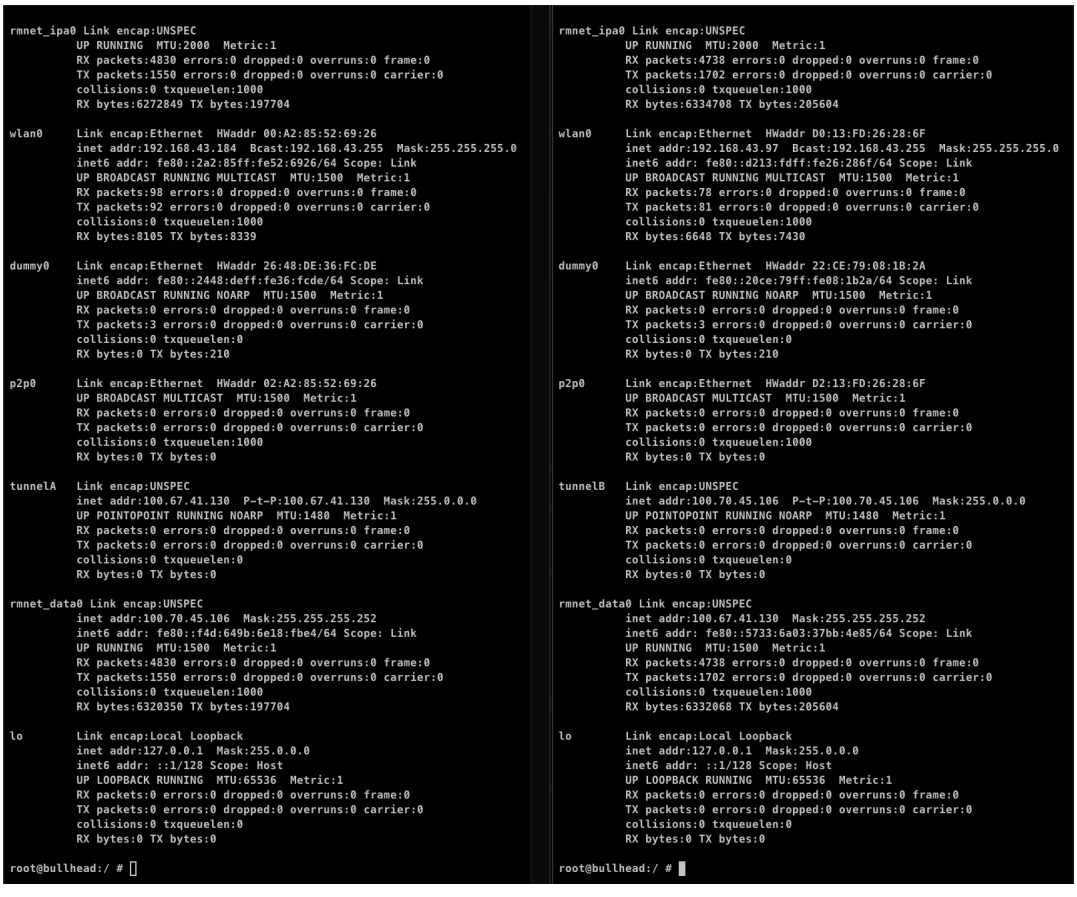

Figure 4.11: Final network configurations of two Android devices

The way that the IP and firewall rules are implemented in the android is not straight forward. So we will be demonstrating this switching process with three Linux machines.

Chapter 5

Results and Evaluation

5.1 Generating the Data

For benchmarking and comparing this hybrid network with other networks needs to be done using parameters like Throughput, Network initialization time and latency properties of these networks. According to the literature review, we have done before these are the main matrices that most of the researchers used in order to compare the performance of these networks with each other. Additionally, we are benchmarking the power consumption of these devices in order to get a good understanding of the real-world implications of this network.

The throughput of the networks was measured with the help of the Linux networking tools wget and Nginx. we hosted an Nginx server on the port 8080 of each device and measured the downloading speeds of each of the devices when using the particular networks. Furthermore, we have taken these measurements multiple times in order to create a more accurate result data set.

Networking tool Ping was used in conjunction with the bash current system time variable in order to calculate the start-up time of the networks. we wrote a bash script to calculate the time difference from the start time to positive ping response time for the currently enabled device. As the cellular and WIFI networks have almost instantaneous response time we are only considering the hybrid network benchmarks for this metric.

Network latency of these networks can be easily calculated using the ping commands. To get more accurate results we repeated the experiment many times at different time intervals using different devices.

It was not possible to measure the exact power draw for these networks as we cannot power off most of the default applications in the mobile device. we have used the battery power draw information to calculate the power draw of the system. To get a more accurate result we have taken multiple readings using multiple devices.

5.2 Results

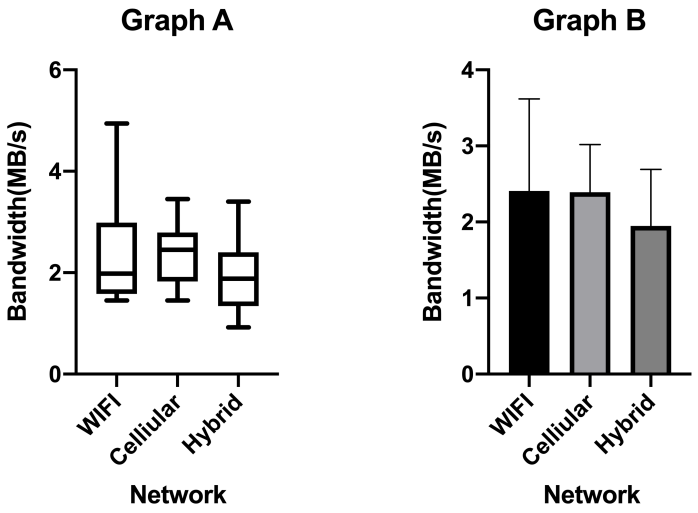

Figure 5.1: Available Bandwidth for different Networks

As you can see in Figure 5.1 WIFI network has the highest bandwidth when compared with the other networks. This is because it has the highest Radio frequency range form the above three wireless networks. Cellular network performance is generally stable than the other two networks. The highest mean value further justifies this argument. According to the graph, the hybrid network has relatively similar bandwidth to that of the cellular and WIFI networks.

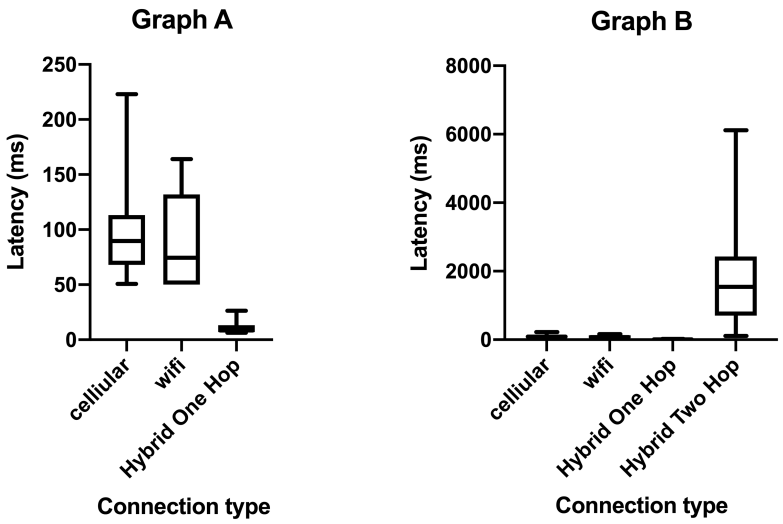

Figure 5.2: Latency of Different Networks

Due to being a peer to peer network, the hybrid network has the smallest latency from these two networks. So this is an improvement form the already available solutions. But when we increase the no of hops in the network this advantage diminishes rapidly. This can be seen by analyzing Figure 5.2 graph A and graph B.

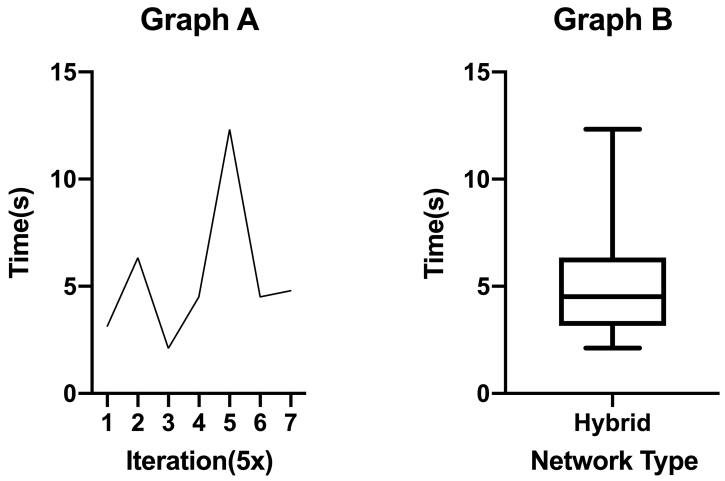

Figure 5.3: Time takes for the Hybrid Network to do network Switching

As the proposed solution is a unique one we were unable to find a similar solution that could be used to benchmark this network switching parameter. According to Figure 5.3 Graph A the actual value for switching fluctuates for different scenarios but the mean value for switching in the network is about 5 seconds when comparing the Figure 5.3 Graph B.

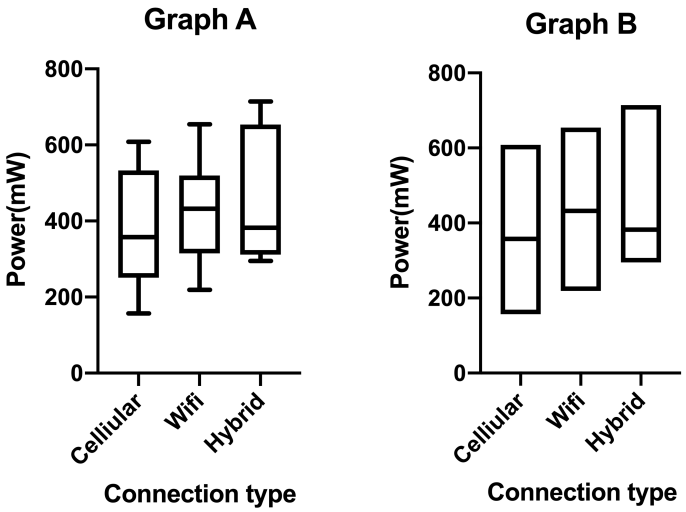

Figure 5.4: Power consumption of the particular Networks

When comparing the power consumption of the devices we used the internal battery power consumption measuring sensor to find the power consumption in real-time. As you can see in Figure 5.4 hybrid solution needs more power compared to the cellular and WIFI networks. This is due to the fact that it needs both of the above networks in order to function properly. As we use already available networking resources the extra layer of the protocol does not affect the energy consumption that considerably.

Chapter 6

Conclusion

6.1 Introduction

In conclusion, we can say that there are some advantages when using this hybrid Network instead of a general one. This network is particularly useful if we have less number of hops in the Ad hoc network. When comparing with the other similar networks using the relevant comparisons matrices we can say that hybrid solution has performed well in these fields.

Due to the protocol overhead, the network switching time for this network is somewhat high. This can be reduced if we do protocol implementation in the OS or Kernel level.

And the power consumption of the system is in the range of the other networks. This is a good trend as most of the hybrid networks tend to consume a higher amount of power than the other infra-structured networks.

6.2 Research questions(aims and objectives)

By implementing a working solution of this Ad-hoc cellular hybrid solution we were able to answer the first part of the research question. Using the working prototype we showed the is feasible to implement a hybrid network using the cellular network as its control plane and Ad hoc network as its data plane.

As you can see from the implementation chapter we found that different newly emerged technologies were used in implementing this network. Even though it is hard it is possible to create such a network using the currently available technologies.

So it can conclude that the hybrid solution we introduce in this research is a valid one when considering chapter 5 and have visible gains to the user from this kind of network.

6.3 Limitations

In this research, the working prototype was implemented using the Nexus 5x devices wish runs on Android OS and Linux kernel. As we adhered to this configuration we were unable to find the applicability of this methodology and techniques to other platforms of mobile devices.

And the testing was done using a limited number of Android devices which can affect the end results. So a wider range of test data is needed in order to examine the quality of the systems further. Due to Android kernels networking implementation being a non-standard one we were unable to change the firewall and IP rules in order to fully implement the hybrid networking protocol in the Android devices. So we demonstrate that part instead of using Linux machines.

6.4 Implications for further research

We can extend the implementation to multiple devices in order to further investigate on the first part of the research question. This will be helpful in generating a wider data set which intern increase the accuracy of the final result.

Further research can be carried out to implement this hybrid network completely in the kernel layer of the Android device to improve the performance further.

As this is a novel solution the routing algorithm for this hybrid network can also be improved in order to give more efficient routing solutions for the end devices by the control plane.

The current implementation described in chapter 5 uses a centralized system for the control plane which leads to problems in the availability of the system. Finding the feasibility of implementing a highly available decentralized control plane can prove further in time.

References

[1] P. Wijesekera and C. I. Keppitiyagama, "Comonet: Community mobile network," University of Colombo School of Computing, Sri Lanka, 2007.

[2] J. Thomas, J. Robble, and N. Modly, "Off grid communications with android meshing the mobile world," pp. 401–405, 11 2012.

[3] "Ictfactsfigures2015.pdf." https://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2015.pdf, 12 2019.

[4] "Global mobile data traffic from 2017 to 2022 by cisco." https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-738429.html, 03 2017.

[5] K. Liu, W. Shen, B. Yin, X. Cao, L. Cai, and Y. Cheng, "Development of mobile ad-hoc networks over wi-fi direct with off-the-shelf android phones," pp. 1–6, 05 2016.

[6] "Gsma - network coverage maps." https://www.gsma.com/coverage/, 03 2019.

[7] E. J. A. Jr, "Mobile communication system - google patents," U.S. Patent US3663762A, May. 1972.

[8] "Download speeds: Comparing 2g, 3g, 4g and 5g networks." https://kenstechtips.com/index.php/download-speeds-2g-3g-and-4g-actual-meaning, 02.

[9] "About mobile technology and imt-2000." https://web.archive.org/web/20080524050117/http://www.itu.int/osg/spu/imt-2000/technology.html#Cellular%20Standards%20for%20the%20Third%20Generation, 02.

[10] "Requirements related to technical performance for imt-advanced radio interface(s)." http://www.itu.int/pub/R-REP-M.2134-2008/en, 02.

[11] C. K. Toh, *Ad Hoc Wireless Networks: Protocols and Systems*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1st ed., 2001.

[12] C.-K. Toh, *Wireless ATM and Ad-Hoc Networks: Protocols and Architectures*. Norwell, MA, USA: Kluwer Academic Publishers, 1996.

[13] "Wi-fi direct | wi-fi alliance." https://www.wi-fi.org/discover-wi-fi/wi-fi-direct, 03 2019.

[14] R. Schollmeier, "A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications," pp. 101 – 102, 09 2001.

[15] H. Jahankhani and S. Yousef, *Evolution of TETRA through the integration with a number of communication platforms to support public protection and disaster relief (PPDR)*, pp. 259–273. 12 2014.

[16] "Multipeerconnectivity | apple developer documentation." https://developer.apple.com/documentation/multipeerconnectivity, 02.

[17] "Wi-fi direct (peer-to-peer or p2p) overview | android developers." https://developer.android.com/guide/topics/connectivity/wifip2p. Accessed: 02/16/2020.

[18] "Batmanconcept - open-mesh - open mesh." https://www.open-mesh.org/projects/open-mesh/wiki/BATMANConcept, 03 2019.

[19] M. TechNet, "Virtual private networking: An overview," September 2001.

[20] "Vpn | android developers." https://developer.android.com/guide/topics/connectivity/vpn, 03 2019.

[21] "The aodv model." https://www.usenix.org/legacy/publications/library/proceedings/osdi02/tech/full_papers/musuvathi/musuvathi_html/node13.html, 03 2019.

[22] "Android common kernels | android open source project." https://source.android.com/devices/architecture/kernel/android-common, 03 2019.

[23] "Ieee standard for local and metropolitan area networks: Overview and architecture," *IEEE Std 802-2014 (Revision to IEEE Std 802-2001)*, pp. 1–74, June 2014.

[24] "Rfc 4251 - the secure shell protocol architecture." https://tools.ietf.org/html/rfc4251, 02 2019.

[25] "Multipath tcp (mptcp) - documents." https://datatracker.ietf.org/wg/mptcp/documents/, 02 2019.

[26] C. Funai, C. Tapparello, and W. Heinzelman, "Enabling multi-hop ad hoc networks through wifi direct multi-group networking," pp. 491–497, 01 2017.

[27] T. Oide, T. Abe, and T. Suganuma, "Infrastructure-less communication platform for off-the-shelf android smartphones," *Sensors*, vol. 18, p. 776, 03 2018.

[28] "The symbian operating system." `https://www.symbianos.org/`, 03 2019.

[29] "iwconfig - debian wiki." `https://wiki.debian.org/iwconfig`, 03 2019.

[30] "Lineageos wiki." `https://lineageos.org/Changelog-23/`, 03 2019.

[31] H. Wu, C. Qiao, S. De, and O. Tonguz, "Integrated cellular and ad hoc relaying systems: icar," *Selected Areas in Communications, IEEE Journal on*, vol. 19, pp. 2105 – 2115, 11 2001.

[32] A. Asadi and V. Mancuso, "Wifi direct and lte d2d in action," pp. 1–8, 11 2013.

[33] H. Luo, R. Ramjee, P. Sinha, L. Erran) Li, and S. Lu, "Ucan: A unified cellular and ad-hoc network architecture," pp. 353–367, 01 2003.

[34] C. E. Casetti, C.-F. Chiasserini, L. C. Pelle, C. D. Valle, Y. Duan, and P. Giaccone, "Content-centric routing in wi-fi direct multi-group networks.," in *WOWMOM* (L. Bononi, G. Noubir, and V. Manfredi, eds.), pp. 1–9, IEEE Computer Society, 2015.

[35] P. Jacquet, P. Muhlethaler, T. H. Clausen, A. Laouiti, A. Qayyum, and L. Viennot, "Optimized link state routing protocol for ad hoc networks," pp. 62 – 68, 02 2001.

[36] N. Alshaer and E.-S. El-Rabaie, "A survey on ad hoc networks," 11 2016.

[37] W. Shen, W. Hong, X. Cao, B. Yin, D. M Shila, and Y. Cheng, "Secure key establishment for device-to-device communications," *2014 IEEE Global Communications Conference, GLOBECOM 2014*, 10 2014.

[38] "Github - termux/termux-packages: Android terminal and linux environment - packages repository." `https://github.com/termux/termux-packages`, 02 2019.

Appendices

Appendix A

Code Resources

Kernel Compiler Toolchain : https://github.com/rmddcr/toolchain
Modified Kernel code : https://github.com/rmddcr/android-kernel-bullhead-3.10-marshmallow-mr1
Android OS Code : https://android.googlesource.com/platform/system/core/+/refs/tags/android-6.0.1