Mobile Guerilla Network

Logan Davis, Joseph Cantrell

***Overview*—This project was designed to solve the problem of communication when cell towers and Wi-Fi routers are unavailable. We overcame this issue by developing an application that allows direct peer to peer communication between devices with a Wi-Fi and/or Bluetooth transmitter. We were able to successfully build a proof of concept that allows communications between 2 or more android devices without the use of cellular or Wi-Fi services.**

I. Introduction

Our team has decided to create a simple messaging app that will send and receive text messages over Bluetooth or Wi-Fi Direct. This concept is like the idea of a walkie talkie app. The inspiration behind this idea came from the fact that most (if not all) walkie talkie apps on the Google Play Store require a cellular connection to work. In making this app work over Bluetooth or Wi-Fi, we are bringing accessibility to the user and furthering communication. For this project, we will be working in the react native environment. Since our team doesn't have much experience working with react native, it will be a learning process. This application will be designed and programmed for the Android platform. If all goes well, we may consider adding cross platform support to iOS devices.

II. Related Work

There are numerous apps that attempt the same goal of wireless communication to people within close proximity. Our Mobile Guerilla Network differs from them in a very important way.

A very popular app that does this is "Zello PTT Walkie Talkie". With this app, you can both send messages to specific people as well as real time audio messages. The downside to this app despite some people believing otherwise, is that this app does indeed require internet to work. This app works over Wi-Fi or cellular but requires an internet connection to be active for it to work.

Another popular app is "Bluetooth Chat". This app is actually very similar since it does not require internet and does transmit over Bluetooth. The downsides to this app over ours are that it only utilizes Bluetooth while ours uses both Wi-Fi and Bluetooth thus making our range slightly longer. Our messaging app also retransmits messages further improving the range. Another downside, depending on how you look at it, is that this app requires you to connect to the specific person you want to talk to. This is worse for the situations our app was designed for, emergency situations. Our app talks to any discovered peer and sends talks as long as the app is open without any user connection acceptance.

Overall, we believe our app satisfies a unique niche that as yet to be fully explored in the app store.

III. Application Design

Our application is designed to be simple and accomplish its purpose quickly and efficiently. As soon as the app is launched, it will immediately start detection for nearby peers to connect to. Every peer it locates, it adds to a discovered peer array and lists the number of peers found to the user.

When a user receives a message, if they are in the application it immediately will populate the received messages list with the most recent message appearing at the top. If the app is not open but is running in the background, the user will receive a notification with the contents displaying a partial of that message. If the notification is clicked, the app is opened.

When the user wants to send out a message, they simply start typing in the message box, and when their message is ready, they will hit the send button. The send button will iterate through all of the peers detected and will attempt to send the message to each peer. When a peer receives the message, if it is there first time receiving said message, they will retransmit out to all peers they're aware of thus further increasing the range of the application. Sent message will also appear in the messages list.

All messages are saved locally on the device so that when the app is launched, the user will see all previously sent and received messages.

IV. Development Cycle

The first steps in developing our app involved getting a base react native application working. Once we had the starter app created and loaded onto an android device, it was time to start basic peer to peer coms to ensure this project was feasible for us. We were able to pretty quickly get a very, very basic user detection working. After we felt good about having a solid path moving forward for the remote coms, we started building the GUI to interact with it.

The GUI starts off pretty basic with a simple layout that allows us to setup the basic properties needed

for the coms. We also made simple buttons to allow for testing of the different actions associated with the library we used. Currently the UI consists of a logo that is associated with the application, a text box that displays the messages sent between devices, a text box that allows users to type their messages, and a send button that will send the message to all devices that are connected.

After getting the UI in its basic form, we started playing with the message transmissions. In order to get our messages to transmit correctly, we had to use JSON syntax and convert to and from strings in order to send messages with userIds for identification, messageIds to avoid redundancies, and the actual message to send. Once we were successful in in getting messaging to work, it was just a matter of adding on the additional features such as notifications, peer counts, message saving, etc.

V. Results

The overall results of the project were a success. Despite the app not being production ready, which you wouldn't expect it to be in the time we had to produce it, it works fairly well.

The application does all the tasks that we originally set out for it to accomplish and more. We are able to send messages directly to any device detected within Wi-Fi / Bluetooth range and forward said message on receive. The app can uniquely identify messages in order to associate them with users, stores messages to a file, has basic settings for message clearing and naming, and other processes.

Even though we were able to accomplish our original goals, our app still has some shortcomings. Too new versions of android seem to lock up the application after a period of time with older versions of android working flawlessly. Peer detection is sometimes hit or miss depending on the device with issues seemingly still related to the version of android. Really old android versions won't even install the application as they lack the necessary components to run it.

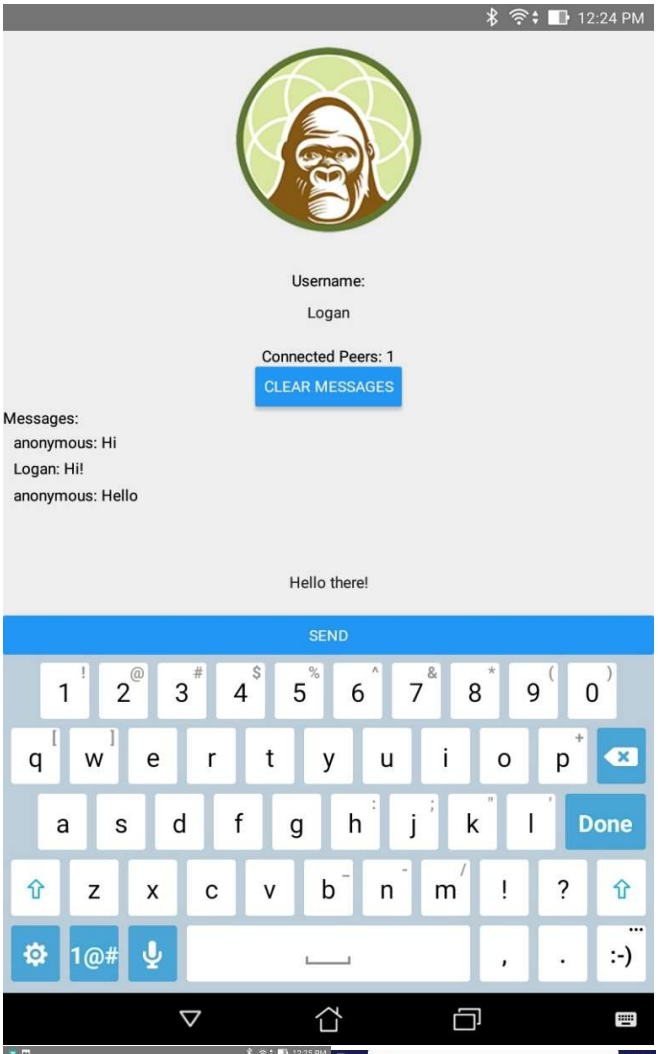
Given these limitations, this application is hardly a universal solution in its current state yet serves as a great proof of concept as to what could be developed with the technology used for emergency situations.

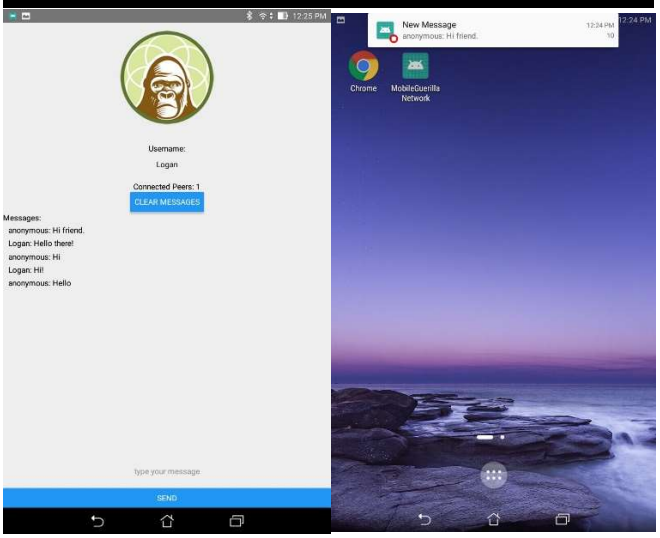
VI. Future Work

Given the time to further work on and develop this project, there are numerous things we would love to improve and further add.

1. IOS compatibility
2. More reliable peer detection
3. Audio message transmission
4. Direct messaging to specific peers
5. Better android version compatibility

VII. Images





VII. References

alexkendall/react-native-bluetooth-cross-platform https://github.com/alexkendall/react-native-bluetooth-cross-platform