COMPUTER SCIENCES AND SOFTWARE ENGINEERING
AUBURN UNVERSITY

COMP 5360/6360
Wireless and Mobile Networks
Spring 2013

Project 1

Mobile Phones without Cell Towers Using WiFi Network Emulation and Socket Programming

Due: 11:55pm February 20, 2013

1 Objective

The goal of this project is to design and implement a mobile phone voice communication application over smart cell phones or computers in a WiFi emulation environment, i.e. without the use of cell towers or access points. This voice chat application will allow users to have two-way voice communication over the network through the socket interface. In the real environment, the two devices will be connected through a WiFi link in ad-hoc (IBSS) mode. In this emulation environment, you may emulate the underlying wireless link layer with exchange of UDP datagrams between two neighboring hosts through AF_INET sockets. However, the protocol you will implement can be ported to an actual network layer. In this first project, audio packets may be transmitted between two smartphones or computers that are connected to the Internet using the TCP/IP protocol suite. However, in subsequent projects, the audio packets will be transmitted over wireless mobile and ad-hoc networks (MANET) using ad-hoc routing protocols or an emulation of the MANET over a fixed network. Your implementation will also use various software that support sound API (Application Programmer Interface).

2 Background

In the events of earthquakes, hurricane, or other wide-spread disasters and even in remote areas such as in the mountains, cell phone towers or access point infrastructures will likely go down or are non-existent. In the absence of cell towers or access points, your sophisticated smartphones will be rendered useless, unless we can make the mobile phones systems more resilient and capable of operating even without cell towers, access points and other communication infrastructures. Our goal is to allow mobile phone users to communicate with each other through a virtual and dynamic network that can be spontaneously established in areas where no cell tower coverage exists. People traveling to remote mountain areas will then be able to communicate with each other for free.

3 Requirements

The objective of this first project is to implement a working two-way voice communication system over the local LAN that will enable users to initiate voice conversations with other users. Using these programs, a user at host computer or smartphone A will be able to speak through a microphone attached to A where a program will extract the voice data and transmit it in a packet through a UDP socket to another

computer or smartphone B attached to the Internet. (Smartphones may be replaced by other wireless Android devices.) The program in B will receive the voice packet and play the voice message in its speaker. Similarly, the user at B will be able to speak through its microphone attached to B and their voice transmitted to and played on A's speaker. Make sure that both A and B are separated over a long distance, otherwise there could be feedback from B's speaker into A's microphone and vice versa. An alternative way to avoid the feedback is to use a headphone.

You can code this program in any language of your choice, although we recommend that you code it in Java. This project is usually more easily implemented with Java Sound API, Java socket API and multithreading.

3.1 Voice Chat Programs

In its simplest form, the voice chat system consists of two programs: (1) a transmitter that extracts the voice data from the microphone and transmits it to the other user through a UDP socket and (2) a receiver that receives the voice packet from a UDP socket and plays it on its speaker. The quality of the voice should be reasonable. Do not be too concerned with the delay of the voice in the receiver's speaker, but the delay should be reasonable. (You can attempt to reduce the delay for extra credit.) You can make use of Java Sound package to process the audio signals from the microphone and playing it on the speakers. You may also use the sample codes in a few helpful websites below.

3.2 Useful websites

The following are some websites that may be helpful in this project:

1. Code example of how to record sound in Java:
   http://www.developer.com/java/other/article.php/1565671

2. Sample codes for Voice chat in Java:
   http://javasolution.blogspot.com/2007/04/voice-chat-using-java.html

3. Sample codes of simple application for transmitting and receiving voice in Java:
   http://javasolution.blogspot.com/2007/04/simple-application-for-voice.html

4. For basic Java tutorials look at the "Trails Covering the Basics" section on Sun's Java website:
   http://java.sun.com/docs/books/tutorial

5. To get started with Java socket programming, the following tutorial is all you'll need:
   http://java.sun.com/docs/books/tutorial/networking

6. Java Datagram tutorials
   http://java.sun.com/docs/books/tutorial/networking/datagrams/index.html

7. For GUI's, the Java Swing tutorial should help:
   http://java.sun.com/docs/books/tutorial/ui

4 Testing

Your voice chat application programs must execute correctly in the College of Engineering Linux tux computers in Davies 123, unless you are doing extra credits and implementing the voice chat system in Smartphones or Android devices. The voice chat application must allow at least two users to communicate with each other, whereby each user must use a different tux computer. You will supply your own microphones; a speaker is already attached to all the tux computers. If you are unable to supply our own microphone, let me know and I'll loan you two microphones per group.

If you choose to implement the extra credit voice chat program over smartphones or Android devices, make sure your programs work in the devices that you provide yourself.

5 Extra Credits

For extra credits, you may choose to do the following. There could be *significant extra credits* given if the following tasks are implemented successfully.

1. Implement your voice chat programs on smartphone (e.g. iphones) or wireless Android devices. These devices include tablets, media player, and smartphones, all of which must have WiFi capability. You may implement this on the Android Operating System or alternatively over Linux that you can install on these devices. Keep in mind that if you choose this implementation that you will need to use these devices to form spontaneous mobile and ad-hoc networks (MANETs) of these devices communicating through WiFi in Ad-hoc (IBSS) mode.
2. Improve the quality of your voice chat application programs by adding the following enhancements:
    a. Improve the Quality of Service (QoS) of the voice communication, e.g. reducing the delay and noise, e.g. using a voice codec, so that it is comparable to typical cell phones services. You may also incorporate echo cancellation program so that you can use speakers as well as headphones.
    b. Implement a capability for dialing a certain user using some form of numbering or naming scheme and provide a service for looking up a number or name in the server.
    c. Provide an efficient and nice GUI for the users to execute the voice chat programs and manage the connections.

6 What to do and turn in

After you have implemented, debugged, and thoroughly verified your voice chat implementation, submit the following in Canvas on or before the due date.

a. Complete two-way voice chat application codes (which should include comments for full credit)
b. A brief (2 pages) report the describes your system architecture, design and implementation issues

You will demonstrate your program to the TA or me on February 21, 2013 in Davis 123 on the COE network of Linux computers.