December 15 - Project 13

Wireless Mesh Networking App

Software Design Document

Group Members

Cole Cummings
Cody Lougee
Ethan Niemeyer
Holden Rehg

Advisor

George Amariucai

Client

George Amariucai

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Purpose

This design document describes the architecture and system design of a mobile ad-hoc network using DSR protocol and encryption techniques. This document is for explaining the processes which we will take when implementing our apple iOS app.

1.2 Scope

The goal of our project is to build a messaging application for mobile devices that supports text, voice and video communications in the absence of a standard wireless network. The main objective is to allow secure and encrypted communication between specific users because this seems to be missing from similar projects. The main benefits would be its usefulness in times when traditional networks fail, for instance, natural disasters may destroy infrastructure like cell towers but responders need to be able to communicate. Some other applications have also been used at times when the government restricted access to traditional internet. Firechat was used in Iraq and Hong Kong protests, but the communications were not encrypted.

1.3 Overview

This document is meant to provide the software development team with guidance to each layer of the application architecture. It is organized by software layers and broken into three layers. The system architecture describes program structure, sub system interfaces, and data flow throughout the application. The data design describes data structure, data organizations, data storage and relationships between data needed. The user interface design describes the visual design and functional design of the interface that the user will interact with. In this case, all section of the document are relative to an iOS mobile application.

1.4 Reference Material

DSR protocol - RFC 7428 http://bit.ly/1E8TasM
Multipeer Connectivity Framework - http://apple.co/1AieVoJ
IEE 802.11s - http://bit.ly/1xhwmof

1.5 Definitions and Acronyms

Node - Each device connected to a mesh network is referred to as a mesh node.

iOS - An operating system used for mobile devices manufactured by Apple Inc.

Multi Peer Connectivity Framework - The iOS library for peer to peer networking between mobile devices.

DSR - Dynamic Source Routing. This is a protocol for mesh networks that forms a route on demand when a transmitting node requests one. It uses source routing at each intermediate device.

UI - User interface and visual design.

UX - User experience and functional design.

Ad Hoc Network - A network where every node on the network has the same status and is free to associate with any other node on the network.

Mesh Network - A network topology in which each node relays data for the network. All nodes cooperate in the distribution of data in the network. Every node in a mesh network is called a mesh node.

Network Packet - Formatted unit of data carried/transferred over a network.

RREQ - Route Request. When a message is sent from one device to another, the sending device will broadcast a routing request to all devices in range in order to find the quickest route

RREP - Route Reply. When a RREQ reaches the destination node, it sends back the route reply which will tell the sender the best path to use to send the message.

RRER - Route Error. If a node drops out of the network and another node attempts to send it a message, the node that attempted to send the message will send a route error after it doesn't receive a response for a set amount of time. The route error is broadcast to all nodes and they remove the affected paths from their routing tables.

Routing Table - Each node keeps one of these tables of known routes to other nodes. This is updated as the node recieves RREQ's, RREP's and RRER's.

2. SYSTEM OVERVIEW

The mobile app should be able to communicate with other peers through wifi without any previous infrastructure. The pathways that the packets go through should be generated dynamically, allowing users to

drop in and out of the network. Data being sent to a node outside of the sender's range should be forwarded by other nodes until reaching the target. By going through peer to peer you are able to send data without paying for previous infrastructure as well as send uncensored data around heavy censored areas.

3. SYSTEM ARCHITECTURE

3.1 Architectural Design

There are three main components to the app, the UI, the networking logic, and the security logic. These three components work together to facilitate ad-hoc communication between mobile devices using the app.

UI

This is the interface through which users use the app. Users will interact with this component to utilize the functionality of the app. Users can switch to different views, send messages, and use the other functions outlined in this document. When a user send data to another device, the networking component is triggered to actually send the data. Data, like sent and received text messages, will be displayed on the UI.

Networking

This component will handle communication between devices. The responsibilities of this component is to build a routing table for packets, update the routing table when changes in the network are detected, and to send and receive data. This component works in the background on its own to discover other devices and receive data, and sends data when a user chooses to. The security component is used to encrypt and decrypt all communication except for when packets are sent out to discover other devices.

Security

The goal of this component is to make the communication between mobile devices using the app secure. Secure communication is accomplished through a public key, private key strategy. That is, a public key is used to encrypt data, and a private is securely stored by the recipient, which is used to decrypt the data. This component works hand in hand with the networking component to encrypt data that is sent, and decrypt data that is received.

3.2 Decomposition Description

3.2.1 UI

- See section 6

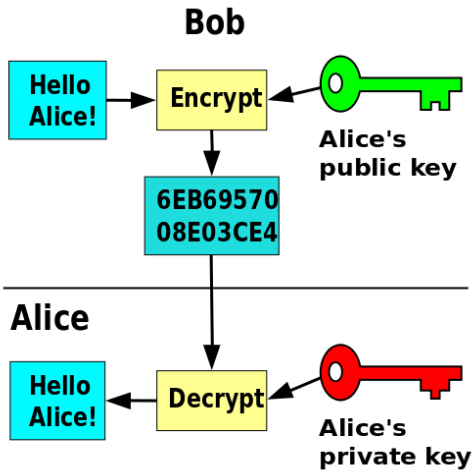
3.2.2 Networking

- Route Discovery
  - Receiving RREQ
    - The RREQ table is checked to see if this RREQ has been received earlier from a faster path, if so nothing is done with it. If it is new, go to the next step
    - The Routing table is updated with the path that this came from.
    - If this device is not the intended recipient, the RREQ has this device's id appended to the list and is sent on. It then waits for an acknowledgment
    - If this device is the intended recipient, it sends a RREP with the list of devices from the RREQ and waits for an acknowledgment
    - If no acknowledgment is received, send a RRER
  - Receiving RREP
    - RREP should contain a path to the intended recipient
    - If this node is the original sender, send the message to the intended recipient
    - If this node is not the original sender, send the RREP to the next node in line and wait for an acknowledgment
    - If an acknowledgment is received, do nothing
    - If an acknowledgment is not received for a set amount of time, send a RRER
  - Handling RRER
    - Update the routing table so that the affected routes are no longer in the table
    - If this device is the original sender, it can attempt to find another route, but the RRER may mean that the intended recipient has fallen out of the network

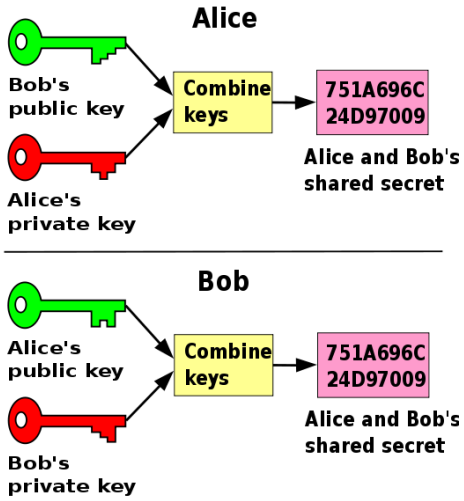
3.2.3 Security

- Encryption
  - Data sent needs to be encrypted
    - Outgoing data to a person will be encrypted based on their public key
  - Data received needs to be decrypted
    - Incoming data to a person will be decrypted based on their private key
- Keys
  - Public keys for individuals are given out to everyone in order to encrypt data

- Private keys will be a per person basis, and be used to decrypt incoming data



- Alternate Design
  - The targets public key will be used along with the sender's private key will be used for encryption
  - The targets private key and the sender's public key will be used for decryption



3.3 Design Rationale

The architecture outlined in section 3.1 was chosen because the components represent the three major requirements of the app. Those are, a user interface to actually use the app, a networking component that facilitates an ad-hoc network, and a security component that encrypts communication. The style of the user interface is meant to be simple since the main use of the app will be reading messages. Therefore, the UI should not be cluttered in order to make reading and navigation easier. For the networking component, iOS's Multi-peer functionality will be used. Multi-peer offers a platform on which to establish the ad-hoc network.

For the security component, a public key, private key strategy will be used since it is a well established way to securely communicate.

4. DATA DESIGN

4.1 Data Description

The data in the system will be stored in a SQLite database on a mobile iOS device. All data processing and storage will be done directly on the mobile device. Each user connected in the application will be stored. The application will use the DSR protocol for the ad hoc network. We will store each node that any given user has come in contact with within the network. This works through route discovery by flooding the network on timed intervals to see who is active. Any given node has a username and unique identifier. A route request or message is a data object containing media with a target node identifier and sending node identifier. As the route request travels through the network, it aggregates routes that it travels. A route reply or response is a confirmation that a route request was successfully received and provides the full route paths from the related route request. A Route Cache will be generated to store all known routes to each given node in a users network. The routes are cached by storing paths returns between users from route requests.

4.2 Data Dictionary

Route - Represents a path between two nodes. Stores a start identifier, stop identifier, and graph of nodes.

Route Cache - A cache of possible paths between nodes in the network. This will only be relative to the users mobile device. Stores and an array of routes.

Route Request - Represents a chat message. Stores target identifier, sender identifier, message media, and route traveled.

Route Response - Represents a confirmation that a chat message was successfully received. Stores the original route request.

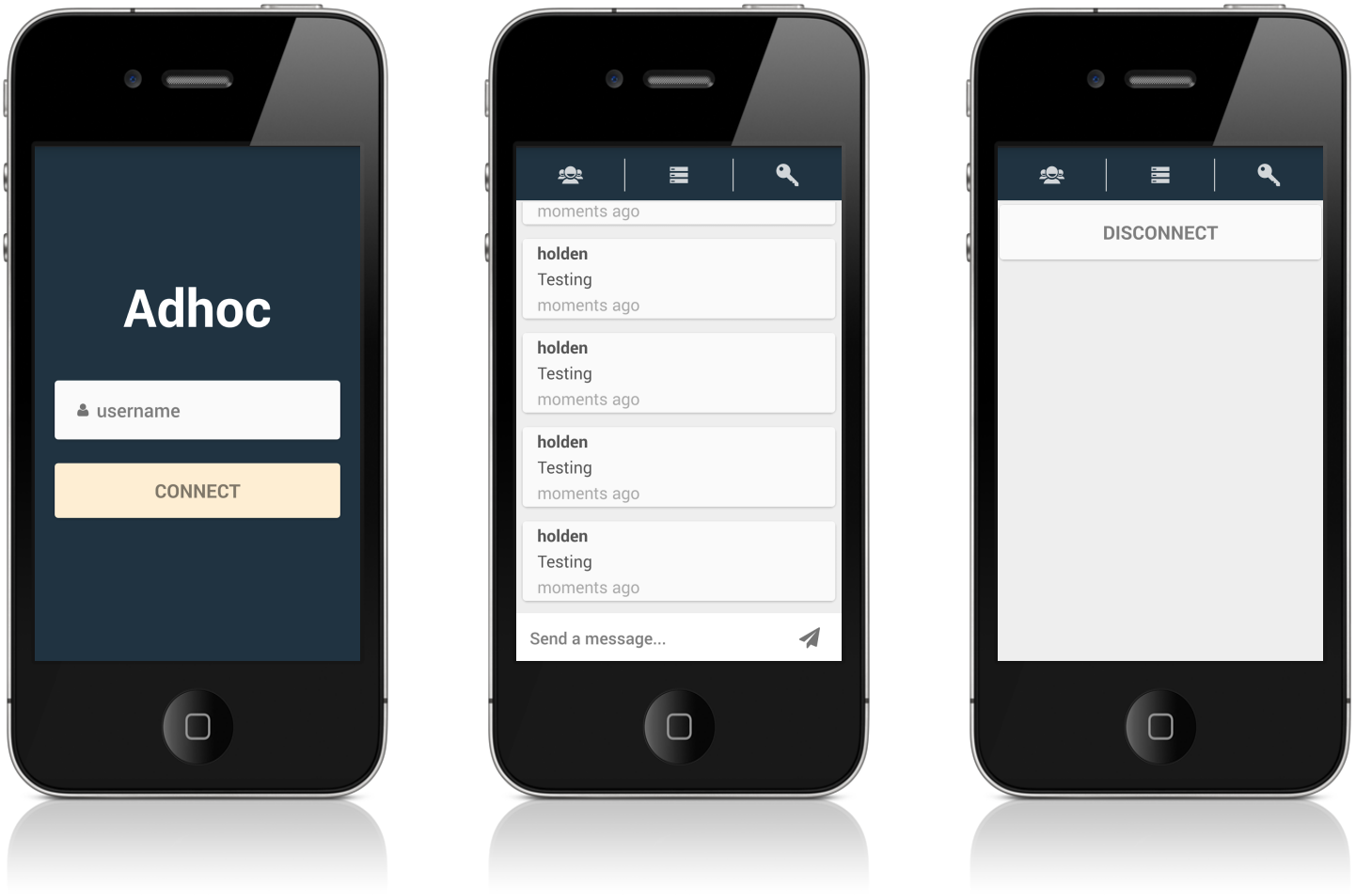
User - Represents a user connected to the network. Stores unique identifier and username.

5. HUMAN INTERFACE DESIGN

5.1 Overview of User Interface

All user interaction will be done through the iOS app. The users will be able to define their name, and see those who are near them. The users will also have a friends list and have the ability to chat with their friends and those online. Data will be sent through a submission box on the phone and data received will be displayed in a text box on the phone.

5.2 Screen Images



The application will provide users with a basic way of both connecting into the ad hoc and disconnecting based on a username that they provide. The user will have a public broadcast tab which submits messages /

media to every available node in the network. Each user that is connected to the user can be favorited and saved for quick lookup in another tab. And the user will have a tab for settings / preferences where they can disconnect from the network.

6. REQUIREMENTS MATRIX

6.1 Functional Requirements

- Ad-hoc network between mobile devices.
- Encrypted communication.
- A user interface that facilitates the following:
    - Text messaging
    - Audio messages
    - Picture messaging
    - Video messaging
- A user can create a favorite list of users.
- A user can public broadcast messages.
- A user can send private direct messages.

6.2 Non Functional Requirements

- Routes established within 5 seconds of connecting.
- Route request sent within 2 seconds over the network.
- Routes repaired correctly for all dropped nodes.
- 100% of messages sent to specific users are encrypted and cannot be read by other users.
- The network can scale to 100 adjacent users.