PEER TO PEER CHAT APPLICATION

A PROJECT REPORT

submitted by

SHAILENDRA SINGH B120588CS

NIKHIL BESRA B120105CS

SHINDE HIMANSHU SUBHASH B120863CS

YOGESH VERMA B120545CS

in partial fulfilment of the requirements for the award of the degree of

Bachelor of Technology
in
Computer Science and Engineering

under the guidance of

MR SUMESH T A

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY CALICUT
NIT CAMPUS P.O, CALICUT
KERALA, INDIA 673601
MAY 2016

ACKNOWLEDGEMENTS

We would like to offer our sincere gratitude to our project guide Mr Sumesh T A who took keen interest in our project work and guided us all along.We express our deep sense of gratitude to the panel members Dr Vinod P,Mr Vinith,Mr Varun,Ms Bijitha for their valuable help and guidance.We are thankful to them for their encouragement and timely support.We would also like to thank our course co-ordinator Dr Saleena N for their guidance and help. We are also grateful to our CSE department for providing the wonderful opportunity to work on project of our choice and also permitting to utilize necessary facilities of the institution.

SHAILENDRA SINGH
NIKHIL BESRA
SHINDE HIMANSHU SUBHASH
YOGESH VERMA

DECLARATION

"We hereby declare that this submission is our own work and that, to the best of our knowledge and belief, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgement has been made in the text".

Place:
Date:

Signature :

Name :
Roll No.:

Signature :

Name :
Roll No.:

Signature :

Name :
Roll No.:

Signature :

Name :
Roll No.:

CERTIFICATE

*This is to certify that the project report entitled: "***PEER TO PEER CHAT APPLICATION***" submitted by* **Shailendra Singh B120588CS,Nikhil Besra B120105CS,Shinde Himanshu Subhash B120863CS** *and* **Yogesh Verma B120545CS** *to National Institute of Technology Calicut towards partial fulfilment of the requirements for the award of Degree of Bachelor of Technology in Computer Science Engineering is a bonafide record of the work carried out by them under my supervision and guidance.*

Signed by Guide(s) with name(s) and date

Place:
Date:

Signature of Head of the Department

Office Seal

Abstract

Instant messaging has become an increasingly popular method for communicating over the Internet.Typical Instant messaging apps are based on client server architecture. Specific to user log-in, user's credentials and also pending messages are stored in the database at the server side. Anyone who has an access to the server and the database can easily access user information.In our peer to peer messaging application,we have no central server through which messages will be passed.The messages will be delivered directly to the recipient.Our application focuses mostly on the security aspect.We have added features like Incognito chat,which very well hides and protect the sensitive information and leaves no traces of the conversation.The pending messages are stored in the user's phone instead of a server. Then there are added benefits of end to end encryption of messages and also encrypted sqlite database.Apart from that we have also implemented conventional chat application features like Group Chat,Sending Pictures,OTP verification etc.

Contents

List of Figures

Chapter 1

Introduction

Instant messaging is a real time supplement and in some regards, a replacement for emailing.Unlike email,instant messaging allows users to see whether a chosen friend or co-worker is connected to the Internet.Instant messaging networks not only provide the ability to transfer text messages but also the transfer of files.On the other hand,recent break-ins into data servers of leading application vendors has caused an alarm among citizens around the world.People have become more concerned about the secrecy and security of their chat histories and file transfers.

This final report aims to provide the reader an insight to the decisions taken and the challenges faced by the group during the course of the project. It also gives the user a brief idea of how the project team intends to improve the app in future versions, if any

1.1 Problem Statement

The problem is to create an secure encrypted messaging application based on peer to peer (server-less) architecture on an android platform which can be used to send/receive text and images.There will be end to end encryption and message will only be stored locally on the device.

1.2 Literature Survey

Our literature survey started with the book [1].The book discusses strengths of P2P-based models against client/server architecture.It contains discussion of various P2P-models such as
1. Pure P2P model,
2. P2P with simple discovery server,
3. P2P with a discovery and look up server,
4. P2P with a discovery look up and content server.

We went through merits,demerits and working of each P2P model eg- Tracing out other peers in each case.Considering our requirements of the application,a P2P model with simple discovery server proved to be the best choice.

Such P2P models do not actually involve a server.The role of the server in this model is

restricted to providing the names of already connected peers to the incoming peer,which notifies the server about it's presence by logging in.

Research paper [2] helped us in understanding the decentralized P2P architecture better as well as choosing appropriate methodology for building a secure chat application.

Research paper [3] showcases the IEEE Recommended Practices for Software Requirements Specifications (SRS). It also describes the content and qualities of a good software requirements specification (SRS) and presents several sample SRS outlines.

The [4] helped us in learning Android app development which was helpful during the implementation phase of the application.

The [5] helped us to sort out any queries regarding android functions or classes.

Chapter 2

Design

2.1 P2P Architecture

A peer-to-peer network does not have the notion of clients or servers, but only equal peer nodes that simultaneously function as both clients and servers to the other nodes on the network.P2P is not an altogether novel concept. It has existed since the Internet was taking form in the 1970s. Recent changes in technology and the improved computing capability of desktops have brought about P2P's revival on a larger scale.

As against the client/server architecture, the greatest strengths of P2P-based models are their decreased dependency on the server and their decentralization of control from servers, which used to be workstations, to peers. Some P2P models do not require servers. End users can directly establish connections with other users without involving servers. Users have more command in P2P-based models than in the typical client /server architecture, in which conventional rules must be followed.Another advantage of P2P is that companies can build collective computing powers and thereby forget servers and expensive storage devices. P2P has shaken the boundaries of networking in terms of sharing resources and costs incurred on servers.

2.2 P2P model with discovery server

The very name of this model suggests its constitution. Such P2P models do not actually involve a server.To affect some administration, server boundaries have been laid down in this model. But the role of the server in this model is restricted to providing the names of already connected peers to the incoming peer, which notifies the server about its presence by logging in. It must be noted that the server only assists peers by providing a list of connected peers and that establishing connection and communication still remains the job of the peers (see Figure 1). Such P2P models surpass the pure P2P model by providing peers the list of already connected peers.

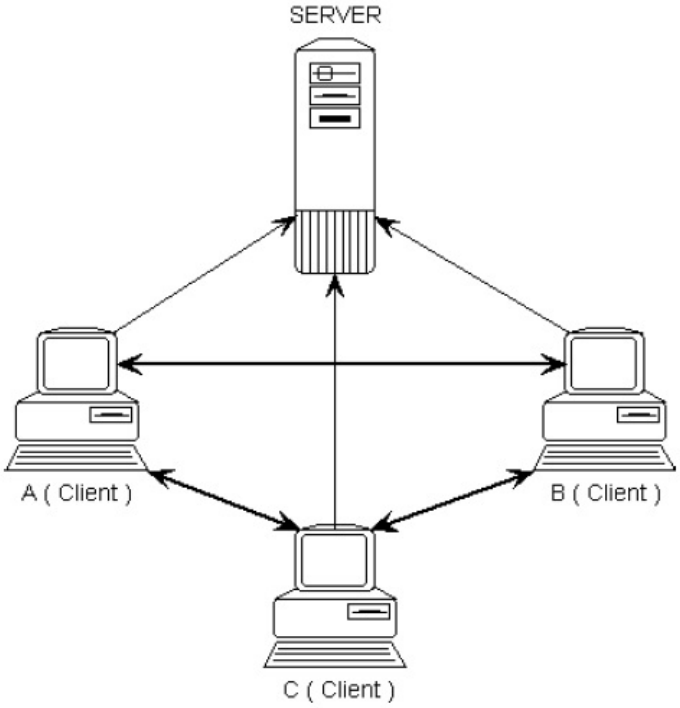


Figure 1-2: Only the discovery of clients occurs via the server; the rest of the communication occurs among peers.

Figure 2.1: Peer to peer network with discovery Server

2.3 Software Requirement Specification Documentation

The purpose of the Software Requirement Specification (SRS) document is to present a detailed description of Peer to peer chat application. It will explain the purpose, the features, the interfaces of the system and the constraints under which it must operate and how the system will react to external stimuli. The document will also explain the functional and non-functional aspects of the software.The intended audience for this document are the users and developers of the software. The SRS document is based on the IEEE standards and would also serve as a guide in the Software Development Life Cycle. (SDLC)

This document gives an overview of the functionality of the product. It describes the system environment to run the application.It contains various Use case diagrams and step by step description of each activity in the app.Requirements Specification section, of this document is describes in technical terms the details of the functionality like what will trigger the activity, pre condition,post condition,Basic path,Alternate path,How it should behave in case of exceptions etc.

2.4 Design Document

The Purpose of this Document is to convey the design of Peer to peer chat application using several modelling diagrams namely Class Diagrams, Sequence Diagrams, E-R Diagrams, Activity Diagrams and Use Cases taking into consideration the SRS of the system.

For creating a Design document,we had to use lot of UML diagrams.The Unified Modeling Language (UML) is a standard visual modeling language intended to be used for modeling business and similar processes,analysis, design, and implementation of software-based systems.

StarUML is an open source software modeling tool that supports UML (Unified Modeling Language). It is based on UML version 1.4, provides eleven different types of diagram and it accepts UML 2.0 notation. It actively supports the MDA (Model Driven Architecture) approach by supporting the UML profile concept and allowing to generate code for multiple languages.

2.5 UI/UX Mockups And Prototyping

To aid with the user interface and to promote a wholesome user experience, mock-ups were designed. UX Mockups allow us to quickly design inferfaces without having to worry about the actual implementation.We did our UX design with the help of the tool WireframeSketcher. WireframeSketcher is a wireframing tool that helps designers, developers quickly create wireframes, mockups and prototypes for desktop, web and mobile applications. Using it,we built a number of UI screens and later merged them for a smooth and convenient user experience.

Chapter 3

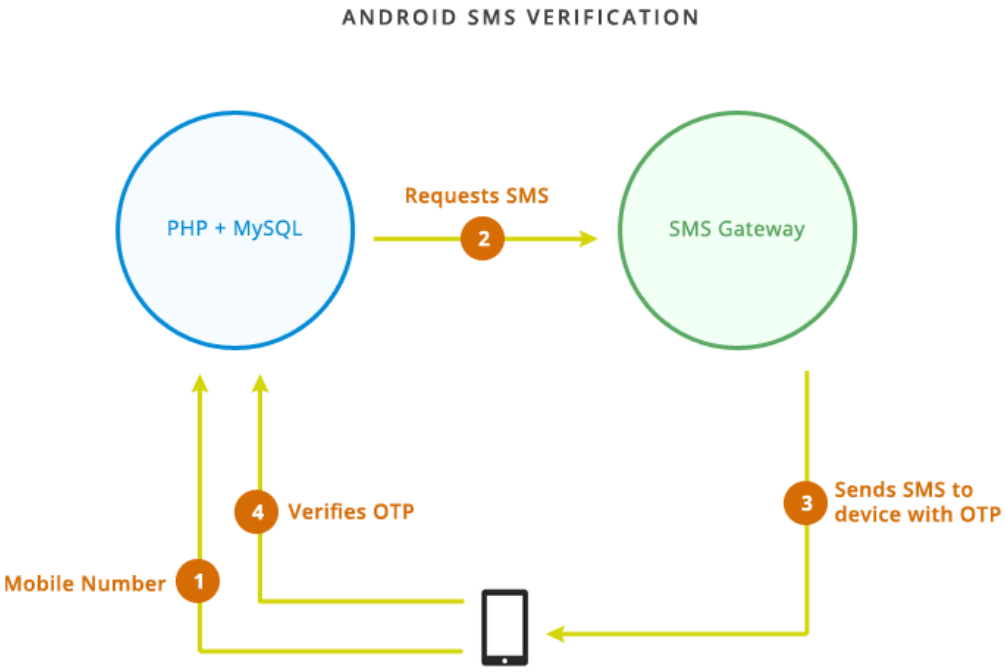
Implementation

Once the design phase was completed in the first semester of the project duration,we proceeded to implement the application in the current semester. The team, in accordance with the course objectives, made a decision to use the latest tools to aid us in development. Android Studio 2.0 was the preferred IDE, with the team using the latest SDK (23) for compilation. The app was continually tested and debugged on phones running on the latest commercially available iteration of the Android OS, Android 6.0 or Marshmallow.

The following sections will bring to light the implementation details and key challenges that the we had to face during the implementation.

3.1 Identity Verification

We implemented a SMS-based identity confirmation to verify the mobile numbers in order to get genuine users and avoid frauds. Below is the simple architecture diagram that explains the process of complete registration which involves the PHP, MySQL server and a SMS gateway.



1. First user mobile number will be sent to our server(hosted in athena.nitc.ac.in) where new user row will be created.A table is maintained in server which contains the name,number,email,online status,ip of registering users.
2. Our server requests the SMS gateway(msg91.com) for an sms to the mobile number with a verification code.
3. SMS gateway sends an SMS to the user device with the 6 digit verification code.
4. The verification code is entered by the user and will be sent back our server again for verification. Our server verifies it and activates the user and leads the user to the homescreen of the app.

3.2 Material Design and Swipe views with tabs

Material Design is a design principle that Google introduced along with Android OS 5.0, commonly known as Lollipop. Material Design is characterized by a more liberal use of grid-based layouts, responsive animations and transitions, padding, and depth effects such as lighting and shadows. We strived to make our app adhere to the design language, and a key component that was included in the app as a result of this decision is the Tabs.

Tabs make it easy to explore and switch between different views or functional aspects of an app or to browse categorized data sets. To navigate between fixed tabs, touch the tab or swipe the content area left or right.In our app,We are using it to provide lateral navigation between three tabs namely Online, Chats and Contacts.

Another interesting component that our app incorporates is the Swipe to refresh option. Swipe to refresh is a swipe gesture available at the beginning of lists, grid lists, and card collections where the most recent content appears .This option provides a clean and hassle free interface for user to manually sync from the list. It is used to refresh the

Online tab in our app which when clicked accesses the central DB and refreshes it's list of online registered users.

3.3 Implementation of Tabs

Online Tab is implemented in onlineFragment.java class.Online Tab is implemented in onlineFragment.java class.It's LoadonlineUsers function takes each entry from central db as jsonStrings and then matches it with contacts in the phone to display the list of online users at the end.

Recent Tab is implemented in RecentFragment.java class.It's onCreateview adds any newly started conversation to ListView adapter in class so that It is displayed on the Recent Tab as soon as someone starts any new conversation.The dbHandler function getAllConv fetches all old conversation from sqlite db of the phone and shows it on recent tab.

Contacts Tab is implemented in ContactsFragment.java class.It takes contacts list from the user's phone and add it to ListView adapter.The advantage of using this is It doesn't load all contacts at once which can severely lag the device instead it loads contacts only when required to show i.e. only when it is scrolled down.

3.4 Simple Chat between two peers

Simple chat between two peers occurs in Chatscreen.java class.As soon as user type something in text area and clicks on send button, a chat message object is created which bundles all the information like content of Message,sender no,receiver no,type of message, deliveryStatus etc inside it.It then adds the message to the user's sqlite database and uses Network utility functions to create socket and send messages through it to recipient's ip.

3.5 Incognito chat

Incognito button is available in action bar at the top of chat screen.When a user A sends an Incog request to user B then it actually sends a message of type Incog to user B.When message is received at user B it checks for the type of message and if it is Incog then it prompts a dialog box asking permission to start incog chat.All incog conversations has the type of message set to incog and hence does not get added to the user's local database.A hashmap IncogModeUsers is maintained to store record of current Incog users.

To close the incog conversation,any of the user has to click incog button and the message type is set to ForceCloseIncog.In the sendReceiveMsg function whenever this type of message is encountered,the entry from Hashmap IncogModUsers is deleted and normal conversation resumes.

3.6 Group Chat

Group chat is implemented in the groupChat.java class.There is a button available in the action bar to form a new group in the app.It asks for Name of the group and ticking the checkbox corresponding to contacts entry adds the contact to the group.When the group creation is successful,the onCreate function adds the group conversation to all member's recent tab. When any member of the group sends any message, then SetGroupForAllMem functiom iterates through group members list and send the messages to every member.

3.7 Encryption

For the end to end encryption of messages during the incognito mode,We decided upon an AES based encryption algorithm.The Advanced Encryption Standard (AES) is a specification for the encryption of electronic data established by the United States National Institute of Standards and Technology in 2001.It is based on the Rijndael cipher developed by two Belgian cryptographers, Joan Daemen and Vincent Rijmen, who submitted a proposal to NIST during the AES selection process. Rijndael is a family of ciphers with different key and block sizes.AES has been adopted by the U.S. government and is now used worldwide. It supersedes the Data Encryption Standard (DES), which was published in 1977. The algorithm described by AES is a symmetric-key algorithm, meaning the same key is used for both encrypting and decrypting the data.For AES, NIST selected three members of the Rijndael family, each with a block size of 128 bits, but three different key lengths: 128, 192 and 256 bits.

In our app,We are using 256-bit AES key.but instead of sending the keys to other party,we are sending a password to other party that can be used to generate key.When the user initiates the Incog request,we are sending a password along with the request.Now here we are are using a random alpha numeric string generator to generate password.

3.8 Challenges faced during implementation

Unfortunately, the team did face a few challenges during the implementation phase of the project.Few things couldn't be done as we planned to do.One such thing was implementing the back-end of the app.We earlier decided upon to use libraries like PeerJS and Sip2peer in our app.but due to it's inherently complex code and lack of resources and tutorials for the above,We opted to go for Socket programming. Another issue that the team faced was during the object creation of DbHandler Class.It was unable to get the correct context for creating object. This led to an unstable user environment replete with force crashes and high memory usage.To solve this problem, We then decided to make the DbHandler class static which meant same object was shared instead of creating new object.

Few of the features that were decided upon during the design phase did not make it to the final implementation of the application. One such feature being sending of pictures as messages.We realized that implementation of this will involve much more amount of work and hopefully future version will be able to achieve it.There are few glitches in Group chat and in some occasions behave unexpectedly.We will try to fix few of those things in later versions.

Chapter 4

Conclusions

Our team started out with the humble objective of learning android application development through a project over the course of the year. We also aimed to implement a secured and safe chat application to prevent hacks into people's conversation, and at this juncture, we believe that we have been able to successfully do so. Over the duration of the project, we believe that each of us has learnt to work together as a team in an organized, systematic manner. The project also provided us with an opportunity to apply theoretical knowledge into solving real world solutions, with the help of modern tools and technologies.

4.1 Possible future work

There are few areas and possible works that can be done to improve this app and provide a better user experience.Addition of few features like sharing video files,audio files,documents etc will make it a wholesome product.

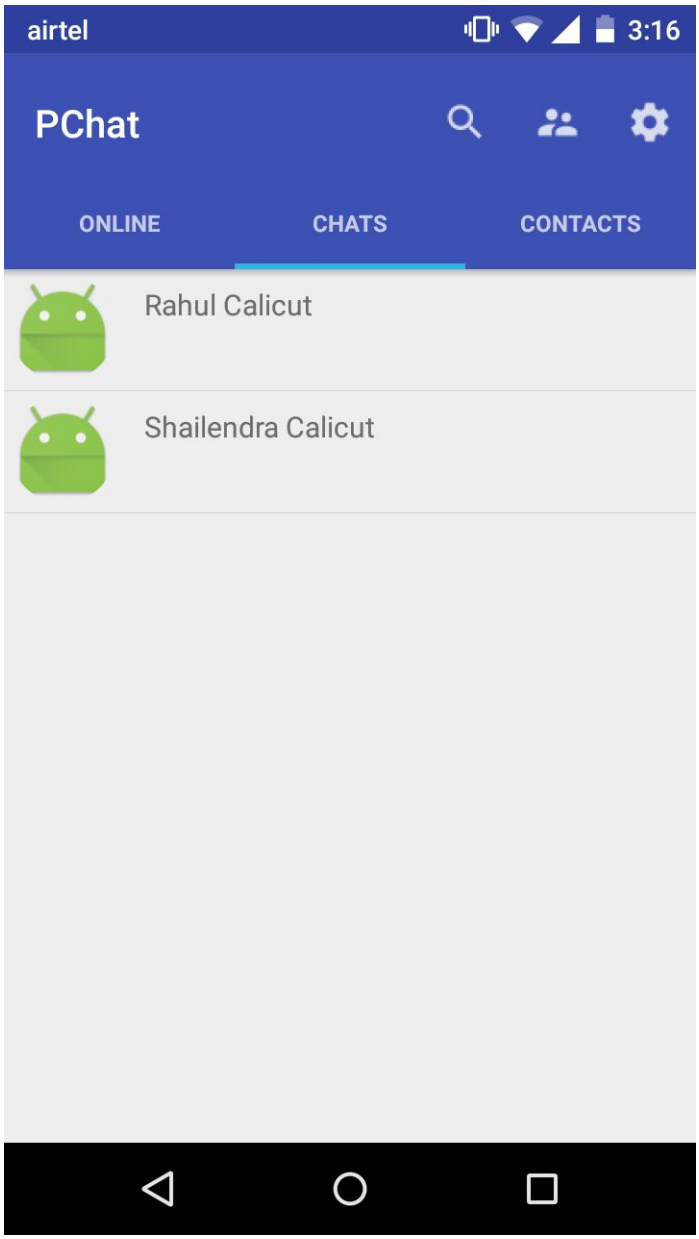


Figure 4.1: Home Screen

Bibliography

[1] *Peer-to-Peer Application Development: Cracking the Code*, Published by Hungry Minds,2002.

[2] A Secure Chat Application Based on Pure Peer-to-Peer Architecture,Mohamad Afendee Mohamed,Abdullah Muhammed and Mustafa Man Accepted: 28-05-2015

[3] IEEE Recommended Practice for Software Requirements Specifications,Approved 25 June 1998

[4] Android Application Development - thenewboston Videos

[5] Android Documentation , www.developer.android.com