Project Phase II : P2P implementation
ChatBuzz - V.2.0

Introduction

The chat application uses the peer to peer architecture for the chat functionality. We have developed the application and tested locally on the XAMPP local server. Each user system acts as a client as well as a server. We use PHP to develop different functionalities. SQL is used to maintain login credentials and IP addresses of the users. JavaScript and AJAX are used to implement the peer to peer architecture and most of the backend chat functionalities are handled here.

Functionality

When user enters username and password to register for the chat application, the credentials are encrypted using MD5 encryption method. When the user logs in, the IP address is captured and the login status is set i.e the flag is set to 1. The few additional features that have been added over the previous version are –

- Encryption of credentials
- Logout option before the user can get into the chat-room
- P2P architecture over client-server architectures

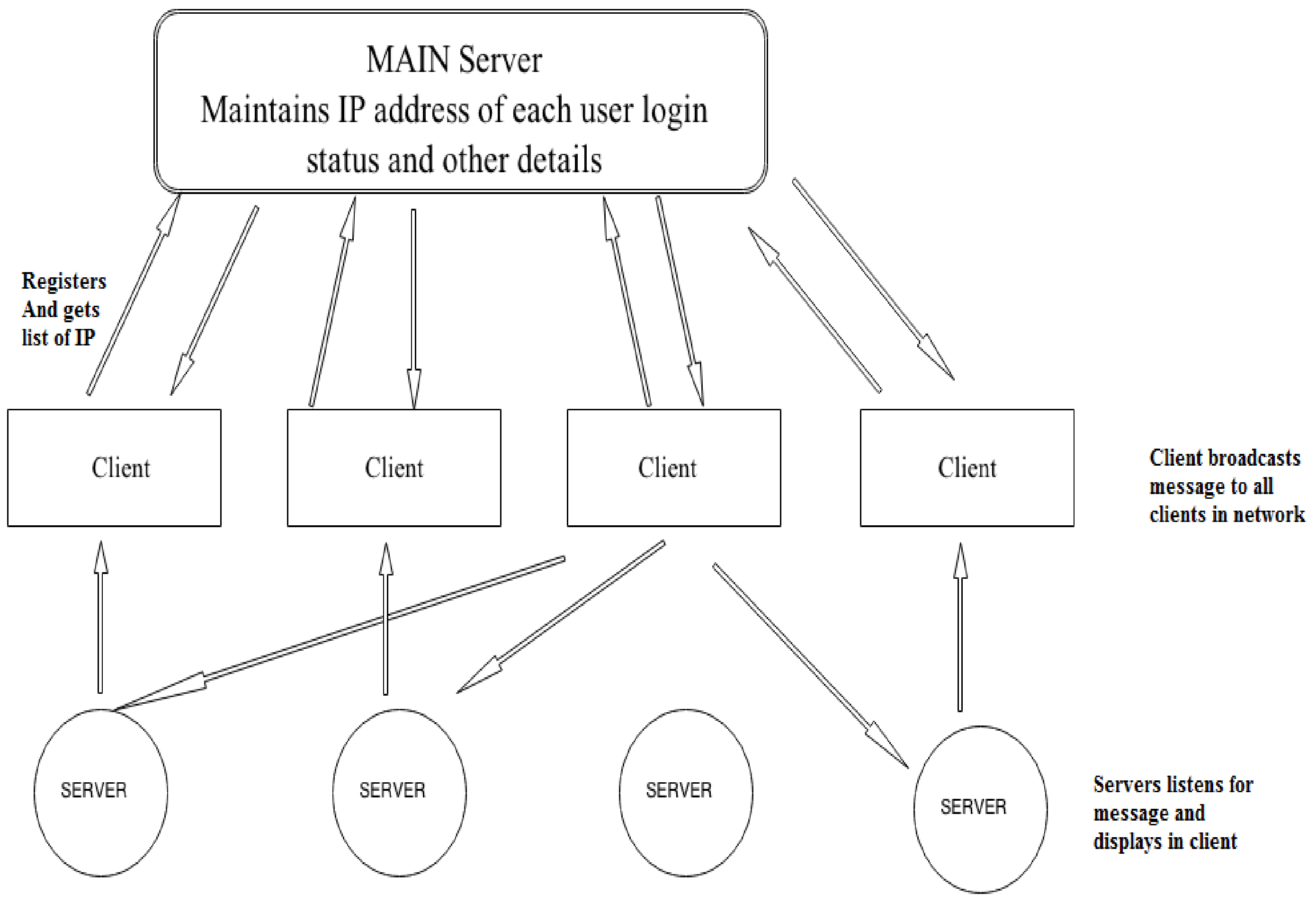
MAIN Server
Maintains IP address of each user login status and other details
Registers And gets list of IP
Client
Client
Client
Client
Client broadcasts message to all clients in network
SERVER
SERVER
SERVER
SERVER
Servers listens for message and displays in client

Figure 1

In the above diagram we show how the application works,

- When the users logs in the credentials is verified from the SQL database.
- The IP address of each user is captured and stored and the login status is set.
- The user gets the list of IP address of all the online users.
- Each user behaves as a client and server in a local network, when a user sends a message, the message is broadcasted to all users in network.
- The server at each user is listening for a message, as and when a message arrives, it displays to the user in the chat room.

In Figure 2 we show what happens when the user in the network enters a chat message.

- The server waits and listens for the broadcast message from other users in the network.
- When a message is received at the server at one of the client from port 9999, it displays the message in the chat room of that user.
- This happens at each and every user's server in the network.

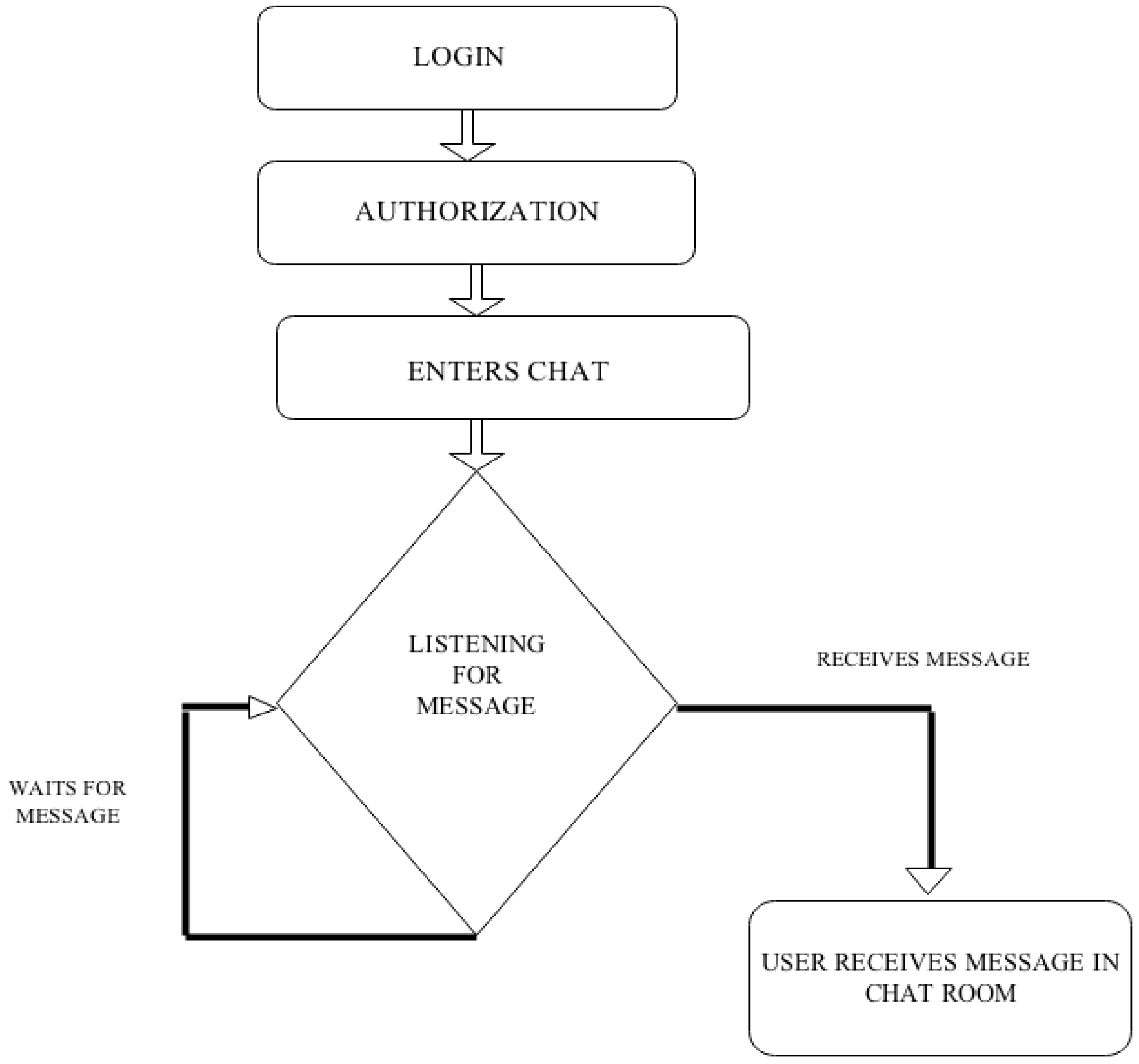

Figure 2

Since our application is a P2P the client plays even the server role here. Other users in the same network can login to the chat similarly and there would be a server running on each client. When a user (client) sends a message, this message is broadcasted to all servers in the network through the web sockets. The server at the client side listens for the broadcast messages, when received, displays that message in the chat room. The server at the client side just waits and listens for incoming message on port 9999 and no routing of message happens through authentication server. Currently the IP address of each user is manually set up in the application for the demo purpose. We could dynamically do this as well if we host it on a web server.

Further implementation

This application can be hosted as a fully fledged chat application on an online server. Here the authentication server will be used to store the user credentials to verify the user, and the corresponding IP address of user is captured and stored in the mySQL database. When a user logs in the login status of that user is set, ie flag is set to 1. The logged in user gets the list of IP address of all online users. When a user sends a message, the message is broadcasted to all online users in that chat room through the IP retrieved from the database. When the user logs out the login status of that user changes ie flag is set to 0 in database in the server. We see that each user is updated with fresh list of IP addresses of online users periodically. In this way each user can communicate in the chat room without the message going through the server and there would be lesser chance of having a single point of failure and we reduce the server over head as well.

<u>Configuring the application</u>

Step 1 : Install xampp software , start apache and Mysql servers in xampp control panel

Step 2 : Unzip the contents of zip file(chatbuzz folder) into /xampp/htdocs folder

Step 3 : Edit the file server.php and update your current local IP in there

Step 4 : Edit the file chatfiles/chatfuntions.js and change the IP there as well

Step 5 : Run your xampp database login credentials in the files ‘checklogin.php’, ‘register_newuser.php’, ‘logout_handle.php’ and ‘chatfiles/chat_form.php’

Step 6: Configure the backend database under ‘test’ database in phpmyadmin using the file in dbsetup folder called ‘members.sql’

Step 7 : Run the server.php file (This would run in a infinite loop so just let it run on a different tab)

Step 8 : You are all set! Run main_login.php to start with registering the user and using the application on a local network ie run localhost/chatbuzz/main_login.php in your browser.

Features

- Login Authentication
- Clear GUI
- On-line users' notification
- Chat room creation
- New user join/leave notification
- Chat sessions stored in a database
- Display users available in Chat room
- Users can join multiple chat rooms
- User can create two chat rooms
- Peer to Peer communication
- Each user can communicate with every other user without the messages actually being routed through the server

References

- http://tutorialzine.com/2010/10/ajax-web-chat-php-mysql/
-]http://forum.codecall.net/topic/58937-simple-chat-system-using-php-mysql-and-ajax
- www.coursesweb.net
- www.github.com
- www.stackoverflow.com
- http://www.sanwebe.com/2013/05/chat-using-websocket-php-socket

Team Members

Hithesh Krishnamurthy (1001096009)

Lavanya Somashekar (1001104262)

Girish Ramesh Babu (1001087481)

Sunayana Suresh Gowda (1001107621)