CS 6349 - NETWORK SECURITY

Instant Secure Messaging System

Project by:
Nevhetha Radhamani Shanmugam, NXR153230
Sai Snigdha Parvataneni, SSP152230
December 4, 2016

SUMMARY

Objective

To build a secure Internet Instant messaging system.

Goals

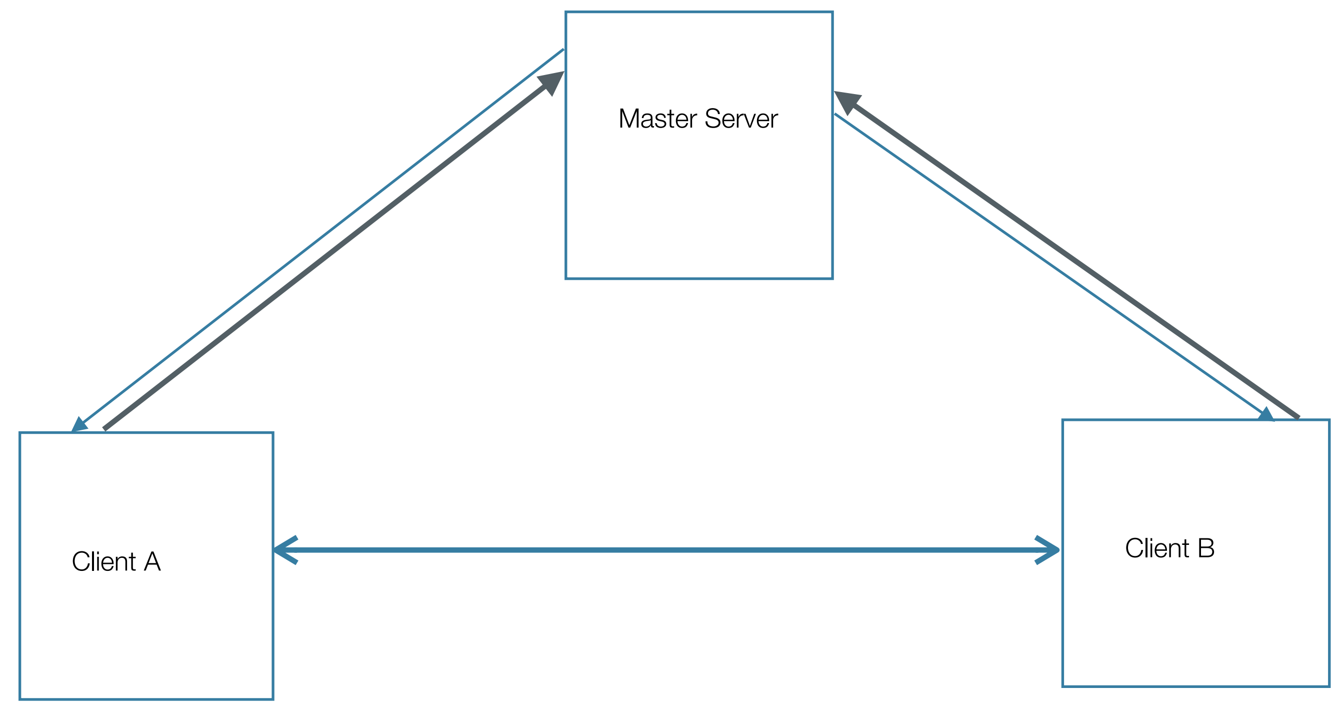
To develop an application in which ensures:

- Confidentiality
- Integrity
- Authentication

Project Outline

- It makes use of java socket programming to establish the connections over TCP
- It is a multi-client server architecture
- There is a database at the backend which stores the user credentials
- The server runs on specific port waiting clients
- Multiple clients can be spawned
- Clients maintain accounts with Server
- Each time client logs in, secret key is generated for Client-Master communication
- The server is involved only during session establishment between end users
- Data flow between the users is direct after the session is established
- Server has a public key/private key pair

LOW LEVEL VIEW



➞ Encrypted using Master's public key - RSA Encryption

→ Encrypted using Client's secret key - AES Encryption

↔ Encrypted using Session key generated by master - AES Encryption

Message Sequence for session establishment:

PRECONDITION: Both client's should be online.

1. Client A sends the session key request to Master encrypted with master's public key

   Emaster (Session Request to Client B + checksum)

2. Master decrypts the service request using his private key

   Decryption : Dmaster (Emaster (Session Request to Client B + checksum)

3. Master generates new session key <clientA><timestamp><clientB> , K_{AB} and sends it to client A encrypted with Client A's secret key shared with master. Master also generates a ticket which helps in authenticating client A to client B. Ticket is encrypted using client B's secret key shared with master.

   $K_{clientA}$ (K_{AB} + ticket to client B + checksum)

   Ticket - $K_{clientB}$ (Client A+ session key)

4. Client A decrypts the connection message using its secret key and gets the session key, K_{AB}. It then sends the ticket to client B.

5. Client B decrypts the ticket using its secret key and gets the session key, K_{AB}.

6. Client A and Client B start communicating using the session key, K_{AB}.

Security Requirements:

- **Confidentiality:**
  1. AES Encryption between clients using session key generated by Master server
  2. RSA Encryption for messages to Master server using master's public key
  3. AES Encryption for messages from Master to Client using Client's secret key

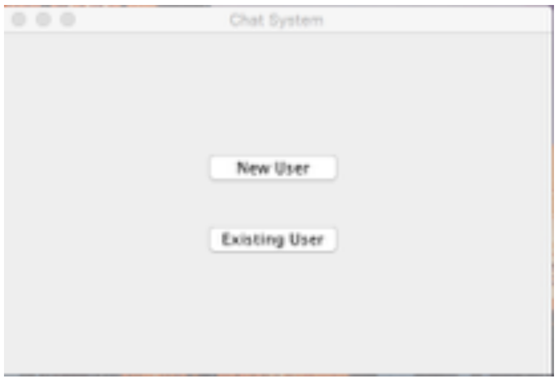
- **Integrity:**
  1. Every message is padded with first 16 characters taken from message digest of that message before encryption which serves as the checksum.
  2. Digest is calculated using SHA-256

- **Authentication:**
  1. Each client authenticates to the Master using password check
  2. Session Requesting client authenticates itself to the Session Receiving Client by giving the ticket produced by master (can be acquired only by authentic client's)
  3. Session Requesting Client authenticates the identity of Session Receiving Client when it receives the ticket for session establishment from master (if Session Receiving client is not genuine, master would not have sent the ticket)

SAMPLE EXECUTION-HIGH LEVEL VIEW

CLIENT HOME :

- New User - Navigates the client to the window that helps the client to create an account with the Master
- Existing User - Navigates the client to the window that helps the client to authenticate himself to the Master

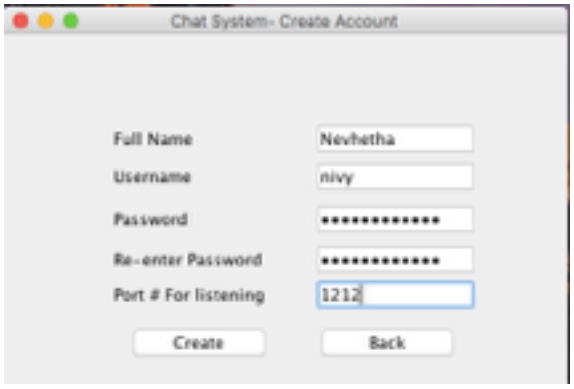


ACCOUNT CREATION:

- Create - Creates an account for the client with the Master using filled details.

    1. All fields are required
    2. Password must be 12 characters long containing at least one upper case letter, one lower case letter, one digit and one special character (other than '@' and ' '). Full name cannot have spaces.
    3. Client accounts cannot be duplicated
    4. 1024 < Port # < 65535

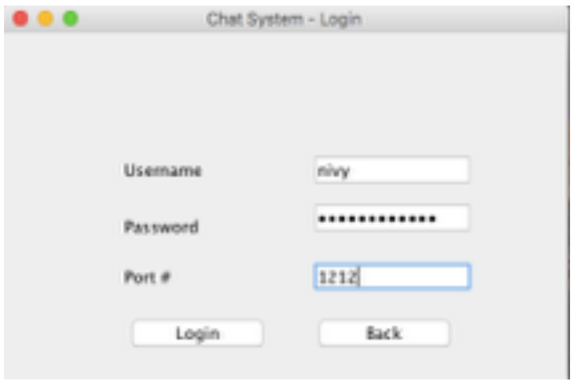
- Back - Navigates to Client home page



Full Name	Nevhetha
Username	nivy
Password	*************
Re-enter Password	*************
Port # For listening	1212

LOGGING INTO CHAT SYSTEM:

- Login - Authenticates client to Master and navigates to chat window on successful authentication
    - All fields are required
- Back - Navigates to client home page.

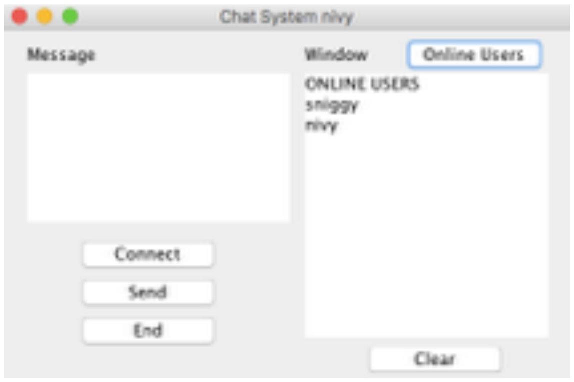


Username	nivy
Password	*************
Port #	1212

CHAT ROOM:

ONLINE USERS:

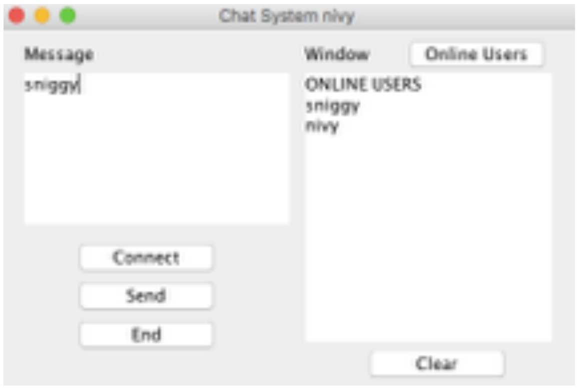
- Online Users - Lists all the users who are online
- Clear - Clears the Window



ONLINE USERS
sniggy
nivy

CONNECT TO OTHER CLIENT:

- Connect - Typing the Username to connect with and clicking the connect button facilitates a new session between the clients
    - Only one session at a time
    - Only one session per login

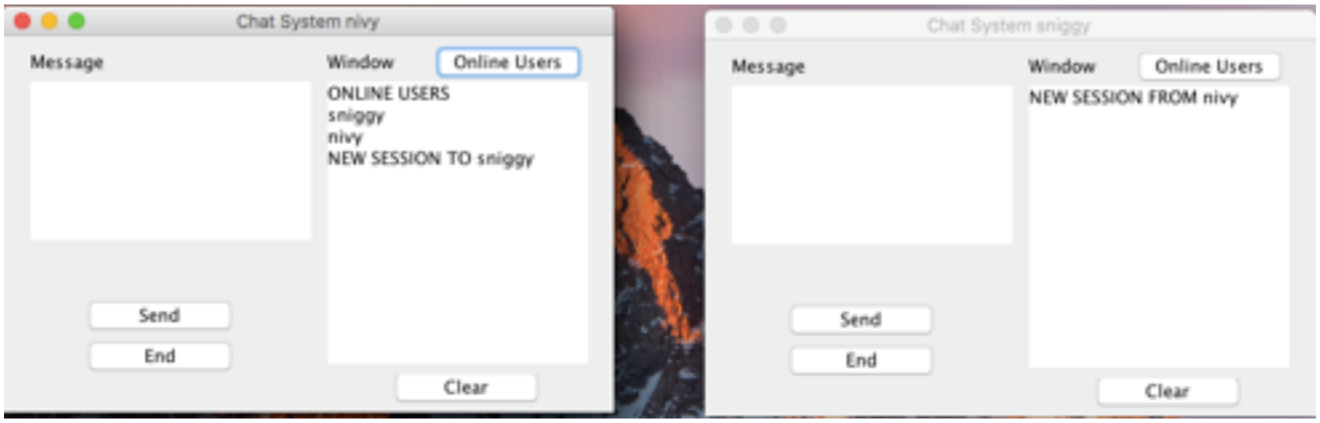


sniggy

ONLINE USERS
sniggy
nivy

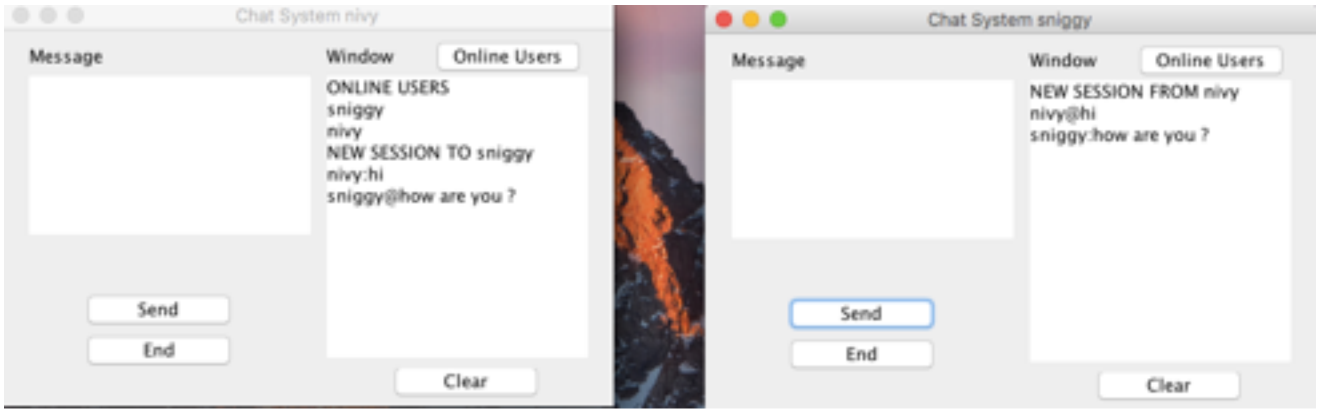
CHAT SESSION:

- Entering the message and clicking on send , sends the message to the other end of commuincation



ONLINE USERS
sniggy
nivy
NEW SESSION TO sniggy

NEW SESSION FROM nivy

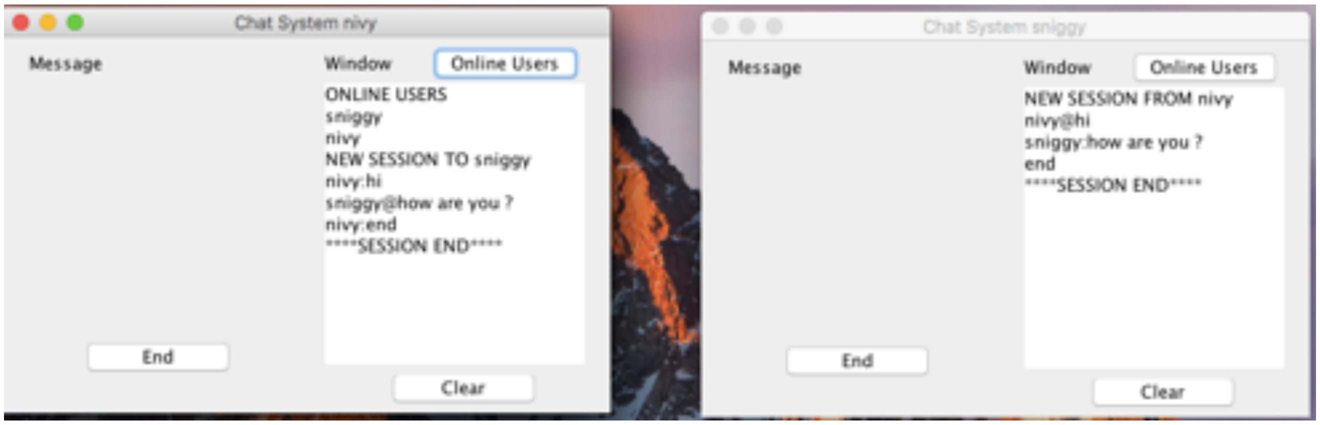


ONLINE USERS
sniggy
nivy
NEW SESSION TO sniggy
nivy:hi
sniggy@how are you ?

NEW SESSION FROM nivy
nivy@hi
sniggy:how are you ?

END SESSION:

- Typing ‘end’ in the message box and clicking on send, ends the session with the message box disabled
- Clicking on end button, ends the session and navigates to login page



ONLINE USERS
sniggy
nivy
NEW SESSION TO sniggy
nivy:hi
sniggy@how are you ?
nivy:end
****SESSION END****

NEW SESSION FROM nivy
nivy@hi
sniggy:how are you ?
end
****SESSION END****

NOTE:

- **The behavior of the system is arbitrary when**
  - Trying to connect to client who is already in a session
  - Using ‘@‘ in messages
  - Using space in full name during account creation

- **The system should be restarted if**
    - Using closed port for server