National Conference on Recent Innovations in Computer Science & Communication Engineering, *July 2016* *ISBN: 978-93-86005-02-1*

Encrypted Chat Messenger

Sadik Hussain*, Vikash Kumar, Manjeet Kumar, Suchana Naskar and Monika Singh

CSE, RCCIIT, Kolkata – 700015, West Bengal, India

Abstract

This project entitled as "Encrypted Chat Messenger" is used basically for chatting purpose with the remote client or user on a network. The word Encryption at the beginning denotes that the project uses techniques of encryption to overcome some security issues. The users or the clients are given user name and password, the user name is simply stored in a file and the password is encrypted and then stored. The encryption algorithm used for encrypting the password is hash algorithm. We have implemented End to End Encryption in our project. This is a special type of encryption in which when a client sends a message to another client, the message is automatically encrypted without the knowledge of the client then passes through the network and reaches the other client in encrypted form. At the other client side the message is decrypted automatically.

Keywords: Chat, Client, Encryption, End to End Encryption, Diffie-Hellman Key, Server

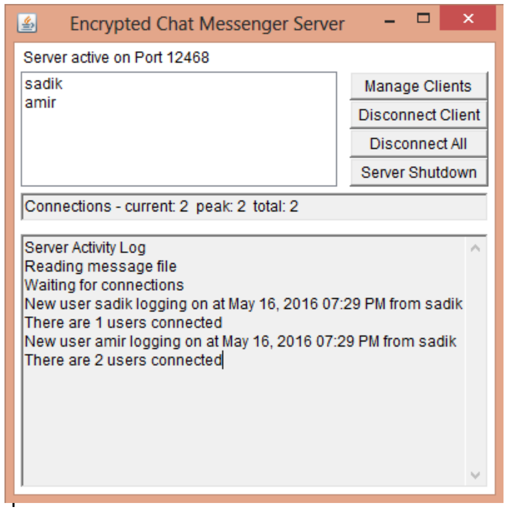
1. Introduction

Since the earliest days of computing, people have used computer systems to enable communication. Simple command-line programs such as write and finger have existed on the earliest time-sharing systems, while computer Bulletin Board Systems (BBS) were flourishing shortly after the introduction of personal computers in the mid-1970s[1].

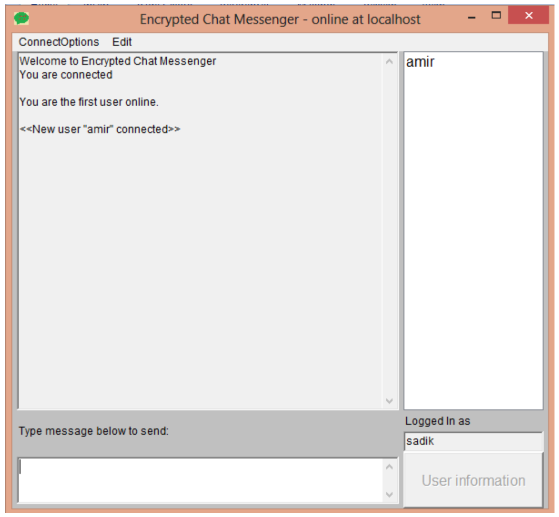
As a communication medium, chat is not without its problems. The conversations in chat tend to be hard to follow. Within a particular room or channel, there are often multiple conversations or "threads" going on at the same time, and a single user often participates in multiple threads at the same time. As a result, chat conversations tend to be confusing and disjointed, with a lot of repetition and corrections.

Java is one of the languages that has been used to create chat application for many devices like Phones, PCs, Wireless chat application etcetera, Java RMI package is used for most of the client/server applications. It provides many tools to create sockets and to implement UDC or TCP protocols[3] for the creation of chat applications. In Java a Client/Server combination is used to chat with remote users. When a client wants to chat with a user on a remote host it sends request to the server with identification. Below is the screen shot of a Server window.

**Author for correspondence*



```
Server active on Port 12468
sadik
amir
Connections - current: 2  peak: 2  total: 2
Server Activity Log
Reading message file
Waiting for connections
New user sadik logging on at May 16, 2016 07:29 PM from sadik
There are 1 users connected
New user amir logging on at May 16, 2016 07:29 PM from sadik
There are 2 users connected
```


Encrypted Chat Messenger - online at localhost
ConnectOptions
Edit
Type message below to send:
Logged In as
User information

```
Welcome to Encrypted Chat Messenger
You are connected

You are the first user online.

<<New user "amir" connected>>
```

```
amir
Logged In as
sadik
```

The security of the system is one of the most important concerns. Attackers can damage the whole system and they may also be able to read the messages of the clients if the security of the system is not good enough. Another concern is the performance of the system. How fast the messages are reached from one user to another and how fast the user is connected to the server. The Client Window looks like.

2. Previous Work

Many different types of chatting applications have been created in past years and all are having huge differences in characteristics as well as in their own features. Media Streaming is implemented and many tools are derived to monitor the chats as well. Few works are mentioned below:

- Chat Client Monitoring: In this System, we will be able to monitor the status of the current user such as offline, online, away and sleep. The system will also detect a type of word that is used by Chat Client and keep that word into the chat record[7].
- Word filtering: The system will be able to detect some certain words and filter it to a new one referring to word database.
- Instant messaging: A user will be able to send the Instant chat message to all users of chat client and the chat client can reply it to the user at a real time and if user will use any blocked words then a warning will be generated.
- Blocking and activate user: Blocking and activating a user is an ability for the system to control the user to use the chat client. The decision to block or activate a user is depending on the user behaviour.
- Media Streaming: Many Chatting applications also provide the ability for streaming media such as audios, videos etcetera[7].

3. Factors Affecting Chat System

There are many factors that impact the efficiency of chat. Here we list several of them and show how they interrelate to each other.

- Lack of recognition: Users find it difficult to associate a nickname with what a user has said in the past if they do not already know the speaker.
- Typing inefficiency: Most people type much slower than they talk, which results in a much slower rate of communication.
- Lack of status information: There is often no way to tell if a user is actively participating in a conversation or has wandered off from the computer all together. This often results in out of order turns, where someone asks a question again in a different ways.
- Lack of context: There is no way to tell what has gone on before in a conversation or what people are talking about now. This is true in a normal spoken conversation as well but is a particular problem in chat because of the number of people constantly entering the conversations.
- High signal-to-noise ratio: since most of the participants in chat do not know each other, there is a great deal of introductory socialization phrases in any chat conversation. In addition, a large percentage of chat turns are corrections, clarifications, or repeats of previous turns.
- General uselessness of the chat history: Although most chat interfaces save the entire history of a particular session, it is rarely used in the chat context. For example, once an answer to a question has scrolled off the screen, users are far more likely to ask the question again rather than scroll back into the history[9].

There are basically two types of attacks on a network:

3.1 Passive Attacks:

These are the types of attacks where in the attacker indulges in eavesdropping or monitoring of data transmission. The following are examples:

- Release of Message contents: In this type of attack the attacker gains the message sent by any user and so the messages are released besides the users wish.
- Traffic Analysis: A passive attacker could also analyze the packets flowing through the network and try to figure out the relation among the packets.

3.2 Active Attacks:

The active attacks are based on the modification of the original messages in some manner or the creation of false messages. For example:

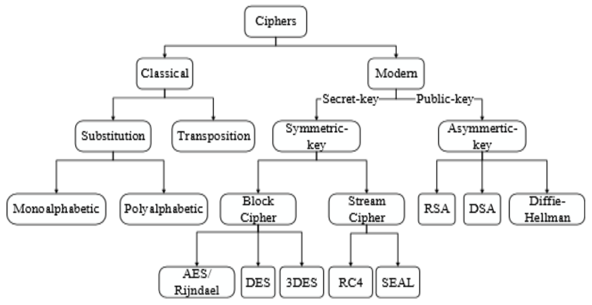
- Masquerading: It is caused when an unauthorized entity pretends to be another entity.
- Modification: In modification two types of attacks can be done.
- Replay attacks
- Alterations
- Denial of Service: DOS attacks make an attempt to prevent legitimate users from accessing some services, which they are eligible for[2,4].

4. Proposed Methods

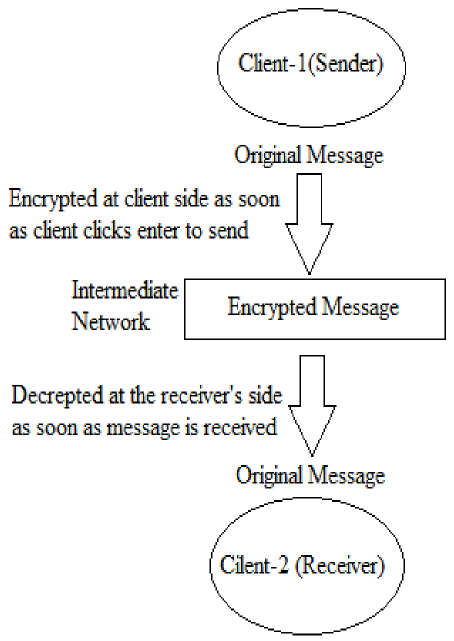
End to End Encryption: In this paper we had implemented End to End encryption method. Encryption is the process of scrambling a message so that only the intended recipient can read it. Encryption can provide a means of securing information. As more and more information is stored on computers or communicated via computers.

There are many different types of encryptions. Below is a chart representing various types of encryption techniques used for encoding messages.

In our project, we have used encryption in a way such that the messages sent by one client to others are automatically encrypted at the



Classification of Encryption Methods


Client-1(Sender)
Original Message
Encrypted at client side as soon as client clicks enter to send
Intermediate Network
Encrypted Message
Decrepted at the receiver's side as soon as message is received
Original Message
Cilent-2 (Receiver)

client side and similarly decrypted at the receiver client side. This type of encryption is called End to End Encryption. We have a pictorial representation of End to End encryption.

We see that the messages passing through the intermediate network between two clients is in encrypted form and thus any third party device cannot understand the message. Since the client needs not to do anything for the process of encryption or decryption, the End to End encryption is completely transparent from clients. Hence our system is a transparent system.

The End to End encryption is very powerful since the attacker after getting the message packets will not be able to understand the messages and the following attacks will be useless on the chatting.

Here we show a piece of code that is used for encrypting messages and then we will describe the attacks that are prevented using this.

```
// Create key and cipher
Key aesKey = new SecretKeySpec
(key.getBytes(), "AES");
Cipher cipher = Cipher.getInstance("AES");


// encrypt the text
cipher.init(Cipher.ENCRYPT_MODE,
aesKey);
byte[] encrypted = cipher.doFinal
(original.getBytes());

StringBuilder sb = new StringBuilder();
for (byte b: encrypted) {
    sb.append((char)b);
}

// the encrypted String
enc = sb.toString();
return enc;
```

Here the key is created using Diffie-Hellman key agreement algorithm (The detailed explanation is given below). After the creation of key the text message that has been typed by the client is converted to the array of bytes and then AES Encryption algorithm (Detail of AES is stated below) is applied. After that the bytes are transmitted over the network. The above process provides the following advantages to the Chatting System.

- Release of message contents: This attack takes place when any client sends a message to another client and the attacker intercepts the message in between the two clients. In this attack the attacker tries to get the message bytes and arrange them to get the actual message. Since the attacker will be able to read the message, so the message contents will be released. Since End to End encryption encrypts a message as soon as it leaves a client so the attacker will not be able to get the contents of any messages and hence this attack is prevented.
- Traffic Analysis: A passive attack may take place on a network such that the attacker will keep quiet and analyze the traffic of the network. This attack is very tough to do since the attacker must be able to get the access of router of any other device of the network.
- The traffic analysis is used to understand the patterns in the messages. As we are exchanging the key of encryption through Diffie-Hellman algorithm, there is no way to get the keys and hence the messages are completely secure from this attack. Since the packets are encrypted and the keys are available only to the sender and receiver, the attacker will not be able to understand the traffic going through network.
- Packet Sniffing: In this type of attack, a passive attacker keeps quiet at any portion of a network and captures the packets passing through his device analyzes it. The attacker also tries to understand the messages kept in those packets but in case of end to end encryption the attacker cannot understand the messages so the attack will be useless.

Regarding the performance issues of the network, the following points must be noted.

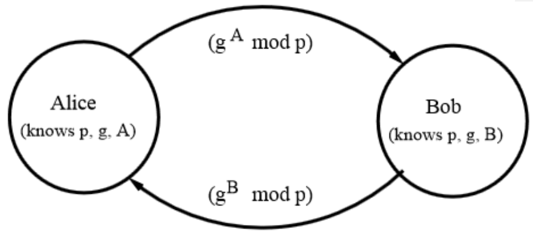
5. Key Agreement Methodology

Suppose Alice and Bob are two parties that want to use the same encryption key for implementing AES encryption algorithm.

The Steps in Diffie-Hellman algorithm are:

- Alice and Bob agree on a prime number p and a base g.
- Alice chooses a secret number a, and sends Bob (g^a mod p).
- Bob chooses a secret number b, and sends Alice (g^b mod p).
- Alice finds ((g^b mod p)^a mod p).
- Bob computes ((g^a mod p)^b mod p).
- For example let us take the following steps
- Alice and Bob agree on p = 23 and g = 5.
- Alice chooses a = 6 and sends 5^6 mod 23 = 8.
- Bob chooses b = 15 and sends 5^{15} mod 23 = 19.
- Alice computes 19^6 mod 23 = 2.
- Bob computes 8^{15} mod 23 = 2.

Then 2 is the shared secret.



Diffie-Hellman Security: The numbers obtained in step-4 and step-5 are both same and those are the keys that are used for encryption.

Suppose p is a prime of around 300 digits, and a, b at least 100 digits each.

Discovering the shared secret given g, p, g^a mod p and g^b mod p would take longer than the lifetime of the universe, using the best known algorithm. This is called the discrete logarithm problem.

The key agreement takes some time and hence if we do the key exchange for each message then the performance of the system will be lowered by a noticeable amount.

To solve this performance issue we will exchange the key for just one time and all the rest of messages will be encrypted by that key till the client sends the last message and disconnects from the server.

- The connection between the nodes must be very good. That is the network must be reliable for better chatting experience.
- Authentication of the user must be very fast. The Server must be able to authenticate a user faster. We have applied searching between log file of the Server so that it can authenticate a user faster.
- The messages between two clients must be routed fast so that messaging takes place faster. A better routing protocol is needed for better chatting.

Two types of encryption algorithms are used in our project. The first one is used to encrypt the password of a user at the Server side. Since any attacker after getting access to the Server may easily understand the password of each user so to prevent this threat the password is stored in encrypted. Some detail about this encryption algorithm is as follows.

6. Hash Algorithm

Hash algorithms are used as components by other cryptographic algorithms and processes to provide information security services. Hash functions are often utilized with digital signature algorithms, keyed-hash message authentication codes, key derivation functions and random number generators. A hash algorithm converts a variable length message into a condensed representation of the electronic data in the message. This representation or message digest, can then be used for digital signatures, message authentication and other secure applications[8].

In java.security package there is a built in method to implement hash algorithm. The method getByte() is used to get the bytes of a String type password and then the bytes are digested by another function and stored in byte array. After that the byte message is encrypted using ISO-8859-1. In this way the password is encrypted and then it is stored in the log file[6].

7. Advanced Encryption Standard (AES) with Diffie-Hellman Key Agreement

To encrypt messages sent by one client to another we have used AES encryption algorithm. Since AES is a symmetric key cryptographic algorithm, both the clients must know the encryption key. As any attacker can access the encryption key and easily decrypt messages, so to solve this problem we have used Diffie-Hellman key agreement algorithm. To understand more about them here is some detail about them.

AES algorithm can support any combination of data (128 bits) and key length of 128, 192 and 256 bits. The algorithm is referred to as AES-128, AES-192 or AES-256, depending on the key length. During encryption-decryption process, AES system goes through 10 rounds for I28-bit keys, 12 rounds for I92-bit keys and 14 rounds for 256-bit keys in order to deliver final cipher-text or to retrieve the original plaintext. We have used the methods provided by java.security package to implement AES algorithm in our project.

The question of key agreement was one of the first problems addressed by a cryptographic protocol. This was prior to the invention of public key cryptography. The Diffie-Hellman key agreement protocol (1976) was the first practical method for establishing a shared secret over an unsecured communication channel. The point is to agree on a key that two parties can use for a symmetric encryption, in such a way that an eavesdropper cannot obtain the key. There are many packages in Java that are used to implement the Key agreement algorithm like java.security, java.security.spec, javax.crypto, com.sun.crypto.provider. SunJCE etc[6]

We can write the following points about Diffie-Hellman algorithm.

- How can two parties agree on a secret value when all of their messages might be overheard by an eavesdropper? The answer is the Diffie-Hellman algorithm.
- It accomplishes the above problem and is still widely used.
- With sufficiently large inputs, Diffie-Hellman is very secure and almost uncrackable[11].

Limitation of AES (Rijndael): AES(R) has no serious weakness; although it was observed that a mathematical property (not an attack) of the cipher might be vulnerable into an attack. Further in AES (Rijndael) the inverse cipher implementation is inappropriate on a smart card than the cipher itself[10].

8. Conclusion and Future Work

This paper presents a study of the factors that affect the efficiency as well as security of chatting system. As the use of internet and network is growing rapidly, there are more requirements to secure the data transmission. To sum up the End to End Encryption is very useful technique to overcome many attacks on the chatting system. According to research done and literature survey it can be found that AES algorithm is most efficient in terms of speed, time, throughput and avalanche-effect and Diffie-Hellman key agreement algorithm ensures the secure transfer of the Encryption Key. Our future work for this project will explore the following things.

- Add a database to store more details of a client.
- Building a registration mechanism for the clients.
- Image and File sharing.
- Make the project work over the Internet.
- Make the project platform independent.

9. References

1. Available from: en.wikipedia.org
2. Kahate A. Cryptography and Network Security. 3rd ed.
3. Forouzan BA. Data Communications and networking. 4th ed. McGraw-Hill.
4. Stallings W. Cryptography and network Security: Principles and Practice. 5th Ed. Pearson Education/Prentice Hall. Available from: www.google.co.in
5. Available from: www.java2s.com
6. Virtual Worlds Group. Microsoft Research One Microsoft Way; Redmond, WA. 98052.
7. Radack S. Ed. Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology.
8. Research papers on A Study of Encryption Algorithm by Gurpreet Singh and Supriya.
9. Symmetric Algorithm Survey: A Comparative Analysis by Mansoor Ebrahim, Sujhaat Khan, Umer bin Khalid.
10. Foundation of Computer Security (Dr. Bill Young, Department of Computer security Texas Austin).