Cipher Talk

Secure message exchange in instant messenger using Rijndael

2001807
Sang won Lee
{swlee@icu.ac.kr}

1. Introduction

In this paper, we describe the cipher talk. Cipher talk is that we exchange message between each another safely in instant messenger or chatting program. I will concern about exchange message in instant message. Therefore I will implement some small instant messenger equipped cipher module.

Instant messenger is program that can exchange information to each another internet user simultaneously. Because instant Messengers have grown from the status of a toy to that of an important corporate communication tool on the Web, The security problem of messenger is important active topic. Before some month, ICQ which have many users in the world is hacked. Nevertheless so far most instant messenger does not have any security module like crypto function except some messenger. Lately messenger has security module is appeared. MSN contains security software licensed from RSA Data Security Inc. and some messenger use SET or SSL protocol for security. But the number of that messenger is rare as before.

In Korea, until now security problem like messenger hacking is not occurred. But professionals point out that messenger used by most internet user is not safe. Because using messenger is spread to business market.

2. Previous work

2.1 AES

The Advanced Encryption Standard (AES) will be a new Federal Information Processing Standard (FIPS) Publication that will specify a cryptographic algorithm for use by U.S. Government organizations to protect sensitive (unclassified) information. NIST also anticipates that the AES will be widely used on a voluntary basis by organizations, institutions, and individuals outside of the U.S. Government - and outside of the United States - in some

cases.

NIST has selected Rijndael as the proposed AES algorithm. The algorithm's developers have suggested the following pronunciation alternatives: "Reign Dahl", "Rain Doll", and "Rhine Dahl".

2.2 Instant Message

Instant messaging (IM) services today are one of the fastest growing web-based technologies in the world. It started off as a toy to facilitate one to one chat on the Web, and today it is also being used by large corporate for internal or across the globe discussions and conversations. From simple chat, it has also grown to enable conferencing, file transfers, voice chat and with some, one can also take over the other user's browser and can lead him to rough a web journey. Many others also have a discussion board and a white board panel. According to a recent study conducted by Forrester's Research in 50 Fortune 500 companies, 46% of them said that they would use Instant Messaging services for intra-corporate discussions by year 2002. It was also recorded by the popular website http://www.download.com that ICQ, by far the -most popular IM tool in India, has been downloaded 100 million times. An estimated 1 billion instant messages travel across the globe everyday and 3 million new users register them with the various IM service providers every month.

All IM services are free. And there is a host of these to choose from. In a recent survey, AOL Instant Messenger (also known as AIM and downloadable from www.aim.com) is the most popular worldwide. It is used by most of the subscribers of AOL, the premier ISP in USA and several other countries, and is also a part of Netscape Communicator. In India, we had done a poll, and ICQ (get it from www.icq.com) emerged as the most popular. The other major players in this arena are MSN Messenger (install directly from messenger.msn.com. You can't download it) from Microsoft and Yahoo Messenger (messenger.yahoo.com) from Yahoo. In the Indian scenario, Rediff on the Net has recently announced the beta version of its IM - Rediff Messenger (you can download it from http://www.rediff.com/messenger/messenger.htm).

How does it work?

The instant in instant messaging is possible because every user sending and receiving the instant messages remain constantly connected to the IM service. Every IM service has its own IM client that the user has to download and install in his computer. The user then is

constantly connected to the server, whenever his computer is switched on or his client is active. The user also has to register himself with the service provider.

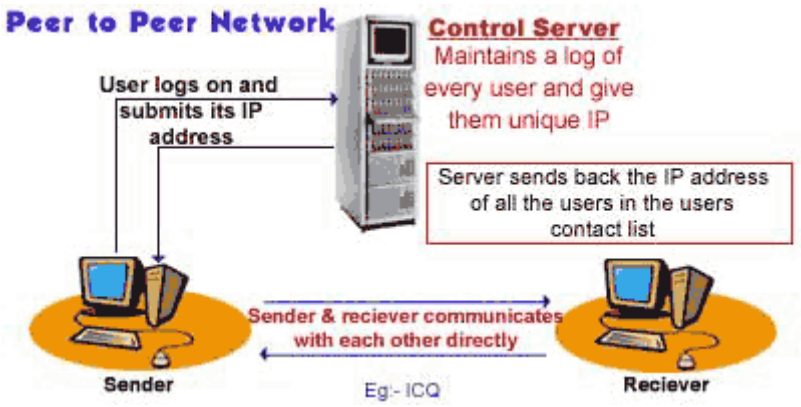
When you log into the IM service, the client in your computer lets a server at the service provider's end know that you are online and ready to receive messages. The server also records your IP address (the unique address of every computer on the Web). Now you are ready to send and receive messages. Sending an IM is just as simple as clicking on the name of the person in your contact list, typing out the message, and then clicking on the send button. The packet of data sent contains the address information of the recipient, the data and an address header identifying you to the recipient.

The IM service network can be any one of the following three types - through server routing, direct communication between users or a combination of both.

Through server routing

In this sort of a setup, you are constantly connected to a network of servers at the service provider's end. These servers track all the connected users and record their unique address. So when you send a message, is sent, it goes to the server, which then routes it through its network and delivers it to the recipient. IM services like MSN Messenger uses this sort of a setup. Nothing wrong with this, but just that file transfers can take slightly long to reach the recipient.

Direct Communication



Let's now look at the other setup. Here, the also is a network of servers. But this server just maintains a log of the various persons who are online. Now, when you log on to the service, the service sends you the IP addresses of the other users marked in your contact list. So when you send the instant message, you send it directly to the recipient, and not through the server. You won't see much of a difference in the speed of transfer of the messages, but file transfer speeds are truly amazing.

Instant messengers like ICQ use this sort of a setup.

Combination setup

The third type of IM setups uses a combination of both. The server records a log of the users, as usual. So when you send an IM, the message routes through the network server and reaches the recipient. But if you are transferring a file, it goes to him directly. IM services like AOL Instant Messenger uses this setup.

IM in corporate

As we told above, IMs are now being used more and more by various corporate and organizations for internal and multi-location communications. And today everyone is paranoid about secrecy, privacy and security on the Net. They might not be comfortable with the thought that a confidential piece of information is being routed through the servers of an external body. This gave rise to a whole breed of enterprise editions of several of these IM servers. Most popular of these are ICQ Groupware server, Wired Red e/pop, NetLert Enterprise and Lotus SameTime. These servers can be deployed and managed at corporate level for more control and security.

Interoperability and standards

IMs today have no standards governing them. They are not interoperable and you can send instant messages only to those who have registered with the same service provider. Different

service providers have different policies on these. MSN tried to make its messenger compatible with AIM, but had to back out after vehement protests from AOL. Till new standards are brought in, interoperability between the various IM services will not be possible. And to make this possible, a group of companies have come together to form a body called IMPPWG (Instant Messaging and Presence Protocol Working Group) and are backed by various biggies in the industry like Microsoft and Lotus.

2.3 Cryptography in Java

Java security software is consisted two parties. One is JDK include crypto class for authentication and the other is JCE supported powerful security. Java security API is set of packages used in security program making. Especially, this follow is security API package.

- javax.security
- javax.security.sert
- javax.security.interfaces
- javax.security.spec
- javax.crypto
- javax.crypto.interfaces
- javax.crypto.spec

The "Java Cryptography Architecture" (JCA) refers to the framework for accessing and developing cryptographic functionality for the Java Platform. It encompasses the parts of the JDK 1.1 Java Security API related to cryptography (currently, nearly the entire API), as well as a set of conventions and specifications provided in this document. It introduces a "provider" architecture that allows for multiple and interoperable cryptography implementations.

The Java Cryptography Extension (JCE) extends the JCA API to include encryption and key exchange. Together, it and the JCA provide a complete, platform-independent cryptography API. The JCE will be provided in a separate release because it is not currently exportable outside the United States.

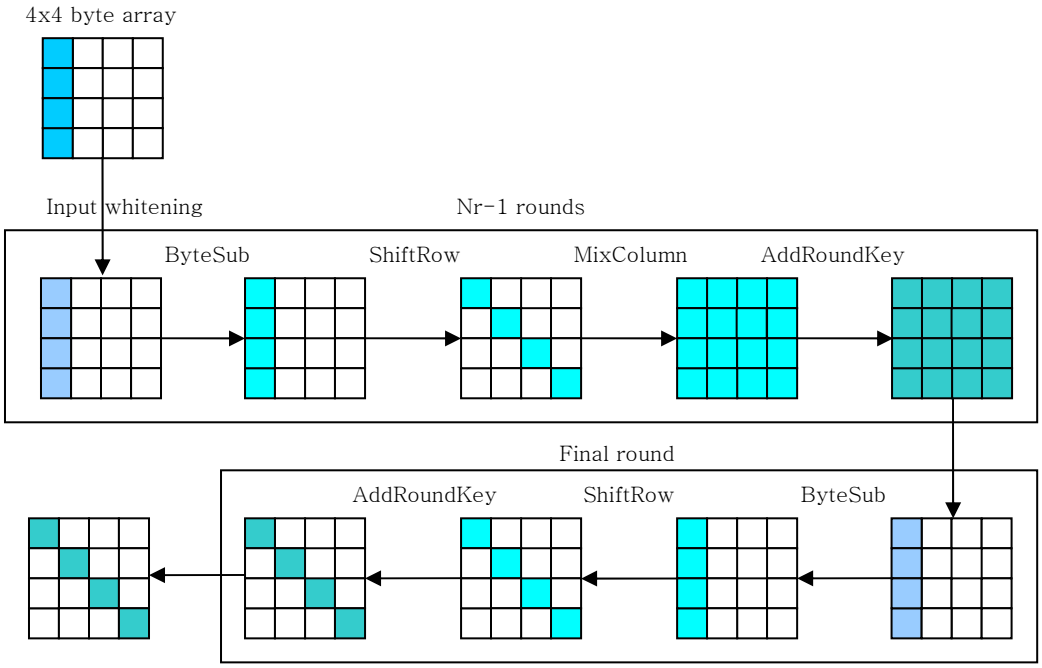
The Java™ Cryptography Extension (JCE) is a set of packages that provide a framework and implementations for encryption, key generation and key agreement, and Message Authentication Code (MAC) algorithms. Support for encryption includes symmetric, asymmetric, block, and stream ciphers. The software also supports secure streams and sealed objects.

JCE is designed so that other qualified cryptography libraries can be plugged in as service providers, and new algorithms can be added seamlessly. (Qualified providers are signed by a trusted entity.)

3. Rijndael

3.1 Specification

The cipher Rijndael consists of an initial round key addition and Nr-1 round and final round.



Author: Joan Daemon, Vincent Rijmen

Data block: 128 bits

Key block: 128, 192, 256 bits

Structure: SPN

Nb: The number of columns

Nk: The number of columns of the cipher key

Nb: The number of rounds

3.1.1 The ByteSub transformation

The ByteSub transformation is a non-linear byte substitution, operating on each of the state bytes independently. The substitution table (or S-box) is invertible and is constructed by the composition of two transformations.

3.1.2 The ShiftRow transformation

In ShiftRow, the rows of the state are cyclically shifted over different offset. Row 0 is not

shifted; Row 1 is shifted over C1 bytes, row 2 over C2 bytes and row 3 over C3 bytes.

NB	C1	C2	C3
4	1	2	3
6	1	2	3
8	1	3	4

Table 1 : Shift offsets for different block lengths.

3.1.3 The MixColumn transformation

In MixColumn, the columns of the state are considered as polynomials over GF(2^8) and multiplied modulo x^4+1 with a fixed polynomial $c(x)$, given by

$$c(x) = \text{'03'}x^3 + \text{'01'}x^2 + \text{'01'}\,x + \text{'02'}$$

This polynomial is coprime to x^4+1 and therefore invertible. This can be written as a matrix multiplication. Let $b(x) = c(x) \otimes a(x)$,

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

3.1.4 The Round Key addition

In this operation, a round key is applied to the state by simple bitwise XOR. The round key is derived from the cipher key by means of the key schedule. The round key length is equal to the length *Nb*

3.2 Key Schedule

The round key is derived from the cipher key by means of the key schedule. This consists of two components: the expansion and the round key selection. The basic principle is the following:

- The total number of round key bits is equal to the block length multiplied by the number of rounds plus 1.
- The cipher key is expanded into an expanded key.
- Round keys are taken from this expanded key in the following way: the first round key consists of the first *Nb* words, the second one of the following *Nb* words, and so on.

3.2.1 Key expansion

The expansion key is a linear array of 4-byte words and is denoted by *W*[*Nb**(*Nr*+*1*)]. The first *Nk* words contain the cipher key. All other words are defined recursively in terms of words with smaller indices. The key expansion function depends on the value of *Nk*

3.2.2 Round Key selection

Round key *i* given by the round key buffer words *W*[*Nb***i*] to *W*[*Nb**(*i*+*1*)].

4. Messenger Design

This messenger is consisted of two parties; server and client. The messenger server is coded in java. It is possible that any computer language can make client program. If supported. But in this project, Client will be coded in java. Basically message exchange is achieved in only two parties and it is possible in online user. And all messages go through server. Exactly when you send a message, is sent, it goes to the server, which then routes it through its network and delivers it to the recipient.

4.1 Messenger Server

The messenger server has the information of all user and function as server for exchange message. All clients should log in server to send message and all message have to pass through server.

4.2 Messenger Client

The client is a program that user can use to send message. It's basic function is to log in server and send message and manage friends' list.

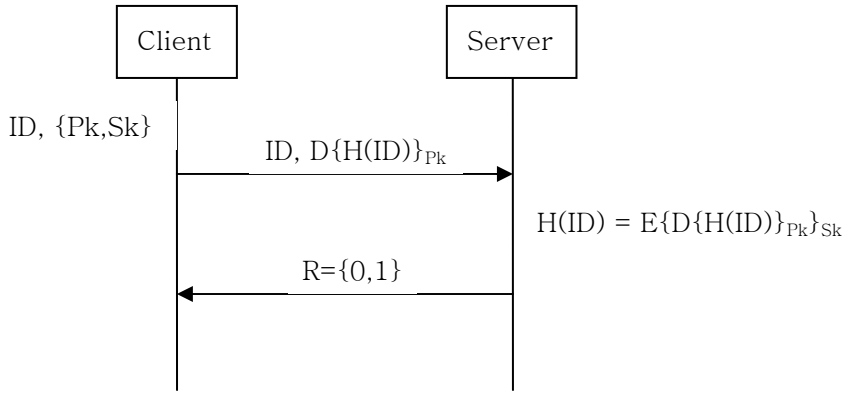
4.3 User authentication & Key agreement

In order to use messenger service authentication is needed. ID and password (or something like key) is needed for log in server. Key is also needed for encryption of message. Key is used in encryption message and decryption message at other user. Therefore key is shared with each other. Diffie-Hellman is used for sharing same key. When client is started, key sharing is occurred with online user. Finally share key is generated each friends. The shared key is used to encrypt and decrypt message in sending

User Authentication

When user is registered in server, client generates key pair and sends user's information and

generated public key. Server received this information and save it. Some signature scheme is used in authentication process. Client generated digested message using some hash function and sends user id and signature decrypted with one's public key. Then server verifies message and signature. If verification is right, server sends success message. Otherwise sends fail message.



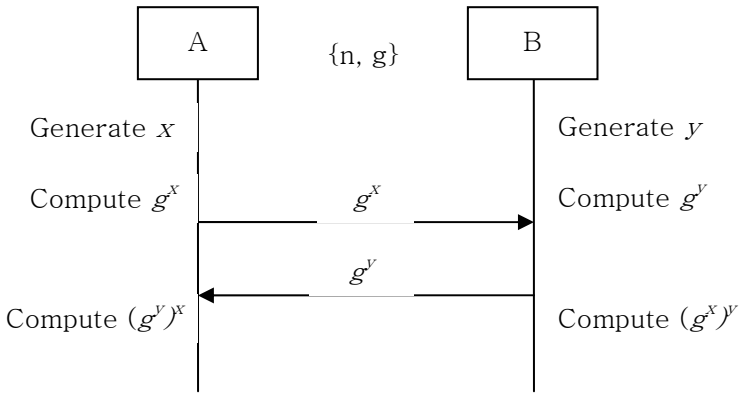
Authentication

Diffie-Hellman Key agreement

When public information is {n,g}.

1. A generates a random integer x and sends to B the elements g^x mod n.
2. B generates a random integer y and sends to A the elements g^y mod n.
3. A can then computes $(g^y)^x$ mod n.
4. B can then computes $(g^x)^y$ mod n.

After all A and B will have the same value.



Diffie-Hellman Key Agreement

In this process java.math.BigInteger class is used importantly.

BigInteger(int bitLength, int certainty, Random rnd);

It will construct a randomly generated positive BigInteger that is probably prime, with the specified bitLength. certainty is a measure of the uncertainty that the caller is willing to tolerate. The probability that the new BigInteger represents a prime number will exceed (1 - $1/2^{certainty}$). The execution time of this constructor is proportional to the value of this parameter.

```
BigInteger n = new BigInteger(bitLength, certainty, rnd);
BigInteger g = new BigInteger(bitLength-1, rnd);
  1.A: BigInteger x = new BigInteger(bitLength-1, rnd);
    computes gx = g.modPow(x, n);
    sends gx to B.
  2.B: BigInteger y = new BigInteger(bitLength-1, rnd)
    computes gy = g.modPow(x, n);
    sends gy to A.
  3.A: Kyx = gy.modPow(x, n);
  4.B: Kxy = gx.modPow(y, n);
```

4.4 Messenger Protocol

There are many different protocol commands. But I will show some important command; related login and message.

Signal

All protocol signal start “+” or “-”. The signal started in “+” is a signal that server send to client. “LF” in UNIX system or “CR/LF” in Windows system is attached at end of signal. The signal started in “-” is a signal that client send to server.

All signal received in server has 3 digit status value. The first digit indicates success or fail and error and the second show more detail information of signal. The final digit is only index of signal.

Time format

Time is consisted 14 string not using default timestamp; year 4digit, month 2digit, day 2digit, hour 2digit, minute 2digit and second 2digit, “20011001120030” means 10-01-2001 12:00:30. Time can changed in each operation or java virtual machine version.

Connection

When client connect to server, client will receive welcome message

+201 Welcome to message server V0.1

This means connection is completed. Then you can log on.

Log on

ID and password are necessary to login.

-login [user id],[user password]

If login success, client receive this signal.

+203 login ok

If you already login, client receives this signal.

+204 already logged in

If you send other protocol without login, server sends this signal.

+205 login first

If you already login and your ID is used, client receives this signal.

+500 This ID is used

If login fails, client receives this signal.

+401 login fails

If protocol signal is wrong, server sends this signal.

+501 Wrong token

Log out

Log out and close socket.

-logout

+211 bye

Messages send & receive

Send message to other client

-message [to user id], [message contents]

+232 message sent user

If user does not exist,

+410 no user

If can not send message,

+420 can not send message

When receive message, sever sends this signal

+230 You've got a message from [to user id]

Receive message from server

If you want to get the most recent message,

-message.getAMessage

If you want to get all message,

-message.getAll

+234 message [from user id], [time], [contents]

If there are no more message,

+235 no more message

Etc

friends.get if receive friends list.

friends.add [user id],[group] if add friends to list.

friends.del [user id] if delete friends to list.

users.regist [user id],[user information] when register user.

users.isconnected [user_id] if want to know user is connected.

5. Progresses

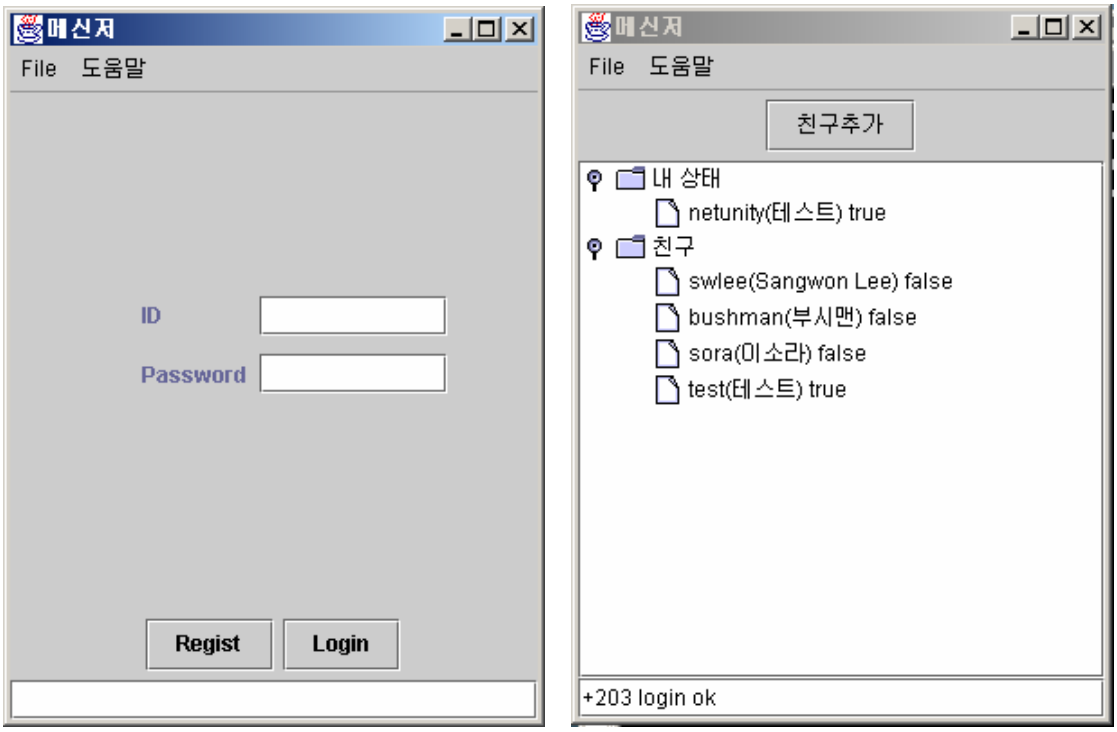
4.1 Implementation of Rijndael

There are so many resources for implementation of Rijndael. I will implement Rijndael in Java. Of course source codes implemented in java exist in internet. I refer to these sources.

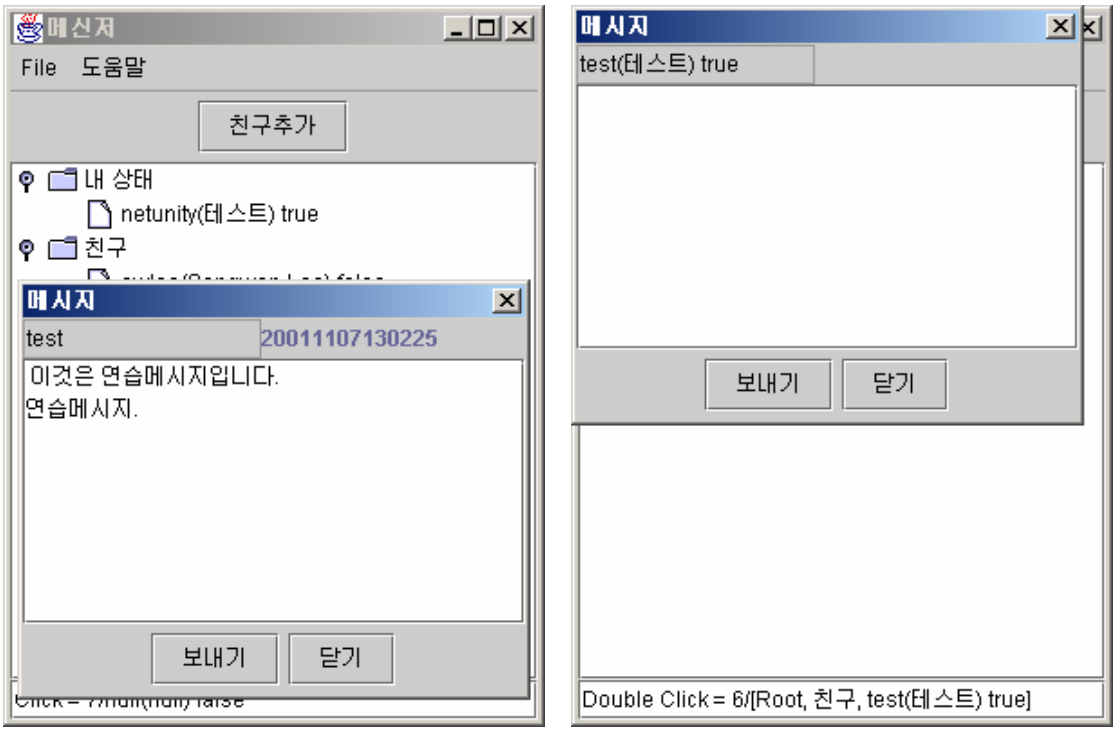
Rijndael is used in encryption and decryption of message. There are two way for implementation of Rijndael in java. One is using provider and another is not use provider. There are already Rijndael block cipher providers in some package and to use provider is relatively easy than make all function by myself. But core implementation is the same.

4.2 Implementation of Messenger

The following picture is messenger client. It is simply and not completed yet. But it can send and receives message. And it has some other function; user registration, add friends to list, etc.



Log on and default message appearance.



When message is received and send message.

6. Analysis & Further works

This messenger works with this way. First, user must register in messenger server in order to use message service. Registration process is that client sends one's information and public key and then server assess this information and save it. After registration process user will log on server. Client sends ID and signature that decrypt ID that hashed by SHA-1 algorithm with one's secret key. Then server verifies hashed ID and information that encrypted with public key and sends response to client. Next authentication user can exchange key with one's friends listed. Diffie-Hellman key agreement scheme is used to share key. When key sharing is completed, client can send message to another user. This time message will be encrypted with shared key using Rijndael.

Now messenger server and client are partially implemented. But cipher function is not attached. So message exchange is possible and all sanded messages are only plain text not encrypted text.

In future it is remained user authentication and message encryption module implementation and attach. For authentication RSA and SHA-1 must be implemented. Now login process use parameter ID and password but this part will be replaced signature scheme.