Secure Massager Protocol using Rijndael

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

1. Introduction

Instant messenger (IM) is program that can exchange information to other 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 of all IM do not have any security module like cytological 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.

Until now security problem like messenger hacking is rare. But professionals point out that messenger used by most internet user is not safe. Because using messenger is spread to business market.

In this paper, we implement secure instant messenger (SIM) protocol and simple messenger. The goal of our secure messenger is to exchange message between each other in internet. And Rijndael will be used in the cytological module used by SIM.

2. Related 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".

Author	Joan Daemon, Vincent Rijmen
Data block	128 bits
Key block	128, 192, 256 bits
Structure	SPN

Table 1 Rijndael Specification

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

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.

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.

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 $c(x)$ = '03'x^3 + '01'x^2 + '01' x + '02'

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.

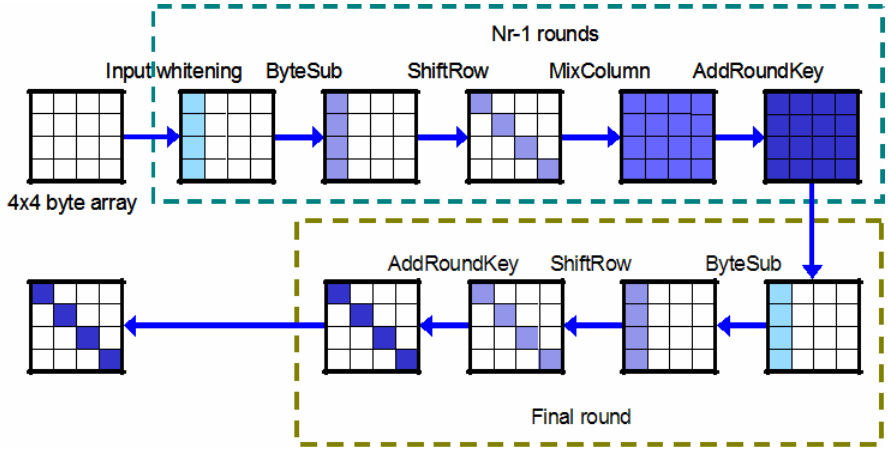


Figure 1 Rijndael Encryption

2.2 Instant Message

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.

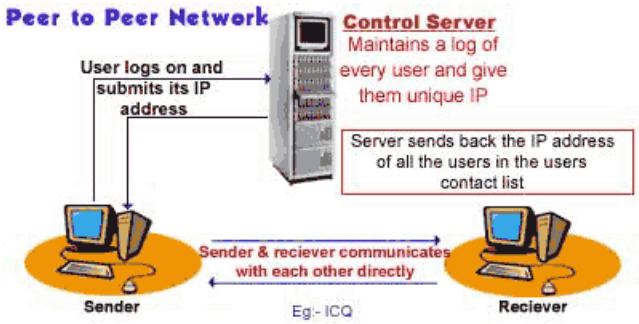
Through server routing

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.

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.cert
- 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. 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.

3. Security Consideration

Desired properties for SIM typically include the following:

Privacy

User's information should be kept safely and not revealed. SIM server should save user information in safe storage.

Confidentiality

Most messages are encrypted and decrypted before sending to other user. Rijndael is used as encryption and decryption algorithm.

Authentication

Only authenticated user can use SIM. RSA is used for authentication.

Integrity

Message sent by user should not be modified or forged by other user. Hash function is used in order to guarantee Integrity of message. SHA-1 algorithm is used as hash function.

Efficiency

SIM use hybrid cryptosystem. Asymmetric cryptosystem is used in key agreement and digital signature. RSA is used as asymmetric. Symmetric cryptosystem is used in message encryption and decryption. Rijndael is used as symmetric.

4. Messenger Protocol Design

This messenger protocol is consisted of two parties; server and client. 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.

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.

The client is a program that user can use to send message. Its basic function is to log in

server and send message and manage friends' list.

N	modulo
G	An element of G, G is a group
PK	User's Public key
SK	User's Private key
$Enc\{\}_{key}$	RSA Encrypt data with key
$Dec\{\}_{key}$	RSA Decrypt data with key
R	Server Response
S	Shared session key
H()	Hash function

Table 2 Notation

Authentication is needed in order to use messenger service. ID and password and key pair is needed for log in server. Key is also needed for encryption of message. Key should be used in encrypting message. Therefore key is shared with each other. Diffie-Hellman is used for sharing same key. When client is started, key sharing is occurred with on-line user. Finally Session key is generated each friend within contact list. This shared key is used for encryption and decryption of exchanged message.

4.1 Registration

User should be register in order to use SIM service. User sends ID, Password and some information to server. Then server checks this information and save storage.

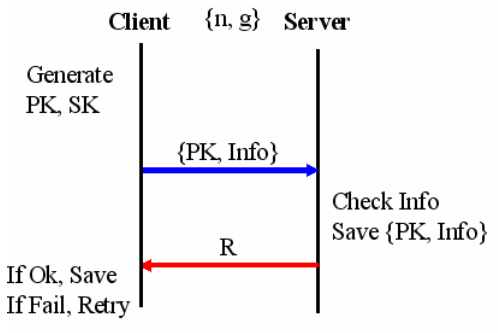


Figure 2 Registration

Initialization

User connects server and receive public information of server, {n, g}

Scheme

1. User generates RSA public key and private key using key generation function.
2. User sends public key with user information of registration.
3. Server receives and checks and saves this information.
4. Server responses to client whether registration is success or not.
5. If server's response is ok, User saves one's key pair.
6. If fail, retry.

4.2 Authentication (in Login)

User should be login to server in order to exchange message to other user. In authentication process, RSA signature scheme and Hash function are used.

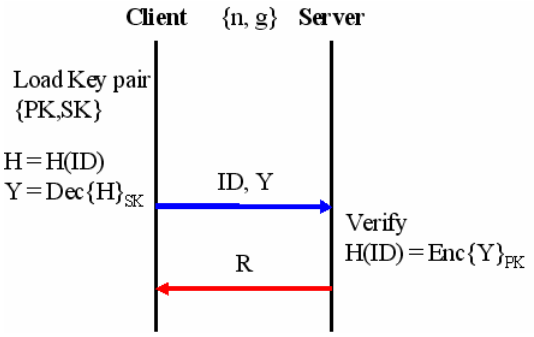


Figure 3 Authentication

Initialization

User loads ones key pair using User's password.

User connects to server and receives {n, g}

Scheme

1. User generate Hash value of ID, H(ID)
2. User computes Y = $Dec\{H(ID)\}_{Sk}$.
3. User sends ID and Y.
4. Server verify $H(ID)=Enc\{Y\}_{PK}$ and response to User

4.3 Key agreement at Login

Key agreement method is used for exchanging message to other user. User should share key with other wanted to send message. Diffie-Hellman key agreement method is used as key agreement scheme.

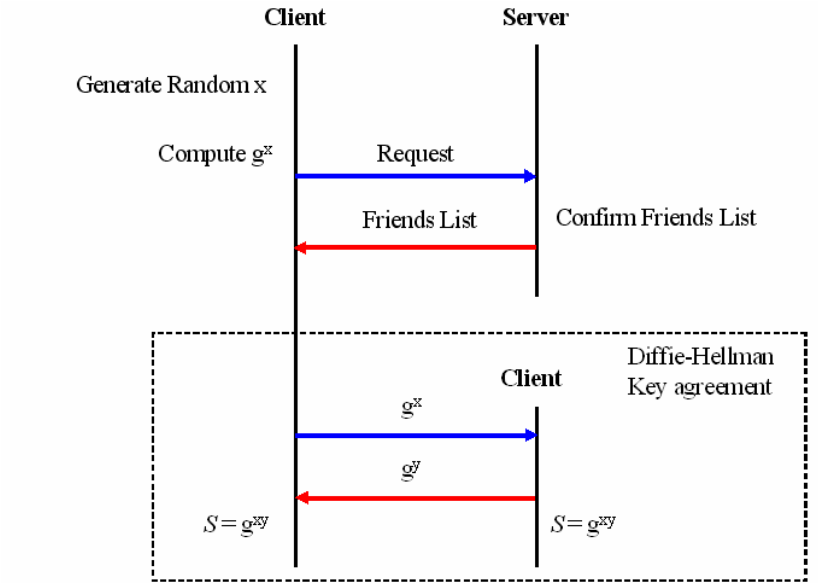


Figure 4 Key Agreement

Initialization

User already know {n, g}

Scheme

1. User generates random integer x.
2. User computes g^x.
3. User request friends list to server.
4. Server confirms whether user's friends is online or not and sends friends list with online conformation information.
5. User receives one's friends list.
6. If friend is online. (DH key agreement)
   A. User sends g^x to friend.
   B. User receives g^y.
   C. User computes session key $S = g^{xy}$.
   D. User saves session key with ID.

5. Implementation of SIM Protocol

In Diffie-Hellman Key agreement process java.math.BigInteger class is used importantly.

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

It will construct a randomly generated positive prime BigInteger, 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);
```

Table 3 Sample code

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.

OS	Windows XP
CPU	Pentium IV 1.4GHz
Memory	512MB
Dev. Kit	JDK 1.3.1, JBuilder 5

Table 4 Implementation environment

6. Conclusion & Further works

Secure instant messenger protocol was implemented. All function of normal instant messenger was not supported, but crypto function is completed and safely operated. Some messenger include crypto module that support only public key cryptosystem. We use hybrid

system in our implementation. Hybrid system is more efficient than system that supports only public key system.

Our implemented messenger is simple and does not use any data base. Data base should be used in order to guarantee security. Also now user can not send message to off-line user. Therefore SIM should be modified to use DB. And another method should be used in order to send message to off-line user

7. Reference

[1] Alfred J. Menezez, Paul C. van Oorschot, and Scott A. Vanstone, *Handbook of applied cryptography*, CRC Press series on discrete mathematics and its application. CRC Press, 1997, ISBN 0-8493-8523-7

[2] Charle Kaufman, Radia Perlman, Mike Speciner, Network Security Private communication in a PUBLIC World, Prentice Hall PTR, *Upper Saddle River, New Jersey 07458*

[3] Merlin Hughes, Michael Shoffner, Derek Hamner and Umesh Bellur. JAVA Networking Programming, MANNING

[4] Jonathan Knudsen, Java security and cryptology, O'REILLY

[5] http://www.cryptix.org/, Cryptix

[6] http://java.sun.com/products/jce/index.html , Java™ Cryptography Extension (JCE)