Final Project RSA Secure Chat Server
CSC 290
Warren Fong
wf007j@mail.rochester.edu

Abstract

Chat servers today are readily available and very useful in conversing with people that might be close by or far away. Internet chat services like AOL provide the convenience of conversing with people in real time. This service provides a host of possibilities for work, school, and connectivity. Unfortunately these widely available chat services do not provide protection/privacy of what is being sent through the chat servers. The objective of this project is to build a secure chat server utilizing Public Key encryption to send secure chat messages across the internet.

Benefits of Chat Service

- Allows for "instant" communications between people.
- Use of the real time chat over the Internet can eliminate costly long distance charges.
- Ability to stay in contact with people who you normally never see.
- Allows for quick questions and quick responses.

Negatives of Chat Service

- Potential security/privacy problems of these Instant Messaging programs.
- Chat in most instances are routed through a server system where the service is provided and that is a single point where all messages can be intercepted.
- Chat programs can provide an open avenue of attack for hackers, crackers, spies and thieves.
  - eavesdrop: intercept messages
  - actively insert messages into connection
  - removing sender or receiver, inserting himself in place

Sample Chat Packet

```
<message from='John'
        to='Jane'
        id='message1'>
        <thread>thread1</thread>
        <body>How are you?</body>
</message>
```

Because the message is in plain text it is easily interpreted. This means if it were intercepted it would be easy for someone to eavesdrop on the conversation between John and Jane.

Implemented Solution

Encryption

- Utilize public key encryption to securely transmit messages between users.
- Encrypt each message with public key of target recipient.
- Since chat messages are normally not long, it requires less processing time for the program to encrypt the message.
- Digitally sign every message

Positives

- Very secure message exchange
- Interception can happen but the interceptor can not decipher the message.
- Insertion of data can happen but the digital signature ensures that message is authentic.

Negatives

- Much slower then symmetric encryption methods, however since messages are relatively short. The difference between the times is negligible.
- Much more hassle to encrypt and decrypt messages.
- Can be subject to man in the middle attack

Comparison of Public Key Algorithms

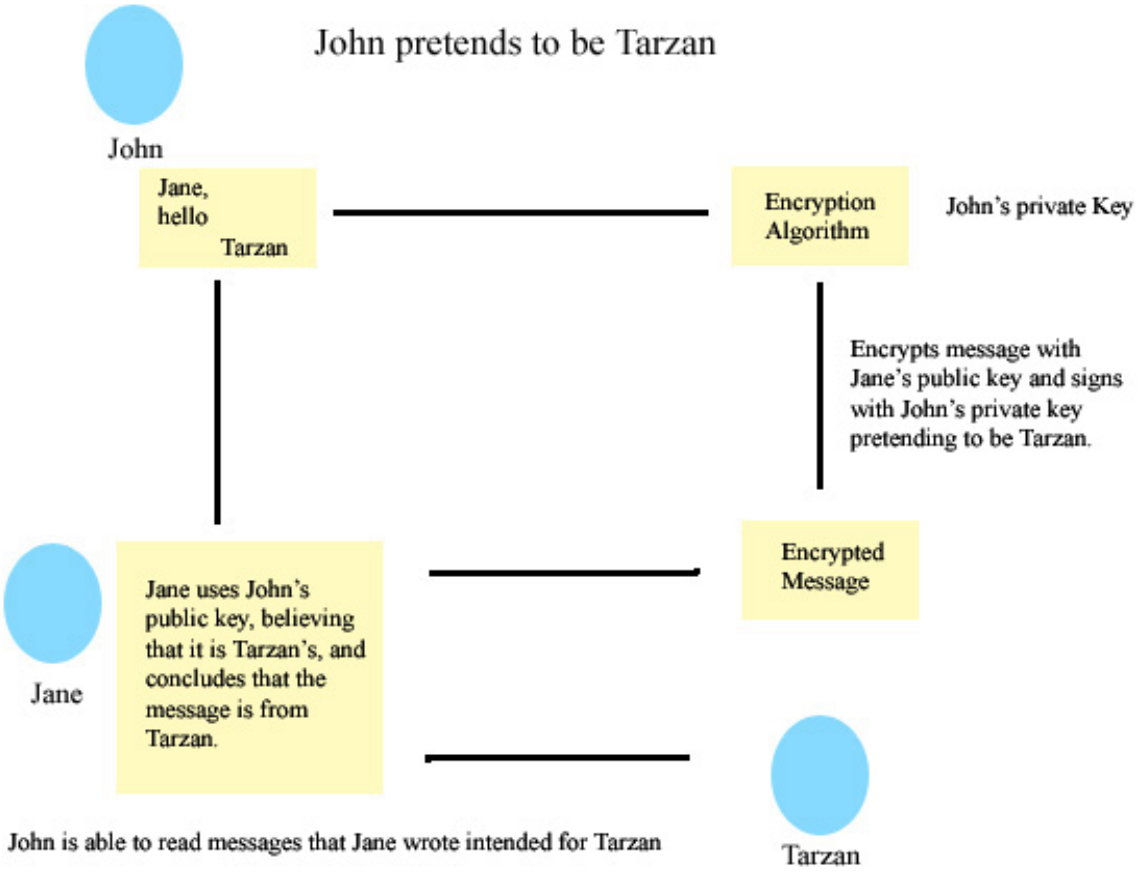
- Diffie Hellman Key Exchange
  - discrete logarithms
  - ~500K 32-bit multiplies / operation for first operation and then speed of symmetric key utilized.
  - 5 step process for successful implementation.
    Key Generation
    Key Generation Symmetric Algorithm
    Encryption
    Decryption
    Encryption/Decryption Symmetric Algorithm
  - 2^{nd} fastest of the three
  - Eventually will be cracked by quantum computing
  - More processes needed for successful implementation
- NTRU
  - Lattices
  - 5 μs / operation
  - 3 step process for successful implementation.
    Key Generation
    Encryption
    Decryption
  - Fastest of the three
  - Can not be cracked by quantum computing.
  - Requires twice the amount of space/bandwidth to maintain security and information than RSA and Diffie Hellman.
  - Not fully proved to be impossible to crack.
  - Industry does not have wide trust for algorithm yet.
- RSA
  - integer factorization
  - ~500K 32-bit multiplies / operation for first operation
  - 3 step process for successful implementation.
    Key Generation
    Encryption
    Decryption
  - Slowest
  - Eventually will be cracked by quantum computing
  - Utilizes public key for all messages not just key.
  - Fully secure for all data
  - Less processes for successful implementation.

Secure Chat Protocol

- 2 Clients connect to a server
- Once connected, each *client generates their public and private keys locally.*
- The public key sent to the server and is set so when a user clicks their name any messages sent will be encrypted with that public key.
- *The private key remains only in the client program.*
- When message is sent out, the client program downloads the public key and encrypts the intended message and then applies the digital signature which is created with the private key and then sends the encrypted message out.
- When the packet is received by the specified person, the client program automatically applies the private key on the text and outputs the message so that the user can see it decrypted and then double checks the digital signature with the public key.
- Once completed with each step we have successfully transmitted a secure message.

Security Threats with Secure Chat Protocol

- Man in the middle attack – Although this protocol provides message security it still does not provide protection against faking to be the person client is taking to.



Keywords

Chat Servers
Public Key Encryption
Secure Chat
Internet Chat
Networks

Process

Utilizing the chat protocol above I created a Java chat program that successfully communicates securely. The RSA algorithm was utilized in encrypting and decrypting the small messages sent between users. The secure chat program allows for two users to connect to the server and encrypts messages with each other's public keys.

How to run program:
Compile the following java files:

ChatClient.java
ChatServer.java
ChatServerThread.java
ChatClientThread.java
Rsa.java

Then start the server with following line:
java ChatServer 1024

Once server is up, open chat.htm and the chatclient applet should appear.
Type name in text box and click connect.

Open another chat.htm and connect another user to the server. Once both connected, type messages to each other securely.

Working

Secure chat between only two people.
Server can only store 2 public keys.
Transmission of secure messages.
Generation of public and private keys for RSA.
Automatic encrypt and decrypt of messages.

Not working

Chat between more than 2 people.
Digital signature check.

The following is exactly what happens when a user connects to the server.

1st user enters name and clicks connect.
Public key generated
Private key generated
Public key is sent to the server
Private key remains on client

2nd user enters name and clicks connect.
Public key generated
Private key generated
Public key is sent to the server
Private key remains on client

1st user types a message and clicks send.
Message is displayed encrypted with 2nd users public key.
Message is also displayed with "Decrypted: (some garbage because the private key applied is not the right key to applied to 2nd users public key.)"

2nd users screen displays encrypted message sent by 1st user.
2nd users screen also displays "Decrypted: (message decrypted with proper key)". Since the message was intended for the 2nd user, the private key of the 2nd user was the correct key applied to the decryption function. So the output is in readable plain text.

2nd user types a message and clicks send.
Message is displayed encrypted with 1st users public key.
Message is also displayed with "Decrypted: (some garbage because the private key applied is not the right key to applied to 1st users public key.)"

1st users screen displays encrypted message sent by 2nd user.
1st users screen also displays "Decrypted: (message decrypted with proper key)". Since the message was intended for the 1st user, the private key of the 1st user was the correct key applied to the decryption function. So the output is in readable plain text.

Success!!! Secure message transaction completed successfully.

References

Basic Information on Public Keys and the OpenPGP Standard
http://www.keyserver.net/en/info.html 2001

Diffie, W., and Hellman, M.E., New Directions in Cryptography, IEEE Transactions on Information Theory, vol. 22, no. 6, November 1976, pp. 644-654.

Garret, Paul. Making, Breaking Codes: An Introduction to Cryptology. Upper Saddle River, NJ: Prentice-Hall, 2001

Hoffstein, Jeffery, Pipher, Jill and Silverman, Joseph H. NTRU: A Public Key Cryptosystem. August 1999.
http://grouper.ieee.org/groups/1363/lattPK/submissions.html#NTRU1

Kurose, James F., Ross, Keith W., Computer Networking: A top Down Approach Featuring the Internet. 2nd edition. Addison Wesley 2002.

Mollin, Richard A., ***RSA*** and public-key cryptography. Boca Raton, Fla.: Chapman & Hall/CRC, 2003.