AN AUTHENTICATION SCHEME FOR INSTANT MESSAGING SYSTEM

***Abeer M. Yousif, **Malath S. Kareem and ***Sattar B. Sadkhan**
*College of Science, Al-Nahrain University, abeermatti@yahoo.com
**College of Science, Al-Nahrain University, Malath.sabri@yahoo.com
***Babel University, drengsattar@yahoo.com

Abstract

This paper presents an instant messaging system designed as LAN supported application with a suggested method to authenticate users. Our system provides simple, fast and secure authentication method. The suggested authenticated method combines two mechanisms: fixed password and digital signature to overcome the problems faced the use of these methods when used alone. The designed instant messaging system has some traditional features like those given by any instant messaging system. It uses TCP/IP protocol to communicate between hosts. All connections between server and clients are based on client/server networking model, while connections between clients is based on peer-to-peer networking model.

The proposed system has been evaluated according to two aspects: user's login security and overall system performance, the evaluation shows quite encouraging results.

Keywords: Instant Messaging System, Authentication methods.

Introduction

Messaging systems are growing in popularity all over the world. The ability for instant communication and the benefit of knowing who is online has added to the messaging systems as a tool together with ordinary phone and e-mail communication [1].

The proposed Instant Messaging System (IMS) presented in this paper has focused on the security of instant messaging system, where two combined different authentication mechanisms have been adopted: fixed password and digital signature.

The system requirements and considerations are given in the next section, and the structure of the system and the main functions of its modules are also presented. The system evaluation results will be presented then. Finally the paper will end with the derived conclusions.

Design Requirements and Considerations

The design considerations of the system are addressed as functional and operational consideration. From the functional point of view, the system should offer the following functions in a simple and fast manner by the users:

- Text chat, to send instant messages back and forth between two users (online chat).
- Group chat, to set up a group text conference between many users (chat rooms). They can be defined immediately when needed by any user in the system.
- Offline messages transfer, using private chat area.
- Password change, in the case of its steeling or forgetting.

From the operational point of view, the system will use TCP/IP protocol for all communications between hosts. All connections between server and clients are based on client server networking model. The server supposed to be as lightweight as possible and yet still be robust enough to handle all requests. It is also supposed to act just as an authentication center to the clients, while the main client responsibility will be manipulating chatting activity without interfering from the server.

System structure

The proposed system consists of five modules; these will be discussed in the following paragraphs:

Initialization Module. This module is executed at server side and it consists of two stages: key generation and start serving.

A) *key Generation stage* creates a table of client's public keys according to ElGamal algorithm [2]. These keys will be used to verify the identity of users at login. Each client must have two numbers P and α created randomly. P Should be prime number, α must be the generator of P.
Concerning to the integer P, the following requirements should be satisfied [3]:
I- The prime P should be large enough to prevent efficient use of index calculus method[4].
II- $P-1$ should be divisible by a prime q sufficiently large enough to prevent a pohlig-Hellman logarithm attack [5].

B) *Start Serving stage* prepares the server to start client's serving which requires opening two windows sockets, one for server registration and one for server login activities. Once server start listening, then it's ready to receive and send data for any client. The exchanged data are transmitted between clients and server in form of packets. Figure 1 shows a representation of the IMS packet.

Packet ID 3 bytes	Version 2bytes	Length 2 bytes	Service Type 5 bytes
Data			

Figure (1): Typical IMS Packet

IMS header part consists of:

- **Packet ID** field**:** represents the messenger name which is always "IMS" (3 bytes).
- **Version** field**:** represents IMS version number; the current version of IMS is 1.0 (2 bytes).
- **Length** field: states how many bytes are in the data section of the packet (2 bytes).
- **Service Type** field: states what kind of service is requested or/and being responded to (5 bytes).

Registration Module. In this module, the client registers himself as new member in IMS. First the client must be connected with server after that, the client must present the following information about the user:
A**)** Register information which includes:
- User ID, which must meet the following requirements:
  - Its length is long enough (to make number of possible UserID very large).
  - It must be unique, i.e. never used by another client before.
  - It doesn't contain any special characters.
- First name, Last name, they shouldn't contain any special characters.
- Birthday date must be valid date.

B) Password information must meet two requirements:
- Its length is long enough.
- There must be at least one alphabetic character and one numeric character.

C) Challenge information represents an answer to a challenge question. It will be used when user needs to change the forgotten password.

After accepting the registration information by the server, the server assign unused pair (P, α) numbers to the new client who in turn will use them for computing another numbers, which are the secret keys (a, a_c) and public keys (y, y_c.). The secrete keys are computed using MD5 hash algorithm[6].

a=MD5(UserID||Password)..... (1)

a_c = MD5(UserID||ChalangeInfo) ..... (2)

$$y = \alpha^a \bmod p \quad \text{..... (3)}$$

$$y_c = \alpha^{a_c} \bmod p \quad \text{..... (4)}$$

All the above information except secret keys will be saved in server DB.

3. Login Module. Successful login requires client to pass authentication test. The test procedure involves the client should send its UserID to the server, the server checks if the UserID already exists in the Online list. If it is, then the server informs the client that the user already is login by warning message. If it isn't, the client compute it secret key a and public key y as in Registration module. Then the client send y with UserID in IMS packet to the server to check their consistency compared with the saved one. If no match occurs then the server sends error message to the client and disconnects the connection. Else the server will send to the client a message (m) to sign it (the message is hashed using MD5 algorithm).
When the client receives the hashed message it will start the signing process. Signing process needs the following computation [3]:

A) Compute a random secret number *k*, satisfying:

$\gcd(k,p-1)=1$ .....(5)

B) Compute *r*, $r = \alpha^k \bmod p$ .....(6)

C) Compute the inverse of *k* using extended Euclidian algorithm

D) Finally, compute the signature s, such that

$$s = k^{-1}\{h(m) - a \bullet r\} \bmod (p-1) \quad .. 7$$

After completing the above computations, the client will send the received message, *r*, and *s* to the server to verify the signature correctness, as follows [3]:

$v_1 = y^r r^s \bmod p$ .....(8)

$v_2 = \alpha^{h(m)} \bmod p$ .....(9)

If $v_1 \neq v_2$ then the client fails to prove its identity and the server will send error message to it and disconnect the link. Otherwise, the client will be allowed to be online, in this case the server saves the UserID of the login client in its DB and sends the client pair (UserID,IP) to all online clients in the system. Finally the server will manipulate the offline messages and present them to the login client by checking its offline table.

4. Chatting Module. In this module the client is allowed to do:

A) Chat with other clients (private and public chat rooms).

B) Adding UserID to a list named Buddy List.

C) Sending offline messages to offline clients.

D) Changing known password.

IMS Evaluation

In this section, the considerations that are addressed in section II are discussed to evaluate the performance of the designed system.

1.**Easy of use**: From the user point of view, IMS client interface provides an easy and simple way to do the following activities: registration, login, chatting, sending offline messages, and changing password.

2.**Fast Execution**: one of the important issues of any instant messaging system is speed. When IMS is tested within LAN of 50 PCs, all chatting functions is performed instantly correct. The only wait by the end user happened when he/she register or log into the system. The waiting time vary from 0.96875 second to 5.671875 second depending on the length of *P*, which is also considered acceptable for security purposes. It is important to mention here that the longer time that needed to generate table of N public keys is performed in offline mode at server side to be invisible to the end user.

3.**Security**: The security of the authentication activity of the system is evaluated as follows:

- The password and secret key *a* are not sent over network neither as plaintext nor encrypted, instead the public key *y* will be sent, so password is protected against eavesdropping, and the secret key *a* is protected against *key only attack* since solving discrete logarithm problem in equation 3 is very difficult [7].
- It takes very long breaking time in brute force attack to know the password. In this system the password was taken between 8-15 character, the character set consists alphabet characters from (A-Z or a-z) and digital characters (0-9) i.e. 36 characters, so the system as whole consist of $(36)^8+(36)^9+\ldots+(36)^{15} \approx 2.273903*10^{23}$ possible password of length between 8-15. At a rate of 10^9 passwords per second, it would take $7.210499*10^6$ years to test all passwords; this time is very enough to make this type of attack not possible.
- The user can change the password when he/she suspects that the password has been broken. Only the real owner of a UserID can change the corresponding password after presenting the correct challenge information, this challenge information is (as with password) not sent over network, but instead its used to generate secret and public keys and then a digital signature which will be verified by server. When verification process succeeds, the user is allowed to change the password.
- The messages to be signed are chosen by server, so any adversary will not be able to choose a message and obtains valid signature for it, because the adversary will not guess which message have to be signed.

Conclusions

As a result of system implementation, the following conclusions have been derived:

1.The combination of password and digital mechanisms offer more strength to the authentication process since the drawbacks of password mechanism were improved by this combination.

2.To achieve IMS scalability, number of IMS's users can be increased by maximizing the number of primes *p* at IMS server, this may cause that the new users will have larger prime *p* and as a result they will get more security.

3.Using windows sockets when designing IMS provides easy of implementation as it is implemented with few statements and need few variables.

During system's design and implementation, the following drawbacks were identified:

1.Knowing the secret key (*a*) by an adversary will let him/her to impersonate the real owner. This can be achieved in two cases only:

A) Solving the discrete logarithm in equation 3. To limit this case, *p* length is chosen greater than 768-bit number.

B) The repeated use of the same *k* by the same client to sign two different messages. Theoretically this attack could be happened when the following conditions are satisfied: **First**, the adversary should be presented each time when the client is logged into the system in order to get all the previous signatures (s_i) that are signed with the same *k*. **Second**, the adversary should perform the following calculations:

I-Keep computing

$s_i - s_{i-1}$ until $s_1 - s_2 \neq 0 \bmod(p-1)$.

II-Compute *k*,

$k = (s_i - s_{i-1})^{-1}(h(m_i) - h(m_{i-1})) \bmod(p-1)$.

2.The instant messages between clients are not encrypted. This may allow someone to intercept the packet and modify the messages exchanged.

References

[1] M. Mannan, P.C. van Oorschot "Secure Public Instant Messaging: A Survey", Carleton University, Ottawa, Ontario, KIS 5B6, September 2004.

[2] T. El-Gamal, "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms". IEEE Trans. Inform. Theory, 31, pp. 469-472, 1985.

[3] A. Menezes, P. van Oorschot, and S. Vanstone, "Handbook of Applied Cryptography", CRC Press, 1996.

[4] R.Crandall, C. Pomerance, "Prime Numbers A Computational Perspective", Second Edition, Springer, 2005

[5] P. Soria-Rodrguez, "Implementation of the Pohlig-Hellman Algorithm for the Discrete Logarithm Problem",2000.

http://alum.wpi.edu/~sorrodp/crypto/report.html

[6] R. Rivest RFC 1321 (Request For Comment), "The MD5 Message-Digest Algorithm", 1992.

[7] J. A. Buchmann, "Introduction to Cryptography", Springer, 2001.