Key Dependent Secure Messenger

Nandanwar Aniket D., Deshmukh Sonali S., Hadawale Rashmi R., Nanaware Gauri B.

Abstract -- 21st century has given rise to Messaging. Instant messaging (IM) is a type of online chat which offers real-time text transmission over the Internet. Messaging allows user to stay in touch with all of his friends, family members, relatives, etc. Messaging facility is provided by messengers. With the fact that many people use messengers, there might a possibility of a user ending up reading messages of other user. There are very few ways by means of which a user can keep his messages secure from other users. In this project we are approaching with a system which will provide a key dependent message security i.e. the encryption scheme remains secure even after encrypting. On confirming his identity, the user is granted access to the encryption key which would enable access to the message. Thus, user is provided with a secure way to communicate with the other user. Algorithm used for encryption is a suggested block cipher encryption algorithm which is a combination of different logical and mathematical operations. To calculate performance of the algorithm, two parameters are used avalanche effect and execution time.

***Index Terms*—- Cryptography, Encryption, Decryption, Security, Algorithm, Network.**

I. INTRODUCTION

The incredible growth of the Internet has excited businesses and consumers alike with its promise of changing the way to live and work. It's extremely easy to get connected to each other all over the world while sitting in front of a laptop or just with the help of internet accessible phone. Messenger available in market offers great ease to stay connected. But there might threat to privacy of a user as other users can access their messages or any information. So privacy and security becomes major concern over the Internet. Especially, when using it to transmit sensitive information between parties.

Messengers offers instant messaging (IM) that is real-time text transmission over the Internet. But almost all of them are more susceptible to privacy and security issues. This paper has presented the design and implementation of the key based secure messenger. The motive of the development of this key based secure messenger is to address privacy and security concern of end users. The reason we are trying to develop new messenger that provides acceptable security, as compared to the messengers available right now in market and at the same time, does not require the intensive processing time. Hence, Encryption can be an effective method of handling privacy and security concern of end users, and is widely used for data security in many applications.

II. TERMINOLOGY

Cryptography is the practice of secure communication in the presence of a third party. It involves mainly the encryption and decryption of data. Encryption is encoding the information so that it is unrecognizable to the third party. This can be done by encrypting the message using a so-called encryption key. Data in an encrypted format is known as a cipher text. Decryption involves decoding the cipher text back to the original plain text. This can be done by decrypting the message using a so-called decryption key. Depending upon the encryption & decryption keys used, cryptography is further classified as symmetric cryptography and asymmetric cryptography. Symmetric key uses the same set of key for encrypting as well as decrypting. Accordingly, different keys are used to encrypt and decrypt in asymmetric cryptography.

An encryption scheme is Key Dependent Message (KDM) secure if it is secure even against an attacker who has access to encryptions of messages which depend on the secret key. Recent studies have revealed that this strong security notion is important both theoretically and practically.[ref] In this paper come up with a good application of KDM security.

III. EXISTING SYSTEM

The current existing messengers avail little security measures in particular. Online messengers like Facebook have only a single security feature which asks the user to log in to communicate with other users, having no further measures to enable secure messaging. Many of the messengers address only smart phone domain. So users are not able to use instant messaging just because of platform. Offline messengers like GO! (For Android Smart Phones only) make use of private boxes for message security in which user has to add contacts to the private contact list to hide those messages. Other features of some of the messengers are to encrypt the complete thread of the contact. Though the contact is viewed, it is not possible to view the messages via that particular contact.

IV. PROPOSED SYSTEM

In this paper, we are going to provide a messenger providing a key dependent security. Encryptions of the message will depend upon the key to be entered by the user. On registering user will have to provide a set of keys – one for encryption of the messages known as secret key and other key for faking other users to be known as hoax key. User will also be provided with two different chat methods viz. public chat session and private chat session.

In a public chat session, user will be able to directly communicate with the other users without the need of any secret key or an encryption key. At the receiver end, receiver will similarly receive directly the message without needing any keys.

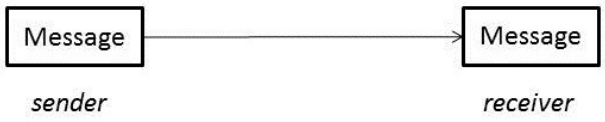

Fig. 1: Public Chat Architecture

For a private chat session, user will have to provide his secret key as means for confirming his identity along with the message and a key to encrypt the message. At the other end, receiver will have to provide his own secret key to decrypt the context.

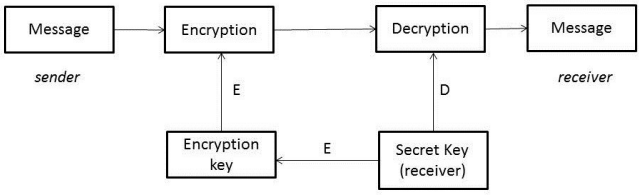

Fig. 2: Private Chat Architecture

V. Aims & Objectives

- Securely transfer messages from one user to the other.
- Along with text, secure transfer of attachments if any.
- Achieve encryption and decryption at the client side.

VI. Achieving Key Dependency

As we are going to build a key dependent secure messenger, security of the keys is also one of the important issues to address. Cryptography has been used to achieve security. Symmetric cryptography has its own disadvantages as the same key is used for encryption as well as decryption. So, to enhance further security we come up with asymmetric cryptography where encryption key and decryption key is different.

In stated project, to achieve asymmetric cryptography and key dependency, the context of message is encrypted with help of Encryption Key (EK) provided by sender. The security of Encryption Key (EK) is also to be maintained as further decryptions are totally depending on this particular key. Hence, encrypting the encryption key is a major concern.

The encryption key (EK) is to be encrypted by the intended receiver's Secret key (SK). At the receiver side, receiver gets notified about the arrival of private message and in response of notification receiver is requested to enter his Secret key (SK) which will decrypt Encryption key which in turn decrypts message.

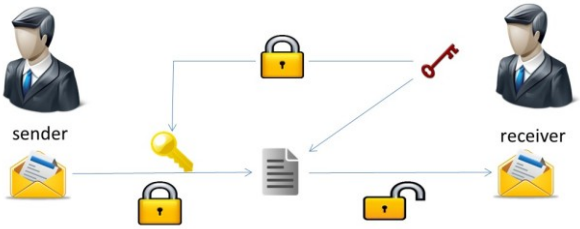

Fig. 3: Key Dependency

VII. Algorithms Used

1. New Encryption algorithm to Enhance Performance Parameters:

For the purpose of encryption as well as decryption, a suitable algorithm is being used. All the encryptions and decryptions will take place at the client side; ensuring transmission of the data will take place only in encrypted format over the complete network.

Stated algorithm works in 3 phases on a 128 bit data. Key required for either of encryption or decryption also has to be 128 bit. In-case the key or the data is less than 128 bit in size, the remaining bits are appended by zeroes. The higher the length of the key entered by the user the higher security would be provided by the stated algorithm.

Encryption Phase:

As stated above, encryption works in 3 phases as follows:

Phase 1:

1. Take any 128 bit plain text.
2. Now divide this plain text into 2 parts, 64 bits each.
3. Reverse each part and then swap both.
4. Now apply a circular shift operation on both the parts twice and again combine parts to get 128 bit data.
5. Select 128 bits key value.
6. Perform XOR operation between plain text and key value and the final results should be in text data form.

Phase 2:

1. The 128 bits obtained after key mixing are divided into 16 equal parts of 8 bits each.
2. Again divide each 8 bit block into 2 parts of 4 bits each.
3. Now combine all the left 4 bit blocks to get a 64 bit block and perform the same with the right ones to get another 64 bit block.
4. XOR both the 64 bit blocks. And the output is combined with the right 64 bit block (without any change) to obtain 128 bit text.
5. Repeat the process 1 to 4 for N number of cycles.
6. Then the final 128 bits are divided into 16 blocks of 8 bits each.
7. Each 8 bit block is then split into 2 parts, of 2 bits and 6 bits, and circular shift is performed on the last 6 bits of each block.
8. Combine the 2 bit part and the modified 6 bit part to get 8 bit block (16 blocks in all).
9. These blocks are combined to finally obtain a 128 bit cipher text.

Phase 3:

1. Select 128 bits key value.
2. Perform XOR operation between 128 bits key values and 128 bit cipher text (final results of phase 1).
3. Now divide this cipher text into 2 parts, 64 bits each.
4. Now apply the reverse circular shift operation on both the parts twice
5. Swap both parts and apply re-reverse operation on both parts.
6. Finally combine both parts to get 128 bit plain text data.

Decryption: Decryption is the just reverse process of the encryption.

Phase 1:

1. Take any 128 bit plain text.
2. Now divide this plain text into 2 parts, 64 bits each.
3. Reverse each part and then swap both.
4. Now apply a circular shift operation on both the parts twice and again combine parts to get 128 bit data.
5. Select 128 bits key value.
6. Perform XOR operation between plain text and key value and the final results should be in text data form.

Phase 2:

1. Select 128 bits cipher text.
2. The 128 bits cipher texts are divided into 16 equal parts of 8 bits each.
3. Each 8 bit cipher block is then split into 2 parts, of 2 bits and 6 bits.
4. Apply reverses circular shift on the second part of 6 bits
5. Combine the 2 bit part and the modified 6 bit part to get 8 bit block (16 blocks in all).
6. These blocks are combined to finally obtain a 128 bit cipher text.
7. Now divide this cipher text into 2 parts, 64 bits each, left and right.
8. XOR both the 64 bit blocks. And the output is a 64 bit block (without any change) to obtain left 64 bit
block 128 bit text.
9. After performing the XOR operation we will get both parts left and right parts of 64 bits each.
10. Now again divide both left and right 64 bits part into 4-4 bits part respectively.
11. Rearrange these 4 bits part in reversely to get original 64 bit parts (see architecture).
12. Finally combine all these blocks to get 128 bits.
13. Repeat process 7 to 12 for N number of cycles.
14. Then the final 128 bits will produce.

Phase 3:

1. Select 128 bits key value.
2. Perform XOR operation between 128 bits key values and 128 bit cipher text (final results of phase 1).
3. Now divide this cipher text into 2 parts, 64 bits each.
4. Now apply the reverse circular shift operation on both the parts twice
5. Swap both parts and apply re-reverse operation on both parts.
6. Finally combine both parts to get 128 bit plain text data.

VIII. Result Comparison

Here two different parameters are used to evaluate performance of the proposed system. First is avalanche effect and second is execution time. Comparison of results have been performed between proposed algorithm and existing algorithm. At the time of results evaluation, plain text and key value both were chosen randomly.

1. Avalanche Effect Comparisons: Avalanche Effect is the important property of security in cryptographic algorithms where, if an input is changed slightly (changing a single bit) the output changes significantly. In our case, we have chosen two different input plain text as “candidate” and “cendidate”

1.1 Proposed Algorithm

Plain Text : candidate
01011010101000000010111100111011000011111111100011101111111010000010011000100110010000111100000100111000101111110000111111000101

Change in Plain Text: cendidate
10100111111111101111011111111101110101000000011111000010000011111111111001110110011011100000111110110001110100000011110001111110101

Avalanche Effect: 69

2. Execution Time Comparison: Execution time is the time during which a program is running. In contrast to other phases of a program's lifecycle such as compile time, link time, load time, etc. i.e., time during which actual activity is done. Proposed algorithm is efficient in this means as well. It takes lesser time than that of other algorithms used for cryptographic techniques**.**

VIII. Future Enhancements

1. Principle of Orthogonality: Orthogonality in a programming language means that a relatively small set of primitive constructs can be combined in a relatively small number of ways to build the control and data structures of the language. The term is most-frequently used regarding assembly instruction sets, as orthogonal instruction set. In simple terms, orthogonality is the property that means "Changing A does not change B". An example of an orthogonal system would be a radio, where changing the station does not change the volume and vice-versa.

The same principle can be applied to the above proposed algorithm. As each of the encryption and decryption processes are carried out via various phases, at any future instant, any new phases can be added or any of the current 3 phases can be deleted in order to enhance future security. Addition of new phases or deletion of the current phases won’t make any difference in the actual overall working of the system.

2. Key: The size of 128 bit key used for encryption and decryption and be increased to any further size enhancing further security as encryptions as well as decryptions are dependent on the key. Using a key of higher size will also allow more data to be encrypted or decrypted at a same time, decreasing the encryption and decryption time further.
paper is styled.

IX. Conclusion

Now-a-days clients are more conscious about their privacy. They want their privacy is to be maintained in any way or the other. And this proposed system does the same with the help of client server architecture. The clients won’t have to worry about their privacy concerns as he/she is able to chat in secure manner using asymmetric cryptography. The encryptions of the text depend upon the encryptions of the key as result which help in achieving key dependency. Although any adversary if gains any access to the cipher, he may not be able to decrypt it into plain text.

Acknowledgement

We are thankful to all helping hands in completion of this project. We would like to express our sincere thanks to all those who have provided us with valuable guidance towards completion of project.

REFERENCES

[1] Tal Malkin, Isamu Teranishi, Moti Yung "Key Dependent Message Security: Recent Results and Applications"
[2] Rajni Jain, Ajit Shrivastava "Design and Implementation of New Encryption algorithm to Enhance Performance Parameters" ISSN: 2278-0661 Volume 4, Issue 5 (Sep.-Oct. 2012), PP 33-39.
[3] William Stallings, Cryptography and Network Security: Principles Practices, Second edition.

Authors

Nandanwar Aniket D.
B.E.Computer
University of Pune
Department of Computer Engg.
Govt. College of Engg, & Research,
Awasari (KD),
Tal- Ambegaon, Dist-Pune. India.

Deshmukh Sonali S.
B.E.Computer
University of Pune
Department of Computer Engg.
Govt. College of Engg, & Research,
Awasari (KD),
Tal- Ambegaon, Dist-Pune. India.

Hadawale Rashmi R.
B.E.Computer
University of Pune
Department of Computer Engg.
Govt. College of Engg, & Research,
Awasari (KD),
Tal- Ambegaon, Dist-Pune. India.

Nanaware Gauri B.
B.E.Computer
University of Pune
Department of Computer Engg.
Govt. College of Engg, & Research,
Awasari (KD),
Tal- Ambegaon, Dist-Pune. India.

