An alternative technique to thwart multiplicity attack in Denning-Sacco protocol using timestamps

Pranav Vyas, Dr. Bhushan Trivedi, Dr. Atul Patel

Abstract— Cryptographic protocols are used to encrypt information and thus securing them while they are being transferred from one host to another. However to understand information they need to be decrypted. Decryption process needs a key by which sender earlier encrypted information. So, the exchange of this key is very crucial to security of communication session. One of protocol which facilitates secure key exchange is Denning-Sacco protocol. The original Denning-Sacco protocol however is vulnerable to multiplicity There is an existing method which solves this problem with use of nonce. In this paper we present an alternative technique to solve the problem without using nonce and thus reducing number of steps of algorithm resulting in faster execution of algorithm.

Index Terms— Key Exchange, Key Distribution, Denning-Sacco, Multiplicity Attack, Timestamps, Security, Key Exchange Protocols,Key Management

1 INTRODUCTION

Today secure communication is need of hour. This can be achieved by encrypting information when exchanging it over insecure communication channel. To decrypt information on the other side a key used earlier to encrypt the information is needed. One way to exchange key is proposed by [1]. In the paper author proposes user to protect a small hardware unit containing his/her secret key. Here, security of key depends on security of hardware unit by user. Another approach to exchange key relies on security and correctness of network and principally its key management facilities. These approaches were initially discussed by [2] & [3].

Hardware key generator systems have limitation of battery and can be used by malicious user easily if he/she can get physical access to key generator device. Hardware key generators have limited computing capacity and can only generate fixed length key which may not be altered for length to make them more secure over the time. Network based protocols for key exchanges do not suffer from these limitations. We limit scope of our discussion to protocols for key exchange over network. In [4], authors analyze number of key exchange algorithms on various parameters. According to simulation results by [4] Denning Sacco protocol is most suitable protocol for key exchange for mobile computers.

In 1978 Needham and Schroeder proposed a key exchange protocol usable for both single key distribution and public key systems [3]. The protocol uses a trusted third party to protect secrecy of keys exchanged between two parties. Each party here has two keys: public and private key. Private key is one known by trusted their party, public key is known to all parties on network According to protocol key generation and distribution are responsibilities of trusted third party. Trusted party uses private key of each party registered with it to encrypt the key. It is however assumed that the private keys are never compromised thus maintaining secrecy of communication [5].

Protocol proposed by Denning and Sacco [5] raises issue of compromising private keys. The authors show how a reply type of attack can be used once private keys are compromised. They propose solution of problem with use of timestamps in communication. Another purpose for using timestamps according to authors is to replace two phase handshake which was used earlier to prevent replay attacks from taking place.

The protocol proposed by Denning and Sacco [5] is still subject to multiplicity attack according to Lowe [6]. It makes use of nonce to thwart the attack by sending nonce encrypted with a private key and receiving it back encrypted in shared session key.

This solution includes two extra steps one for each party. Considering the fact that we are in age where mobile devices are used for access and sharing of information. These devices have limited battery power. To make most efficient use of power available optimization system processes.

This paper proposes an alternative solution to thwart multiplicity attack. We propose solution using timestamps that are already being used in the algorithm. In order for this solution to work, both the parties need s to be synchronized. We propose this synchronization to be done at the time of key distribution by trusted third party.

This paper is divided into 5 segments. We start with introduction, giving idea on problem and how we propose to solve it. Second section discusses the protocol proposed by [5]. In section 3 we discuss multiplicity attack and technique to thwart it as proposed by [6]. In section 4 we discuss alternative solution proposed. Section 6 deals with results of experiment where we compare how protocol [5], [6] and proposed protocol performs. In section 7 we present findings and draw conclusion.

2 ORIGINAL DENNING-SACCO PROTOCOL

2.1 Review Stage

The original Needham-Schroeder protocol was designed to exchange secret keys on public key systems. However, the protocol had a crucial design flow which left it vulnerable to "Reply attack" [5]. Denning-Sacco protocol proposes a solution to this vulnerability. The original Needham-Schroeder protocol is as follows:

1. A →S: A, B, I_A
2. S →A: { I_A, B, KAB, {A,KAB}KBS}KAS
3. A →B: {A, KAB}KBS

Here, IA is a unique identifier which is used to make sure that the response from S is not reply of previous response. Here, B has no way to determine identity of A. B cannot know if the messages are coming from A are not reply of previous messages.

The original Needham-Schroeder proposes a handshake mechanism shown below to solve the problem discussed earlier.

1. B →A: { I_B}KAB
2. A →B: { I_B+1}KAB

As Denning-Sacco points out, the handshake mechanism cannot work if C intercepts message in step (1).

Denning-Sacco [1] tries to solve this problem by use of timestamps when sending messages. They argue that an interval of about 1-2 minutes should suffice if all nodes in network set their clocks manually by reference to standard clock. They give two solutions for key distribution: public keys and communication keys.

Following are steps for distribution of public key between A, B and trusted third party S:

1. A → S: A, B
2. S → A: {A, P_A, T}KSS, {B, P_B, T}KSS
3. A →B: {A, P_A, T}KSS, {B, P_B, T}KSS

Here, PA and PB are public keys of A and B. KSS is secret keys of S. To prevent forgery S certifies message by signing it with its secret key. Timestamps are used to thwart reply attacks.

Following are steps for distribution of communication key to A and B from trusted third party S:

1. A →S: A,B
2. S →A: {A, P_A, T}KSS, {B, P_B, T}KSS
3. A → B: {A, P_A, T}KSS, {B, P_B, T}KSS, {{SAB, T}SA}KBS

Here, SAB is secret key which will be used to encrypt all further communication in the current session. KSS is used here to deliver message from trusted third party S.

3 MULTIPLICITY ATTACK

As discussed by Lowe [6] initial version of Denning-Sacco protocol was vulnerable to a type of reply attack. Following is original version of Denning-Sacco protocol:

1. A→AS: A,B
2. AS→A: {B, KAB, T, {KAB, A, T}KBS}KAS
3. A→B: {KAB, A, T}KBS

In step 3 of the above protocol, B has no means to authenticate message coming from A. If we assume that malicious user captures this message and then resends it then B will still assume that message as is coming from A and will open another channel for communication (Step 3(a)).

a. M → B {KAB,A, T}KBS

Multiplicity attack was first mentioned by Lowe in [6].

3.1 Lowe's solution to multiplicity attack

Lowe also suggests possible solution to thwart this attack. They suggest using nonce based handshake to verify identity of A. This approach adds two new steps to original protocol (Step 4).

1. B → A {Nb}KAB

Here, B sends a nonce encrypting it using the secret shared key between A and B. It expects reply from A in decrypted form. If the received nonce is the same as the one sent by B, it authenticates A (Step 5).

5. A → B {dec{Nb}}KAB

Lowe's solution uses two steps more than proposed by original protocol algorithm. When we consider the possibility of using Lowe's modified protocol on mobile device, it results in increased utilization of processor for encrypting and decrypting activity. Since, mobile devices have limited battery it should be conserved as much as possible. This solution uses more battery then original protocol and thus making battery utilization on mobile device inefficient.

3.2 Out Alternative solution of multiplicity attack

We propose an alternative solution to thwart multiplicity attack. We make use of timestamp in the protocol. We compare value of timestamp with value of system time on receiving system. We check the difference and determine freshness of message.

However, it is possible that all nodes in network may not be synchronized. So there is a possibility of different nodes running different clocks. To solve synchronization problem we synchronize clocks of both A and B with trusted third party AS. At the time of sending message, AS adds synchronization sequence which can be used by A to synchronize its clock with AS. AS also sends a separate message to B with synchronization sequence. This sequence is encrypted by private key of B which is only known to AS and B.

The complete protocol with alternative solution is as follows:

1. A → S: A,B
2. S → A: {B, KAB, T, Sync, {KAB, A, T} KBS}KAS
3. S→ B: {A, Sync} KBS
4. A→B:{KAB, A, T}KBS

5 CONCLUSION

In this paper we show an alternative approach to solve the problem of multiplicity attack. Earlier solution aims to solve problem by means of nonce. It resulted in two extra steps in algorithms and thus extra messages in protocol, thus using more processing power and battery backup of mobile devices.

Our solution uses timestamp that is part of protocol. So no extra messages are exchanged between two hosts. We solve clock synchronization problem by synchronizing clocks of both A and B with S by using synchronization sequence which is sent to both parties, this will synchronize their system clocks. This will allow efficient usage of existing timestamps and thwart any type of reply attacks.

The proposed alternative solution satisfies all the properties that are satisfied by earlier solution proposed by Lowe [6]. Security of the solution lies in its ability to communicate with inclusion of timestamps. If the malicious user intercepts a message and then tries to send the same message to B, based on timestamp included in message B will be able to detect the message and discard it, if the difference of time more than threshold value. However, usage of timestamp is only effective if the machines which are taking part in communication process are synchronized. Synchronization is achieved by synchronizing both the nodes with clock of trusted third party server. Once their clocks are synchronized, the nodes can easily detect duplicate messages and discard them.

REFERENCES

[1] D. E. Denning, "Secure Personal Computing in Insecure Network," Communications of ACM, vol. 22, no. 8, pp. 476-482, 1979.

[2] C. S. K. G. J. Popek, Operating Systems - Design Issues for Secure Computer Networks, London: Springer-Verlag, 1978.

[3] R. M. Needham and M. Schroeder, "Using encryption for authentication in large networks of computer," Communication of ACM, vol. 21, no. 12, pp. 993-999, 1978.

[4] B. T. A. p. Pranav Vyas, "Simulation Analaysis of Session Key Exchange Protocols based on Key Parameters," International Journal of Computer Applications, vol. 68, no. 1, pp. 46-52, 2013.

[5] D. E. Denning and G. Sacco, "Timestamps in key distributed protocols," Communication of ACM, vol. 24, no. 8, pp. 533-535, 1981.

[6] G. Lowe, "A family of attacks upon authentication protocols," Department of Mathematics and Computer Science, University of Leicester, Leicester, 1997.