A Key Distribution Protocol Applying a Haar-like Transform (KDPH)

Magdy Saeb

Computer Engineering Department,

Arab Academy of Science, Technology and Maritime Transport

mail@magdysaeb.net

Abstract: Key distribution is a central cryptographic problem in data communication security. In this work, we present a key distribution protocol employing a Haar-like transform. The shared secret is a number of elements of the reconstructing trend and fluctuations vectors. The protocol security is based on the fact that the reconstruction of the elements of the vector, used to build the secret message, is double the number of the elements constituting the shared secret. This shared secret is updated whenever a new key is generated. The proposed technique uses simple arithmetic operations that provide uncomplicated and efficient software and hardware implementations.

Keywords: Key Distribution, Haar Transform, Cryptography, Authentication, Encryption.

1. Introduction

The main objective of cryptography is to provide security and authentication between communicating entities. In order to achieve this in the presence of an active adversary, the two communicating parties must have a shared secret. This shared secret is an encryption key. The key should be available to the communicating parties before the communication session starts. Key distribution is one of the most analyzed problems in data security. The problem was first studied by Needham and Schroeder [1]. Other works followed such as Baur et al. [2] who presented a key distribution protocol using event markers where communicants do not have to keep absolute sense of time. Bellare and Rogaway [3, 4] provided a discussion on the session key distribution in the three party setting of Nedham and Schroeder. They presented a definition, a protocol and a proof that the protocol satisfies the definition assuming minimal assumption of a pseudorandom function. In their second paper they discussed the problems of two-party mutual authentication and key distribution in the complexity-theoretic framework of modern cryptography, Huang [5] presented a key pre-distribution scheme for secure wireless sensor networks. The paper provides an approach that any pair of sensor nodes can find a common pair wise secret key between them with simple calculation. Chang et.al [6] presented a conference key distribution based on interpolation polynomials. A sealed lock is used to lock the conference key in such a way that only the private keys of the invited members are matched. The sealed lock is then made public or distributed to all. Only legitimate users can disclose it and obtain the conference key. Liu et al. [7] proposed a practical deployment model, where sensor nodes are deployed in groups, and the nodes in the same group are close to each other after the deployment. Based on this model, the paper develops a novel group-based key pre-distribution framework, which can be combined

Received 12/6 , Reviewed 12/25

with any of existing key redistribution techniques. Wahiddin et. al. [8] used satellite random transmissions, Von Neumann corrector and a hash function to generate the key between communicating entities. Cervesato et al. [9] discussed a method that assembles the security properties of a protocol by composing the guarantees offered by embedded fragments and patterns. It sheds light on fundamental notions such as challenge-response and fed a growing taxonomy of protocols. Sun et al. [10] proposed three secure authentication and key distribution protocols to provide perfect forward secrecy of these three classes. All these protocols are used in protecting poorly-chosen passwords chosen by users from guessing attacks and replay attacks.

In the present work we focus on the two communicating parties' symmetric case. We propose a key distribution protocol based on a Haar-like transform where the shared secret is some elements of the reconstructing vectors. The relative simplicity of the method contributes to uncomplicated software and hardware implementations. In the following sections we discuss the methodology of the proposed technique, the adversary model and the proposed protocol implementation. Finally, we provide a summary and our conclusions.

2. Methodology

Like all wavelet transforms, the Haar transform decomposes a vector of discrete set of integers into two vectors of half its length. One vector represents the running average or trend and the other represents running difference or the fluctuations. For example one considers the vector $\mathbf{V} = (v_1, v_2, v_3, v_4, v_5, v_6)$, then the trend vector $\mathbf{t} = (t1= (v_1 + v_2)/2, t2= (v_3 + v_4)/2, t3= (v_5 + v_6)/2)$ and the fluctuation vector $\mathbf{f} = ((v_1 - v_2)/2, f2= (v_3 - v_4)/2, f3= (v_5 - v_6)/2))$. Now, if **t** and **f** are known, then the vector **V** can be reconstructed**,** simply, by adding and subtracting the respective elements of **t** and **f.** The original Haar transform multiply the trend and fluctuations by a factor of $\sqrt{2}$ to keep the energy before and after the transform invariant. However, for key distribution, we will not use this factor. Now, if some of the values of the trend were available only to the communicating entities, namely Alice and Bob, then the eavesdropper Eve, we hypothesize, will not be able to reconstruct the vector **V** even if she was able to intercept and store the other values of the trend and fluctuations vectors. As a matter of fact for a number of **t** and **f** elements hidden from the attacker, say k, there are 2k vector **V** elements that will not be computable by Eve. This is quite apparent since to generate the two elements v_i and v_{i+1}, one needs to add and subtract one element from the **t** vector and one element from the **f** vector. As a result the secret is doubled and accordingly the security is appreciably enhanced. In addition, these hidden values serve as an authentication tool since the two communicating entities are the sole owners of these secret vector elements. The addition and subtraction operations are quite easy to implement. Actually the data is to be represented in two's complement. The two's-complement system has the advantage of not requiring that the addition and subtraction circuitry examine the signs of the operands to determine whether to add or subtract. This property makes the system both simpler to implement and capable of easily handling higher precision arithmetic. In addition, the zero will have a single representation, eliminating the required adjustments associated with negative zero, which exists in ones'-complement systems. In the next few lines, we formally describe the protocol as shown in Table 1. The shared secret between Alice and Bob can be considered as a group key or an initialization value of the protocol. This value is updated, using part of the pseudo-randomly generated key.

Table 1: Formal Description of the Key Distribution Protocol Using a HAAR-like Transform (KDHP)

Step	Alice	Bob
1	Acquire, through a secure channel, the shared secret (t_k, f_j)	Acquire, through a secure channel, the shared secret (t_k, f_j)
2	t_i , f → B Send t_i , f_j, $i \neq k$, $i \neq J$	Receive t_i , f_j, $i \neq k$, $i \neq j$
3	Assemble **V** (by adding and subtracting t_i, f_i including t_k, f_k)	Assemble **V** (by adding and subtracting t_i, f_i including t_k, f_k)
4	Concatenate elements of **V** (using operator ||) to get the secret message σ where $\sigma = v_1\|\|v_2\|\|\ldots\ldots\|\|v_n$	Concatenate elements of **V** (using operator ||) to get the secret message σ where $\sigma = v_1\|\|v_2\|\|\ldots\ldots\|\|v_n$
5	K = h(σ), where h is a hash function	K = h(σ), where h is a hash function
6	Update the Shared secret (using part of the generated key)	Update the Shared secret (using part of the generated key)

3. The Adversary Model

Assume that there are two nodes, the system generates a number of secret pairs of t_k and f_j and distributes the shared secret through some secure means. In this scheme, the system uses distinct secret pairs to create the secret message that is to be hashed to generate the key. The security of the presented scheme is based on these secret pairs. By applying the Haar-like procedure mentioned above, the node which owns the secret pairs, can reconstruct the vector **V**. This vector provides the seed for the key to be generated locally in this node. In other words, a set of pairs lacking the secret pairs cannot be used to reconstruct the vector **V**. There are two extreme cases in this protocol; the first one is that all the elements of the vectors **t** and **f** are secret. In this case the scheme is provably secure since Eve cannot get any information regarding the key. The second extreme case is that none of the elements of **t** and **f** are kept secret. In this case the protocol is not secure since Eve can construct the secret message based on the vector **V**. In practice, the protocol requires that a finite number of the trend and fluctuations vectors are kept secret and updated whenever a new key is locally generated. The effort that Eve has to spend to guess these secret pairs is, for say a 16-bit integer pairs, is equivalent to perform $2 \times 2^{16} = 2^{17}$trials. However for 32-bit integers, the number of trials is increased to $2 \times 2^{32} = 2^{33}$ trials. These computations are based on the assumption that we have two unknown integers only. In practice, if we use k secret integers, the result increases dramatically since we have to use the correct integer in the correct location to get **V**. Therefore, we hypothesize that the protocol is secure for all practical purposes and Eve will spend an indefinite amount of time trying to generate the key. This can be demonstrated using the following theorem.

Theorem 3.1:

Assume that the size of the vector **V** is n elements and there are $k_f = k_t$ elements representing the shared secret. Then the probability (P) of guessing the correct locations of these pairs of the secret is given by:

$P = Pr\{\text{guessing the correct locations of the secret pairs } k_f, k_t\} = (1/n/2)^{k}_{t} .(1/n/2)^{k}_{f}$

$$= (2/n)^{(k_t + k_f)}$$

$$= (2/n)^{(k_t + k_f)}$$

$$= 2^{(k_t + k_f)} \cdot (1/n)^{(k_t + k_f)}$$

For example, if n = 200 and $k_f = k_t = 10$, then P is equal to $2^{20} \cdot (1/200)^{20} = 1 \times 10^{-40}$

Based on this theorem, one concludes that Eve can guess the locations of the shared secret pairs with an infinitesimal probability. Taking this into consideration along with the number of trials required to guess the values of the secret pairs, one arrives at the conclusion that for Eve to guess the values and the correct locations of the shared secret pairs, she will spend a very long undetermined time.

4. Implementation

Suppose that **V** is composed of, say, n elements then the trend and fluctuations vectors will be of length n/2 elements. Now if the shared secret used k elements of the trend and k elements of the fluctuations, then to reconstruct the original vector the number of the elements that Eve cannot reconstruct will be equal to 2k. Other elements are available to Eve since their trend and fluctuations were transmitted on an unsecure channel. Therefore from Eve perspective, the secret message that is to be hashed to generate the key is not computable as was shown in section 3. We recommend using 16-bit integer data types represented in two's complement format to implement this protocol. However, larger size integers will add to the security, however, on the expense of added storage requirements. As a simplified example, shown here for illustration purposes only, one considers the trend vector **t** = (1, 2, 3) and the fluctuation vector **f** = (4, 5, 6) then the original vector **V** is given by ((1+4)/2, (1-4)/2, (2+5)/2, (2-5)/2, (3+6)/2, (3-6)/2) = (5/2, -3/2, 7/2, -3/2, 9/2, -3/2). One can get the secret message from this vector by discarding the sign and taking the ceiling of the result of the vector elements and concatenate them together. The result, in decimal format, would be 324252. If this message is hashed, say using SHA1, then the resulting key will be 6cb93958ab6eb42f3669a2fb3e7a190c770a73b1. One notices that if the elements t_3 and f_1 were the shared secret between Alice and Bob, then the vector elements v_1, v_2, v_5, v_6 cannot be computed by Eve. In addition, one can use the binary representation of the vector elements directly as elements of the secret message. This implementation will provide better security, with its underlying random behavior, than the previously mentioned method.

Summary & Conclusions

In this work, we have proposed a secure and simple method for symmetric key exchange between two communicating entities. We have demonstrated that an active attacker will possess infinitesimal probability of guessing the key. The operations involved in producing the secret message are simply addition and subtractions of the elemental values. This will result in simple hardware and software implementations.

References

[1] R. M. Needham, M. D. Schroeder, "Using Encryption in Large Networks of Computers," Communication ACM21, pp.993-999, 1978.

[2] R. K. Bauer, T. A. Berson, R. J. Freietag, " A Key Distribution Protocol Using Event Markers," ACM Transactions on

Computer Systems, Vol. 1 Number 3, pp. 249-255, August 1983.

[3] M. Bellare, P. Rogaway, " Entity Authentication and key Distribution," Advances in Cryptology, Crypto93, Proceedings, Springer-Verlag, 1993.

[4] M. Bellare, R. Canetti, H. Krawczyk, "A Modular Approach for the Design and Analysis of Authentication and Key Exchange protocols," Proceedings of the 30 th Annual Symposium On the Theory of Computing, ACM, 1998.

[5] H. Huang, " A Pairwise key Pre-distribution for Wireless Sensor networks," ISI 2008 Workshops, LNCS 5075, pp. 77–82, 2008.

[6] C. Chang, C. Lin, C. Chen, "A Conference Key Distribution Scheme Using Interpolating Polynomials," International Journal of Security and its Applications, Vol. 1, No. 2, October, 2007.

[7] D. Liu, P. Neng, W. Du, "GroupBased Key PreDistribution in Wireless Sensor Networks," WiSE'05, Cologne, Germany. September 2, 2005.

[8] M. R. Wahiddin, N. S. Noor Sham, M. Saeb, M. Hamdan, " A Protocol for Secret Key Infusion from Satellite Transmissions," International Journal of Computer and Network Security(IJCNS), Vol.2, No.7, July 2010.

[9] I. Cervesato, C. Meadows, and D. Pavlovic, "An Encapsulated Authentication Logic for Reasoning About Key Distribution Protocol," Eighteenth Computer Security Foundations Workshop, IEEE Computer Society, Press, CSFW-18, pages 48–61, Aix-en-Provence, France, 2005.

[10] H. Sun, H. Yeh, "Password-based authentication and key distribution protocols with perfect forward secrecy," Journal of Computer and System Sciences 72, 1002–1011, 2006.

Magdy Saeb received the BSEE, School of Engineering, Cairo University, in 1974, the MSEE, and Ph.D. degrees in Electrical & Computer Engineering, University of California, Irvine, in 1981 and 1985, respectively. He was with Kaiser Aerospace and Electronics, Irvine California, and The Atomic Energy Establishment, Anshas, Egypt. Currently, he is a professor in the Department of Computer Engineering, Arab Academy for Science, Technology & Maritime Transport, Alexandria, Egypt; He was on-leave working as a principal researcher in the Malaysian Institute of Microelectronic Systems (MIMOS). His current research interests include Cryptography, FPGA Implementations of Cryptography and Steganography Data Security Techniques, Encryption Processors, Mobile Agent Security.

www.magdysaeb.net.