ENHANCED VERIFIER-BASED PASSWORD AUTHENTICATED KEY AGREEMENT PROTOCOL FOR THREE-PARTIES

Dina Nabil Shaban, Maged H. Ibrahim and Zaki B.Nossair
Department of Electronics, Communications and Computers Engineering, Faculty of Engineering at Helwan, Helwan University.
eng.dinanabil@yahoo.com, mhii72@yahoo.com, znossair@yahoo.com

(Received August 21, 2008 Accepted October 15, 2008)

Most of password authenticated key agreement protocols have focused on the two-party setting where two communicating parties share a password. However, in the two-party setting any user may want to communicate with other users who have not shared the same password. In this paper we present an efficient verifier-based password authenticated key agreement protocol for three-parties. In the three-party setting each user only shares a password with a trusted server which authenticates two users and helps the users with different passwords share a common session key. Our proposed protocol is secure against several attacks and provides perfect forward secrecy.

KEYWORD: *authentication, cryptanalysis, key agreement, password, three-party.*

1. INTRODUCTION

To communicate securely over an insecure public network, it is essential that secret session keys be securely exchanged. The shared session key may be subsequently used to achieve some cryptographic goals such as confidentiality or data integrity. In the public-key based and symmetric-key based key agreement protocols, a party has to keep long random secret keys. However, it is difficult for a person to memorize a long random string. Thus, a party uses an additional storage device to keep the random string. On the other hand, password-authenticated key agreement protocols allow two or more specified parties to share a secret session key using only a human-memorable password. Password-authenticated key agreement protocols provide a new and unique way to authenticate parties and derive high-quality cryptographic keys from low-entropy passwords. Most of designs for password-authenticated key agreement protocols for two-party setting assume that the two parties have a pre-shared password. In this setting any user may want to communicate securely with many other users who have not shared the same password. If a user shares a password with the other user, the number of passwords that the user has to memorize linearly increases with the number of possible partners. Password-authenticated key agreement in the three-party setting surmounts all the above mentioned problems. In three-party setting, each user only shares a password with a trusted server. The trusted server authenticates two users and helps the users with different passwords share a common session key which means that each user only has to share a single password with the trusted server. Consequently, three-party password-authenticated key agreement protocols can limit the number of

passwords that each user must memorize. However, the server has to participate during the protocol run to help the two users share a session key.

Password-based authenticated key agreement protocols can be classified into two types of models according to the difference of knowledge of a shared password between each user and a server: symmetric and asymmetric (or verifier-based) models. In a symmetric model, each user and a server use the same knowledge related with a password to authenticate each other. In a verifier-based model, each user has a password; whereas a server has an image (called a verifier) of the password which is computed using a one-way function instead of the password itself.

If in a symmetric model the server is compromised, an adversary with the server's password file can immediately masquerade as a legitimate user by using the password in the password file .This is called a server compromise problem [15]. However in a verifier-based model a server has verifiers of the passwords instead of the passwords themselves, so the server compromise does not directly reveal the passwords.

2. RELATED WORK

The well best-known protocol for key agreement is the Diffie-Hellman protocol [1], which allows two parties to establish a shared secret by exchanging messages over an insecure channel without the need for any prior communication. However, the basic Diffie-Hellman protocol has a weakness of possible man-in-the middle attack. To solve this problem, many authenticated key agreement protocols using certificates [2] or pre-shared secret passwords between two parties [3, 4, 5] have been put forward over the past years. Although two-party setting is quite useful for client-server architectures, it is not suitable for client-client architectures where the two-party setting requires every pair of communication entities to share a password, it is very inconvenient in key agreement for large scale communication environments. To avoid this inconvenience, some three-party password-based authenticated key agreement protocols have been extensively studied in the last few years**.** In 1995, Steiner et al. [9] proposed the first three-party encrypted key exchange protocol (3PEKE) based on Encrypted Key Exchange called EKE [6]. In Steiner et al.'s protocol, called STW-3PEKE, clients share a password with a trusted server S only and in which S mediates between two parties to allow their mutual authentication. However, Ding and Horster [10] showed in 1995 that the STW-3PEKE is vulnerable to undetectable on-line password guessing attacks. Lin et al. modified STW-3PEKE, named LSH-3PEKE, by employing the public-key cryptosystem to avoid the password guessing attack. In 2001, Lin et al. [11] proposed a new encrypted key exchange protocol for three parties, called LSSH-3PEKE, which is resistant to password guessing attacks and does not require server's public key. Recently, Sun et al. [12] proposed two improved 3PEKE protocols, called SCH-3PEKE, respectively based on the password and the verifier. A security weakness of the SCH-3PEKE scheme was recently revealed by Nam et al. [13]. Also Kulkarni et al. [14] has proposed a three-party key agreement protocol in which each user stores one way hash function of the password at the server rather than storing the password itself. Kulkarni et al.'s protocol is secure against online and dictionary attacks. However, as it will be shown in section 7, Kulkarni et al.'s protocol is computationally more expensive than our proposed protocol.

3. MOTIVATION AND CONTRIBUTION

3.1. Motivation

Recently, Lu et al. [8] have proposed a modified protocol to Kim et al.'s protocol [7] which does not resist the off-line password guessing attack. The Lu et al.'s protocol is a two-party setting for client-client architectures which require a shared password between every pair of communicating entities and these will increase the number of passwords that the client has to memorize. Therefore, in this paper we introduce Lu et al.'s protocol in the three-party setting in which each client needs to memorize only single password with trusted server.

3.1. Contribution

In order to limit the number of passwords that each user needs to remember, in this paper we propose a new efficient verifier-based key agreement protocol for three parties, where each user only shares a password with a trusted server. The main advantage of this solution is that it provides each user with the capability of communicating securely with other users in the system while only requiring him to remember a single password. This seems to be a more realistic scenario in practice than the one in which users are expected to share multiple passwords. In the proposed protocol we assume that the server is honest but curious, which means that, even though the server is helping to establish a session key between two users, the server should not be able to gain any information on the value of that session key. The proposed protocol can resist various attacks such as password guessing, replay attacks, known-key attacks, and server compromise and provide a perfect forward secrecy.

The remainder of this paper is organized as follows. In Section 4, we describe the model of the proposed protocol. In Section 5, we present the proposed protocol. In Section 6, we show security analysis of our protocol. In Section 7, we analyze of the efficiency of the proposed protocol. Finally, Section 8 gives our conclusions.

4. THE MODEL

In the communication model there are three parties, Alice, Bob (clients) and AS (Authentication Server). Each client holds password π, while the AS does store a verifier of the password in its database. All the parities are connecting over an insecure public network but the verifier of the password is sent to AS via a secure and authenticated channel.

In the adversary model, we assume passive adversary and active adversary. A passive adversary is the adversary can eavesdrop on the honest parties' communication, but cannot actively modify it. An active adversary is the adversary that, in addition to eavesdropping, can insert, deletes, or arbitrarily modify messages sent from one user to another.

5. OUR PROPOSED PROTOCOL

In this section, we present an efficient verifier-based key agreement protocol for three parties.

In the proposed protocol, each client uses a memorable password, while the server does store verifiers instead of plaintext-equivalent passwords to resist to server compromise [15]. The proposed protocol is shown in Fig.1.

5.1. Notations

The notations used in the proposed protocol are described as follows:

A, B, and S	The identifiers of Alice, Bob, and AS.
V	A verifier that is computed from a password π.
n	Large prime number.
g	Generator in the cyclic group Z^*_n.
$H(\ )$	One-way hash function.

5.2. The protocol.

5.2.1. Initialization Phase

For registering for AS, Alice and Bob respectively choose passwords π_A and π_B, compute verifiers $V_A = g^{H(A, S, \pi_A)}$ and $V_B = g^{H(B, S, \pi_B)}$, and then send V_A and V_B to AS over a secure channel then AS stores V_A and V_B in a password's table. The registration method is out-of-scope in the protocol design.

5.2.2. Session key agreement phase

Assume Alice wants to establish a session key with Bob.

1. Alice computes $X_A = g^a \bmod n$ by choosing $a \in_R Z^*_n$ and then sends A, B and X_A to AS, similarly Bob computes $X_B = g^b \bmod n$ by choosing $b \in_R Z^*_n$ and sends B, A and X_B to AS.

2. After receiving the messages from Alice and Bob, AS retrieves V_A and V_B from a password table, checks whether X_A or X_B equal to V_A or V_B, if they hold, AS terminates otherwise moves to the next step which is compute $X_{SA} = (V_A)^c \bmod n$ and $X_{SB} = (V_B)^d \bmod n$ by choosing c, $d \in_R Z^*_n$ and sends X_{SA} , X_B to Alice and X_{SB} , X_A to Bob, respectively. While waiting for messages from Alice and Bob, AS computes $K_{SA} = (X_A)^c = g^{ac} \bmod n$ and $K_{SB} = (X_B)^d = g^{bd} \bmod n$.

3. After receiving the messages from AS and Bob, Alice computes $K_{AS} = ((X_{AS})^{t_A^{-1}})^a = (g^c)^a = g^{ca} \bmod n$ and $V_{AS} = H(A, B, S, X_A, X_B, K_{SA}, 0)$ and sends V_{AS} to AS. Similarly, after receiving the message from AS, Bob computes $K_{BS} = ((X_{BS})^{t_B^{-1}})^b = (g^d)^b = g^{db} \bmod n$ and $V_{BS} = H(A, B, S, X_A, X_B, K_{SB}, 0)$ and sends V_{BS} to AS.

4. After receiving the messages from Alice and Bob, AS checks whether $V_{AS} = H(A, B, S, X_A, X_B, K_{SA}, 0)$ and $V_{BS} = H(A, B, S, X_A, X_B, K_{SB}, 0)$ hold or not. If they hold, AS is convinced that Alice and Bob are validated. Then, AS computes $V_{SA} = H(A, B, S, X_A, X_B, K_{SA}, 1)$ and $V_{SB} = H(A, B, S, X_A, X_B, K_{SB}, 1)$ and sends V_{SA} and V_{SB} to Alice and Bob, respectively.

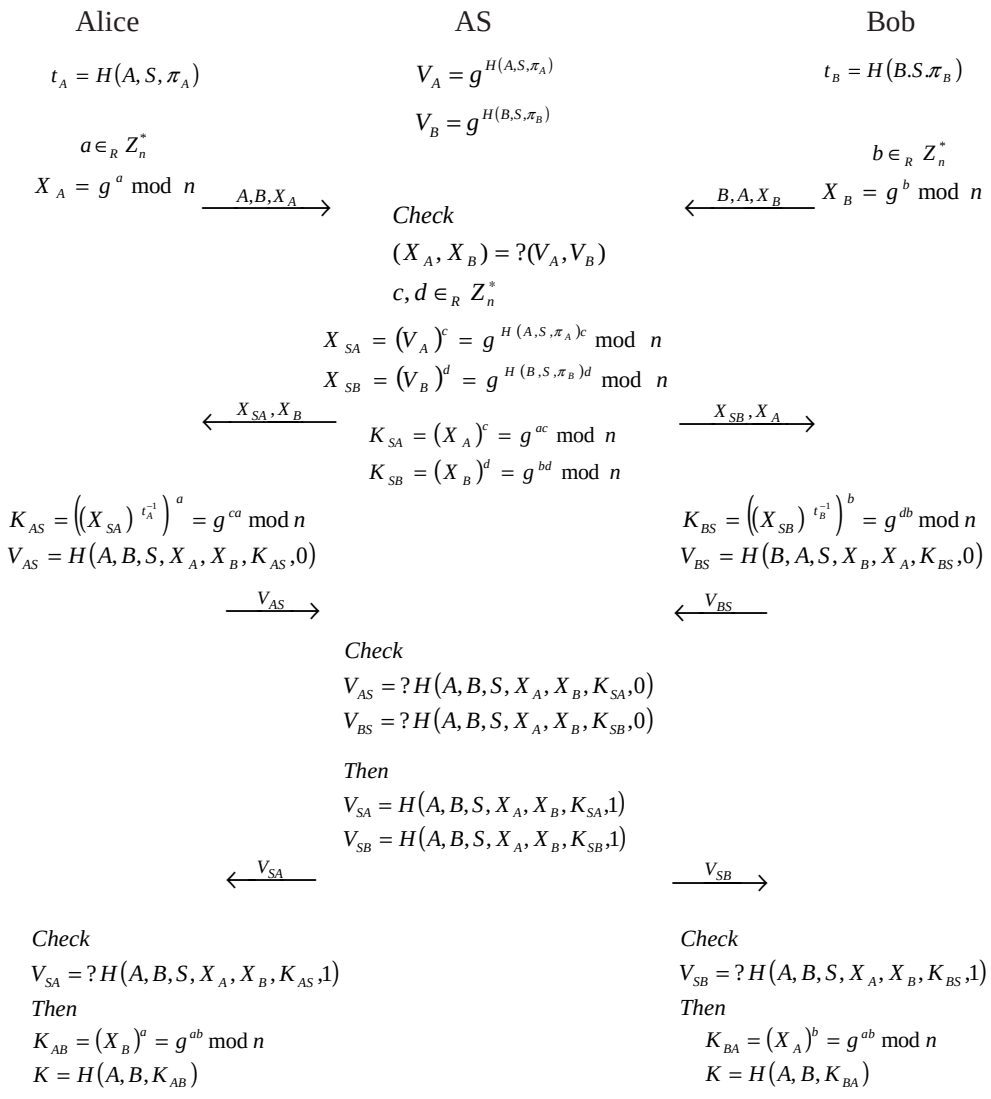

Fig. 1. The proposed protocol.

5.2.3. Key computation Phase

After receiving the message from AS, Alice checks whether $V_{SA} = H(A, B, S, X_A, X_B, K_{AS}, 1)$ holds or not. If they hold, Alice is convinced that both Bob and AS are validated. Similarly, after receiving the message from AS, Bob checks whether $V_{SB} = H(A, B, S, X_A, X_B, K_{BS}, 1)$ holds or not. If they hold, Bob is convinced that both Alice and AS are validated. Finally, Alice and Bob compute $K_{AB} = (X_B)^a = g^{ab} \bmod n$ and $K_{BA} = (X_A)^b = g^{ab} \bmod n$, and then compute a common session key $K = H(A,B,K_{AB}) = H(A,B,K_{BA}) = H(A,B,S, g^{ab} \bmod n)$, respectively. If they hold, Bob is convinced that both Alice and AS are validated.

Finally, Alice and Bob compute $K_{AB} = (X_B)^a = g^{ab} \bmod n$ and $K_{BA} = (X_A)^b = g^{ab} \bmod n$, and then compute a common session key $K = H(A,B,K_{AB}) = H(A,B,K_{BA}) = H(A,B,S, g^{ab} \bmod n)$, respectively.

To enhance the efficiency of the protocol t, t^{-1} and V can be pre-computed by each client before the protocol runs. The proposed protocol needs message exchanges of four rounds as follows:

(1) Alice → AS: $\{A, B, X_A\}$, Bob → AS: $\{B, A, X_B\}$, (2) AS → Alice: $\{X_{SA}, X_B\}$, AS → Bob: $\{X_{SB}, X_A\}$, (3) Alice → AS: $\{V_{AS}\}$, Bob → AS: $\{V_{BS}\}$, (4) AS → Alice: $\{V_{SA}\}$, AS → Bob: $\{V_{SB}\}$.

6. SECURITY ANALYSIS

First, suppose that all communications among the interacting parties is under the control of an attacker, called Eve. That is, Eve can read the messages produced by the parties, provide messages of her own to them, modify messages before they reach their destination, delay messages or replay them, and make new instances of any parties. The security of our protocol is based on the difficulty of the Discrete Logarithm Problem (DLP) [16] and the Diffie-Hellman problem (DHP) [1].

We analyze the security of our protocol with regard to several attacks.

1. Assume that after capturing the transmitted messages, A, X_A, B, X_B, X_{SA}, X_{SB}, V_{AS}, V_{BS}, V_{SA}, and V_{SB}, Eve directly tries to compute the passwords or the session key of the clients from them. However, it is computationally infeasible due to the difficulty of DLP and DHP and the properties of one way hash function.
2. Assume that Eve tries to masquerade Alice or Bob. However, AS can detect this attack when verifying $V_{AS} = H(A, B, S, X_A, X_B, K_{SA}, 0)$ or $V_{BS} = H(A, B, S, X_A, X_B, K_{SB}, 0)$, because Eve cannot compute the valid K_{SA} or K_{SB} due to not knowing their correct passwords.
3. Assume that Eve tries to masquerade AS. However, Alice and Bob can respectively detect this attack when verifying $V_{SA} = H(A, B, S, X_A, X_B, K_{SA}, 1)$ and $V_{SB} = H(A, B, S, X_A, X_B, K_{SB}, 1)$, because Eve cannot compute the valid K_{AS} and K_{BS} due to not knowing their correct verifiers.
4. Password guessing attacks succeed when there are pieces of information in communications that can be used to verify the correctness of the guessed password.
   4.1. On-line password guessing attacks: On-line password guessing attacks are detectable in the proposed protocol. If Eve tries to obtain the password of Alice or Bob, she shall use the guessed password π'_A or π'_B to compute t'_A or t'_B and she need to verify her guess. As long as her guess is wrong she can be detected by AS because of $V'_{AS} \neq V_{AS}$, $V'_{BS} \neq V_{BS}$.
   4.2. Off-line password guessing attacks: Off-line password guessing attacks can be avoided in our proposed protocol because Eve can not obtain π_A or π_B from any transmitted values due to the one-way hash function, DHP and DLP, therefore she tries to guess π'_A or π'_B but because of the uncertainty of a, b and the difficulty of solving the DLP she can not verify her guess.
5. Perfect forward secrecy is provided in the situation that even though passwords are compromised, Eve cannot derive previous session keys. To consider this, suppose that Eve knows π_A or π_B. Eve tries to find previous session keys from the

information collected in past communication sessions. However, it is infeasible due to the difficulty of DLP and DHP and the properties of one-way hash function.

6. For the protocol being secure against server compromise which means that an attacker not being able to pose as a client after the server is compromised. In the proposed protocol, if AS is compromised, Eve may knows two clients' verifiers $V_A = g^{H(A, S, \pi_A)}$ and $V_B = g^{H(B, S, \pi_B)}$. However, she cannot pose as the clients because of not knowing $t_A = H(A,S, \pi_A)$ and $t_B = H(B,S, \pi_B)$ being used in step 4. Therefore, the proposed protocol is secure against server compromise.
7. The proposed protocol is secure against known-key attack since a session key is constructed by ephemeral random numbers. That is the value of a and b are randomly and independently selected in each session. Thus, the compromised session keys are not helpful for an adversary in guessing other unknown session keys.
8. For the protocol being secure against unknown key-share resilience means an attacker not being able to trick a client to share a key with him instead of the wanted one. In the proposed protocol as long as t_A, t_B, V_A and V_B are safe not compromised Eve can not trick Alice or Bob because of V_{AS},V_{BS} which are verified by AS and V_{SA},V_{SB} which are verified by Alice and Bob.
9. No key control is provided in this protocol. Bob has the possibility to choose a value b after receiving X_A from Alice. The session key is computed by the hash value of b and the received value from Alice. Finding a value of b which gives the wanted output from the hash function is considered computationally infeasible.
10. The proposed protocol preventing the man-in-the-middle attack. In this attack we assume that the adversary Eve is a legitimate user who is registered with the authentication server AS. The goal of adversary Eve is to share a session key with Alice by masquerading as Bob and to share another session key with Bob by masquerading as Alice. To achieve this goal, Eve faces the AS with her true identity, while sitting in between the clients and the server to intercept and inject messages for her own sake. This attack can be detected by AS because Eve can trick AS that both Alice and Bob want to share a session key with her, respectively, but she can not modify V_{AS} and V_{BS} which are verified by AS, because of both values are one-way hash functions and contain the identities A and B and K_{AS} /K_{BS} which are computed by t_A / t_B.
11. The proposed protocol is secure against reflection attack because $V_{SA} \neq V_{AS}$ and $V_{SB} \neq V_{BS}$ so that Eve can not intercept the messages from the sender (Alice or Bob or AS) and sends them back to the sender in the proper sequence.

7. EFFICIENCY ANALYSIS

In this section, we compare the proposed protocol with related verifier-based protocols. Table 1 shows the comparison regarding with several efficiency factors such as the number of rounds, random numbers, exponentiations, asymmetric encryption/ decryption, symmetric encryption/decryption, and hash functions.

As shown in Table 1, the proposed protocol, compared with the related other protocols, requires acceptable modular exponentiation computations and does not

require any encryption/decryption operation. Consequently, the comparison in Table 1 clearly indicates that the computation cost for both parties and the server side is comparatively light in the proposed protocol, moreover the scheme of SCH-3PEKE [12] needs to obtain and validate the server's public-key which puts a burden on the clients because the clients have to obtain, verify, and keep safely the public key of the server, while the proposed protocol requires clients only to remember their own passwords, This indicates that the proposed protocol is efficient and practically more usable.

Table 1
Comparison with the related verifier-based protocol

Protocol	Efficiency factors																	
	Round	Random no.			Modular Exp.			Asym. enc./de.			Sym. enc./de.			Hash function				
		A	B	S	A	B	S	A	B	S	A	B	S	A	B	S		
Proposed protocol	4	1	1	2	3	3	4	0	0	0	0	0	0	3	3	4		
Kulkarni *et al*. [14]	7	1	2	2	3	3	4	0	0	0	3	3	4	3	3	4		
SCH-3PEKE [12]	4	1	1	3	2	2	4	1	1	2	2	2	0	0	0	0		

On the other hand, when we come to security feature, the scheme of SCH-3PEKE [12] is suffering from man-in- the middle attack, insecure against unknown key-share resilience and not providing a mutual authentication. While the proposed protocol overcoming all of these weaknesses as the above analysis (Section 6) has shown.

8. CONCLUSION

In this paper, we have proposed a new verifier-based key agreement protocol for three-party, which does not require server's public key but requires each client only to remember a password. The proposed protocol is preventing various attacks and provides the perfect forward secrecy. Besides, compared with other related verifier-based schemes, the proposed protocol not only needs fewer rounds to perform the protocol but also has considerably lower computational cost and it is more secure. Briefly, this paper proposes a secure and efficient verifier-based key agreement protocol for three parties and practically usable.

REFERENCES

[1] W. Diffie and M. Hellman, “New directions in cryptography,” IEEE Trans. On Information Theory, vol. 22, no. 6, pp. 644-654, 1976.

[2] W. Diffie, P. C. Van Oorschot, and M. J. Wiener, " Authentication and authenticated key exchanges”, Design, Codes and Cryptography, vol. 2, pp. 107–125, 1992.

[3] V. Boyko, P.D. MacKenzie, S. Patel. Provably secure password-authenticated key exchange using Diffe-Hellman// Proceedings of the 2000 Advances in Cryptology (EUROCRYPT'2000), pp. 156-171, Springer-Verlag, 2000.

[4] E. Bresson, O. Chevassut, D. Pointcheval. New security results on encrypted key exchange// Proceedings of the 7th International Workshop on Theory and Practice in Public Key Cryptography (PKC'2004), pp. 145-158, Springer-Verlag, 2004.

[5] R. Gennaro, Y. Lindell. A framework for password-based authenticated key exchange// Proceedings of the 2003 Advances in Cryptology (EUROCRYPT'2003), pp. 524-543, Springer-Verlag, 2003.

[6] S.M. Bellovin, M. Merrit, Encrypted key exchange: password-based protocols secure against dictionary attacks, in: Proceedings of the IEEE Symposium on Research in Security and Privacy, pp. 72–84, 1992.

[7] Y. S. Kim, E. N. Huh, J. H. Hwang, and B. W. Lee, "An efficient key agreement protocol for secure authentication", in ICCAS 2004, LNCS 3043, pp. 746–754, Springer-Verlag, 2004.

[8] R. Lu and Z. Cao "Off-Line Password Guessing Attack on an Efficient Key Agreement Protocol for Secure authentication" International Journal of Network Security, Vol.3, No.1, PP.35–38, July 2006. **http://isrc.nchu.edu.tw/ijns**

[9] M. Steiner, G. Tsudik, M. Waidner, "Refinement and extension of encrypted key exchange". ACM Operating Systems Review 29 (3), pp. 22–30, 1995.

[10] Y. Ding, P. Horster, Undetectable on-line password guessing attacks, ACM Operating Systems Review 29 (4) 77–86, 1995.

[11] C.L. Lin, M. Steiner and T. Hwang, "Three-party Encrypted Key Exchange without Server Public-keys", IEEE Communications Letters, Vol. 5, No. 12, pp.497-499 2001.

[12] H.M. Sun, B.C. Chen, T. Hwang, "Secure key agreement protocols for three-party against guessing attacks", The Journal of Systems and Software 75, pp. 63–68, 2005.

[13] J. Nam, S. Kim, and D. Won,"A weakness in Sun-Chen-Hwang's three-party key agreement protocols using passwords", Cryptology ePrint Archive, Report 2004/348, 2004. **http://eprint.iacr.org/2004/348.pdf**

[14] S. Kulkarni, D. Jena, and S.K. Jena, "A Novel Secure Key Agreement Protocol using Trusted Third Party", Computer Science and Security Journals, IJCSS, volume1. issue1, pp. 11 – 18, 2007.
http://ijcss.org/Volume1/Issue1/V1_I1_011.pdf

[15] S. Bellovin and M. Merritt, "Augmented encrypted key exchange: a password based protocol secure against dictionary attacks and password-file compromise," ACM Conference on Computer and Communications Security, pp. 244-250, 1993.

[16] SearchSecurity.com-Discrete Logarithm Problem.
http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci214532,00.html

نظام المفتاح المتفق عليه المحسن لثلاثة مستخدمين الموثق بواسطه محقق معتمد على كلمة السر

معظم بروتوكولات المفتاح المتفق عليه الموثق بواسطة كلمة السر قد ركزت على مشاركة اثنين من المستخدمين على كلمة سر واحدة و من عيوب هذه البروتكولات أنه اذا أراد مستخدم الاتصال بمستخدمين أخرين لا يشاركهم نفس كلمة السر فعليه أن يحفظ العديد من كلمات السر الخاصة بهؤلاء المستخدمبن. لهذا نقدم فى هذا البحث بروتوكول فعال لثلاثة مستخدمين لنظام المفتاح المتفق عليه الموثق بواسطه محقق وهو دالة ذات اتجاه واحد معتمدة على كلمة السر. فى هذا البروتوكول كل مستخدم يتشارك بكلمة سر واحده فقط مع خادم موثوق فيه والذى يتأكد من شخصية المستخدم ويساعد اثنين من المستخدمين اللذين يتشاركان معه بكلمتين سر مختلفتين على الاتفاق على مفتاح مشترك بينهم. البروتوكول المقترح أمن ضد عدة هجمات ويوفر السرية.