A Study on Encrypted Key exchange using password

Deok-Gyu Lee, Im-Yeong Lee
Division of Information Technology Engineering
Soonchunhyang University
#646, Eupnae-ri, Shinchang-myun, Asan-si, Choongchungnam-do
KOREA

Abstract: - A traditional password protocol was based on user-chosen key. However, this method has password-guessing attack to attacker. Ways that is proposed in existing protected password solidifying protection about password. Quote user in network that is not safe from these problem and propose new method to share session key between each other. Proposed protocol designed safety Dictionary attack by active attacker, password-guessing attack, forward secrecy, server compromise, and client compromise and session key loss.

Key-Words: - Key Exchange, Password, Authentication

1 Introduction

As the practical application of the Internet expands to a wide range of areas, there has been an increasing demand for more user-friendly password-based communication, as compared to the more difficult to use key-based methods. For security reasons, password-based communication must satisfy confidentiality, integrity, and other considerations about password.

Password-based key exchange communication protocol must provide efficient and secure password management functions. To securely store password, it must provide forward and backward secrecy, which can protect the password in case of session key exposure, whenever each session key is generated. Moreover, when considering efficiency and scalability of the password, password-specific renewal of the key and efficiency-specific generation of the key must be convenient.

The classic encoding protocol basically uses a user-chosen key method, which allows users to choose the key. But such a method allows attackers to guess the password.

Two models, DH-EKE and A-EKE, initially proposed by LGSN, have been proposed. How-ever, they do not use a certificate using EKE (Encrypted Key Exchange). On the other hand, GLNS is a protocol that evolved from LGSN and many modified and improved protocols have been proposed based on LGSN.

In this article, we propose authentication and key exchange protocol. This method uses pass-word and as a user and server each have different values, it also hides user password informa-tion. Likewise, we present three methods (one-flow model includes two proposed methods), and the asymmetric trust model, which maintains security for users even if secret user informa-tion is compromised in each methods. We have designed our protocol in such a way it includes certain characteristics, which will be mentioned in the later chapters.

- Password-based authentication: It must authenticate users through password, and un-dergo the stage in which a new session key is established through the password.
- Diffie-Hellman-based key approval: In establishing a session key, the key must be approved based on D-H, and, this generated session key offers a password security, as well as forward and backward secrecy.
- Easy generation of session key: Since a session key is generated through the password, its generation does not need any other session elements. Users establish a session key only as an element provided for the key agreement based on D-H.

This article is composed of six chapters. Chapter 2 summarizes the requirements for password-based key distribution method. Chapter 3 examines the three methods using existing methods. Chapter 4 gives a detailed description for each method. Chapter 5 analyzes the security and efficiency of the proposed methods and compares them with the existing methods. Finally, chapter 6 concludes this article.

2 Requirements of Password-based Key Distribution Method

The password-based key distribution method proposed in this article has the following requirements. It should be designed to provide security and efficiency.

- It must to be an asymmetric trust model. In other words, the server must not have a client password should have a verifier that must not have a client password, and verifies the client password only.
- Confidentiality of the password is the most fundamental attribute. It must be mathematically impossible for any attackers to guess the password.
- Dictionary attack poses a threat from an impersonating user continuously guessing the password on-line. It is important to block any attempt to illegally access the password.
- Attacker must not be able to acquire a session key in any way other than by calculating it by participating in the protocol interactively.
- It must provide safeguards against active attackers manipulating multiple sessions.
- Knowing the previously distributed session key must not assist any attack at all.
- It must not use previously distributed session key in the protocol.
- There must be no way of knowing previously distributed session key even if client PB is exposed.
- It must be impossible to know current session key by simply bugging communication information between the server and client even in case of a client PB exposure.
- Even if an attacker acquires the password authentication information (L, Z), he/she must launch a dictionary attack on PB to be able to pose as a legitimate client.
- The protocol must be simple and made from a well-known encoding primitive.
- Fundamentally, the protocol must allow the server to authenticate the client, and the client to selectively authenticate the server.

These attributes must be considered in relation to each other. The proposed method must pro-vide forward secrecy for the session key through password, backward secrecy to protect pass-word in case of

3 Analysis of Existing Methods

This chapter in an introduction to the password-based key distribution method of AMP[9], A-EKE[4], and AuthA[12]. Some existing methods are designed to allow authentication. There are entity and user authentication. Entity authentication is required for validating the identity of the communicating entity when initializing communication, and provided with key generation systems such as key agreement or key transfer among entities. User authentication is based on known elements. Characteristics of the elements are simplicity, convenience, adaptability, mobility, low hardware specification, and requiring user knowledge only.

The following is the list of coefficients used in the protocol.

$Alice$: Client, Bob : Server

G : basic group, q : Size of group

H, H' : Mask-generation functions

$\varepsilon^1, \varepsilon^2$:encoding functions used in A and B

π : Password, τ : Salt

k : Security parameter,

$\stackrel{?}{=}$: Comparison of the two values

$pwa = H'(A \| B \| Password)$: element of Group

$pwb = g^{pwa}$

3.1 AMP

In general, it authenticates passwords based on expanded password. This method can also make key agreement in the authentication process. Expanded password proof can attain zero-knowledge proof.[9]

Diagram 1 summarizes the entire AMP protocol. This method includes a total of five processes, two of them designed for password security against server compromise and dictionary attack. Aside from the two protocols, the rest of them have several weaknesses: they cannot prevent server dictionary attack, server impersonation, and client impersonation.

The following is the analysis of each protocol.

In case of dictionary attack, the password is located in the client and the attacker supplies π to the server thereby client impersonation attack is enabled. Finally, server impersonation using random π is also enabled. Password guessing, however, may be prevented to a certain degree by securely storing g^{π}.

[Protocol]

1. Alice and Bob are sharing g, p, q; id represents the user's name; Alice and Bob represent the address.

2. Alice selects $\pi \in_R \{0,1\}^{\omega(k)}$, and announces it to Bob in an authentic and confidential manner.

3. Bob selects $\tau \in_R \{0,1\}^k$ and stores $(id, \tau, \upsilon = g^{(\xi+\tau)^{-1}\upsilon})$ (only when $\upsilon = h(id, \pi)$).

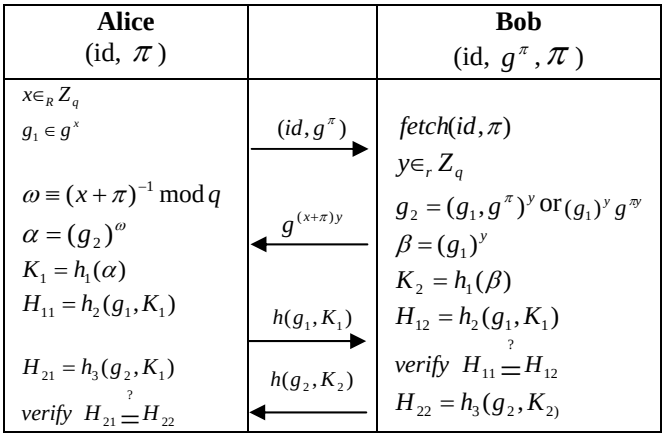

Fig 1. AMP Protocol

3.2 The AuthA protocol for Password-Based Authenticated Key Exchange

The characteristics of AuthA method are as follows: it is designed based on the asymmetric trust model, and offers security against active attackers who can control various sessions. Attackers may not be able to acquire any information about the distributed session key in any way other than testing random password bidirectionally.

The distributed session key cannot be used within the protocol. In spite of these advantages, this method has several vulnerabilities.

If the server is damaged, it can be vulnerable to dictionary attacks, as well as server threats.

[Protocol]

1. Client A sends AuthA to server B. Server B receives AuthA.

⇒ *SessionKeyB: B accepts* ⇔ *AuthA = AuthAcheck*

2. Server B sends AuthB to A. Client A receives AuthB.

⇒ *SessionKeyA : A accepts* ⇔ *AuthB = AuthBcheck*

3. Server B does not send AuthB. In this protocol, A is accepted when session key A is calculated.

4 Proposed method

Our plan is to design a method that satisfies new requirements, as well as the existing considerations about password.

- Password-based authentication: It must authenticate users through password, and undergo the stage in which a new session key is established through the pass word.

- Diffie-Hellman-based key approval: In establishing a session key, the key must be approved based on D-H, and, this generated session key offers a password security, as well as forward and backward secrecy.

- Easy generation of session key: Since a session key is generated through the password, its generation does not need any other session elements. Users establish a session key only as an element provided for the key agreement based on D-H.

- Password file protection: Only the user knows the password, and sending a modified value of the password prevents password guessing. At the same time, with the exception of password exposure by user mistakes, security is guaranteed since there is no password exposure in the transaction.

- Password secrecy: As the most basic attribute, it must be mathematically impossible for any malicious attacker to guess the password,

- Dictionary attack: This attack poses as a threat from an impersonating user who continuously guesses passwords on-line. It is important to block any attempt to illegally access a password.

- Forward Secrecy: It must be impossible for malicious attackers to gain access to data even if they know the previous session key, by making calculation of next session key impossible.

- Backward Secrecy: Malicious attackers must not gain access to data. They must not be able to calculate previous session key using later acquired session key information.

Based on these eight considerations, we now propose this proposed method.

Moreover, we propose this method, as a solution to the problems of existing password-based AMP and A-EKE. We assume that the proposed method is an asymmetric trust model; Client A has password pw, and server B has pb, special value of pw. The two elements use the Diffie-Hellman key exchange.

4.1 System Parameters

The following is the list of system parameters used in this proposed method.

ID_*: ID of *, pw: Password

$seed$: Value shared by A and B initially

q: Big decimal defining the finite body GF(q)

r: Decimal r|q-1, g: Primitive element on GF(p)

m: Public information

δ: Value generated by user A to authenticate server B

β : Hash value transferred to server B for authentication

pb : Generated by user A as $pb \equiv g^{pw} \bmod q$

ρ_A , δ : Value provided to server B by user A

θ, θ'' : Value for key generation

ρ_B : Value generated by server B and provided to user A

K' : Generated session key authentication value

Ξ : Value provided to user A by server B

E^1, E^2 : encoding functions used in A and B

$E_\omega^{-1}, E_\omega^{-2}$: decoding functions used in A and B

G_A, G_{A+1}, G_B : Random value generated by user A and B

ω : Initially used encoding values

Z : Value provided to server B by user A

Z' : Value decoded by server B

L : Value decoded and provided to server B by user A

L' : Value decoded by server B

γ : Value generated from A' by server B

K : Generated session Key

α : Hash value transferred to authenticate user A

P : Value provided to user A by server B

α' : Value validating α acquired from server B

$\stackrel{?}{=}$: Comparison of the two values

4.2 Proposed Method - 1

The following is the detailed description of each stage.

4.2.1 Preliminary stage

This is the preliminary stage between the user and the server; the user shares password with the server. At this stage, the user shares pw, and the server shares pb with each other. They also share $seed$ value to increase password security.

4.2.2 Key exchange and authentication stage

In this key exchange stage, ω, which encodes session key between the user and the server, is generated and the encoded session key is transferred.

step 1. The user generates expression (1-1), (1-2), and (1-3), encodes using expression (1-1), (1-2), and (1-3), generates expression (1-4), and, transfers expression (1-5).

$$G_A = g^x \quad (1\text{-}1)$$

$$\omega = H(pb \| seed) \quad (1\text{-}2)$$

$$Z = G_A \cdot H(g^{pw})^{-1} \quad (1\text{-}3)$$

$$L = E_\omega(Z) \quad (1\text{-}4)$$

$$(ID_B, L) \quad (1\text{-}5)$$

step 2. The server acquires Z by receiving expression (1-5) and encoding it. With acquired Z and pb it already has, it generates session key (expression (1-9)) as server-generated expression (1-7) and (1-8) by using expression (1-8). It then hashes generated session key K, γ, Z, and sends them to the user (expression (1-12)).

$$L' = E_\omega^{-1}(E_\omega^1(Z)) = Z' \quad (1\text{-}6)$$

$$G_B = g^y \quad (1\text{-}7)$$

$$\gamma = pb \cdot Z = g^{pw} \cdot H(g^{pw}) \cdot g^x \cdot H(g^{pw})^{-1} = g^{pw+x} \quad (1\text{-}8)$$

$$K = (\gamma)^y \quad (1\text{-}9)$$

$$\alpha = H(Z, \gamma, K) \quad (1\text{-}10)$$

$$P = E^2(G_B) \quad (1\text{-}11)$$

$$(ID_A, P, \alpha) \quad (1\text{-}12)$$

step 3. The user calculates γ' in advance (expression (1-13)), generates session key (expression (1-15)) by calculating G_B from P acquired from the server (expression (1-13)), and validates α (expression (1-16), (1-17)). The user generates expression (1-18), and hashes and sends expression (1-19) to the server.

$$\gamma' = g^{pw} \cdot g^x = g^{pw+x} \quad (1\text{-}13)$$

$$P' = E_\omega^{-2}(E_\omega^2(G_B)) = G_B \quad (1\text{-}14)$$

$$K' = (G_B)^{pw+x} \quad (1\text{-}15)$$

$$\alpha' = H(Z, \gamma, K) \quad (1\text{-}16)$$

$$verify\ \alpha \stackrel{?}{=} \alpha' \quad (1\text{-}17)$$

$$\delta = (G_B)^\omega \quad (1\text{-}18)$$

$$\beta = H(G_B, \delta, K) \quad (1\text{-}19)$$

step 4. The server generates value δ in advance (expression (1-20)), and validates β value acquired from the user.

$$\delta' = (G_B)^\omega \quad (1\text{-}20)$$

$$\beta' = H(G_B, \delta, K) \quad (1\text{-}21)$$

$$verify\ \beta \stackrel{?}{=} \beta' \quad (1\text{-}22)$$

4.3 Proposed Method - 2

We now propose this method as a solution to the problems of the existing password-based AMP and A-EKE. We assume that the proposed method is an asymmetric trust model. Client A has password pw, and server B has pb, special value of pw. The two elements of the protocol use the Diffie-Hellman key exchange.

4.3.1 Preliminary stage

This is the preliminary stage between the user and the server; the user shares password with the server. At this stage, the user shares pw, and the server also shares pb with each other. They also share the public m value to increase password security.

4.3.2 Key exchange and authentication stage

In this key exchange stage, ω, which encodes session key between the user and the server, is generated and the encoded session key is transferred.

step 1. The user generates expression (2-1), (2-2), (2-3), and (2-4), generates expression (2-6) by encoding based on expression (2-6), and sends expression (2-7).

$$G_A \equiv g^x \bmod q \quad (2\text{-}1)$$

$$\omega \equiv H(g^{pw+m} \bmod q) \quad (2\text{-}2)$$

$$\rho_A \equiv g^{x-pw} \bmod q \quad (2\text{-}3)$$

$$\theta \equiv (\rho_A)^m \bmod q \quad (2\text{-}4)$$

$$\delta \equiv H(\theta \| \rho_A) \quad (2\text{-}5)$$

$$L = E_\omega^1(\rho_A \| \delta) \quad (2\text{-}6)$$

$$(ID_B, L) \quad (2\text{-}7)$$

step 2. The server acquires $\rho_A' \| \delta'$ by receiving and encoding expression (2-7), validates expression (2-11) and (2-13) by using acquired $\rho_A' \| \delta'$ and pb it already has, and generates session key (expression (2-16)) as the server generated expression (2-15) using the expression (2-11). The server then hashes and encodes the generated session key K, ρ_B, and transfers them to the user.

$$G_B \equiv g^y \bmod q \quad (2\text{-}8)$$

$$\omega \equiv H((pb)^m \bmod q \equiv H(g^{pw+m} \bmod q) \quad (2\text{-}9)$$

$$L' = E_\omega^{-1}(E_\omega^1(\rho_A \| \delta)) = \rho_A' \| \delta' \quad (2\text{-}10)$$

$$\theta' \equiv (\rho_A)^m \bmod q \quad (2\text{-}11)$$

$$\delta' \equiv H(\theta' \| \rho_A') \quad (2\text{-}12)$$

$$verify \ \delta \stackrel{?}{=} \delta' \quad (2\text{-}13),$$

$$\rho_B \equiv pb \cdot G_B \bmod q \equiv g^{pw+y} \bmod q \quad (2\text{-}14)$$

$$K \equiv \theta' \cdot \rho_B \bmod q \equiv g^{x-pw} \cdot g^{pw+y} \bmod q \equiv g^{(x+y)} \bmod q \quad (2\text{-}15)$$

$$\tau \equiv H(K \| \rho_B) \quad (2\text{-}16)$$

$$\Xi = E_\omega^2(\rho_B \| \tau) \quad (2\text{-}17)$$

$$(ID_A, \Xi) \quad (2\text{-}18)$$

step 3. The user generates session key (expression (2-21)) by calculating K using ρ_B from Ξ acquired from the server, and validates τ (expression (2-22)).

$$\Xi' = E_\omega^{-2}(E_\omega^2(\rho_B \| H(K \| \rho_B)) = \rho_B \| \tau \quad (2\text{-}19)$$

$$K' \equiv \rho_B \cdot \rho_A \bmod q \equiv g^{x-pw} \cdot g^{pw+y} \bmod q \equiv g^{x+y} \bmod q \quad (2\text{-}20)$$

$$verify \ \tau \stackrel{?}{=} \tau' \quad (2\text{-}21)$$

4.4 Proposed Method - 3

This proposed method is the process in which the server receives the key acceptance process by generating information and providing it to the server at the time of connection.

4.4.1 Preliminary stage

In this preliminary stage between the user and the server, the user shares the password with the server: the user shares pw and the server shares pb with each other.

4.4.2 Key exchange and authentication stage

In this key exchange stage, ω that encodes the session key between the user and the server is generated and the encoded session key is transferred.

step 1. The server generates the expression (4-1), (4-2), and (4-4), generates the expression (4-6) by encoding based on the expression (4-3), and transfers the expression (4-7).

$$G_A = pb \equiv g^{pw} \bmod q \quad (4\text{-}1)$$

$$G_{A+1} \equiv g^x \bmod q \quad (4\text{-}2)$$

$$\omega \equiv H((pb)^m \bmod q \equiv H(g^{pw+m} \bmod q) \quad (4\text{-}3)$$

$$K \equiv G_A \cdot G_{A+1} \bmod q \equiv g^{pw+x} \bmod q \quad (4\text{-}4)$$

$$L \equiv H(\omega \| K) \quad (4\text{-}5)$$

$$Z = E_{\omega}^{1}(G_{A+1} \| L) \quad (4\text{-}6)$$

$$(ID_B, m, Z) \quad (4\text{-}7)$$

step 2. The server acquires $G_{A+1}{}' \| L'$ by receiving and decoding the expression (4-7). Using the acquired $G_{A+1}{}' \| L'$ and pb it already has, it generates the expression (4-10), and (4-11), and, validates the expression (4-12).

$$\omega' \equiv H((pb)^m \bmod q \equiv H(g^{pw+m} \bmod q) \quad (4\text{-}8)$$

$$Z' = E_{\omega}^{-1}(E_{\omega}^{1}(G_{A+1} \| L)) = G_{A+1}{}' \| L' \quad (4\text{-}9)$$

$$K' \equiv G_A \cdot G_{A+1}{}' \bmod q \equiv g^{pw+x} \bmod q \quad (4\text{-}10)$$

$$L' \equiv H(\omega \| K') \quad (4\text{-}11)$$

$$verify \ L \stackrel{?}{=} L' \quad (4\text{-}12)$$

5 Characteristics and Analysis of the Proposed Methods

This article has proposed a total of three methods in 3, 2, 1-flow format. Each method has different characteristics. We now look into the characteristics and analysis of the proposed methods in general.

5.1 Characteristics

Briefly viewing, the characteristics are as follows. First, the three-flow method has L and P as the biggest messages. The rest send the value through Hash function, solving the problem of authentication through small messages. Furthermore, encoding between the user and the server is processed in two steps, and in case of exponential operation, each encoding is processed in three operations. If g^x and g^{pw} are calculated in advance, the number of operation is reduced to two for the user, and three for the server. Next, the biggest messages in 2-flow format proposed method are L and Ξ. The rest send the value through the Hash function, solving the problem of authentication through small messages. Likewise, encoding between the user and the server is processed in two steps, and, in case of exponential operation, each encoding is processed in five operations. If g^x and g^{pw} are calculated in advance, the number of operation is reduced to four for the user, and four for the server.

Primarily, the general characteristic of the proposed method is the provision of key agreement as this allows key generation between the server and the client by exchanging D-H key. In this proposed method, we have proposed the protocol based on one-pass, and the server and the client generate G_A, G_{A+1}, G_B by exchanging their key information.

5.2 Characteristics

We now examine the requirements for each type of attacks, similar to how we have examined the requirements of password-based key distribution in chapter 2.

5.2.1 Server impersonation

Since the client and the server share $pb \equiv g^{pw} \bmod q$ securely between them, the server is protected from server impersonation when the client requests and the server sends, and when the user sends a message to the server using $pb \equiv g^{pw} \bmod q$ because the attacker doesn't know $pb \equiv g^{pw} \bmod q$.

5.2.2 Client Impersonation

The attacker does not know anything about $pb \equiv g^{pw} \bmod q$. Thus, the proposed method is safe from client impersonation attacks. Even if the attacker generates and sends a random pb, the client may cancel it since the pb is not the pb to the original pw.

5.2.3 Dictionary Attack

A dictionary attack on pw can compromise integrity to a certain degree, if the client has generated pw by dictionary combination, and not by random combination. But this proposed method does not use pw, but generates pb based on pw, and transfers using pb. Thus, we can assume it is secure. The evidence for our assumption is based on a hard-problem.

5.2.4 Guessing Attack

A) On-line Guessing Attack

An attack on pw and session key K are the two broad categories for on-line guessing attack. On-line guessing attack on pw is a guessing attack on pb at the time of initialization of communication. Since pb is based on a hard-problem, it is safe from on-line attacks. We will examine the guessing attack on

session key K in PFS and backward secrecy, which will be mentioned later.

B) Off-line Guessing Attack

Similarly, off-line guessing attack can be categorized into two categories, ^{pb} and K. Attacks on pb are impossible, for it is impossible to disassemble pb. For K, there is no way of knowing anything about later key generation, even if the existing session key is exposed.

5.2.5 Replay Attack

When the attacker transfers information in exactly the same pattern using the previous information (ID_A, m, Z), the information generated from the client and the server is changed every time. Thus, an attack on the information is impossible.

5.2.6 Loss of Session Key

Even if session key is lost, it is impossible to acquire new information using the previous information. The server and the user generate new G_A, G_{A+1}, G_B in every session and exchange them. It is difficult to guess pw and pb from the lost session key. Moreover, the attacker cannot generate $Z = (E_\omega^1(G_{A+1} \| L)$ based on collected session keys. Thus, generating a new one can solve loss of session key.

5.2.7 Loss/damage of pb

If securely shared pb is lost, future key exchanges cannot be done securely, for all security is based on pw and pb.

pb can be used as the initializing point of a dictionary attack on password. It is closely related to PFS. It may be used as a meaningful data threatening other elements. When Bob loses pb, attackers will calculate a new Z'. When $Z' = Z = E_\omega^{-1}(E_\omega^1(G_{A+1} \| L)$, attackers will be able to know that pb_i is pb. But even if they know pw, pb is still secure.

5.2.8 Backward Secrecy

There is no way of knowing information about previous session key even if the attacker knows the current session key. Even if the attacker collects and acquires the information on current session key or future session key, there is no way of knowing the previous message from the collected session key information. It is impossible to acquire pw, pb information from the collected session keys.

5.2.9 Partition Attack

A partition attack may be a partition attack on password itself, as well as on pb. Security can be guaranteed if the password-generating group is not a small group. Alice and Bob use DH Exponential Key Exchange on Z_P^*. In this case, p is a big decimal, and $p-1$ divides q. Basically, g needs to block partition attack. A third person can attempt to encode $Z' = Z = E_\omega^{-1}(E_\omega^1(G_{A+1} \| L)$ using the dictionary of pw. If g is not a primitive, incorrect guess pb_i will not be validated.

Table 1. Analysis of the Conventional Schemes vs. Proposed Schemes

	Dictionary attack	authentication	Forward secrecy	Backward secrecy	Server compromise	pb loss
AMP	×	○	○	○	×	△
A-EKE	○	○	○	○	×	×
AuthA	○	○	○	○	×	×
Proposed – 1	○	○	○	○	△	△
Proposed – 2	○	○	○	○	○	△
Proposed – 3	○	△	○	○	○	△

5 Conclusion

With current researches on password-based key distribution methods that are safe from on/offline dictionary attacks, password has been applied in many practical areas. As the application of password expands, the importance of user password has been increasing gradually.

As we have seen in this article, there are many difficulties in securely re-configuring password between the server and the user. To solve these problems, instead of using the existing method, we have sent modified password value, not sending the value as it is, and proposed a password-based protocol designed for secure authentication and key exchange. We have attempted to design a protocol that can deal with dictionary attack, password guessing, etc. efficiently.

Our proposed protocol offers security for forward secrecy, backward secrecy, replay attack, guessing attack, etc.

The key exchange protocol is currently used in many fields including authentication and key exchange, and

will be used in many fields including key roaming and key recovery. Currently, there are many plans for key-exchange protocol-related research.

References:

[1] M. Ballare, D. Pointcheval, and P. Rogaway. "Authenticated key exchange secure against dictionary attack," *In EUROCRYPT 2000*

[2] S. Bellovin and M. Merritt. "Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks." *Proc. of Symposium on Security and privacy*. page 72-84. IEEE, 1992

[3] S. Bellovin and M. Merritt. "Augmented Encrypted Key Exchange : A Password-Based Protocol Secure against dictionary Attacks and Password File Compromise." *Proceeding of the 1st Annual Conference on Computer and Communications Security*, ACM, 1993

[4] V. Boyko, P. Mackenzie and S. Patel. "Provably Secure Password Authenticated Key Exchange Using Diffie-Helllman." *in Eurocypt 2000*

[5] D. Jablon. "Strong Password-Only Authenticated Key Exchange." *ACM Computer Communications Review*, October 1996

[6] P. Mackenzie and R. Swaminathan, "Secure network authentication with Password identification," *Presented to IEEE p1363a*, August 1999

[7] S. Lucks, "Open Key Exchange: How to defeat Dictionary Attacks without encrypting public-keys," *The Security Protocol Workshop `97, April 7-9*, 1997

[8] M. Lomas, L. Gong, J. Saltzer, and R, Needham, "Reducing risks from poorly chosen keys," *Proceedings of the 12th ACM Symposium on Operating System principles, ACM Operating Systems Review*, 1989, pp. 14-18

[9] L. Gong, M. Lomas, R. Needham, and J. Saltzer, "Protecting poorly chosen secrets from guessing attacks," *IEEE journal on SAC.*, vol. 11, no.5, pp.648-656, June 1993

[10] T. kwon, "Ultimate solution to authentication via memorable password"

[11] M. Bellare, and P. Rogaway, "The AuthA Protocol for Password-Based Authenticated Key Exchange", *contribution to the IEEE P1363 study group for Future PKC Standards*