T.C.
B A H Ç E Ş E H İ R Ü N İ V E R S İ T E S İ

A PASSWORD BASED KEY ESTABLISHMENT PROTOCOL WITH SYMMETRIC KEY CRYPTOGRAPHY

Master Thesis

RECEP GÖKÇELİ

İSTANBUL, 2008

T.C.
B A H Ç E Ş E H İ R   Ü N İ V E R S İ T E S İ

THE INSTITUTE OF SCIENCE
COMPUTER ENGINEERING

A PASSWORD BASED KEY ESTABLISHMENT PROTOCOL WITH SYMMETRIC KEY CRYPTOGRAPHY

Master Thesis

RECEP GÖKÇELİ

Supervisor: PROF. DR. EMİN ANARIM

İSTANBUL, 2008

T.C.
B A H Ç E Ş E H İ R Ü N İ V E R S İ T E S İ

INSTITUTE OF SCIENCE
COMPUTER ENGINEERING

Name of the thesis: Password Based Key Establishment Protocol With Symmetric Key Cryptography
Name/Last Name of the Student: Recep GÖKÇELİ
Date of Thesis Defense: 01.08.2008

The thesis has been approved by the Institute of Science.

Prof. Dr. Erol SEZER
Director

____

I certify that this thesis meets all the requirements as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Adem KARAHOCA
Program Coordinator

____

This is to certify that we have read this thesis and that we find it fully adequate in scope, quality and content, as a thesis for the degree of Master of Science.

Examining Committee Members	Signature
Prof. Dr. Emin ANARIM	___________________
Assoc. Prof. Dr. Adem KARAHOCA	___________________
Asst. Prof. Dr. Olcay KURŞUN	___________________

ACKNOWLEDGEMENTS

Firstly, I would like to thank **my family** for their helps on various topics throughout my life.

I would like to express my gratitude to my supervisor **Prof. Dr. Emin ANARIM** for encouraging and challenging me throughout my thesis studies.

I also thank **İmran ERGÜLER** and **Ş. Çağlar TOKLU** for their helps throughout my thesis studies.

ABSTRACT

PASSWORD BASED KEY ESTABLISHMENT PROTOCOL WITH SYMMETRIC KEY CRYPTOGRAPHY

Gökçeli, Recep

The Institute of Sciences, Computer Engineering

Supervisor: Prof. Dr. Emin Anarım

August 2008, 40 pages

In 2005, Laih, Ding and Huang proposed a password-based key establishment protocol such that a user and a server can authenticate each other and generate a strong session key by their shared weak password within a symmetric cipher in an insecure channel. In this protocol, a special function, which is a combination of a picture function and a distortion function, is combined to authenticate the user and protect the password from the dictionary attacks that are major threats for most of the weak password-based protocols. They claim that the proposed protocol is secure against some well known attacks. However, Tang and Mitchell show that the protocol suffers from an offline dictionary attack requiring a machine based search of size 2^{23} which takes only about 2.3 hours. So designing such a protocol with providing practical security against offline attack is still an open problem. In this study, a password-based authenticated key establishment protocol is proposed that provides practical security against offline dictionary attacks by only using symmetric cryptography.

Key Words: Key establishment protocol, protocol, cryptography, authentication

ÖZET

SİMETRİK ŞİFRELEME İLE PAROLA TABANLI ANAHTAR OLUŞTURMA PROTOKOLÜ

Gökçeli, Recep

Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği

Tez Danışmanı: Prof. Dr. Emin Anarım

Ağustos 2008, 40 sayfa

2005 yılında Laih, Ding ve Huang güvenli olmayan bir kanalda kullanıcı ve sunucunun birbirlerinin kimliklerini doğrulayabileceği ve zayıf bir paroladan güçlü bir oturum anahtar üretimini sağlayan bir protokolü tanıttılar. Bu protokolde, kullanıcıların kimliğini doğrulamak ve şifreyi zayıf birçok şifre tabanlı protokolü tehdit eden çevrimdışı sözlük saldırısına karşı korumak için bir imge fonksiyonuyla bir biçim bozma fonksiyonun birleşimi olan üzel bir fonksiyon kullanılmıştır. Her ne kadar protokolün çevrimdışı sözlük saldırılarına karşı dahi güvenli olduğunu iddia etseler de, Tang ve Mitchell 2^{23}'lük bir makine işlemiyle yaklaşık olarak 2.3 saat içerisinde sistemin kırılabileceğini göstermişlerdir. Dolayısıyla çevrimdışı sözlük saldırılarına karşı güvenli bu tür bir protokol tasarımı hala açık problem olarak durmaktadır. Bu çalışmada bu problemi yalnızca simetrik kripto kullanarak çözecek bir model önerilmektedir.

Anahtar Kelimeler: Anahtar oluşturma protokolü, protocol, şifreleme, kimlik doğrulama

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

In 1976 Diffie and Helman (Diffie and Hellman 1976) introduces a key agreement protocol in which two parties can establish a secret session key over insecure channel. It makes use of the difficulty of computing discrete logarithms over a finite field. Diffie-Hellman key exchange does not authenticate the participants. Several methods of integrating authentication into the scheme have been proposed.

One method involves incorporating certificates (e.g. digital signatures) into the key agreement protocol, thus providing authentication of the session key. A certificate from a trusted authority is presented to the user along with the public key to certify ownership of the keys. Now an attacker cannot impersonate both Alice and Bob (the participants) and cannot substitute the original public keys with her own because they are signed. A public key system such as RSA can be used for this purpose. One example of this scheme is the authenticated Diffie-Hellman key agreement protocol, or station-to-station (STS) protocol, which was developed by Diffie et al. (Diffie et al. 1992).

As key exchange schemes with certificates require some trusted authority to verify the integrity of the received messages, the extension to a larger system may be difficult. They need a large storage for certificates and more bandwidth for the verification of the signature as the number of user increases. Furthermore, if the authority is compromised then the total system would be in danger.

Another method for achieving an authenticated key agreement protocol which does not require a trusted authority, involves two users (Alice and Bob) who pre-share a secret password. In encrypted key exchange (EKE) (Bellovin and Merritt 1992) a shared password P is used as a key to encrypt a randomly generated number. This scheme defeats man-in-the-middle attacks, as attacker has no method to disguise herself as Alice and Bob without knowing the password P. But this algorithm is complicated and is also patented, obstructing wide usage. Another example of this type of scheme is fortified key negotiation (Anderson and Lomas. 1994). In 1999, Seo and Sweeney (Seo and Sweeney 1999) proposed the password-based authenticated key agreement scheme, which is a slight modification of the Diffie–Hellman scheme, and based on a pre-shared password method for user authentication. After the scheme of (Seo and Sweeney 1999), there have been a sequence of works to improve the scheme. Tseng (Tseng 2005) pointed out that Seo and Sweeney's scheme is not secure against the replay attack, in which an adversary can successfully make a honest party compute a wrong session key. Tseng (Tseng 2005) also proposed an improved scheme to remedy this vulnerability. Later, Ku and Wang (Ku and Wang 2000) showed that Tseng's scheme is weak to two attacks, called the backward replay attack and the modification attack, and proposed a new enhancement to eliminate these weak points. However, Hsu et al. (Hsu et al. 2003) showed that Ku and Wang's scheme is weak to the modification attack, in which an adversary fools two communicating parties into sharing a wrong session key, and proposed an improvement to solve this weakness. Then, Lee and Lee (Lee and Lee 2004) found that Hsu et al.'s scheme has a weakness against the modification attack of (Hsu et al. 2003) and proposed an improved scheme to repair this security flaw. Recently, Lee et al. (Lee et al. 2005) argued that Lee and Lee's scheme is also vulnerable to a password guessing attack and proposed an improved scheme. Very recently, Kwon, Hwang, Kim, Lee (Kwon et al. 2005) show that Lee et al.'s scheme (denoted by LKY) is still vulnerable to a password guessing attack.

The remaining of this thesis is organized as follows: In section 2, the related works which are Seo and Sweeney's simple authenticated key agreement protocol, Tseng's modified key agreement protocol, Ku-Wang key agreement protocol, Hsu et al. key agreement protocol, Lee and Lee key agreement protocol, Lee-Kim-Yoo password based key agreement protocol, Laih-Ding-Huang password based key agreement protocol, Tang-Mitchell key agreement protocol and their cryptanalysis are reviewed. In section 3, the proposed protocol and its software simulation is given and section 4 is conclusion of the thesis.

2. RELATED WORKS FOR PASSWORD BASES KEY AGREEMENT PROTOCOL

2.1. SIMPLE AUTHENTICATED KEY AGREEMENT ALGORITHM

Seo and Sweeney proposed a simple authenticated key agreement protocol that Alice and Bob (two users) share a common password P before the protocol begins and uses the same public values of g and n as the original Diffie-Hellman. In the Diffie-Hellman key agreement protocol, the system uses public values n and g where n is a large prime number and g is a generator with order $n-1$ in $GF(n)$.

1. Alice and Bob each compute two integers Q and $Q^{-1} \bmod (n-1)$ from the password P. Q could be computed in any predetermined way from P, provided it yields a unique value, relatively prime with $(n-1)$, and with low probability that two different passwords will give the same value of Q. For example, Q could be the smallest such integer that is greater than a numeric representation of the password P.
2. Alice chooses a random large integer a and sends Bob

$$X_1 = g^{aQ} \bmod n$$

3. Bob chooses a random large integer b and sends Alice

$$Y_1 = g^{bQ} \bmod n$$

4. Alice computes

$$Y = (Y_1)^{Q^{-1}} \bmod n$$

$$Key_1 = Y^{a} \bmod n = g^{ab} \bmod n$$

5. Bob computes

$$X = (X_1)^{Q^{-1}} \bmod n$$

$$Key_2 = X^{a} \bmod n = g^{ab} \bmod n$$

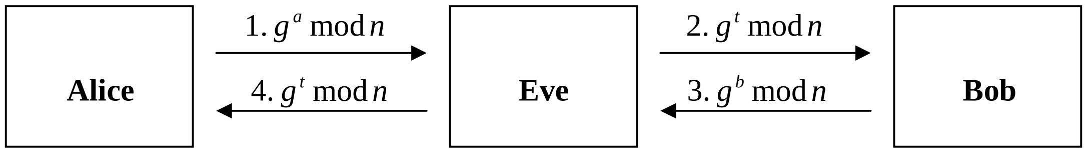
It is clear that $Key_1 = g^{ab} \bmod n = Key_2$. A common session key is thus established.
To check the validity of the session key, Alice and Bob may perform the following steps:

1. Alice computes $(Key_1)^Q \bmod n$ and sends it to Bob
2. Bob computes $(Key_2)^Q \bmod n$ and sends it to Alice
3. Alice and Bob each compute the other's key by applying Q^{-1} and compare it with his/her own session key.

2.1.1. Cryptanalysis Of The Simple Authenticated Key Agreement Algorithm

2.1.1.1. Man-In-The Middle Attack

With the original Diffie-Hellman, Eve (attacker) can alter the public values such as $g^a \bmod n$ and $g^b \bmod n$ from Alice and Bob with her own values. Then Eve and Alice share one key, and Eve and Bob share another key without notice.



$Key_1 = (g^t)^a \bmod n$ $\quad$ $Key_1 = (g^a)^t \bmod n$ $\quad$ $Key_2 = (g^t)^b \bmod n$

$Key_2 = (g^b)^t \bmod n$

Figure 1.1 Man-In-The Middle Attack To The Diffie-Hellman Key Agreement Protocol

But, with simple authenticated key agreement algorithm, when Eve receives (X_1 = g^{aQ} mod n) in step (2), she cannot guess ‘g^a mod n’ and Q, since the problem is combined with the discrete logarithm and a secret password. If she still tries to eavesdrop, she has to make $(g^{a^* Q^*} \bmod n)$ and send it to Bob. If Bob tries to solve $((g^{a^* Q^*} \bmod n)^{Q^{-1}} \bmod n)$, he will obtain a wrong value, which it is impossible for Eve to know.

Eve does not know Q or Q^{-1} and therefore cannot send values that will result in Alice and Bob re-computing the same key values as before.

However, Tseng (Tseng 2005) showed that the simple authenticated key agreement protocol of Seo and Sweeney is vulnerable to the replay attack.

2.1.1.2. Replay Attack

Seo and Sweeney (Seo and Sweeny 1999) proposed a simple authenticated key agreement protocol that is based on a pre-shared password method and modifies the Diffie-Hellman scheme to provide user authentication. They claimed that established session key between two users is also verified. However, Tseng (Tseng 2005) pointed out that verification of the session key cannot be achieved in their protocol. If an opponent replies to the received message after receiving the honest user's message, the honest user cannot determine the invalidity of the session key. That is, verification of the session key cannot be achieved in the Seo-Seweeney protocol (Seo and Sweeny 1999).

In the Seo-Seweeney protocol (Seo and Sweeny 1999), although an attacker (Eve) cannot impersonate Bob to compute a common session key shared with the Alice, according to Tseng (Tseng 2005), the verifying process of the session key in their protocol has the following weakness: Eve may re-send it to Alice after receiving the message Key_1^Q mod n sent by Alice. Alice then computes the key $(Key_1^Q)^{Q^{-1}} \bmod n$ by applying Q^{-1} and it must be equal to Key_1 because $Q.Q^{-1} = 1 \bmod (n-1)$. Therefore, although Alice obtains a wrong session key and Eve cannot compute the same wrong session key, Alice will still believe it. That is, verification of the session key cannot be achieved using this protocol.

2.2. TSENG'S MODIFIED KEY AGREEMENT PROTOCOL

By using a pre-shared password technique, Seo and Sweeney (Seo and Sweeny 1999) proposed a simple key agreement protocol which was intended to act as a Diffie-Hellman scheme (Diffie and Hellman 1976) with user authentication. In the Seo-Sweeney protocol, two parties who have shared a common password can establish a session key by exchanging two messages. The authors also claimed that key validation can be achieved by exchanging two more messages. Later, Tseng (Tseng 2005) addressed a weakness in the key validation steps of the Seo- Sweeney protocol. By replying to the message sent from the honest party, the adversary can fool the honest party into believing a wrong session key. Tseng modified the key validation steps of the Seo-Sweeney protocol and claimed that key validation can be achieved in the modified protocol.

In the Tseng's modified protocol, as in the original Diffie-Hellamn scheme (Diffie and Hellman 1976), the system possesses two public values *n* and *g*, where *n* is a large prime number and *g* is a generator with order *n-1* in *GF(n)*. Let Alice and Bob denote the two parties who have shared a common password P. The protocol has two phases, the key establishment phase and key validation phase, and can be describe as follows:

Key establishment phase:

(e.1) Alice and Bob each compute two integers Q and Q^{-1} *mod (n-1)* from *P,* where Q is computed in a predetermined way and is relatively prime to *n-1*.

(e.2) Alice selects a random integer *a* and sends Bob

$$X_1 = g^{aQ} \bmod n$$

(e.3) Bob also selects a random integer *b* and sends Alice

$$Y_1 = g^{bQ} \bmod n$$

(e.4) Alice computes the session key Key_1 as follows:

$$Y = (Y_1)^{Q^{-1}} \bmod n = g^b \bmod n$$
$$Key_1 = Y^a \bmod n = g^{ab} \bmod n$$

(e.5) Bob computes the session key Key_2 as follows:

$$X = (X_1)^{Q^{-1}} \bmod n = g^a \bmod n$$
$$Key_2 = X^b \bmod n = g^{ab} \bmod n$$

Key validation phase:

(v.1) Alice sends Y to Bob.

(v.2) Bob sends X to Alice.

(v.3) Alice and Bob check whether $X = g^a \bmod n$ and $Y = g^b \bmod n$ hold or not, respectively.

2.2.1. Cryptanalysis Of Tseng's Modified Key Agreement Protocol

From Tseng's point of view (Tseng 2005), with the modified protocol, when Eve (attacker) receives $X_1 (X_1 = g^{aQ} \bmod n)$ from Alice (user), Eve must compute $X = X_1^{Q^{-1}} \bmod n$ = $g^a \bmod n$ and then sends it to Alice in the verification steps of the session key. However, it is impossible to obtain g^a mod n and Q, since the problem combined with the discrete logarithm and a secret password. Eve cannot therefore compute the correct X from X_1. Moreover, in the modified protocol, X and Y computed in the session key establishment phase. Compared with the original protocol, the modified protocol reduces the computational time by two exponentiations.

However, Ku and Wang (Ku and Wang 2000) showed that Tseng's scheme is vulnerable to two attacks, called the backward replay attack and the modification attack.

2.2.1.1. Backward Replay Attack

Upon seeing X_1 sent by Alice in step (e.2), the adversary (Eve) can masquerade as Bob to re-send it back to Alice in step (e.3) as Y,. Consequently, Alice will compute

$$Y = Y_1^{Q^{-1}} \bmod n = X_1^{Q^{-1}} \bmod n = g^a \bmod n$$
$$Key_1 = Y^a \bmod n = g^{a^2} \bmod n$$

and send Y to Bob in step (v.1). Then, Eve can masquerade as Bob to re-send Y back to Alice in step (v.2) as X. Since Y = g^a mod n holds, Alice will be fooled into believing the wrong session key Key$_1$. It should be noted that if step (v.1) and step (v.2) are exchanged, the protocol is still vulnerable to the replay attack, in which Eve masquerades as Bob to start another protocol run with Alice by using X$_1$. The message sent by Alice in the first key validation step of the new protocol run can be used by Eve in the second key validation step of the original protocol run. Again, Alice will be fooled into believing the wrong session key.

2.2.1.2. Modification Attack

Upon seeing X$_1$ sent by Alice in step (e.2), Eve can replace it with any number ε $[1, n-1]$, say $X_1^{'}$. In step (e.3), Bob sends Y, to Alice, and then Alice sends the corresponding response Y to Bob in step (v.1). In step (v.2), Bob will send $X^{'}$, which equals $(X_1^{'})^{Q^{-1}} \bmod n$, to Alice. Because $X^{'} \neq g^a \bmod n$, Alice will not believe Key_1. However, since $Y = g^b \bmod n$ holds, Bob will believe the wrong session key $Key_2^{'}$ which equals $(X_1^{'})^{bQ^{-1}} \bmod n$ Although Eve cannot compute $Key_2^{'}$, she can still fool Bob into believing this wrong session key. Note that if step (v.1) and step (v.2) are exchanged, the protocol is still vulnerable to the modification attack in the opposite direction, i.e. it is Alice rather than Bob who will be fooled into believing a wrong session key.

2.3. KU-WANG KEY AGREEMENT PROTOCOL

Ku and Wang (Ku and Wang 2000) demonstrated two attacks on Tseng's enhancement. The first one is called backward replay without modification, in which the adversary can masquerade as one communicating party to fool the other one into believing the wrong session key by replaying the exchanged message. The second one is called modification attack, in which the adversary interposing in the line between two communicating parties can manipulate the exchanged message to convince one party of a wrong session key. They further proposed a countermeasure to eliminate these security flaws inherent in Tseng's improved protocol.

Brief description of Ku-Wang key agreement protocol is given below:

Let *n* be a large prime and *g* Є Z_n a generator with order n-1. Assume that two communicating parties, Alice and Bob, share a common password P in advance. Alice and Bob can pre-compute two integers Q and Q^{-1} (mod n) from P in any predetermined way before performing the key agreement protocol. Detailed description of this protocol is given below.

(1) Key establishment: Procedure of establishing the session key shared between Alice and Bob is described as follows.

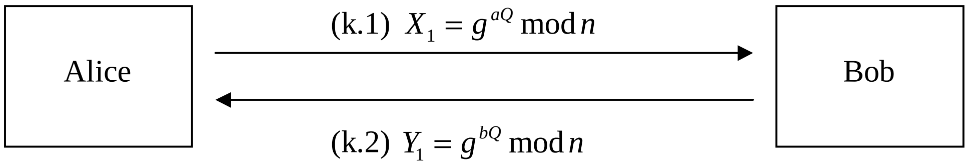


Figure 2.1: Key Establishment Phase of Ku-Wang Key Agreement Protocol

Alice randomly chooses an integer *a*, computes $X_1 = g^{aQ} \bmod n$, and then sends message (k.1) to Bob. By the same way, Bob sends message (k.2) to Alice, where *b* is a random number chosen by Bob. After that, Alice first computes $Y = Y_1^{Q^{-1}} = g^b \bmod n$ and then derives the session key K_1 by $K_1 = Y^a \bmod n$. Similarly, Bob can obtain the session key $K_2 = X^b \bmod n$, where $X = X_1^{Q^{-1}} = g^a \bmod n$. Note that the shared session key is regarded as $K_1 = K_2 = g^{ab} \bmod n$.

(2) Key validation: To check the validity of the established session key, Alice and Bob should cooperatively perform the following protocol:

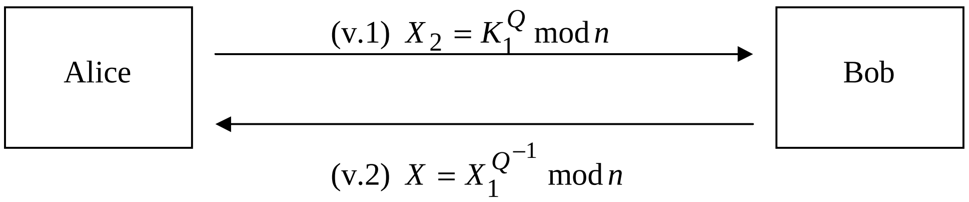


Figure 2.2: Key Validation Phase of Ku-Wang Key Agreement Protocol

Alice computes $X_2 = K_1^Q \bmod n$ and then sends message (v.1) to Bob. Upon receiving message (v.1) from Alice, Bob checks whether if $X_2^{Q^{-1}} \bmod n = K_2$. If it holds, Bob believes that he has obtained the correct X_1 and Alice has obtained the correct Y_1, i.e. Bob is convinced that K_2 is validated and then sends message (v.2) to Alice. On the other side, Alice checks whether X = g^a mod n holds or not. If it holds Alice believes that she has obtained the correct Y_1 and Bob has obtained the correct X_1, i.e. Alice is convinced that Key_1 is validated.

2.3.1. Cryptanalysis Of Ku-Wang Key Agreement Protocol

The weakness of the Seo-Sweeney protocol is due to the same values of the two key validation messages. One problem within Tseng's modified protocol is that the values of the two key validation messages will be the same once $Y_1 = X_1$. Another problem within Tseng's modified protocol is that Bob cannot judge the correctness of X_1 from the received Y. In the enhanced key validation steps, the first key validation message is directly inherited from the Seo-Sweeney protocol while the second key validation message is adopted from Tseng's modified protocol. The use of asymmetric messages in the enhanced key validation steps is one of the methods of resisting the attack of backward replay without modification (Gong 1993). In addition, according to the (Ku and Wang 2000) the first key validation message, X_2, can alternatively be generated from $X_2 = (Y_1)^a$ mod n and verified by checking whether $X_2 = (X_1)^b$ mod n. This alternative is useful if the protocol is implemented in hardware. As the generation (or verification) of Y_2 can be performed in parallel with the session key generation, the computation delay can be reduced.

However, Hsu et al. (Hsu et al. 2003) showed that Ku and Wang's scheme is weak to the modification attack, in which an adversary fools two communicating parties into sharing a wrong session key.

2.3.1.1. Modification Attack

Let Eve be an active adversary who interposes the communication between Alice and Bob. In the key establishment, Eve could manipulate the exchanged messages to plot the modification attack as follows.

$$\begin{tikzcd}
\boxed{\text{Alice}} \arrow[r, shift left, "(k.1) X_1 = g^{aQ} \bmod n"] & \boxed{\text{Eve}} \arrow[l, shift left, "(k.2') Y_1' = Y_1^{u} \bmod n"] \arrow[r, shift left, "(k.1') X_1' = X_1^{t} \bmod n"] & \boxed{\text{Bob}} \arrow[l, shift left, "(k.2) Y_1 = g^{bQ} \bmod n"]
\end{tikzcd}$$

Figure 2.3: Manipulation In Key Establishment Phase For Modification Attack.

Upon intercepting message $(k.1)$ sent by Alice, Eve can replace it with message $(k.1')$, where *t* is a random integer arbitrarily chosen by her. Similarly, Eve chooses another random integer *u*, computes $Y_1' = Y_1^u \bmod n$, and replaces message $(k.2)$ sent by Bob with message $(k.2')$. Here, the session key obtained by Alice is $K_1' = g^{abu} \bmod n$, while that obtained by Bob is $K_2' = g^{abt} \bmod n$. To convince Alice and Bob of K_1' and K_2', Eve will intervene in the key validation as follows.

$$\begin{tikzcd}
\boxed{\text{Alice}} \arrow[r, shift left, "(v.1) X_2 = (K_1')^{Q} \bmod n"] & \boxed{\text{Eve}} \arrow[l, shift left, "(v.2') X' = X^{t^{-1}} \bmod n"] \arrow[r, shift left, "(v.1') X_2' = X_2^{tu^{-1}} \bmod n"] & \boxed{\text{Bob}} \arrow[l, shift left, "(v.2) X = (X_1')^{Q^{-1}} \bmod n"]
\end{tikzcd}$$

Figure 2.4: Manipulation In Key Validation Phase For Modification Attack.

On seeing message $(v.1)$ sent by Alice, Eve replaces it with message $(v.1')$. Similarly, Eve replaces messages $(v.2)$ with $(v.2')$. Since $(X_2')^{Q^{-1}} = g^{abt} = K_2' \bmod n$, Bob will be fooled into believing that his obtained key (K_2') is verified. Similarly, Alice is also deceived that K' is validated, since $X' = X^{t^{-1}} = g^a \bmod n$, where $X = (X_1)^{Q^{-1}} = g^{at} \bmod n$. It is to see that although Eve cannot obtain K_1' or K_2', she can still fool Alice and Bob into believing their wrong session keys. So the Ku and Wang's scheme is vulnerable to the modification attack

2.4. HSU ET AL. KEY AGREEMENT PROTOCOL

In 2000, Ku and Wang (Ku and Wang 2000) pointed out that the Tseng (Tseng 2005) scheme suffers from two kinds of attacks: the backward replay attack without modification and the modification attack. In the first attack, an attacker can masquerade as one communicating party and replay the exchanged messages to cheat the other one. In the second attack, an attacker can alter the exchanged messages to cheat one party into believing a wrong session key. Ku and Wang (Ku and Wang 2000) also proposed a modified authenticated key agreement scheme to defeat these two attacks. Unfortunately, in 2003, Hsu et al. (Hsu et al. 2003) showed that the Ku–Wang scheme is still vulnerable to the modification attack and gave an improvement to enhance the security of the Ku–Wang scheme. Moreover, the Hsu et al. scheme is more efficient than the previous schemes (Ku and Wang 2000, Seo and Sweeny 1999, Tseng 2005).

Some notations which are used in Hsu et al. scheme and protocol description are given below.

i. Alice, Bob : two communicating parties
ii. Eve : an attacker
iii. id_A, id_B : the identities of Alice and Bob
iv. n : a large prime number
v. g : a generator Є Z_n^* with order $n-1$
vi. P : the common password shared between Alice and Bob
vii. Q : an integer computed from P
viii. Q^{-1} : the inverse of Q $(\bmod n)$
ix. a : a random number chosen by Alice
x. b : a random number chosen by Bob
xi. $H(.)$: a one-way hash function

There are two phases in the Hsu et al. scheme which are key establishment phase and key validation phase.

Key establishment phase:

(e.1) Alice computes $X_1 = g^{aQ} \bmod n$ and sends X_1 to Bob, where *a* is a random number.

(e.2) Bob computes $Y_1 = g^{bQ} \bmod n$ and sends Y_1 to Alice, where *b* is a random number.

(e.3) Alice computes the session key K_1 as follows:

$$Y = Y_1^{Q^{-1}} \bmod n = g^b \bmod n,$$

$$K_1 = Y^a \bmod n = g^{ab} \bmod n.$$

(e.4) Bob computes the session key K_2 as follows

$$X = X_1^{Q^{-1}} \bmod n = g^a \bmod n$$
$$K_2 = X^b \bmod n = g^{ab} \bmod n$$

After the step *(e.4),* two communicating parties, Alice and Bob, compute same session key $K_1 = K_2 = g^{ab} \bmod n$.

Key validation phase:

(v.1) Alice computes $X_2 = H(id_A, K_1)$ and sends X_2 to Bob.

(v.2) Bob verifies the validation of the equation $X_2 = H(id_A, K_2)$

(v.3) If it holds, Bob computes $Y_2 = H(id_B, K_2)$ and sends Y_2 to Alice

(v.4) Alice also verifies the validation of the equation $Y_2 = H(id_B, K_1)$.

After the step *(v.4),* Alice and Bob are now convinced the common secret key $K_1 = K_2 = g^{ab} \bmod n$.

2.4.1. Cryptanalysis Of Hsu et al. Key Agreement Protocol

Hsu et al. claimed that their scheme can withstand the modification attack. Eve must compute $g^{abt} (\bmod n)$ and send $X_2^{'} = H(id_A, g^{abt} (\bmod n))$ to Bob. However, it is impossible to obtain $g^{abt} (\bmod n)$ since the problem is based on the intractability of solving the discrete logarithm problem and the difficulty of compromising the password. Hence, Eve cannot fool Bob into believing a wrong session key. For the same reason, Eve cannot cheat Alice to accept a wrong session key. Thus, the proposed improvement is secure against the modification attack. As compared with that of Ku and Wang's key validation, the computation complexities of the proposed improvement reduces two exponentiation operations but requires two more one-way hash function operations.

However, Lee and Lee (Lee and Lee 2004) found that the Hsu et al.'s claim is not correct. They showed that Hsu et al.'s scheme has a weakness against the modification attack.

2.4.1.1. Modification Attack

According to Lee and Lee (Lee and Lee 2004) Hsu et al. scheme still suffers from the modification attack. An attacker Eve can alter the exchanged messages in the Key establishment phase to plot the modification attack as follows.

$(e.1')$ Eve replaces X_1 with $X_1' = X_1^t \bmod n$ in the step $(e.1)$, and then sends X_1' to Bob.

$(e.2')$ Eve replaces Y_1 with $Y_1' = Y_1^t \bmod n$ in the step $(e.2)$, and then sends Y_1' to Alice.

Finally, Bob computes $X' = X_1^{'Q^{-1}} \bmod n \ (= g^{at} \bmod n)$ and the wrong session key $K_2' = X^{'b} \bmod n \ (= g^{abt} \bmod n)$. Alice computes $Y' = Y_1^{'Q^{-1}} \bmod n \ (= g^{bt} \bmod n)$ and the wrong session key $K_1' = Y^{'a} \bmod n \ (= g^{abt} \bmod n)$. Since K_1' is equal to K_2', the message digest $X_2' = H(id_A, K_1')$ will also be equal to $H(id_A, K_2')$. Eve can cheat Bob into accepting the wrong section key K_2'. Similarly, Eve can cheat Alice into accepting K_1'. Thus, the Hsu et al. scheme is still vulnerable to the modification attack.

2.5. LEE AND LEE KEY AGREEMENT PROTOCOL

Recently, Lee and Lee showed that Hsu et al.'s authenticated key agreement scheme is vulnerable to the modification attack and then proposed an improved scheme.

Assume that two communication parties, called Alice and Bob, share a common password P before the scheme begins, and that the system parameters are n and g, where n is a large prime and g is a generator with order $n-1$ in $GF(n)$. Alice and Bob each compute two integers

Q and $Q^{-1} \bmod (n-1)$ from the password P, where Q could be computed in any predetermined way from P. The Lee and Lee's scheme (Lee and Lee 2004) is as follows:

Key establishment phase:

- Alice computes $X_1 = g^{aQ} \bmod n$ and sends X_1 to Bob, where a is a random number.
- Bob computes $Y_1 = g^{bQ} \bmod n$ and sends Y_1 to Alice, where b is a random number.
- Alice computes the session key K_1 as follows:

$$Y = Y_1^{Q^{-1}} \bmod n = g^{b} \bmod n,$$

$$K_1 = Y^{a} \bmod n = g^{ab} \bmod n.$$

- Bob computes the session key K_2 as follows

$$X = X_1^{Q^{-1}} \bmod n = g^{a} \bmod n$$

$$K_2 = X^{b} \bmod n = g^{ab} \bmod n$$

Key validation phase:

- Alice computes X_2 and sends it to Bob

$$X_2 = H(id_A, X_1, K_1)$$

- Bob checks whether X_2 is equal to $H(id_A, X_1, K_2)$.

$$X_2 \overset{?}{=} H(id_A, X_1, K_2)$$

If it holds Bob computes Y_2 and sends it to Alice

$$Y_2 = H(id_B, Y_1, K_2)$$

- Alice checks whether Y_2 is equal to $H(id_B, Y_1, K_1)$.

$$Y_2 \overset{?}{=} H(id_B, Y_1, K_1)$$

To establish a common session key, Alice randomly chooses an integer a, computes $X_1 = g^{aQ} \bmod n$, and then sends message X_1 to Bob. By the same way, Bob sends the message $Y_1 = g^{bQ} \bmod n$ to Alice, where b is a random number chosen by Bob. After that, Alice computes $Y = (Y_1)^{Q^{-1}} = g^b \bmod n$ and derives the session key $K_1 = (Y)^a = g^{ab} \bmod n$. Similarly, Bob can obtain the session key $K_2 = (X)^b = g^{ab} \bmod n$. To validate the established session keys, Alice computes $X_2 = H(id_A, X_1, K_1)$, where H is a one-way hash function and id_A is her identifier. Then, Alice sends the message X_2 to Bob. Bob validates the equation $X_2 \overset{?}{=} H(id_A, X_1, K_2)$. If it holds, Bob computes $Y_2 = H(id_B, Y_1, K_2)$ where id_B is his identifier, and then sends the message Y_2 to Alice. Alice validates the equation $Y_2 \overset{?}{=} H(id_B, Y_1, K_1)$. Finally, the shared session key is $K_1 = K_2 = g^{ab} \bmod n$.

2.5.1. Cryptanalysis Of Lee And Lee Key Agreement Protocol

2.5.1.1. Modification Attack

According to the modification attack, we assume that the transmitted messages in the *Key establishment phase* have been altered by an attacker Eve. That is, Eve replaces X_1 and Y_1 with $X_1^{'}$ and $Y_1^{'}$. Then, Eve has to compute $X_2 = H(id_A, X_1^{'}, K_2^{'})$ and $Y_2 = H(id_B, Y_1^{'}, K_1^{'})$ to convince Bob and Alice in the *Key validation phase*. Obviously, Eve needs to know $K_1^{'}(K_1^{'} = K_2^{'})$ before computing X_2 and Y_2. To find $K_1^{'}(= g^{abt} \bmod n)$ from $X_1(= g^{aQ} \bmod n)$ and $Y_1(= g^{bQ} \bmod n)$ is computationally infeasible, because that the attacker has to solve the discrete logarithm problem and the difficulty of compromising the password. Therefore, according to Lee and Lee (Lee and Lee 2004), the *modification attack* cannot work in the Lee and Lee's scheme. Moreover, Lee and Lee's scheme keeps the same efficiency as compared with the Hsu et al. scheme.

However, Lee et al. (Lee et al. 2005) argued that Lee and Lee's scheme is also vulnerable to a password guessing attack

2.5.1.2. Password Guessing Attack

Suppose that an adversary, called Eve, interposes the communication between Alice and Bob. Eve may not only eavesdrop messages but also masquerade Bob and defraud Alice to gain any verifiable data for user password. In the Lee and Lee's scheme, Alice first computes $X_1 = g^{aQ} \bmod n$ and then sends it to the other party. Upon receiving the message, Eve to pretend to be Bob computes $Y_1^{'} = g^{bQ} \bmod n$, where b is a random number chosen by Eve, and sends it to Alice. After receiving $Y_1^{'}$, Alice computes $Y = (Y_1^{'})^{Q^{-1}} = g^{bQ^{-1}}$, $K_1 = (Y)^a = g^{abQ^{-1}} \bmod n$ and $X_2 = H(id_A, X_1, K_1) = H(id_A, g^{aQ}, g^{abQ^{-1}})$ in sequence and then sends X_2 to the other party. On the other hand, Eve can obtain $K_2 = (X_1)^b = g^{abQ} \bmod n$ using the received message X_1. After receiving X_2 from Alice, Eve guesses a candidate password $P^{'}$ and computes $Q^{'}$ and $Q^{'-1}$ from $P^{'}$. Then she can verify the correctness of $P^{'}$ by computing $H(id_A, X_1, (K_2)^{(Q^{'-1})^2}) = H(id_A, g^{aQ}, g^{abQ^{'-1}})$ and comparing it with X_2. If they are equal, the user's password P is guessed. Otherwise, Eve tries the next candidate password until they are equal. Therefore, the Lee and Lee's scheme is vulnerable to the password guessing attack.

2.6. LEE-KIM-YOO PASSWORD BASED KEY AGREEMENT PROTOCOL

Lee and Lee found Hsu et al.'s scheme still suffers from the modification attack, and then proposed an improved scheme to repair the security flaw (Lee and Lee 2004). However, Lee, Kim, Yoo will show that the Lee and Lee's scheme cannot withstand the password guessing attack. Moreover, they propose an improved scheme to solve this problem of the scheme.

Key establishment phase:

- Alice computes $X_1 = g^{aQ} \oplus Q \bmod n$ and sends X_1 to Bob, where a is a random number.
- Bob computes $Y_1 = g^{bQ} \oplus Q \bmod n$ and sends Y_1 to Alice, where b is a random number.
- Alice computes the session key K_1 as follows:

$$K_1 = (Y_1 \oplus Q)^{aQ^{-1}} \bmod n = g^{ab} \bmod n.$$

- Bob computes the session key K_2 as follows

$$K_2 = (X_1 \oplus Q)^{bQ^{-1}} \bmod n = g^{ab} \bmod n$$

Key validation phase:

- Alice computes X_2 and sends it to Bob

$$X_2 = H(id_A, X_1, K_1)$$

- Bob checks whether X_2 is equal to $H(id_A, X_1, K_2)$.

$$X_2 \overset{?}{=} H(id_A, X_1, K_2)$$

If it holds Bob computes Y_2 and sends it to Alice

$$Y_2 = H(id_B, Y_1, K_2)$$

- Alice checks whether Y_2 is equal to $H(id_B, Y_1, K_1)$.

$$Y_2 \overset{?}{=} H(id_B, Y_1, K_1)$$

To establish a session key, Alice selects a random number a, computes $X_1 = g^{aQ} \oplus Q \bmod n$, and then sends it to Bob. Bob also computes $Y_1 = g^{bQ} \oplus Q \bmod n$ and sends the message to Alice, where b is a random number chosen by Bob. Upon receiving Y_1 from Bob, Alice computes a session key $K_1 = (Y_1 \oplus Q)^{aQ^{-1}} \bmod n = g^{ab} \bmod n$ using a and Q. Similarly, Bob computes a session key $K_2 = (X_1 \oplus Q)^{bQ^{-1}} \bmod n = g^{ab} \bmod n$ using b and Q. To validate the computed session key in the key confirmation phase, Alice computes $X_2 = H(id_A, X_1, K_1)$ and sends it to Bob, who checks if $X_2 = H(id_A, X_1, K_2)$; if it holds, Bob computes $Y_2 = H(id_B, Y_1, K_2)$ and sends it back to Alice. Otherwise, the protocol halts. Alice finally checks if $Y_2 = H(id_B, Y_1, K_1)$ using the message Y_2 received from Bob. If everything works correctly, the session key computed by Alice and Bob is $K_1 = K_2 = g^{ab} \bmod n$.

2.6.1. Cryptanalysis Of Lee-Kim-Yoo Password Based Key Agreement Protocol

According to Known, Hwang, Kim, Lee (Kwon et al. 2005), there are two flaws of Lee-Kim-Yoo protocol (Lee et al. 2005). First one is incompleteness of a key-computation process and second one is vulnerability to an off-line dictionary attack.

2.6.1.1. Incompleteness of Lee-Kim-Yoo Password Based Key Agreement Protocol

To show that Lee-Kim-Yoo protocol (Lee et al. 2005) is incomplete, i.e., two parties may not share a common session key; an important observation is that, in some cases,

$$(g^{aQ} \oplus Q \bmod n) \oplus Q \neq g^{aQ} = (g^{aQ} \oplus Q) \oplus Q \bmod n$$

where $g^{aQ}\ \varepsilon\ GF(n)$ and so $g^{aQ} \leq n-1$. The above case happens when $g^{aQ} \oplus Q$ is larger than $n-1$ since a modular multiplication in $GF(n)$ and a bit-wise XOR operation are not associative. That is, if $g^{aQ} \oplus Q \geq n$ then applying the modular operation $(\bmod n)$ to this value results in $g^{aQ} \oplus Q = q \cdot n + r (\bmod n) = r$, which is a random number. Hence Bob cannot obtain an intended value $g^{aQ} \bmod n$ from $X_1 = g^{aQ} \oplus Q \bmod n$ by computing $X_1 \oplus Q$. After all, Alice and Bob cannot compute a common session key.

Example:

Now we consider a concrete toy-example in which Lee-Kim-Yoo is incomplete. In the following description we denote an integer by its binary representation, additionally. The arithmetic is in $GF(11)$. Suppose that g^{aQ} and Q computed by Alice are $8(01000_{(2)})$ and $4(00100_{(2)})$, respectively. Then Alice computes $X_1 = g^{aQ} \oplus Q(\bmod 11) = 8(01000_{(2)}) \oplus 4(00100_{(2)}) = 12(01100_{(2)})(\bmod 11) = 1(00001_{(2)})$ and sends X_1 to Bob. Upon receiving X_1 from Alice, Bob computes $X_1 \oplus Q = (0001_{(2)}) \oplus (00100_{(2)})$ and $K_2 = (X_1 \oplus Q)^{bQ^{-1}}$. But Bob obtains $g^{aQ} = 5(00101_{(2)})$ instead of the correct value $g^{aQ} = 8(01000_{(2)})$ intended by Alice. After all, Alice and Bob cannot share a common session key $g^{aQ} \bmod n$.

2.6.1.2. Offline-Dictionary Attack

Known, Hwang, Kim, and Lee (Kwon et al. 2005) show that Lee-Kim-Yoo is vulnerable to an off-line dictionary attack even if Lee-Kim-Yoo functions correctly, i.e., $g^{aQ} \oplus Q \le n-1$. They omit $\bmod n$ in obvious cases since $g^{aQ} \oplus Q = g^{aQ} \oplus Q \pmod n$ if $g^{aQ} \oplus Q \le n-1$. The vulnerability to an off-line dictionary attack is also, like the incompleteness of the scheme, caused by two different types of group operations used to make a flow. An adversary can get redundancy information by checking if a flow is in a domain. Thus an adversary can mount an off-line dictionary attack by using this membership information. The attack of Known, Hwang, Kim, and Lee is given below:

Suppose that the goal of an adversary Eve is to discover the password of Alice and Bob running the scheme Lee-Kim-Yoo. Normally, Alice and Bob generate password- injected values, i.e., $X_1 = g^{aQ} \oplus Q \bmod n$ and $Y_1 = g^{bQ} \oplus Q \bmod n$, in the key establishment phase of Lee-Kim-Yoo and exchange the values each other. Eve overhears all the communication flows between Alice and Bob and obtains X_1, especially.

Now Eve mounts an off-line dictionary attack using X_1 as follows: Eve selects a candidate password $P^{'}$ from the dictionary of passwords and computes $Q^{'}$ using $P^{'}$ and $X_1^{'} = X_1 \oplus Q^{'} = (g^{aQ} \oplus Q) \oplus Q^{'}$. Eve checks whether $X_1^{'}$ is a member of $GF(n)$ or not. Note that the value $g^{aQ} \bmod n$ computed by an honest user Alice is always in $GF(n)$. If $X_1^{'}$ is not in $GF(n)$ then Eve can be certainly convinced that the guessed password $P^{'}$ is wrong. Eve runs through all the passwords $P^{'}$ from the dictionary by checking the membership of a value $X_1^{'} = (g^{aQ} \oplus Q) \oplus Q^{'}$ in $GF(n)$. The iterative works for the set of remaining candidate passwords in the different sessions will give the correct password.

2.7. LAIH-DING-HUANG PASSWORD BASED KEY AGREEMENT PROTOCOL

In 2005, Laih, Ding and Huang(Laih and Ding 2005) proposed a password-based key establishment protocol(referred to as the LDH protocol) such that a user and a server can authenticate each other and generate a strong session key by their shared weak password within a symmetric cipher in an insecure channel. In the LDH protocol, a special function, which is a combination of a picture function and a distortion function, is adopted to authenticate the user and protect the password from offline dictionary attacks. The CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) scheme (Ahn et al. 2003) is an example of such a special function.

We first introduce some notation. The special function used in (Laih and Ding 2005) is defined as $\varphi(r,s) = g(p(r,s))$, where g is a distortion function and p is a picture function. Specifically, given inputs r and s, where r is a random string of characters or bits and s is a random number, p generates a random picture which depicts r in some way. Given an input $p(r,s)$ (a picture) the distortion function g generates a distorted version $R^{'} = g(p(r,s))$ such that humans have the ability to recognize r from $R^{'}$ while a machine typically cannot.

Suppose $\{E_{pw}, D_{pw}\}$ denotes a pair of symmetric encryption/decryption functions, where pw is the secret key. H denotes a one-way hash function, n is a security parameter, and B_n denotes the set of all strings of length n, with elements drawn from some set of characters (e.g. all letters or all alphanumeric symbols). All these system parameters except pw are made known to all relevant parties. The secret key pw (a password) is only known to the user and the server.

- User (U) generates a random number t and sends it to the Server (S) with its identity ID_U as $\{ID_U, t\}$.
- S randomly selects a string r from B_n and produces a random number s. Then S computes $\varphi(r,s)$ and calculates $C_1 = E_{pw}(\varphi(r,s))$. Next it sends C_1 with the hash

value $C_2 = H(pw \| r \| t)$ to S where, as throughout, $\|$ represents the concatenation operator.

- U first decrypts the message C_1 by using his password pw as $D_{pw}(C_1)$ and obtains the distorted image. At this point, U recognizes r' from the image. Then it gets the hash $H(pw \| r' \| t)$ and checks whether or not it is equal to C_2. If it is true, $r = r'$ with high probability and U can authenticate S. However if it is not equal, then U ends the protocol. Following the authentication, it transmits $C_3 = H(1 \| pw \| r' \| t)$ to S.
- After receiving C_3, S compares it with $H(1 \| pw \| r \| t)$ to see whether or not they are equal. If yes, S authenticates U. Otherwise protocol is terminated. After authentication of both parties, U and S can share the session key $SK_U = SK_S = H(2 \| pw \| r \| t)$.

2.7.1. Cryptanalysis Of Laih-Ding-Huang Password Based Key Agreement Protocol

In their analysis of the LDH protocol, Laih, Ding and Huang claim that the protocol is secure under the condition $|B_n| + |C_{pw}| > 70$, where C_{pw} is the set of passwords that people select and $|C_{pw}|$ is the size of the password set. They assume that the brute force attack fails when the entropy of the searching space is larger than 70 bits and $|C_{pw}|$ equals 23 bits in their security analysis. In these assumptions, we select the string length $n = 8$ (i.e. the number of B_n is $62^8 > 2^{47}$ with 26 upper (lower)-case letters and ten digits) for $|B_n| + |C_{pw}| > 70$ bits. Specifically they make the following two security claims.

1. Exhaustive search by a machine

The machine first needs to compute $C_1' = E_{pw}(\varphi(r', s'))$ by guessing the values of r' and pw', and then compares C_1' and C_1 in order to verify this guess. There are 62^8 possible values for r', and 2^{23} possible values for pw', i.e. the total search space is of size

$$62^8 * 2^{23} > 2^{70}$$

So, based on the assumption that, it is computationally infeasible for the machine to compute pw.

2. Exhaustive search by a human being and a machine.

If a valid message $C_1 = E_{pw}(\varphi(r,s))$ is obtained, the machine first guesses a password pw' and computes $A = D_{pw'}(C_1)$; then the human being decides whether or not A contains a string from B_n, which indicates whether or not pw' equals pw. This process is repeated until the correct password is found. This would require the human to check $|C_{pw}| = 2^{23}$ possible values for pw'. Based on this, Laih, Ding and Huang estimate that in this case it will take about 3.2 months for a human being and a machine to successfully search for the password.

However in (Tang and Mitchell 2005), Tang and Mitchell points out that in the LDH protocol, the protection of the password is based on the security of the function φ, i.e., the assumption that a machine (without a human being involved) cannot effectively recognise r from $\varphi(r,s)$. As Laih, Ding and Huang point out in (Laih and Ding 2005), the string reorganization CAPTCHA schemes (Ahn et al. 2003) are potentially suitable choices for the function φ. However, the security of these artificial intelligence (AI) problems is based on the state of the art in pattern reorganization research, and is thus essentially heuristic. Mori and Malik (Mori and Malik 2003) have recently developed efficient methods based on shape context matching that can identify, with a high success rate (83%), the word in an ez-gimpy image, a type of CAPTCHA scheme currently in use. Thayananthan et al. (Thayananthan et a. 2003) developed a program that can achieve a ninety three percent correct recognition rate against ez-gimpy. Recently Moy et al. (Moy et al. 2004) developed a program that can achieve a seventy eight percent accuracy against gimpy-r, another type of CAPTCHA scheme.

Apart from the above problems, Tang and Mitchell exhibit a number of security vulnerabilities in the LDH protocol which exist almost regardless of the choice of φ. These vulnerabilities are based on the following observations.

1.A human being must be able to easily recognize r from $D_{pw}(\varphi(r,s))$, which implies that $D_{pw}(\varphi(r,s))$ is very different from a completely random picture.

2. If $pw^{'} = pw$ then $D_{pw^{'}}(\varphi(r,s))$ will resemble a random image. This implies that it is possible to determine whether or not a guessed password $pw^{'}$ is correct merely by deciding whether $D_{pw^{'}}(C_1)$ is a (distorted) image or a random pattern.
3. It is likely to be very simple to develop software to distinguish between a distorted image and a random pattern (for example, a compression algorithm should be able to compress an image whereas a random pattern will be incompressible). This is certainly a much simpler problem than automatic string recognition.

Specifically, the following attacks might be mounted by a machine or a human being.

2.7.1.1. Password Guessing Attack

In some cases it might be feasible for a machine to mount an offline password guessing attack. The machine works through all possible passwords and, for each guessed password $pw^{'}$, the machine computes $A = D_{pw^{'}}(C_1)$. By some means (see fact 3 above) the machine then checks whether or not A resembles a distorted image rather than a random bit pattern. Because of fact 2 above, the correct password can be identified from the unique case where A is a distorted image rather than a random bit pattern. This attack only requires a machine-based search of size $|C_{pw}|$. If, for example, it takes a millisecond to check one value of A, then checking through a password space of size 2^{23} will take only 2.3 hours. Therefore LDH protocol can not be considered as a secure protocol.

2.8. TANG-MITCHELL KEY AGREEMENT PROTOCOL

In the enhanced protocol, they make the following assumptions. Suppose a user (U) with identity ID_U and a server (S) with identity (ID_S) share a secret password pw. We also suppose that p and q are two large prime numbers, where $p = 2q+1$, and H is a secure one-way hash function. When U and S want to negotiate a session key, they first compute $g = H(pw \| ID_U \| ID_S \| i) \bmod p$, where $i(i > 0)$ is the smallest integer that makes g a generator of a multiplicative subgroup of order q in $GF(p)^*$. U and S then perform the following steps.

1. U generates a random number $t_1 \varepsilon\ Z_q^*$, and sends $m_1 = g^{t_1} \bmod p$ to S.
2. After receiving m_1, S generates a random number $t_2 \varepsilon\ Z_q^*$, and sends $m_2 = g^{t_2} \bmod p$ to U. S uses a CAPTCHA scheme to construct a distorted picture $\varphi(r)$, where r is a random string, and also sends $\varphi(r)$ to U. We suppose that the selected CAPTCHA scheme has not be broken.

   S computes $z = g^{t_1 t_2} \bmod p$ as the shared key material, and computes $K = H(z \| 1)$ as the shared key.
3. After receiving m_2, U recognizes r from the distorted picture $\varphi(r)$, computes $z = g^{t_1 t_2} \bmod p$ as the shared key material, and computes $K = H(z \| 1)$ as the shared key. Then U constructs and sends the following confirmation message to S:

$$C_1 = H(\varphi(r) \| r \| 3 \| m_1 \| m_2 \| g^{t_1 t_2} \| ID_U \| ID_S)$$

4. After receiving C_1, S checks that the received message equals

$$H(\varphi(r) \| r \| 3 \| m_1 \| m_2 \| g^{t_1 t_2} \| ID_U \| ID_S)$$

If the check fails, S terminates the protocol execution. Otherwise, S computes and sends the following confirmation message to U:

$$C_2 = H(4 \| m_1 \| m_2 \| g^{t_1 t_2} \| ID_U \| ID_S)$$

5. After receiving C_2, U checks that it equals:

$$C_2 = H(4 \| m_1 \| m_2 \| g^{t_1 t_2} \| ID_U \| ID_S)$$

If the check fails, U terminates the protocol execution. Otherwise U confirms that the protocol execution has successfully ended.

3. THE PROPOSED PROTOCOL

3.1. PASSWORD-BASED KEY ESTABLISHMENT PROTOCOL WITH SYMMETRIC KEY CRYPTOGRAPHY

In 2005, Laih, Ding and Huang proposed a password-based key establishment protocol such that a user and a server can authenticate each other and generate a strong session key by their shared weak password within a symmetric cipher in an insecure channel. In this protocol, a special function which is a combination of a picture function and a distortion function, is combined to authenticate the user and protect the password from the dictionary attacks that are major threats for most of the weak password-based protocols. They claim that the proposed protocol is secure against some well known attacks. However Tang and Mitchell shows that the protocol suffers from an offline dictionary attack requiring a machine based search of size 2^{23} which takes only about 2.3 hours. So designing such a protocol with providing practical security against offline attack is still an open problem. In this study, we introduce two password-based authenticated key establishment protocols that provide practical security against offline dictionary attacks by only using symmetric cryptography.

Passwords are the most widely used authentication method although use of them has many well-known security weaknesses such that they can be easily guessed by automated programs running dictionary attacks. The scenario in which a user and a server authenticate each other and produce a strong session key through symmetric cryptography from the low entropy password known by the both parties is very practical and convenient in the real world. However, designing a secure protocol for this scenario has been an open problem due to effectiveness of offline dictionary attacks. In (Laih and Ding 2005), C.S. Laih et. al. proposed a password-based authenticated key establishment protocol (referred as LDH) to resolve this problem. Actually, the major difference of the protocol from some well-known proposals (Bellovin and Merritt 1992, Gong et al. 1993) is it does not use public key cryptography to combine a space with password space to form a large enough space to resist the offline dictionary attack. The key idea behind this protocol is use of a special function which is consisted of a picture function and a distortion function. This function is defined as $\varphi(r,s) = g(p(r,s))$,where g is a distortion function, p is a picture function which takes random string of characters/digits r and a random number s as input arguments. The CAPTCHA (Ahn et al. 2003) which is used by several companies (Yahoo, Microsoft etc.) to avoid too many free account application from machine alone is an example of this function. A sample picture of CAPTCHA is depicted in Fig.1. By means of use of such a function, distorted picture can be easily recognized by a human, while this is a very hard problem for a machine. So according to the authors, the strength of the attacks based on only the power of machine computation can be weakened and with their proposed protocol practical security is provided. They also analyze the security of the protocol considering the scenario if both human and machine work together to crack the system and claim that such an attack takes about 3.2-month. However in (Tang and Mitchell 2005), Q. Tang and C. Mitchell investigate the security power of LDH and state that it suffers from offline dictionary attack. According to (Tang and Mitchell 2005), the basic weakness in the protocol stems from the fact that: A machine can realize the difference between a distorted image pattern and a random image pattern, so when the machine works through all possible passwords, a successful decryption means getting a non-random image pattern. As a result, for 2^{23} password search space, by using a machine which can make one check per millisecond, one can capture the password only about 2.5 hours. In fact, for this attack strategy there is no need of human assistance.

In this study, we propose a new protocol with obeying the main steps and rules of LDH to provide practical security against offline dictionary attacks. The main idea behind the proposed model is use of a CAPTCHA like problem which can be solved easily by human intelligence while it is very hard for a machine. The best attack against this model requires collaboration of both a human and a machine. Thus, we intend to realize security claims of LDH with avoiding its present weakness by introducing a different problem set instead of using image pattern recognition problem.

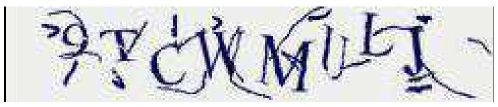


Figure 3.1: An Example Picture Of CAPTCHA.

The Proposed Protocol

We can fulfill the security claim of LDH, if we construct a problem with the following properties:

- Problem is created directly related to pw.
- One can easily solve the problem, if he knows pw; this is a hard problem otherwise.
- For the given problem exhaustive search of pw by a machine respects: Human participation must be needed to verify each guess and it must result less complexity than without the human case.

Note that, in LDH protocol the problem is mainly based on the lack of image pattern recognition capabilities of the machines. However it suffers from the last item of the above properties, because the machine has capability to check correctness of each pw individually and it requires less time compared to with human case. For the proposed model, firstly hundreds of concrete object images (a farm, a monkey, a ball, a computer, an insect etc.) are selected. Then for each image, information strings that are related the image are deduced and labeled with it. For example, for a farm picture the following strings can be written: “There are three chickens”, “I can see the apple trees”, “The color of the tractor is red” etc. These strings are stored in a database. Also, there can exist strings that are not correlated to any image in the database or some strings can be related more than one picture. For example a text string containing word ”net” can be correlated with more than one picture as shown in Fig. Of course such a picture selection increases security of the system as mentioned in the next section. We can summarize the proposed model as follows:

- U generates a random number t and sends $\{ID_U, t\}$ to S.
- Suppose the function $\Phi(N, pw, t)$ permutes the order of an image set N, then reduces its size $n = |N|$ to m and outputs image set M according to pw and t. Also, in a similar way the function $\theta(R, pw, t)$ permutes the order of a string set R with |R|= r, and gives the string set J, where $|J| = j$ and $m < j < r$. There exists a relation between M and J such that m out of j strings are previously labeled with the images in M. For example, 2nd image is related with 6th string,

3rd image is with 1st string, mth image is correlated with 1st string etc. It is expected that U can easily solve this matching problem for given M and J. Assume answer of the problem is represented by P_S. S sends the image set N, the string set R and C_1 to U, where $C_1 = H(P_s \| pw \| t \| ID_U \| ID_S)$.

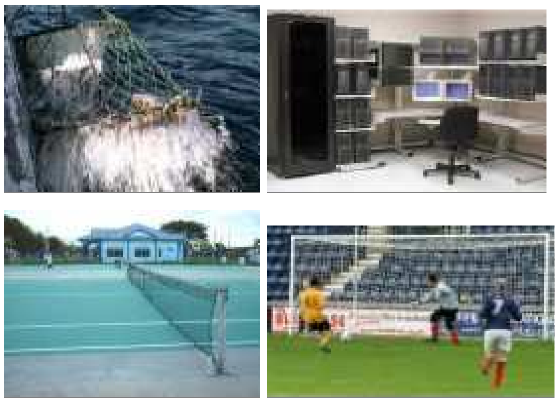


Figure 3.2: Different pictures related with the word "net".

- U evaluates $\Phi(N, pw, t)$ and $\theta(R, pw, t)$, obtains M and J respectively. U perceives the relation between images and strings and gets an answer $P_S^{'}$. Then checks whether $C_1 = H(P_S^{'} \| pw \| t \| ID_U \| ID_S)$ holds. If U does not get any sensible answer or the equality is not satisfied, then U ends the protocol. Otherwise, it authenticates S with concluding $P_S = P_S^{'}$ and transmits $C_2 = H(1 \| P_S^{'} \| pw \| t \| ID_U \| ID_S)$ to S.
- S computes $C_2 = H(1 \| P_S \| pw \| t \| ID_U \| ID_S)$ and checks it with C_2 to authenticate U. If they are equal, S agreed that U is the valid user, otherwise the protocol is terminated. After authentication of both parties they generate the session key $SK_U = SK_S = H(2 \| P_S \| pw \| t)$ and send $E_{SK_S} = (SK_S \| t)$ and $E_{SK_U} = (SK_U \| t)$ to each other to confirm that the produced session keys are equal. In case of any mismatch, the protocol is terminated.

3.1.1. Cryptanalysis Of The Proposed Protocol

3.1.1.1. Using Message C_1

Suppose for the first scenario the attack only uses machine. It is assumed that C_1, N and R are available to the attacker. Then for each pw' firstly $\Phi(N, pw', t)$ and $\theta(R, pw', t)$ are computed and corresponding M' and J' sets are obtained. For each candidate M' and J', there exists a total of $(\,{}^{j}_{m}\,)\ (m!)$ possible answers. Let B' be the possible solution set for a given pw'. Then the cost of such a process is $|C_{pw}||B'| = 2^{23+\log_2(|B'|)}$ with assuming $|C_{pw}| = 2^{23}$. For example, if $j = 17$ and $m = 8$, then $|B'| = \begin{pmatrix} 17 \\ 8 \end{pmatrix}.(8!) \cong 2^{29.87}$. In (Laih and Ding 2005), it is assumed that machine can handle 10^9 guessed password verifications per second. Hence for this example the attack requires $2^{29.87}.2^{23} = 2^{52.87}$ password verification operations that needs about 3.2 months. As it can be seen, such an attack is impractical under condition that the change period for the password is in the order of weeks or months.

3.1.1.2. Labeling N and R

For this type of attack, we assume a human can assist to the machine. In this technique, human firstly classifies strings in R and groups them with images in N. Let Δ_i denote string group of ith image, also $\Delta_{i,j}$ stand for jth string element of Δ_i. For example, suppose $string1,\ string4,\ string102, \ldots\ stringR$ correlates with $image8$, then these strings are added to the string group of $image8$ as $\Delta_8 = \{string1, string4, string102 \ldots, stringR\}$. Note that a string in the group of an image can be a common element, i.e. it can belong to other groups of images. Hence, the attacker has to analyze one by one whether any element in R relates with element in N. In other words, grouping process results in $O(|N||R)$ time complexity. Suppose, for a given image and a string human spends 1 s for recognition and to say whether they are correlated or not. Thus, grouping step requires about $|N|.|R|$ seconds. After the grouping phase of the attack, the machine evaluates $\Phi(N, pw', t)$ and $\theta(R, pw', t)$ and gets corresponding M' and J' sets for each pw'. If $\exists \Delta_{i,j} \varepsilon J' \forall i \varepsilon M'$ for some positive integer j, where $0 \le j \le |\Delta_i|$, then it is highly possible pw' is the correct password.

The complexity of the attack is mainly stemmed from the grouping step; the machine search process requires 0.01 s, so we ignore cost of this part. For example, if $|N| = 256$ and $|R| = 32768$, then the time complexity of the attack becomes $|N|.|R| = 2^{23} s$. Therefore, in this case it takes about 3.2 months for a human being and a machine to successfully search for the password. This is enough to prevent the attacker from using the offline dictionary attack.

3.1.1.3. Using Human and Machine Together

In the previous attack, although human and machine collaborate, they do separate parts of the attack. On the other hand, in this attack they work together serially such that: For each guess of pw', machine outputs corresponding M' and J' to human by computing $\Phi(N, pw', t)$ and $\theta(R, pw', t)$. Human decides whether a correlation between elements of M' and J' exists or not. If decision is yes, then he gets an answer P_s' and checks $C_1 = H(P_S' \| pw \| t \| ID_U \| ID_S)$ holds or not. Holding of equation means guessed pw' is correct. In case of a wrong guess, simply a new guess is made and process is repeated. We assume human needs 1 s to check correlation between elements of M' and J'. Thus, total complexity of the attack becomes $O(|C_{pw}|)$, which is equal to 2^{23} s with assuming $|C_{pw}| = 2^{23}$ i.e. attack requires about 3.2 months. Note that the attack complexity is not better than that of the previous attack.

Following table shows the strength of the protocols against some known attacks.

Table 3.1: The Strength Of the Protocols Against Some Known Attacks

	Replay Attack	**Modification Attack**	**Offline Dictionary Attack**	**Man In the Middle Attack**
Diffie Hellman Protocol				WEAK
Seo-Sweeney Protocol	WEAK			
Tseng's Protocol	WEAK	WEAK		
Ku-Wang Protocol		WEAK		
Hsu et al. Protocol		WEAK		
Lee and Lee Protocol			WEAK	
Lee-Kim-Yoo Protocol			WEAK	
Laih-Ding-Huang Protocol			WEAK	

3.1.2. Simulation Of The Study

To demonstrate of this study, a client-server application is developed using with Microsoft Visual Studio 2008 tool. C# language is used to developed software and Microsoft Sql Server 2005 is used to create a database for storing images and related and unrelated strings for images. The reason why Microsoft .NET Framework is used is that it includes Base Class Library which covers a large range of programming needs in a number of areas including database connectivity, cryptography, data access, etc.. Especially using with cryptography library, SHA512 class is selected to compute hash value of the data which is transmitted between client and web service.

In software engineering, three layered architecture is commonly used. These layers are presentation layer, business layer and data access layer.

Presentation Layer : The presentation layer provides the application's user interface (UI). Following figures are user interfaces of the developed software.

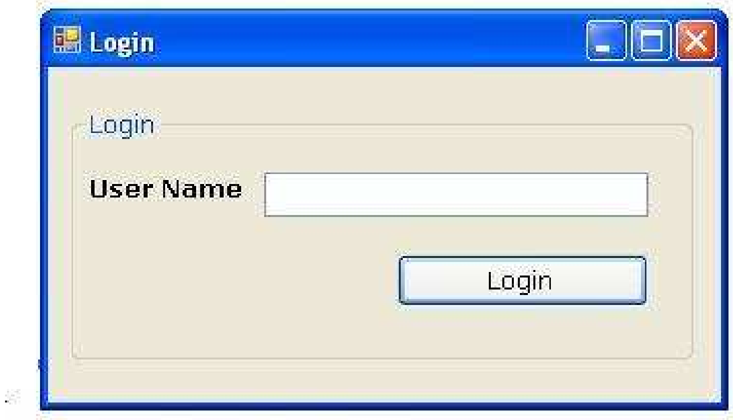


Figure 3.3: Login window of the program.

Figure 3.3 represents the login window of the program. User can enter his/her user name and send it to the server through this screen. When user presses the login button, the program gets the login time and sends it with user name to the server. Then server checks whether this user is valid user or not. If it is valid user then gets its password from database and run the proposed algorithm and send the images and strings to the user and ask him/her his/her password.

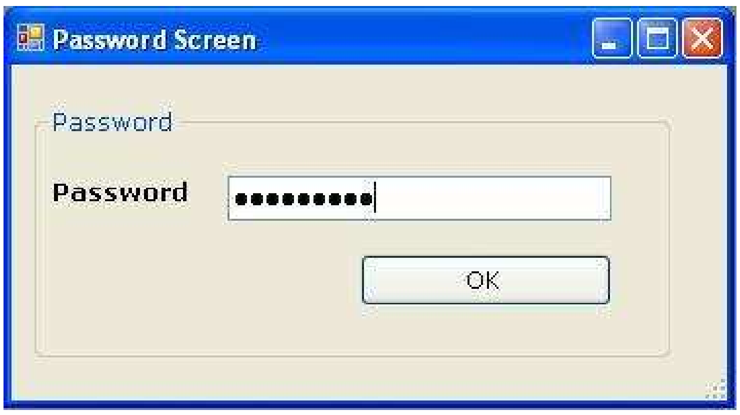


Figure 3.4: Password window of the program

Figure 3.4 shows the password screen of the program. When the user enter the OK button program display the images depends on the user’s user name and password and login time. Figure 3.5 shows that user matches the images with related strings. The program runs other steps and decides whether the session key is established or not.

Business Layer: The business layer implements the business functionality of the application. It provides the communication between presentation layer and data access layer. This layer calls the data access layer and gets the data from it and sends the data to the presentation layer.

In this application there are four classes in the business layer. The first class is WorkOnMyImages class which responsible for getting images from database through data access layer. Second class is WorkOnMyImageString. This class gets the image and its string from database. The third class is WorkOnMyString class which gets the string from database. The fourth class is WorkOnMyUser which get user information from database.

Data Access Layer: A data access layer (DAL) is a layer of a computer program which provides simplified access to data stored in persistent storage of some kind, such as an entity-relational database. For example, the DAL might return a reference to an object (in terms of object-oriented programming) complete with its attributes instead of a row of fields from a database table. This allows the client (or user) modules to be created with a higher level of abstraction. This kind of model could be implemented by creating a class of data access methods that directly reference a corresponding set of database stored procedures. For example, instead of using commands such as insert, delete, and update to access a specific table in a database, a class and a few stored procedures could be created in the database. The procedures would be called from a method inside the class, which would return an object containing the requested values. Also, business logic methods from an application can be mapped to the Data Access Layer. So, for example, instead of making a query into a database to fetch all users from several tables the application can call a single method from a DAL which abstracts those database calls. Data Access layer will make project database independent.

In proposed application, there are three classes in Data Access Layer. First one is DAOL class which is responsible for connecting database and executing queries. Second one is DataBaseFactory class which provides connection provider to desired database such as SqlClient

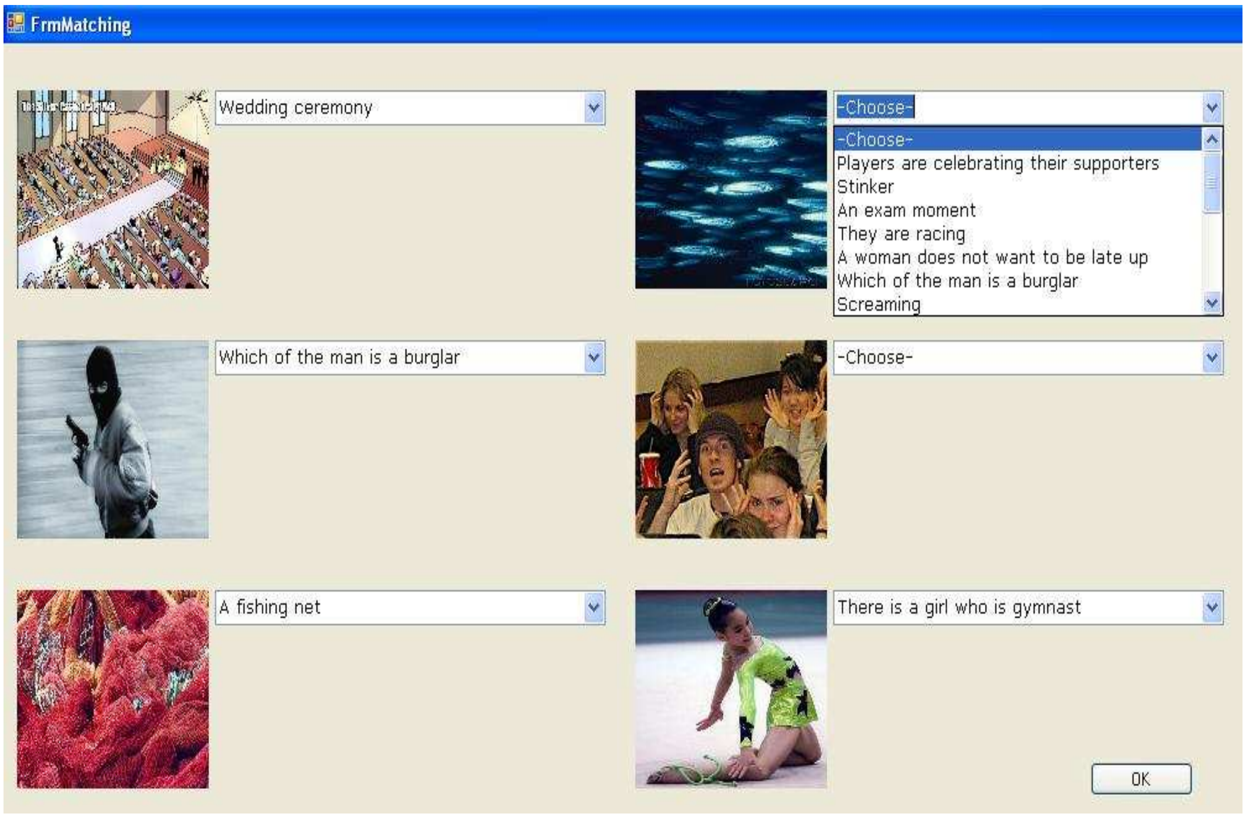

FrmMatching
Wedding ceremony
-Choose-
-Choose-
Players are celebrating their supporters
Stinker
An exam moment
They are racing
A woman does not want to be late up
Which of the man is a burglar
Screaming
Which of the man is a burglar
-Choose-
A fishing net
There is a girl who is gymnast
OK

Figure 3.5: Matching window of the program.

provider for SQL Server, Oledb and Odbc provider for other database tools. Third one is Parameter class that parameters for queries can be added to queries using with this class.

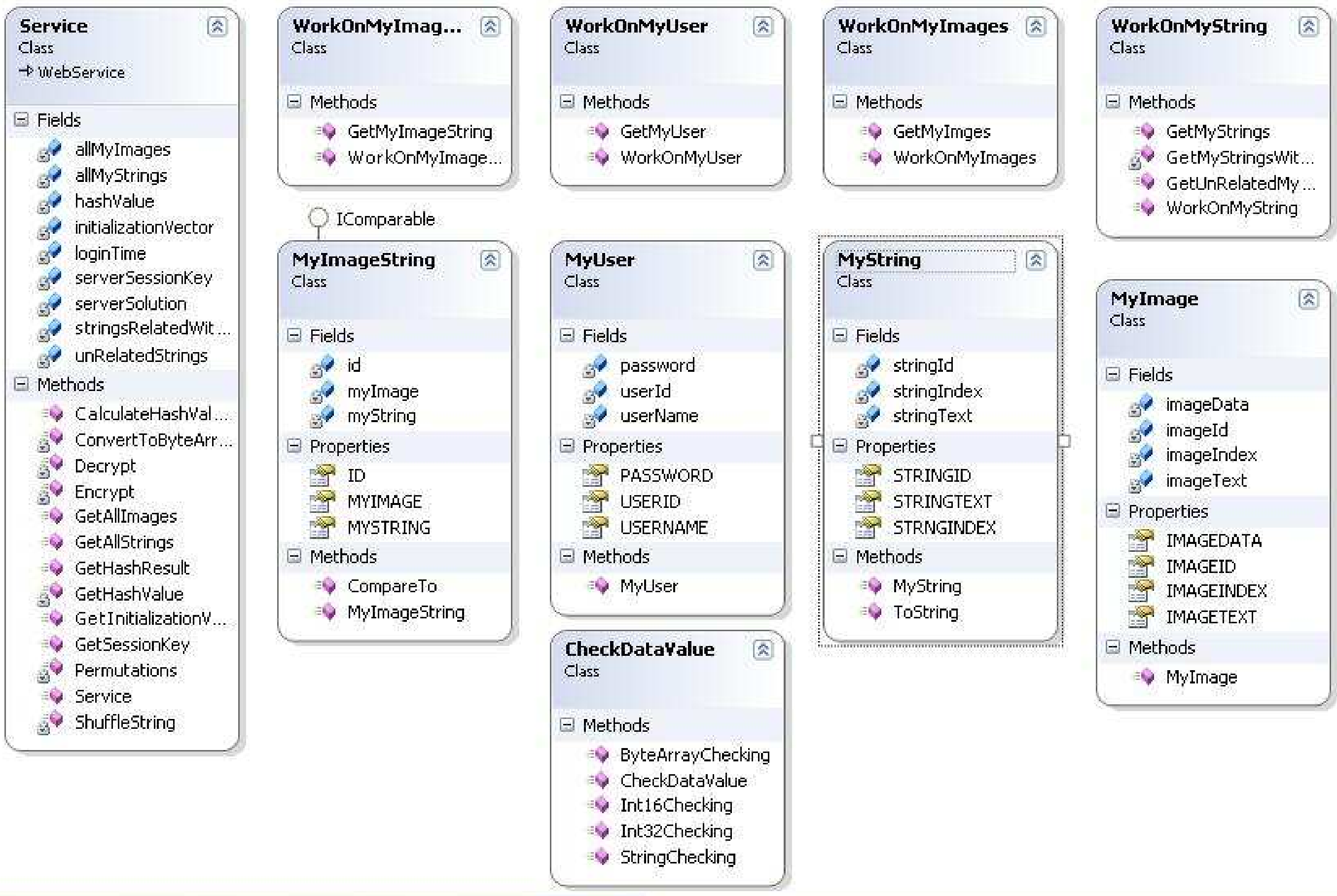


Figure 3.6: Class Diagram Of The Web Service Program

4. CONCLUSION

In this study, a new password-only authenticated key establishment protocols is presented that use only symmetric cryptography. The proposed protocols provide a practical solution to problem of offline dictionary attack from which LDH protocol suffers. By customizing and scaling the protocols they become very convenient and practical without facing the problem of public key certificates.

REFERENCES

Periodical Publications

Ahn, L., Blum, M., Hopper, N., Langford, J., 2003, Captcha: Using hard AI problems for security. In E. Biham, editor, *Advances in Cryptology - EUROCRYPT 2003, volume 2656 of Lecture Notes in Computer Science*, *Springer-Verlag*, pp. 294–311.

Anderson, R.J., Lomas, T.M.A., 1994, Fortifying key negotiation schemes with poorly chosen passwords, *Electronics Letters*, 30, pp. 1040-1041

Bellovin, S., Merritt, M., 1992, Encrypted key exchange: password-based protocols secure against dictionary attacks, in: *IEEE Symposium on Security and Privacy*, pp. 72–84.

Diffie, W., Hellman, M.E., 1976, New directions in cryptography, *IEEE Trans.*, IT-22, (6), pp. 644-654

Diffie, W., Oorschot, P.C.V., Wiener, M.J., 1992, Authentication and authenticated key exchanges, *Des. Codes Cryptography*, 2, pp. 107-125

Gong, L., Lomas, M., Needham, R., Saltzer, J., 1993, Protecting Poorly Chosen Secrets from Guessing Attacks, *IEEE J. Sel. Areas Communications*., 11, (5), pp. 648-656.

Gong, L., 1993, Variations on the themes of message freshness and replay. *Proc. IEEE Computer Security Foundations Workshop* VI, pp. 131-136

Hsu, C.L., Wu, T.S., Wu, T.C., Mitchell, C., 2003, Improvement of modified authenticated key agreement scheme, *Applied Mathematics Computation* 142, pp. 305–308.

Ku, W.C., Wang, S.D., 2000, Cryptanalysis of modified authenticated key agreement scheme, *Electronics. Letters*. 36 (21) 1770–1771.

Kwon, J.O., Hwang, J.Y., Kim, C., Lee, D.H., 2005, Cryptanalysis of Lee–Kim–Yoo password-based key agreement scheme, *Applied Mathematics and Computation* 168, pp. 858–865

Laih, C. S., Ding, L., Huang, Y. M., 2005, Password-only Authenticated Key Establishment Protocol without Public Key Cryptography, *Electronics Letters*, 41 (4), pp. 185-186.

Lee, N.Y., Lee, M.F., 2004, Further improvement on the modified authenticated key agreement scheme, *Applied Mathematics Computation* 157 pp. 729–733.

Mori, G., Malik, J., 2003, Recognizing objects in adversarial clutter: Breaking a visual CAPTCH,. *In Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR ’03), volume 1, IEEE Computer Society,* pp. 134–141.

Moy, G., Jones, N., Harkless, C., Potter, R., 2004, Distortion estimation techniques in solving visual CAPTCHAs, *In Proceedings of 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE Computer Society*, pages 23–28.

S.W. Lee, H.S. Kim, K.Y. Yoo, 2005, Improvement of Lee and Lee’s authenticated key agreement scheme, *Applied Mathematics Computation*, 162, pp. 1049-1053.

Seo, D.H., Sweeney, P., 1999, Simple authenticated key agreement algorithm, *Electronics. Letters* 35 (13) pp. 1073–1074.

Tseng, Y.M., 2005, Weakness in simple authenticated key agreement scheme, *Electronics. Letters* 36 (1) pp. 48–49.

Tang, Q., Mitchell, C., 2005, Enhanced Password-based Key Establishment Protocol, *Cryptology ePrint Archive,* Report 2005/141.

Thayananthan, A., Stenger, B., Torr, P., Cipolla, R., 2003, Shape context and chamfer matching in cluttered scenes, *In Proceedings of 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE Computer Society,* pp. 127–133.

CURRICULUM VITAE

Name Surname	: Recep GÖKÇELİ
Address	: Bahçeşehir Üniversitesi Mühendislik Fakültesi Çırağan Cd. Osmanpaşa Mektebi Sk. No: 4 – 6 34349 Beşiktaş / İstanbul / Türkiye
Birth Place / Year	: İstanbul - 1982
Languages	: Turkish (native) - English
High School	: İbrahim Turhan High School - 2000
BSc	: Bahçeşehir University - 2005
MSc	: Bahçeşehir University – 2008
Name of Institute	: Institute of Science
Name of Program	: Computer Engineering
Publications	: Karahoca A, Karahoca D., İnce İ.F., **Gökçeli R.**, Aydın N., Güngör, 2007, Intelligent Question Classification for e-learning by ANFIS, *SOCRATES – ERASMUR 114046-CP-1-BG-ERASMUS-TN EUROPEAN THEMATIC NETWORK for DOCTORIAL EDUCATION in COMPUTING, e-learning conference'07,* August 2007.
Work Experience	: Bahçeşehir University Software Engineering Department Teaching and Research Assistant (August 2005 – Today)