# Performance Analysis of Key Establishment Protocols for Secure System

Tej Bahadur Shahi
Tribhuvan University
Central Department of
Computer science and IT
Kathmandu, Nepal

Narendra Bahadur Bohara
Tribhuvan University
Sanothimi Campus
Department of Information and
Communication Technology
Kathmandu, Nepal

## ABSTRACT
Providing security over open and large distributed networks has always been both intriguing and challenging. There is a great chance for malicious individuals to perform disruptive and unethical tasks. Malicious users may attempt to obtain valuable information. So we require "secure channel" over insecure network. The secure communication channel should achieve primary security goals like confidentiality, integrity, authentication and non-repudiation and shared session keys are incorporated for the purpose. Therefore, it is of great interest and most challenging to devise effective mechanisms to establish these shared session keys, called key distribution problem. Much work has been done in recent years on mechanisms for key establishment. Many cryptosystems rely on cryptographically secure keys and therefore have to deal with issues like key management. A number of key establishment protocols have been proposed by different researchers as solutions to the key distribution problem and password based scheme is one of them. A password is shared between the entities in password based schemes. However, because users choose small and frequently used words as passwords, these schemes are suffered from password guessing attacks. Especially these schemes are subject to offline dictionary attacks. This work focus on password based key establishment. Even though there are a lot of password based schemes, the LDH, enhanced LDH, and PP-TAKE seems to be widely accepted mechanisms. In this context, this study includes performance evaluation of the above mentioned protocols.

## General Terms
Security, Key Management, Protocols, cryptosystems, Key Establishment, GF (p).

## Keywords
LDH, CAPTCHA, PP-TAKE, RSA, key establishment protocols.

## 1. INTRODUCTION
When we transmit confidential messages from one user to another over an insecure network, an eavesdropper can be able to record, alter, delete, insert, reorder and reuse past or current messages i.e. one can disclose its secrecy and modify as well. So, an eavesdropper can break the security of the system in different ways. If the attacker attempts only to defeat a cryptographic technique by simply recording data and thereafter analyzing it, then this attack is called passive attack and an active attack involves modification of messages. Therefore the security of the cryptosystem is of great importance and challenging as well. To prevent from such attacks and keep the cryptosystem secure, the system should ensure data confidentiality, data integrity, data origin authentication and non-repudiation [20]. To gain these security goals, we need to transmit messages in such a form that an eavesdropper might not be able to read the messages. For the purpose, some shared session key is required and incorporated it to keep the system secure. The session keys help to make the communication secure because it would vary with every execution round (session) of protocol in order to make the adversaries unable to know the messages exchanges in later communications even though the key in previous session is compromised. So session keys are supposed to be ephemeral (temporary) ones. The secret key can subsequently be used to create a secure communication channel among the entities. The problem of finding effective mechanisms to establish these shared session keys is called key distribution problem. Various key establishment protocols are used as solutions to the problem and these are not efficient equally from security and execution time aspects. So finding an efficient protocol is of great interest.

## 2. PROBLEM FORMULATION
To achieve data confidentiality in a session established by party *A* with intended recipient *B*, one may use a cryptographic algorithm, called symmetric encryption, which given a plaintext message *m*, produces a cipher text *c* that *A* can subsequently send to *B* over the network. This cipher text has the property that it does not reveal any information about the original plaintext to anyone except *A* and *B*. This property can be achieved since the algorithm requires *A* and *B* to share a piece of secret information, known as a shared session key, that is fresh and unique for each session. Similar reasoning show that the other four goals of secure communications can be accomplished, provided that shared session keys are readily available to each pair of parties. Therefore, it is of great concern to devise effective mechanisms to establish these shared session keys, called key distribution problem.

The solution to the problem is to have the two parties share secret information (i.e. password) for key establishment, without the need for a trusted party, i.e. password based key establishment protocols are solutions to key distribution problem. Therefore, key establishment protocols are procedures to securely establish shared session keys over distributed networks. There are two major techniques of key establishment, key transport and key agreement and our study focuses on key agreement protocols.

## 3. RESEARCH METHODOLOGY
### 3.1 Literature Review
Key agreement protocols have a very long history; the first well known protocol was Diffie-Hellman proposed in 1976. The security of Diffie-Hellman algorithm is based on the

difficulty of computing discrete logarithms over a finite field in reasonable time frame. it suffers from man-in-the-middle attacks[ REFERENCE]. After Diffie-Hellman protocol came to existence, many other key agreement protocols have been proposed by various researchers. The concept of password-authenticated key exchange was first proposed by Bellovin and Merritt in 1992[8]. In their paper, Encrypted Key Exchange (EKE) protocol was proposed as a solution for remembering long keys for users. EKE uses private and public cryptographic techniques and it provides the adversary no sufficient information to verify his guessed password and therefore it is resistant to online-dictionary attacks. Later on the authors of EKE extended their protocol to Augmented EKE (A-EKE) to detect the attack on the host where the password is stored by avoiding storing the password in clear text format. Instead of storing passwords in form of plain texts, these are stored as hash values. Encrypted Key Exchange protocols easily work with Diffie-Hellman key exchange (DH-EKE) [26] and the messages that needs to be exchanged between the parties is first encrypted using the shared password and are transmitted. But the EKE protocols do not work with other public-key cryptosystems like the RSA and ElGamal. In 1996, David Jablon proposed SPEKE (Simple Password Encrypted Key Exchange) [1] as the extension of EKE. Besides preventing password from dictionary attack, DH-EKE and SPEKE protocols achieve perfect forward secrecy; it means that the remaining session keys are not compromised even though the password is disclosed. Bellovin and Merritt pointed out in [1] that the RSA-based EKE variant is subject to a special type of dictionary attack, called e-residue attack. In 1997, Lucks [2] proposed an RSA-based password-authenticated key exchange protocol (called Open Key Exchange) which was claimed to be secure against the e-residue attack.

Later, Mackenzie et al. [3] found that the OKE protocol is still subject to the e-residue attack. So in [4], Mackenzie et al. proposed an RSA-based password-authenticated key exchange protocol, called SNAPI and provided a formal security proof in the random oracle model. The SNAPI protocol makes compulsory that the public exponent e be a prime number larger than the RSA modulus n. This ensures that e is relatively prime to phi_(n).

To avoid using this large public exponent e, Zhu et al. [23] proposed an "interactive" protocol which is revised from an idea of [4]. In the interactive protocol, sender sends to receiver a number of messages encrypted using receiver's public key.

Further in [24], Wong et al. revised the interactive protocol to reduce the message size involved in the interactive protocol.

A drawback of the interactive protocols is the large communication overhead involved in the verification of RSA public key. To win the drawback of interactive protocol, Muxiang Zhang proposed RSA-based password-authenticated key exchange protocols that can use both large and small primes as RSA public exponent, and it does not require large communication overhead on communicating parties. For this purpose, a new protocol was proposed for password-authenticated key exchange based on RSA. The new protocol is called PEKEP, which involves two entities sharing a short password and receiver possesses a pair of RSA keys, n, e and d, where $ed \equiv 1 \pmod{phi\_(n)}$.

Unlike the protocol SNAPI, however, the new protocol PEKEP allows receiver to select both large and small primes for the RSA public exponent e. In the protocol PEKEP, sender does not need to verify if e is relatively prime to phi_(n), and furthermore, sender does not have to test the primality of a large public exponent selected by receiver. Thus, the protocol PEKEP improves on SNAPI by reducing the cost of primality test of RSA public exponents. The protocol PEKEP also improves on the interactive protocol by reducing the size of messages communicated between the entities. To further reduce the computational load on entities, same writers proposed computationally efficient key exchange protocol (called CEKEP). In 1999, Seo and Sweeney proposed a simple authenticated key agreement protocol in which two users share a common password *P* before the protocol execution starts and uses the same public values of *g* and *n* as the original Diffie-Hellman [19]. It is the slight modification of Diffie-Hellman. It also protects the man in middle attack. 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*). It incorporates the authentication based on pre-shared password. After the scheme of Seo and Sweeney, a lot of works has been done to improve the scheme. In 2005, Tseng stated that Seo and Sweeney's scheme also suffers from replay attack and attacker can successfully make an honest party compute a wrong session key. Tseng also proposed an improved scheme to remedy this vulnerability [4]. Later, Ku and Wang also showed that Tseng's scheme is also vulnerable to the backward replay attack and the modification attack, and further he proposed a modification to eliminate these attacks. However, Hsu et al. in 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 [7]. Then, Lee and Lee 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. Again, Lee et al. in 2005 argued that Lee and Lee's scheme is also vulnerable to a password guessing attack and proposed an improved scheme. In 2005, Kwon, Hwang, Kim, Lee showed that Lee et al.'s scheme (denoted by LKY) is still vulnerable to a password guessing attack [9].

In 2005, Laih, Ding and Huang proposed a password- based key establishment protocol in which a user and a server authenticate each other and negotiate a session key. It uses a special function which is a combination of picture function and distortion function to protect the protocol from offline dictionary attack. It was followed by Enhanced LDH protocol, proposed by Qiang, Tang and Chris J. Mitchell in 2005. It includes the identities of participants.

After a lot of research works in one factor authentication protocols, researchers felt that is no more secure than two-factor authentication protocols. So researchers gave more focus on two factor authentication protocols. Park and Park First [18] proposed a two factor authenticated key exchange (PP-TAKE) protocol with two factors including a password and a token (e.g., a smart card with a stored secret key) to make system more secure. This scheme provides mutual authentication.

## 3.2 Key Establishment Protocols

### 3.2.1 LDH Protocol

***Start***

***Client***

*Generate_random(t), $t \varepsilon Z$*

*SendtoServer($ID_U$,t)*

***Server***

*Generate_random(s)*
*Select_string(r)*
*Compute C1 = $E_{pw}$ (φ(r, s)) and $C_2$ = h (pw||r||t)*
*SendtoClient ($C_1$ & $C_2$)*
***Client***
*Receive ($C_1$ & $C_2$)*
*D= Decrypt ($C_1$)*
*Recover r' from D*
*Check if C2=h(pw||r'||t)*
*Then compute $C_3$= (1||pw||r'||t)*
*SendtoServer($C_3$)*
*else*
*terminate protocol*
***Server***

*Receive ($C_3$)*
*Check if $C_3$ = h (1||pw||r||t)*
*Then confirm U as a valid user.*
*Else terminate protocol.*

*If the protocol successfully ends, S and U compute their session key as h (2||pw||r||t).*
***End***

#### 3.2.2. Enhanced LDH Protocol

***Start***
***Client***
*Generate_random ($t_1$), $t_1 \varepsilon Z^*_q$*
*Compute $m_1 = g^{t1} \bmod p$*
*SendtoServer($m_1$)*
***Server***
*Receive ($m_1$)*
*Generate_random($t_2$), $t_2 \varepsilon Z^*_q$ and r is a string.*
*Compute $m_2 = g^{t2} \bmod p$*
*Send($m_2$)*
***Client***
*Receive ($m_2$)*
*Recover(r) from φ(r)*
*Compute $C_1$ = h (φ(r) ||r||3||$m_1$||$m_2$||$g^{t1t2}$ ||g||$ID_U$||$ID_S$)*
*SendtoServer($C_1$)*
***Server***

*Receive ($C_1$)*
*Check if C1=h(φ(r)||r||3||$m_1$||$m_2$||$g^{t1t2}$||g||$ID_U$||$ID_S$)*
*Compute $C_2$=h(4||$m_1$||$m_2$||$g^{t1t2}$ ||g||$ID_U$||$ID_S$)*
*SendtoClient($C_2$)*
*else*
*Terminate the protocol*
***Client***
*Receive ($C_2$)*
*Checkif*
*$C_2$ = h (4||$m_1$||$m_2$||$g^{t1t2}$ ||g||$ID_U$||$ID_S$)*
*U confirms that the protocol execution has successfully ended*
*Else*
*Terminate Protocol*

*If protocol successfully ends U and S compute z = $g^{t2t1}$ mod p as the shared key material, and computes K = h (z||1) as the shared key.*

*End.*

#### *3.2.3 PP-TAKE Protocol*

***Begin***
***Client***
*π,t*
*Pre_Compute x ε $Z_q$, c = $g^{xb}$ in advance*
*Compute h ($ID_A$, $g^b$)*
*Sendtoserver h ($ID_A$, $g^b$)*
***server***
*π,t,b*
*Receive (h ($ID_A$, $g^b$)*
*Obtain_realdentity ($ID_A$)*
*If Identity found Select (r), r ε Zq SendtoClient (r)*
***Client***
*Receive(r)*
*Compute f = h (π, t, r) and e = $E_f$ ($g^x$).*
*$sk_A$ = h(c, $g^x$, r, $ID_A$) ,$M_A$ = h ($sk_A$, π, t, $g^b$)*
*SendtoServer(e & $M_A$)*
***Server***
*Receive (e & $M_A$)*
*f = h(π, t, r) and $g^x$ = $D_f$(e), c = $g^{xb}$, $sk_B$ = h(c, $g^x$ , r,$ID_A$)*
*Check If $M_A$ = h($sk_B$, π, t, $g^b$)*
*Then computes $M_B$ = h ($sk_B$, π, t, $ID_A$)*
*Sendtoclients($M_B$)*
*else terminate protocol*
***client***
*Receive ($M_B$)*
*Checkif $M_B$ = h ($sk_A$, π, t, $ID_A$)*
*Then B is the valid server and B's authentication is successful*
*End*

## 3.3 Experimentation and Analysis

### 3.3.1 Execution time analysis

The results shown below were acquired on machines with Microsoft Windows 7 operating system, Pentium 2 GHz processor and 3 GB RAM. The table 1 shows the average time needed for successful completion of each protocol. All time measurements are in seconds. To find more accuracy, 20 readings of each protocol execution have been recorded and average value is taken as result. The result shows that the PP-take protocol is 12.62% better than LDH and 88.49% better than enhance LDH. The prime numbers selected are of 512 bits. The graph in figure 1 shows the execution time comparison of three protocols for secure communication.

**Table 1: Execution time comparison in seconds**

| S.N | LDH | Enhanced LDH | PP-Take |
|---|---|---|---|
| 1 | 11 | 9 | 9 |
| 2 | 4 | 52 | 4 |
| 3 | 5 | 51 | 4 |
| 4 | 4 | 13 | 4 |
| 5 | 7 | 89 | 4 |
| 6 | 9 | 48 | 4 |
| 7 | 5 | 12 | 4 |

| | | | |
|---|---|---|---|
| 8 | 5 | 83 | 4 |
| 9 | 5 | 16 | 4 |
| 10 | 4 | 1 | 4 |
| 11 | 4 | 15 | 4 |
| 12 | 4 | 17 | 6 |
| 13 | 5 | 112 | 5 |
| 14 | 5 | 22 | 4 |
| 15 | 4 | 9 | 4 |
| 16 | 5 | 7 | 4 |
| 17 | 4 | 27 | 5 |
| 18 | 4 | 89 | 5 |
| 19 | 5 | 53 | 7 |
| 20 | 4 | 27 | 4 |
| **Average time** | **5.15** | **39.1** | **4.5** |

### *3.3.2 Security Analysis*

**The security function φ**

In LDH, the protection of password is totally based on the security of the function φ which assumes that human only (no machine) can identify distorted words but it is possible to identify these words with software with high success rate. But there are various methods that can identify with high success rate like Mori and Malik which gave 83 % success rate against ez-gimpy image, Thayananthan developed program achieved 93 % success rate against ez-gimpy [14,22]. So CAPTCHA schemes are no more secure.

**Offline dictionary attack**

Let 'a 'be the number of symbols used to form $B_n$ (set of strings), '$C_{pw}$' is the set of passwords and $|C_{pw}|$ is the size of password set. When a=62 (characters from a-z, A-Z and 0-9) and n=4, the set of passwords $|C_{pw}|$ equals $2^{23.}$

In addition, there are a number of vulnerabilities regarding LDH protocol based on following facts:

1. A human being is able to recognize r from $D_{pw}(\varphi(r , s ))$ means that $D_{pw}(\varphi(r , s ))$ is very different from a random image.
2. If password is not matched then $D_{pw}$ $(\varphi(r, s))$ will resemble a random image. This means that it is possible to determine whether or not a guessed password pw' is correct only by deciding A is a distorted image or random pattern.
3. Software can be developed to distinguish between a random pattern and a distorted image. For example, a compression algorithm that will compress an image but not the random pattern, which will be much simpler than string recognition.
4. Humans repeat some password than others, it means that $|C_{pw}|$ will be less than $2^{23}$.

On the basis of above facts, there is a chance of offline dictionary attack in LDH as follow:

The machine checks all possible passwords. For each guessed password pw', the machine computes $A = D_{pw'}(C1)$. By some means like software, the machine checks whether A is a distorted image or a random pattern. If distorted image is obtained, the password is correct. So this requires only search of $|C_{pw}|$. Suppose that machine takes 1 millisecond to check one value of A, the total time required to check a password space of size $2^{23}$

$= 2^{23}$millisecond

$= 2^{23}/ (10^3 *60*60)$ hrs

$= 2.3$ hrs

So the password can be guessed in only 2.3 hrs, which is reasonable amount of time and it shows that LDH protocol suffers from offline password guessing attack.

These security vulnerabilities arise because the protection of the secrecy of a password is based on the assumption of machine's inability to read distorted words. So it is not appropriate and the CAPTCHA scheme is better to use only to guarantee the presence of human being (not more than that i.e. not to protect the secrecy of passwords. In enhanced LDH, CAPTCHA scheme has been used only to guarantee the participation of human being in protocol execution.

The enhanced LDH is more secure because it is based on intractability of Computational Diffie Hellman Problem which states that given two values of $g^{x1}$ and $g^{x2}$, it is hard to recover $g^{x1x2}$ in reasonable amount of time. So in case of enhance LDH, even an adversary knows the $m1 = g^{t1}$ mod p and $m2 = g^{t2}$ mod p, he cannot find $z = g^{t2t1}$ mod p. For finding m1, an adversary should know either t1 or t2 which is hard due to

DLP problem. Using mod p operations make more secure because result of each mod p operation looks just as an element from {1, 2, 3,…….p-1} and no one can guess what the message more than an element is.

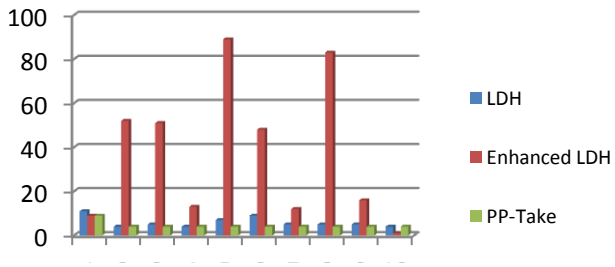


**FIgure 1: The comaprative chart of three protocals**

**Identities of participants**

In case of LDH protocol, no identities of participants have been included in the massages exchanged between them, so client and server cannot assure the presence of intended entity in the communication.

However in case of enhanced LDH protocol, it includes identities of participants in the messages exchanged between the client and the server which results into the mutual authentication. So the client and the server can assure that the message has come from intended user.

## 4. RESULT AND CONCLUSION

From the implementation part and analysis part in this chapter, it has been found that that enhanced LDH protocol is secure for generic systems with comparison to LDH. The time taken by LDH, Enhanced LDH, and PP-TAKE protocol for key establishment were found to be 5.15, 39.1 and 4.5 seconds respectively. So PP-TAKE was found to be efficient for handheld devices because it takes less time for key establishment. PP-TAKE was found 8.68 times faster than enhanced LDH and 1.10 times faster than LDH.

When we transmit secret messages from one user to another over an insecure network, an eavesdropper can capture messages unethically resulting into the breakdown of the security of the system. To keep the cryptosystem secure, messages need to be transmitted in encrypted form that an eavesdropper might not be able to read the messages. For the purpose, some shared session key is required and incorporated it to keep the system secure. However the secure distribution of key is most challenging task. Key establishment protocols have been used as solutions to the key distribution problem. Here in the study, LDH, Enhanced LDH and PP-TAKE protocols were implemented for the purpose. Furthermore, the security of the former two protocols were analyzed based on various security parameters like security function CAPTCHA, factors defining offline dictionary attack. Against the claim of proposers of LDH protocol, this study shows that it still suffers from offline dictionary attack and password can be guessed only in 2.3 hours. Furthermore, enhanced LDH has been found more secure than LDH protocol. Studying these protocols on the basis of execution time, PP-TAKE protocol has been found 8.68 times faster than enhanced LDH and 1.10 times faster than LDH.

## 5. FUTURE WORK

Key management is interesting research field of cryptography and a lot of research work is in progress in key management. Regarding to the thesis topic, the results were based on the security function text CAPTCHA. The study is limited to distortion free, sound free CAPTCHA. So this study may be prolonged for the visual CAPTCHA incorporating distortion and audio CAPTCHA as well. This study is restricted to multiplicative cyclic group, so it may be extended in the group of a point of an elliptic curve. An efficient protocol has been suggested in terms of execution time only, it may be continued to find efficient protocol on the basis of communication time, computation time, occupation of bandwidth (i.e. in terms of number of data units transmitted) etc.

## 6. ACKNOWLEDGEMENT

Authors would like to thanks Dr. Tank Nath Dhamala, Professor of mathematics and former head of department, central department of computer science and information technology and Asst. Prof. Jagadish Bhatt for their supervision during the completion of this work.

## 7. REFERENCES

[1] Stallings William Cryptography and Network Security Principles and practices, Fourth Edition, 2009.

[2] Jablon David P., Strong password-only authenticated key exchange, ACM, 1996.

[3] Zhu F., Wong D., Chan A and Ye R., More efficient password authenticated key exchange based on RSA Springer-Verlag, 2003.

[4] Aloul1 Fadi, Zahidi1 Syed, El-Hajj Wasim, Multifactor Authentication using Mobile Phones , International Journal of Mathematics and Computer Science, 2009.

[5] Bellovin and Merritt, Encrypted Key Exchange: password based protocols secure against dictionary attacks, IEEE, 1992.

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

[7] Wu Shuhua and Zhu Yuefei, Improved Two-Factor Authenticated Key Exchange Protocol, The International Arab Journal of Information Technology, Vol. 8, No. 4, October 2011.

[8] Seo Dong hwi, Sweeney P., Simple authenticated key agreement algorithm, Electronics Letters, 1999.

[9] Bellovin and Merritt, Encrypted Key Exchange: password based protocols secure against dictionary attacks, IEEE, 1992.

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

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

[12] Pilla K.R. Chandrashekhar and Sebastian M. P., An Authenticated Session Key Establishment Protocol for High Security Applications, Canadian Journal on Network & Information Security, Canada, Vol.1, No.2, pp.5-15, March 2010.

[13] Mori G. and Malik J., Recognizing objects in adversarial clutter: Breaking a visual CAPTCHA, IEEE Computer Society, 2003.

[14] Thayananthan A., Stenger B., Torr P., and Cipolla R., Shape context and chamfer matching in cluttered scenes, IEEE Computer Society, 2003.