*IOSR Journal of Computer Engineering (IOSR-JCE)*
*e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 16, Issue 2, Ver. XI (Mar-Apr. 2014), PP 42-47*
*www.iosrjournals.org*

# A Secure and Efficient Two-Server Password Only Authenticated Key Exchange

Ms. Sandra Bhavadas A.B[1], Ms. Jini K.M[2]
[1]*Department of Computer Science and Engineering, Nehru College of Engineering and Research Center, India*
[2]*Department of Computer Science and Engineering, Nehru College of Engineering and Research Center, India*

***Abstract*:** *Password-authenticated key exchange (PAKE) is an authentication mechanism where a client and a server who share a password and authenticate each other with that password and hence both will agree on a cryptographic key. Normally, the passwords which are required to verify the clients are stored on a single server. If the server is compromised, due to some malicious operations like hacking or installing a Trojan horse, passwords which are stored in the server gets revealed. In this paper two servers cooperate to authenticate a client and if one server is cooperated, the attacker still cannot act as a client with the evidence from the conceded server. Current solutions for two servers PAKE are either symmetric in the way that the two server correspondingly contribute to the authentication or asymmetric in the sense that one server confirms the authenticity of legal client with the assistance of another server. This paper presents the development of symmetric protocol for two-server PAKE, where the client can establish different cryptographic keys with the two servers. In addition to that a nonce will be generated during the period of authentication and this will act as a timer. If the timer does not expire with in the period limit, the authentication procedure will be carried out within the limit which provides security to replay attacks.*

***Keywords*:** *PAKE, Dictionary Attack, Diffie-Hellman Key Exchange, Elgamalencryption, Nonce*

## I. Introduction

Password-based user authentication systems are low cost, user friendly and ease of access makes it easy to use among common people. A user only needs to remember a short password and can be genuine anywhere, anytime, regardless of the types of access devices he/she employs.A password is a secret code comprising a word or string of characters for user authentication to prove the identity of an individual or to access resources. Passwords are commonly used by people during a log in process [1] for accessing computer operating systems, mobile phones, and automated teller machines. A computer user may require passwords for many purposes for logging in to computer accounts, retrieving e-mail from servers, accessing programs, databases, networks, and websites. Before few years ago the password based authentication methods transferred a cryptographic hash of the password through a public channel which gives the possibility of hash value available to an attacker. When this is possible, the attacker will work offline, curiously testing possible passwords against the true password hash value. Studies have constantly shown that a large portion of user chosen passwords are readily predicted spontaneously.

Recent advances in the password based authentication have allowed a client and a server jointly to authenticate with a password and for the meantime to establish a cryptographic key for authentication. Password only authentication protocol is both real and provably to be secure under ordinary cryptographic assumptions. The encryption and decryption key pairs for the two servers are generated by the client side and will be delivered to the servers through secure channels. Nonce is a number which is generated only once and will be delivered to the servers during the first step in authentication phase. The nonce will be generated randomly and will not get repeated. The servers will be keeping track of all those nonce which has been previously generated. If suppose the attacker is trying with the same nonce the servers can identify that intruder is working beneath it. An asymmetric two-server PAKE protocol runs in series and only the front end server and the client need to establish a secret session key at the end. Current asymmetric protocols need two servers to exchange messages for several times in series. The asymmetric protocol is not much efficient when compared to the symmetric design which allows two servers to authenticate in series.

However, the use of passwords has several weaknesses. The main problem is that the user chosen passwords are inherently weak since most of them choose short and easy one in order to remember passwords. In particular, passwords are normally drawn from a relatively small dictionary, so it will be vulnerable against brute-force dictionary attacks, where an intruder will tally every possible password in the dictionary to find out the original password. Dictionary attacks can be classified as two types online and offline. The online dictionary attack is where the intruder attempt to log in to a server by trying all passwords from the dictionary until they find a correct one. In an offline dictionary attack, attackers track the record of a past successful login attempt session and then check all the passwords in the dictionary against the login transcript session. .

## II. Related Works

In 2005, Katz *et al.* suggested the first two server password only authenticated key exchange protocol with an evidence of security in the standard model. Their protocol stretched and built upon the Katz- Ostrovsky-Yung PAKE protocol called KOY protocol. In their protocol, a client C randomly chooses a password pwC, and two servers A and B are delivered random password parts pw1 and pw2 subject to pw1+ pw2=pwC. At high level, their protocol can be observed as two implementations of the KOY protocol, one between the client C and the server A, using the server B to support with the confirmation, and one between the client C and the server B, using the server A to assist with the authentication. The assistance of the other server is needed since the password is split between two servers. In the end of their protocol, each server and the client agree on a secret session key. KOY protocol is symmetric where two servers correspondingly contribute to the authentication and key exchange. For their basic protocol secure against a passive adversary, everyone performs roughly twice the amount of works as the KOY protocol. For the protocol secure against active adversaries, the work of the user remains the same but the work of the servers increase by a cause of roughly 2-4. The advantage of KOY protocol is the protocol structure which maintenances two servers to compute in parallel, but its main disadvantage is ineffectiveness for practical use [2].

Yang *et al.* claimed that most password based authentication systems place total expectation on the authentication server where clear text passwords or easily derived password verification data are stored in a common central database. Such systems are by no means resistant against offline dictionary attacks initiated at the server side. Compromise of the authentication server by either outsiders or insiders subject all user passwords to exposure and may have serious legal and financial consequences to an organization. Recently, several multi-server password systems were planned to circumvent the single point of defenselessness natural in the single-server architecture. However, these multi server methods are tough to deploy and operate in practice since either a user has to join simultaneously with multiple servers or the protocols are quite expensive. The system has a number of appealing features. A front-end service server engages straight with users while a control server stays behind the scene. Therefore, it can be directly applied to strengthen surviving single-server password systems.

Yang suggested an asymmetric setting, where a front end server called service server (SS), cooperates with the client, whereas a back end server, called control server (CS), helps SS with the authentication, and only SS and the client agree on a session key at the duration of completion. They suggested a PKI based asymmetric two-server PAKE protocol in 2005 and several asymmetric password-only two-server PAKE protocols in 2006. In their password only protocol the client initiates a request, and SS rejoins with B=B1B2 where B1= g1b1g2π1 and B2= g1b2g2π2 are created by SS and CS on the basis of their random password shares π1 and π2 separately, and then the client can obtain g1(b1+b2) by eradicating the password π= π1+ π2 from B, i.e, calculating B/ g2π. Next, SS and the client authenticate each other by inspecting if they can approve on the same secret session key, either g1a(b1+b2) or g1aa1(b1+b2), with the help of CS, where a, (a1, b1) and b2 are randomly chosen by the client, SS and CS, respectively.

The advantage of Yang *et al.*'s protocols is its effectiveness for practical use. Yang et al.' s protocol are more proficient than KOY protocol in terms of communication and computation complexities, but its disadvantage is the protocol structure which needs two servers to compute in series and desires more communication rounds [3]. Jin et al. further improved Yang et al.'s protocol where a two-server PAKE protocol with less communication rounds. In their protocol, the client refers B= g1ag2π to SS; SS forwards B1=B/ g1b1g2π1 to CS, CS returns A1= g1b2, B2=(B1/ g2π2)b2= g1(a-b1)b2 to SS, SS calculates B3=(B2 A1b1)b3= gab2b3 and responds A2=A1b3, S1=H(b3) to the client where H is a hash function Next, SS and the client authenticate each other by proving if they can agree on the same secret session key gab2b3, where a,(b1; b3) b2 are randomly designated by the client, SS and CS. respectively. The advantage of Jin et al.'s protocol is that it needs less communication rounds than Yang et al.'s protocol in without presenting additional computation complexity. Like Yang et al.'s protocols, the disadvantage of Jin et al.'s protocol is the protocol structure which requires two servers to calculate in series [4].

Joblon [5] detached the condition for PKI and proposed a protocol with the related property in the password-only model. Both the threshold PAKE protocols were not shown to be secure formally. In2002, MacKenzie *etal*.[6]gave a protocol in the PKI-based setting, which necessitates only t out of n servers to collaborate to authenticate a client and is safe as long ast−1orfewer servers are cooperated. They were the first to offer a formal confidence proof for their threshold PAKE protocol in the random oracle model. In 2003, Di Raimondo and Gennaro[7] proposed a protocol in the password-only setting, which needs less than 1/3ofthe servers to be compromised. The security of Yang et al.'s protocol is based on an assumption that the back end server cannot be compromised by an adversary. This assumption was later impassive at the cost of more computation and communication rounds.

Diffie *et al*.[8] concept is based on discrete logarithm problem. Discrete logarithm problem are logarithms defined with regard to multiplicative cyclic groups. If G is a multiplicative cyclic group and g is a

generator of G, then from the explanation of cyclic groups. Every element h in G can be written asgx for some x. The discrete logarithm to the base g of h in the group G is defined to be x. The discrete logarithm problem is defined as: given a group G, a generator g of the group and an element h of G, to find the discrete logarithm to the base g of h in the group G. Discrete logarithm problem is not always hard. The hardness of finding discrete logarithms depends on the groups. The main advantage of this concept is that if the prime is too large, then it is difficult to break. The discrete logarithm problem is defined as a group G, a generator g of the group and an element h of G, to find the discrete logarithm to the base g of h in the group G. Discrete logarithm problem is not always tough. The hardness of finding discrete logarithms depends on the groups. For example, a popular choice of groups for discrete logarithm based crypto systems is $Z_p^*$ where p is a prime number, if p−1 is a product of small primes. $g^x \bmod p=y$, consider x mod p (g=3, p=17) $3^x \bmod 17=1$…….16, $3^x \bmod 17=12$, It is difficult to find the value of x. $3^{29} \bmod 17 \rightarrow 12$ It is easy to compute the value of 12. But, $3^x \bmod 17=12$ it is hard to find out the value of x.

Diffie-Hellman key exchange protocol can be used as follows
1. Alice and Bob settle on a cyclic group 𝔾 of large prime order q with a generator g.
2. Alice randomly picks an integer a from $Ż_q$ and calculates $X=g^a$ while Bob randomly chooses an integer b from $Ż_q$ and computes Y=gb. Hence, Alice and Bob interchange X and Y.
3. Alice computes the secret key $k_1= Y^a=g^{ba}$ while Bob computes the secret key $k_2= X^b = g^{ab}$

It is noticeable that k1= k2 and thus Alice and Bob have settled on the same secret key, by which the succeeding communications among them can be protected. Diffie-Hellman key exchange protocol is safe against any passive adversary, who cannot cooperate with Alice and Bob, endeavoring to define the secret key exclusively built upon experiential data.

The Elgamal encryption scheme [9] was developed by Elgamal in 1985 on the foundation of Diffie-Hellman key exchange procedure. It consists of key generation, encryption, and decryption algorithms. ElGamal encryption arrangement is a probabilistic encryption scheme. If encrypting the similar message with ElGamal encryption scheme numerous times, it will produce diverse ciphertexts.

1. Key generation. On input a security parameter k, it distributes a cyclic group 𝔾 of large prime order q with a generator g. Then it picks a decryption key x arbitrarily from $Ż_q$ and calculates an encryption key $Y=g^x$.
2. Encryption. On inputs a message m€G and the encryption key y, it picks an integer r arbitrarily from $Ż_q$ and yields a ciphertext $C=\varepsilon(m,y)=(A,B)=( g^r,m,y^r)$.
3. Decryption. On inputs a ciphertext (A;B), and the decryption key x, it outputs the plaintext $m=D(C,x)=B/A^r$.

## III. Pake Model

In pake model, there exist two servers S1 and S2 and a group of clients. The two servers work jointly to confirm clients and offer services to genuine clients. Prior to confirmation, each client C chooses a password pwC and produces the password authentication information $Auth^{(1)}_C$ and $Auth^{(2)}_C$ for S1 and S2, respectively, such that nothing can determine the password pwC from $Auth^{(1)}_C$ and $Auth^{(2)}_C$ unless S1 and S2 conspire. The client sends $Auth^{(1)}_C$ and $Auth^{(2)}_C$ to S1 and S2, respective, through diverse secure channels through the client registration. After that only the client recollects the password and the two servers keep the password confirmation evidence. The protocol runs mainly in three stages Initialization, Registration and Authentication.

### *3.1. Initialization*

The two peer servers S1 and S2 together select a cyclic group 𝔾 of large prime order q with a generator g1 and a secure hash function $H : \{0,1\}^* \rightarrow Z^*q$ which maps a message of varying length into an l-bit integer, where $l=\log_2 q$. Next, S1 randomly chooses an integer s1 from Z*q and S2 arbitrarily chooses an integer s2 from Z*q and S1 and S2 swap $g_1^{s1}$ and $g_1^{s2}$. After that, S1 and S2 together publish public system parameters 𝔾 q; g1; g2; H where $g_1=g_1^{s1s2}$. In most of existing two server PAKE protocols it is inferred that the discrete logarithm of g2 to the base g1 is unfamiliar to any person. The initialization can make sure that no one is able to identify the discrete logarithm of g2 to the base g1 except the two servers collude. The discrete logarithm problem is hard, and the model assumes that the two servers not at all conspire.

### *3.2. Registration*

Before authentication, each client C is essential to register to both S1 and S2 through different secure channels. First of all, the client C produces decryption and encryption key pairs (xi,yi) where $yi= g_1^{xi}$ for the server Si (i=1, 2) using the public parameters available by the two servers. Next, the client C picks a password

pwC and encrypts the password using the encryption key yi, i.e., $\varepsilon(g_2^{pw},yi)=(Ai,Bi)= (g_1^{ai}, g_2^{pwC}yi^{ai})$ (i=1,2) where ai is randomly chosen from Z*q according to Elgamal encryption. Then, the client C arbitrarily chooses b1 from Z*q and lets b2=H(pwC)⊕b1, where ⊕stands for exclusive OR of two l-bit blocks. After that, the client C recalls the password pwC. The two secure channels are essential for all two server PAKE protocols, where a password is encrypted by means of two different encryption keys, which are safely broadcasted to the two servers, during registration. Although, the idea of public key cryptosystem, the encryption key of one server should be unfamiliar to another server and the client needs to memorize the secret code or password just behind registration. The two servers S1 and S2 have settled on the password confirmation information of the client C during registration.

### *3.3. Authentication and Key Exchange*

Authentication and key exchange is the key exchange method by which the exchange of session key and thus also authenticate the identities of parties involved in the key exchange. The two servers S1 and S2 have received the password authentication information of a client C during the registration. There are five steps for the two servers S1 and S2 to authenticate the client C and establish secret session keys with the client C in terms of parallel calculation. The two peer servers S1 and S2 equally contribute to the authentication and key exchange. Therefore, the protocol is symmetric.

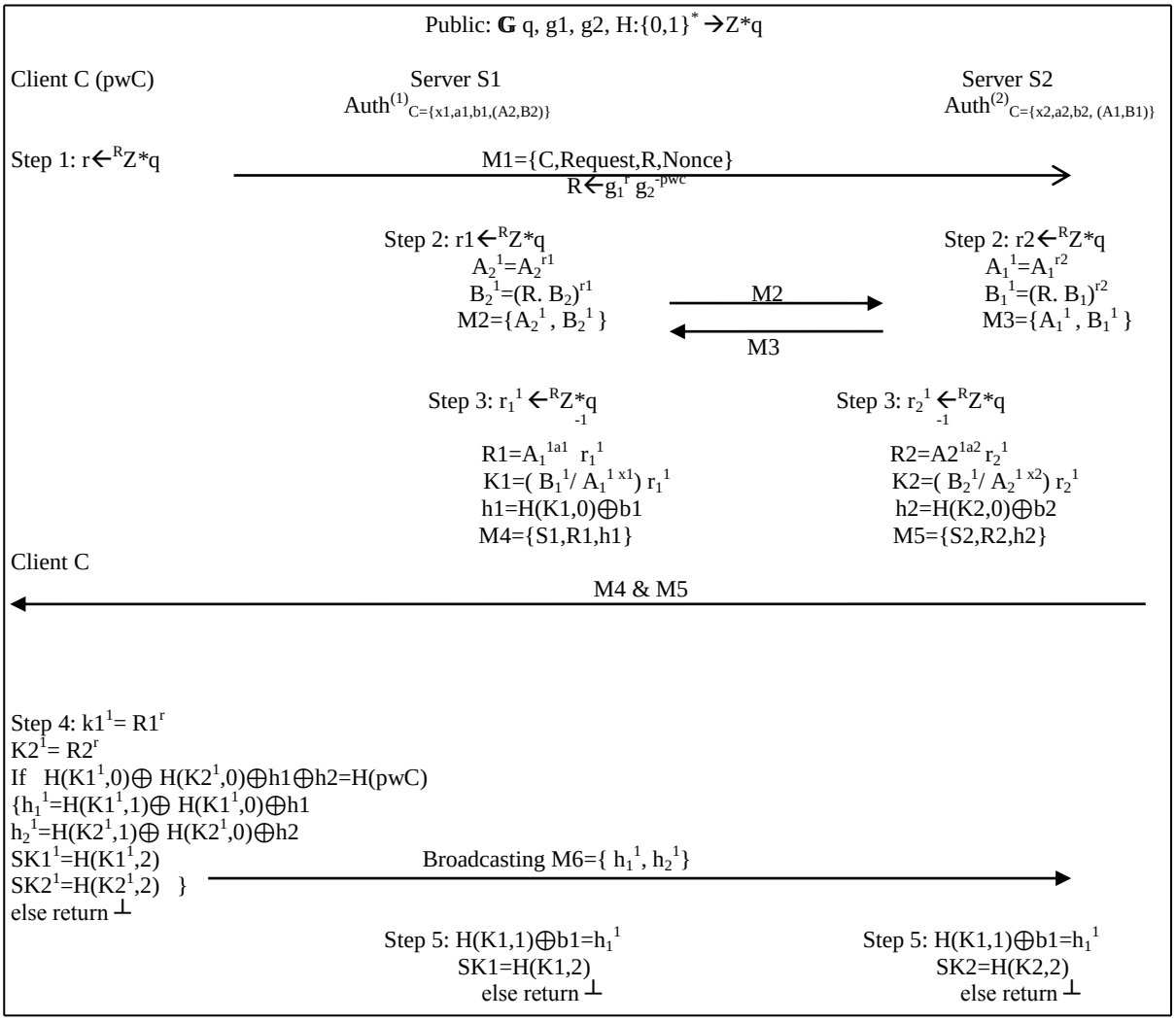


Fig 1.Authentication & Key Exchange of Symmetric Protocol

1. The client C broadcast a request message M1 to the two servers. The message includes the authentication information of the client and a nonce.

2. The two servers exchange messages M2 and M3 based on the authentication information gathered during the registration phase.
3. The servers compute their keys based on the information of messages on step 2. Then the two servers compute the hash of the computed keys and deliver the message M4 and M5 to the client C.
4. On receiving message M4 & M5, The client computes a key. Now, the client compare whether the key matches with the keys of the servers. If it found to be matched, the client confirms that the 2 servers are authentic and sets a secret session key. In addition, the client computes the hash of its computed keys and sends a message in the form of M6.
5. On receiving M6, the server checks whether the hash value computed in M4 & M5 matches with the hash value of client in M6. If it found to be matched, the 2 servers also confirm that the client is authentic and hence sets the secret session key.

In terms of parallel computation, the protocol requires only four communication rounds. The client C broadcasts $M_1$to the two servers $S_1$ and $S_2$ in the first round; $S_1$ and $S_2$ exchange $M_2$and $M_3$ in the second round; $S_1$ and $S_2$both reply to the client C with $M_4$ and $M_5$ in the third round; C broadcasts $M_6$ in the last round. The client C thus participates in three communication rounds. The protocol is efficient in the sense that it requires only 5 communication rounds for authentication and key exchange. The 4 rounds are for the communication between client and server. The remaining round is for the communication between the 2 servers. In addition, the protocol is secure in the sense that the authentication and key exchange must be completed within a limited period. A naive details for two server password only authentication and key exchange can be implemented by running two server password authenticated key exchange(PAKE) sessions among the client and two servers. Towards the end mutually the two servers authenticate to each other as the outcome of the verification process. This result can be constructed with any existing two party PAKE protocol.

### *3.4. Correctness*

If the two servers and the client all follow the protocol is correct, then $SK1^1$= SK1 and $SK2^1$=SK2.
Since $R= g_1^{r} g_2^{-pwC}$,
$A1= g_1^{a1}$,
$B1= g_2^{pwc} y_1^{a1}$
Because $y_{1=g1}^{x1}$,
$A_1^1=A_1^{r2}$, $B_1^1=(R. B_1)^{r2}$
$A_1^1= ( g_1^{a1} )^{r2}= g_1^{a1r2}$,
$B_1^1= ( g_1^{r} g_2^{-pwC} g_2^{pwC} y_1^{a1}) ^{r2}= g_1^{r\,r2} y_1^{a1\,r2}$
$(A_1^1, B_1^1)$ is an Elgamal encryption of $g_1^{r\,r2}$ by the encryption key y1 of the S1
$K1=( B_1^1/ A_1^{1x1})_1^{r\,1}$
$=( g_1^{r\,r2} y_1^{a1\,r2}/ (g_1^{a1r2})^{x1})_{r1}^{\;1}$
$=( g_1^{r\,r2} y_1^{a1\,r2}/ y_1^{a1\,r2}) _{r1}^{\;1}= g_1^{rr\,1\,r2}$.
In addition,

$$R1=A_1^{-1\,a1}\ r_1^{-1\,1}=(A_1^{r2})^{a1}=g_1^{r1r2}.$$

$$K_1^1= R_1^r= g_1^{r1r2r}.$$

Therefore, $K_1^1= K_1$. By the symmetric property, $K_2^1= K_2$, Because

h1=H(K1,0)⊕b1
h2=H(K2,0)⊕b2
$H(K1^1,0)\oplus H(K2^1,0)\oplus h1\oplus h2$
$=H(K1^1,0)\oplus H(K2^1,0)\oplus H(K1,0)\oplus b1\oplus H(K2,0)\oplus b2$
=b1⊕b2
=H(pwC)
In view of this, the client C accepts the messages M4 and M5, broadcasts
$h_1^1=H(K1^1,1)\oplus H(K1^1,0)\oplus h1$
$h_2^1=H(K2^1,1)\oplus H(K2^1,0)\oplus h2$ to two servers S1 and S2, and computes two secret session keys.
$SK1^1=H(K1^1,2)$
$SK2^1= H(K2^1,2)$

## IV. Our Contribution

During the authentication phase, in addition to the request message M1, the client C will add a nonce. The nonce is the number used only once and it can be used as a timer. The timer will expire on every second. The authentication procedure should be completed within the generated period. The advantage is that the nonce which is generated randomly by the client side will have different values. If the attacker is able to capture the message M1, each and every time the same nonce will be trying for authentication, by which the 2 server can identify that an intruder is trying to authenticate as if it is a legal user. When the attempt is being continued, the server will immediately shut down. Thus prevents the replay attacks.

## V. Conclusion

The paper presents a symmetric protocol for two server password only authentication and key exchange. Security analysis has shown that the protocol is secure against passive and active attacks in case that one of the two servers is compromised the intruder cannot find out the password. Performance analysis has shown that the protocol is more efficient than current symmetric and asymmetric two server PAKE protocols in terms of parallel computation. In addition to the efficiency the authentication and key exchange should be completed within a time limit. Hence, the protocol is secure against replay attacks.

## References

[1]. Xun Yi, San Ling, Hauxiong Wang, "Efficient Two-Server Password Only Authenticated Key Exchange", IEEE Transactions on Paralleland Distributed Systems,24,no.9, Sep. 2013.
[2]. J. Katz,P.MacKenzie,G.Taban, andV.Gligor, "Two-ServerPassword-Only Authenticated Key Exchange," Proc. AppliedCryptographyandNetworkSecurity(ACNS'05),pp. 1-16,2005.
[3]. Y.Yang, R.H. Deng, and F.Bao,"A Practical Password-Based Two-Server Authenticationand keyExchangeSystem," IEEE Trans DependableandSecureComputing, vol.3,no.2,pp. 105-114, Apr.2006.
[4]. H.Jin,D.S.Wong, andY.Xu,"AnEfficientPassword-OnlyTwo- Server AuthenticatedKey Exchange System," Proc.Ninth Int'l Conf.Information andComm.Security(ICICS'07),pp. 44-56,2007.
[5]. D. Jablon, "Password AuthenticationUsing Multiple Servers,"Proc.Conf.TopicsinCryptology: TheCryptographer's TrackatRSA (RSA-CT'01),pp. 344-360,2001.
[6]. P.Mackenize,T.Shrimpton,and M.Jakobsson,"Threshold PasswordAuthenticated key Exchange," Proc.22ndAnn. Int'l CryptologyConf.(Crypto'02),pp. 385-400,2002.
[7]. M. Di Raimondoand R.Gennaro,"ProvablySecure ThresholdPassword Authenticated Key Exchange,"Proc.22ndInt'l Conf. TheoryandApplications ofCryptographic Techniques(Eurocrypt'03)pp. 507-523,2003.
[8]. W. DiffieandM.E. Hellman, "New Directions in Cryptography,"IEEETrans.Information Theory,IT-22,no. 6,pp. 644-654, Nov. 1976.
[9]. T.ElGamal, "APublicKeyCryptosystemand aSignatureScheme Based on Discrete Logarithms,"IEEETrans.InformationTheory, vol.IT-31,no.4,pp. 469-472,July1985.