Three-Party Password-Based Authenticated Key Establishment Protocol Resisting Detectable On-Line Attacks

[1]Weijia Wang, [2]Lei Hu
[1*] *School of Science, Beijing Jiaotong University, wangwj@bjtu.edu.cn*
[2] *State Key Laboratory of Information Security,*
Institute of Information Engineering, Chinese Academy of Sciences, hu@is.ac.cn

Abstract

Three-party password-based authenticated key establishment (three-party PAKE) protocols, which enables two clients to authenticate each other and build a session key with the help of an on-line server, has received much attention in recent years. Until now, designing a secure three-party PAKE protocol resisting detectable on-line password guessing attacks is still a challenging problem. To prevent detectable on-line password guessing attacks, ones usually take additional precautions such as logging failed protocol attempts and invalidating the use of the password after a certain number of failures. However, in the three-party PAKE scenario such precautions are high cost, affordable for the server side but not for the lightweight client sides, which may cause security threats on the clients. In this paper, we propose an efficient three-party PAKE protocol, which is able to resist detectable on-line attacks without using additional precautions and whose security is based on both hardness assumptions on the artificial intelligence (AI) problem and the computational Diffie-Hellman problem.

Keywords: *Authenticated Key Establishment, Password Guessing Attack, Artificial Intelligence Problem.*

1. Introduction

Three-party password-based authenticated key establishment (three-party PAKE, or 3PAKE) protocols assume a scenario in which two communication parties want to establish a shared session key through the cooperation of a trusted server, with which each of the two parties shares a predetermined human-memorable short key or password. This kind of protocols is well-suited for applications that communication parties are human beings who are equipped with light-weight or mobile client machines which cannot afford a heavyweight infrastructure such as public key infrastructure and shared secrets with other parties. In the three-party PAKE settings, the trusted server is assumed to be on-line during the execution of key establishment protocols.

Related work. The first three-party password-based key establishment protocol was proposed by Gong, Lomas, Needham and Saltzer [5], called the GLNS protocol. Since then, password-based protocols for three-party settings have received much attention. Tsudik and Herreweghen [15], and later Gong [4] proposed simplification and improvement variants of the GLNS protocol. However, Ding and Horster [3] soon found that there exists an undetectable on-line attack on these versions. Also, it is not always guaranteed in these protocols that the privacy of two communication parties is not leaked to the server. To overcome this drawback, Steiner et al. [12] presented a three-party password-based encrypted key exchange (Three-party PEKE, or 3PEKE) protocol in which the server does not generate and cannot obtain the session key of the two clients. But the protocol is not resistant to off-line and undetectable on-line password guessing attacks [3], [9]. Subsequently, Lin *et al.* [10] proposed a protocol called LSSH-3PEKE which is resistant to off-line and undetectable on-line password guessing attacks. In 2005, Abdalla *et al.* [1] presented a generic construction of three-party PAKE protocols, initializing the formal treatments on the security of three-party PAKE protocols. Following this work, researchers introduced several new three-party PAKE protocols [8] [14] [17] [18] [19] [20] [21], focusing on the building of the model of the adversary and the formal proof of the protocol security. However, all protocols and constructions cited above still suffer from detectable on-line attacks, especially on clients. On the other hand, since von Ahn *et al.* [16] in 2003 introduced hard artificial intelligence (AI) problems as security primitives and provided several novel constructions of CAPTCHA (completely automated public Turing test to tell computers and humans apart), the related techniques on preventing on-line attacks [2] [6] have developed very quickly, which naturally influences the password-based authentication field. Pinkas and Sander [11] utilized reverse turning tests as additional precautions to prevent on-line attacks. Laih *et al.* [7] designed a two-party password-based authenticated key establishment protocol by using a CAPTCHA scheme [16]. But Tang and Mitchell [13] quickly showed that only depending on a certain hard AI problem cannot keep the protocol safe, the protocol of Tang and Mitchell still suffering from off-line dictionary attacks. At the same time, many Internet companies such as Yahoo and Microsoft also used CAPTCHA schemes to prevent free accounts from being registered by machine alone.

Our contribution. As mentioned above, a number of three-party PAKE protocols have been proposed. However, none of these adequately captures an important characteristic of three-party setting that the two clients are humans who remember a weak (low-entropy) password shared with the trusted server, and operate the client machines participating in the three-party scenario. In fact, exerting human being's special abilities different from computers will benefit the efficiency and the security of three-party PAKE protocols.

In this paper, we present the first three-party password-based authenticated key establishment protocol, WH-3PAKE, whose security is based on both difficulties of the computational Diffie-Hellman problem and a hard artificial intelligence problem. The string recognition such as the CAPTCHA scheme used in Yahoo is an hard AI problem. The advantages of this protocol are as follows:

1. Detectable on-line password guessing attacks are still dangerous, especially on clients. WH-3PAKE guarantees that humans are actually involved in the execution of the protocol. Humans' participation can effectively resist continual detectable and undetectable on-line attacks on the clients as well as the trusted server.
2. Since partial authentications in the client side are shared by human being participators, the complexities of clients computing and communicating are reduced largely. So, this protocol is essentially well-suited for light-weight or mobile clients.
3. In WH-3PAKE, we combine hard artificial intelligence problem and computational Diffie-Hellman problem together to provide resistance to all known attacks and to provide perfect forward secrecy.
4. With a small modification, WH-3PAKE can be changed into a three-party EKE protocol with same communication steps.

2. Preliminaries

2.1. Attack in the Three-Party Scenario

- Communication circumstance. The parties interact in a network in which an active adversary has full control over the communication links. This means that the parties cannot communicate directly with each other; rather all communication is carried out via the adversary.
- Insider attacks. A main difference between the security in three-party scenario and the two-party case is the existence of insider attacks. An insider adversary who plays the role of one of the involved client parties and tries to obtain the secret of the other party is more dangerous than an outsider adversary. An insider adversary can do any what an outsider adversary can attack.
- Password guessing attacks. We can divide password guessing attacks into three categories:
    1. Detectable on-line password guessing attacks: An attacker, who may be one party of the two clients and can also masquerade as the trusted server, attempts to use a guessed password in an on-line transaction. Using the response from the honest client or the server, he verifies the correctness of his guess. A failed guess can be detected by the honest client or the server.
    2. Undetectable on-line password guessing attacks: Similarly, an attacker tries to verify a password guess in an on-line transaction. However, a failed guess cannot be detected by the honest client or the server, since one of them is not able to distinguish a malicious request from an honest one.
    3. Off-line password guessing attacks: Only by using the eavesdropped information, an attacker guesses a password and verifies his guess off-line. No participation of the honest client or the server is required, so they does not notice the existence of the attack.

Among the three classes of attacks, three-party protocol designers always pay more attentions to how to resist off-line password guessing attacks and undetectable on-line password guessing attacks and consider that detectable on-line password guessing attacks cannot be avoided in the three-party scenario and additional precautions, such as delaying response or introducing exponentially increasing delays after failed attempts and locking the password after an excessive amount of failures, are needed. However, we think additional precautions are very costly and may introduce additional risks such as denial of service attacks. In the following sections, we will show that the precautions can be handled appropriately by our method of combining hard AI problems and computational Diffie-Hellman primitives.

- Perfect forward secrecy. A protocol is called perfect forward secure if a compromise of a password does not compromise any session key obtained in the before execution of the protocol.

2.2. Hard AI Problem and CAPTCHA

von Ahn*et al.* [16] suggested the use of hard AI problems as security primitives. Now we restate the definitions of AI problems and hard AI problems.

An AI problem is a triple $\mathcal{P} = (S; D; f)$, where S is a set of problem instances, D is a probability distribution over the problem set S, and $f: S \to \{0,1\}^*$ answers the instances. Let $\delta \in (0,1]$. We require that for an $\alpha > 0$ fraction of the humans H, $Pr_{x \leftarrow D}[H(x) = f(x)] > \delta$. An AI problem $\mathcal{P}$ is said to be (δ, τ)-solved if there exists a pro- gram $\mathcal{A}$, running in time at most τ on any input from S, such that $Pr_{x \leftarrow D, r}[\mathcal{A}_r(x) = f(x)] \geq \delta$. ($\mathcal{A}$ is said to be a (δ, τ) solution to $\mathcal{P}$.) $\mathcal{P}$ is said to be a (δ, τ)-hard AI problem if there is no current program being a (δ, τ) solution to $\mathcal{P}$, and the AI community agrees it is hard to find such a solution.

Based on the above mentioned hard AI problems, A CAPTCHA is a program that can generate and grade tests that most humans can pass but current computer programs can not do. In the next section, we introduce a CAPTCHA, which asks users to read a distorted string, into our three-party protocol.

3. The Protocol

In the section, we describe the WH-3PAKE protocol, which is efficient and can resist on-line and off-line attacks effectively.

3.1 Computational Assumption and Notations

- **A** and **B** are two persons equipped with light-weight or mobile client machines, each of them shares a human-memorable password with the trusted server **S**.
- p and q are two sufficiently large primes such that $q|p-1$ and the computational Diffie-Hellman assumption on the subgroup of Z_p^* of order q holds. The assumption says that given g, g^a, and g^b, g^{ab} cannot be computed out by any probabilistic polynomial-time algorithm, where g is a generator element of the subgroup and a and b take their values randomly in the range $[0, p-1]$.
- Powers of g are calculated in the subgroup, i.e., operating exponentiations modulo p.
- $E_K(M)$ is the encryption of M using a symmetric encryption scheme with a key K. $E_{pw_u}(M)$ is the encryption of M using the symmetric encryption scheme with the key derived from the password pw_u of $\boldsymbol{U}$.
- $H(M)$ is the hash value of M under a one-way hash function.
- $\varphi(r, t)$ is a distorted picture function and a specific CAPTCHA scheme, where $r \in \Omega_n$, Ω is the set of all 52 upper-case and lower-case characters and 10 digits, Ω_n is the set of all strings of symbols in Ω of length n, and t is a random integer to generate a random distorted picture of r such that people can recognize r from the picture but machines cannot. $\varphi(r, t_1)$ and $\varphi(r, t_2)$ are different to machines due to different values of t_1 and t_2, but to humans they are the same string. To avoid an off-line machine attack on session keys, we choose the length of the string r as $n = 14$. So, the size of Ω_n is 62^{14} and $|\Omega_n| > 2^{80}$.

3.2 Description of the Protocol

In the initialization phase of the protocol, the trusted party $\boldsymbol{S}$ generates and publishes the public information such as generator $g \in G$, symmetric encryption schemes and hash function, and etc. Execution of the protocol is as follow (see also Figure 1):

1. **A** chooses a random integer x from Z_p^*, computes $M_1 = E_{pw_a}(g^x)$ and sends M_1 to **B** along with identities of clients.
2. Upon receiving the message M_1 from **A**, **B** chooses a random integers y from Z_p^*, computes $M_2 = E_{pw_b}(g^y)$, and sends M_1 and M_2 to **S** along with identities of clients.
3. **S** obtains g^x and g^y by decrypting $E_{pw_a}(g^x)$ and $E_{pw_b}(g^y)$, chooses $s_1, s_2 \in_R Z_p^*$, and computes $K_{AS} = g^{xs_1}$ and $K_{BS} = g^{ys_2}$. **S** also selects a string $r \in_R \Omega_n$ and two random numbers t_1 and t_2. Next **S** computes $M_3 = E_{K_{BS}}(\varphi(r, t_1))$, $M_4 = E_{pw_b}(g^{s_2})$, $M_5 = E_{K_{AS}}(\varphi(r, t_2))$ and $M_6 = E_{pw_a}(g^{s_1})$. **S** finally return M_3, M_4, M_5 and M_6 to **B**.
4. **B** decrypts $E_{pw_b}(g^{s_2})$ to find g^{s_2} and computes $g^{s_2 y}$. Next, **B** gets $Pic_1 = \varphi(r, t_1)$ by decrypting $E_{K_{BS}}(\varphi(r, t_1))$ with $g^{s_2 y}$ and then checks Pic_1 to see whether Pic_1 contains a recognizable string r. If doesn't contain, the execution of the protocol terminates. Otherwise **B** computes $M_7 = H(1||r||B||A)$. Finally **B** sends M_5, M_6 and M_7 to **A**.
5. **A** decrypts $E_{pw_a}(g^{s_1})$ to find g^{s_1} and computes $g^{s_1 x}$. **A** also gets $Pic_2 = \varphi(r, t_2)$ by decrypting $E_{K_{AS}}(\varphi(r, t_1))$ with $g^{s_1 x}$. Next **A** checks Pic_2 to see whether Pic_2 contains a recognizable string r and verifies $H(1||r||B||A)$ by using r. If the check or the verification fails, the execution of the protocol terminates. Otherwise **A** computes $M_8 = H(1||r||A||B)$ and the session key $sk = H(2||r||A||B)$. Finally **A** sends M_8 to **B**.
6. After receiving the message M_8 from **A**, **B** firstly verifies the validation of $H(1||r||A||B)$ by r obtained from **A**. If it is true, **B** also computes $sk = H(2||r||A||B)$ as the session key. Otherwise, **A**'s request is rejected.

$\boldsymbol{A}$		$\boldsymbol{B}$		$\boldsymbol{S}$
$x \in_R Z_p^*$ $M_1 = E_{pw_a}(g^x)$				
	$\xrightarrow{M_1, A, B}$			
		$y \in_R Z_p^*$ $M_2 = E_{pw_b}(g^y)$		
			$\xrightarrow{M_1, M_2, A, B}$	
				$r \in_R \Omega_n, t_1, t_2 \in_R N$ $K_{AS} = g^{xs_1}, s_1 \in_R Z_p^*$ $K_{BS} = g^{ys_2}, s_2 \in_R Z_p^*$ $M_3 = E_{K_{BS}}(\varphi(r, t_1))$ $M_4 = E_{pw_b}(g^{s_2})$ $M_5 = E_{K_{AS}}(\varphi(r, t_2))$ $M_6 = E_{pw_a}(g^{s_1})$
			$\xleftarrow{M_3, M_4, M_5, M_6}$	
		$M_7 = H(1\|\|r\|\|B\|\|A)$		

$$\xleftarrow{M_5, M_6, M_7}$$

$M_8 =$
$H(1||r||A||B)$
$sk =$
$H(2||r||A||B)$

$$\xrightarrow{M_8}$$

$sk =$
$H(2||r||A||B)$

Figure 1. The WH-3PAKE protocol

4. Security Analysis

In this section, we present that the WH-3PAKE protocol can prevent all known attacks. Since humans are really involved in the execution of WH-3PAKE, there are two scenarios for adversary, one is that a machine works alone as an adversary, and the other is that a human being and a machine work together as an adversary. It is obvious that we need only to consider the latter scenario.

- Off-line password guessing attacks: The password pw_a of **A** is used only in M_1 and M_6 to encrypt g^x and g^{s_1} and to authenticate the status of **A** and **S**. There are no any other verifiable information encrypted by the password. So, to get the Diffie-Hellman one-time key $K_{AS} = g^{xs_1}$ from g^x and g^{s_1} is the only way to verify the decrypting result of $g^{x'}$ and $g^{s_1{}'}$ by using the guessing password. But this is the computational Diffie-Hellman problem, which is generally considered to be infeasible to solve. A similar analysis exists for the password pw_b of **B**. Therefore, WH-3PAKE is immune to off-line password guessing attacks.
- Undetectable on-line password guessing attacks: Since each execution of the protocol requires humans' participation, the protocol resists on-line attack essentially. Suppose φ is omitted from M_3 and M_5 and the two messages are $M_3 = E_{K_{AS}}(r)$ and $M_5 = E_{K_{BS}}(r)$. The insider adversary **B** can masquerade as **A** in a protocol run so that it looks normal to **S**. Specifically the adversary guesses $pw_a{}'$, selects a value for x', computes $M_1{}' = E_{pw_a{}'}(g^{x'})$, and then sends $M_1{}'$ to **S** instead of M_1. On receipt of M_6 from **S**, the adversary can obtain $g^{s_1{}'}$ by decrypting M_6 using the guessing password $pw_a{}'$. Finally, using $g^{s_1{}'x'}$ and $g^{s_2 y}$, the adversary decrypts both encrypted messages $M_3{}'$ and M_5 and checks if they give the same value for r: if so then the guess for pw_a was highly probably correct. Notice that **S** must accept random values in M_1. Otherwise, an off-line attack is possible. Furthermore, this attack may be extended to an outsider attack by guessing both candidate passwords together. But if φ is used, the comparison between $M_3{}'$ and $M_5{}'$ only performed by the machine is infeasible since the values of $\varphi(r, t_1)$ and $\varphi(r, t_2)$ are always different due to the different values of t_1 and t_2. If a human being participate to recognize them, he need to work on recognition for each guessed pw'. Laih *et al.* [7] estimate that it will take about 3.2 months for a human being and a machine to successfully search the correct password. It is enough to prevent undetectable on-line attack.
- Detectable on-line password guessing attacks: For the same reason as above, even if the adversary masquerades as both the trust server and one of the two clients, it will fail on carrying out detectable on-line password guessing attacks on the other client.
- Perfect forward secrecy: In the protocol, even when the passwords of both **A** and **B** are revealed, the adversary cannot obtain g^{xs_1} (and respectively g^{ys_2}) from $E_{pw_a}(g^x)$ and $E_{pw_a}(g^{s_1})$ ($E_{pw_b}(g^y)$ and $E_{pw_b}(g^{s_2})$, respectively). Under the computational Diffie-Hellman assumption, no information about g^{xs_1} and g^{ys_2} can be found. Hence, the adversary cannot decrypt M_3 or M_5 and get no information about the session key sk .

5. Comparison

In this section, we compare WH-3PAKE with some related protocols. The result of the comparison is listed in Table 1, where item "symmetric cryptographic operation" means a symmetric encryption or decryption, a one-way hash evaluation, or a computation of a message authentication code.

Table 1. Comparisons with Related Protocols

Protocols	Communication steps	Key distribution or agreement	Resistance to attack			Perfect forward secrecy	Computational cost			
			Off-line password guessing attack	Undetectable on-line password guessing attack	Detectable on-line password guessing attack		Modular exponentiation	Asymmetric en(de)-cryption	Rando Number	Symmetric crypto-graphic operation
Optimal GLNS	5	Key distribution	Yes	No	No	Yes	0	4	7	12
LSSH-3PEKE	7	Key agreement	Yes	Yes	No *	Yes	10	0	4	14
WH-3PAKE	5	Key distribution	Yes	Yes	Yes	Yes	8	0	7	16

*Note: In LSSH-3PEKE, the adversary who masquerades as both the trust server and the other client can succeed in applying detectable on-line password guessing attacks to the other client.

Although the optimal GLNS protocol also needs only 5 communication steps, it is weak in resisting both detectable and undetectable on-line password guessing attacks and does not specify the asymmetric en(de)cryption scheme. Comparing with LSSH-3PEKE, it is obvious that the WH-3PAKE protocol is more suitable for lightweight or mobile clients because it provides more securities but requires less communication steps and computational costs, especially on the clients.

WH-3PAKE is essentially a key distribution protocol. By a small modification, it can be changed into a three-party EKE protocol with the same security as WH-3PAKE. See Figure 2 for the modified protocol.

A **B** **S**

$x \in_R Z_p^*$

$M_1 = E_{pw_a}(g^x)$

$\xrightarrow{M_1, A, B}$

$y \in_R Z_p^*$

$M_2 = E_{pw_b}(g^y)$

$\xrightarrow{M_1, M_2, A, B}$

$r \in_R \Omega_n, t_1, t_2 \in_R N$

$K_{AS} = g^{xs_1}, s_1 \in_R Z_p^*$

$K_{BS} = g^{ys_2}, s_2 \in_R Z_p^*$

$M_3 = E_{K_{BS}}(\varphi(r, t_1))$

$M_4 = E_{pw_b}(g^{s_2})$
$M_5 =$
$E_{K_{AS}}(\varphi(r, t_2))$
$M_6 = E_{pw_a}(g^{s_1})$

$\xleftarrow{M_3, M_4, M_5, M_6}$

$z_1 \in_R Z_p^*$
$M_7 =$
$H(1||g^{z_1}||r||B||A)$

$\xleftarrow{M_5, M_6, M_7, g^{z_1}}$

$z_2 \in_R Z_p^*$
$M_8 =$
$H(1||g^{z_2}||r||A||B)$
$sk =$
$H(2||g^{z_1 z_2}||r||A||B)$

$\xrightarrow{M_8, g^{z_2}}$

$sk =$
$H(2||g^{z_1 z_2}||r||A||B)$

Figure 2. The WH-3PEKE protocol

6. Conclusion

Detectable on-line password guessing attacks on three-party protocols are usually considered unavoidable and additional precautions against them are necessary. In this paper, by combining a hard artificial intelligence problem with the computational Diffie-Hellman primitive, we propose the first three-party password-based authenticated establishment protocol resisting detectable on-line password guessing attacks and other known attacks. Benefiting from the participation of human beings, the WH-3PAKE protocol is of less cost on communication and computation, and is especially suitable for lightweight or mobile client settings.

7. Acknowledgment

The work is partially supported by the Fundamental Research Funds for the Central Universities under Grant Nos.2012JBM104.

8. References

[1] Michel Abdalla, Pierre-Alain Fouque, David Pointcheval, “Password-based authenticated key exchange in the three-party setting”, In proceedings of the 8th International Workshop on Theory and Practice in Public Key Cryptography, pp. 65–84, 2005.
[2] Elie Bursztein, Matthieu Martin, John C. Mitchell, “Text-based CAPTCHA strengths and weaknesses”, In Proceedings of the 18th ACM conference on Computer and communications security, pp. 125– 138, 2011.
[3] Yun Ding, Patrick Horster, “Undetectable on-line password guessing attacks”, Operating Systems Review, 29(4): 77–86, 1995.
[4] Li Gong, “ Optimal authentication protocols resistant to password guessing attacks”, In proceedings of the 8th IEEE Computer Security Foundations Workshop , pp. 24–29, 1995.
[5] Li Gong, T. Mark A. Lomas, Roger M. Needham, Jerome H. Saltzer, “Protecting poorly chosen secrets from guessing attacks”, IEEE Journal on Selected Areas in Communications, 11(5):648–656, 1993.
[6] Abishek Kumarasubramanian, Rafail Ostrovsky, Omkant Pandey, Akshay Wadia, “Cryptography using captcha puzzles ”, In Proceedings of the Public-Key Cryptography , pp. 89–106, 2013.

[7] C.S. Laih, L. Ding, Y.M. Huang, " Password-only authenticated key establishment protocol without public key cryptography", Electronics Letters, 41(4):185–186, 2005.
[8] Tian-Fu Lee, Tzonelih Hwang, "Simple password-based three-party authenticated key exchange without server public keys", Information Sciences, Volume 180, pp. 1702-1714, 2010
[9] Chun-Li Lin, Hung-Min Sun, Tzonelih Hwang, "Three-party encrypted key exchange: Attacks and a solution", Operating Systems Review, 34(4):12–20, 2000.
[10] Chun-L Lin, Hung-Min Sun, Michael Steiner, Tzonelih Hwang, "Three-party encrypted key exchange without server public-keys", IEEE Communication Letters, 5(12):497–499, 2001.
[11] Benny Pinkas , Tomas Sander, "Securing passwords against dictionary attacks", In Proceedings of the 9th ACM Conference on Computer and Communications Security , pp. 161–170, 2002.
[12] Michael Steiner, Gene Tsudik, Michael Waidner, "Refinement and extension of encrypted key exchange", Operating Systems Review, 29(3): 22–30, 1995.
[13] Qiang Tang, Chris J.Mitchell, "Enhanced password-based key establishment protocol", In Cryptology ePrint Archive, 2005. http://eprint.iacr.org/2005/141.pdf.
[14] Hao-Chuan Tsai, Chin-Chen Chang, "Provably secure three party encrypted key exchange scheme with explicit authentication", Information Sciences, Volume 238, pp. 242–249, 2013
[15] Gene Tsudik, Els Van Herreweghen, "Some remarks on protecting weak keys and poorly-chosen secrets from guessing attacks", In Symposium on Reliable Distributed Systems, pp. 136–142, 1993.
[16] Luis von Ahn, Manuel Blum, Nicholas J. Hopper, and John Langford, " Captcha: using hard AI problems for security", In EUROCRYPT, pp. 294–311, 2003.
[17] Ang Gao , "Provable Password-Authenticated Key Exchange Protocol against Imposter Attack on Ad Hoc Networks", JDCTA, Vol. 4, No. 8, pp. 150–163, 2010
[18] Weijia Wang , Lei Hu, "Efficient and provably secure generic construction of three-party password-based authenticated key exchange protocols," In Proceedings of INDOCRYPT 2006, LNCS 4329, pp. 118–132, 2006.
[19] Weijia Wang, Lei Hu, Yong Li, "How to construct secure and efficient three-party password-based authenticated key exchange protocols", In proceedings of Information Security and Cryptology, pp. 218–235, 2011
[20] Jianjie Zhao, Dawu Gu," Provably secure three-party password-based authenticated key exchange protocol", Information Sciences, Vol. 184, pp. 310–323 , 2012.
[21] Wang Tao, Chen Wen qing, "Three-Party Strong Password Authenticated Key Exchange Protocols", IJACT, Vol. 3, No. 11, pp. 39–46, 2011