Journal of Information Hiding and Multimedia Signal Processing ©2018 ISSN 2073-4212
Ubiquitous International Volume **9**, Number **3**, May **2018**

A Novel Two-party Password-Authenticated Key Retrieval Scheme

Dan Zhu

School of Foreign Languages
Shenyang Jianzhu University
No.9, HunNan East Street, HunNan District, Shenyang, P.C 110168 China
zhudan413@163.com

Hongfeng Zhu*, Shuai Geng and Yanlin Meng

Software College, Shenyang Normal University
No.253, HuangHe Bei Street, HuangGu District, Shenyang, P.C 110034 - China
1036103490@qq.com; 569072432@qq.com
*Corresponding author: zhuhongfeng1978@163.com

Received October, 2017; revised December, 2017

Abstract. *A Password-authenticated Key Retrieval scheme (PAKR) allows two parties, such as a client and a server, to authenticate each other, and to retrieve a long-term static key in an exchange of messages with at least one other party (say, server) that has a private key associated with a memorable password. In this paper, we analyze the only PAKR (named as PKRS-1) which cannot be withstanded by off-line dictionary attacks. Then, we firstly construct a novel Password-authenticated Key Retrieval scheme based on chaotic maps, which is a high efficient cryptosystem. Compared with the existing schemes, our scheme effectively prevented the dictionary attacks. Finally, we give the formal security proof about our scheme in the random oracle model.*
Keywords: Key retrieval, mutual authentication, password-guessing attack, chaotic maps

1. **Introduction.** The problem of secure storage client long-term static keys can be resolved through a voucher service, which also addresses many of the client's availability limitations. Consider a client that accesses the network from a different location to retrieve his / her static key (for example, for temporary use of PKI (public key infrastructure)). This mode can be supported by the certificate server of the authentication client and then the client can download the static key.

For the authenntication phase in the model, several works [1-4] used the password authentication key exchange (PAKE) protocol, which is one of the most commonly known authentication key exchange (AKE) protocols, providing password authentication and setup time session key to protect the later communication. The PAKE protocols concept was introduced by Bellovin and Merritt in [5], where the client only need to records a short password set by themselves, the corresponding server uses the password or its authentication data to verify the client's password. However, you should be very careful with two major password attacks: online[16] and offline dictionary attacks. An online dictionary attack is performed by an imitating side attacker so that the attacker can filter the possible candidate for the password one by one. On the other hand, offline dictionary

attacks are performed off-line, and the parallel execution of the attacker thoroughly enumerates all possible cryptographic candidates to try to determine the correct dictionary attack. The latter attack is possible because the password is selected from a relatively small dictionary so that a detailed search can be made. While it is possible to prevent online dictionary attacks by taking appropriate countermeasures (such as locking accounts after three or four times wrong), the offline dictionary attacks can not be avoided by this countermeasure.

In order to solve the above problems, As an approach for the roaming model, Ford and Kaliski [6] proposed some protocols (later, named as Password-Authenticated Key Retrieval. Password-authenticated Key Retrieval Scheme, referred to as PKRS. Jablon proposed a PAKR protocol in[7], that uses multiple servers that do not require a secure channel for previous server authentication. In addition, reference [8] another PKRS is based on a unique blind signature [9]. Unlike the PKI [18] passwords that use the PAKE protocol, the static keys retrieved in PKRS [8], [6], [7] are from the client's password and the servers private key. PKRS-1 proposed a solution between the client and the server.

However, in 2017, Shin et al. in [11] proposed PKRS-1 (Clause 10.2 and Annex D.2.2.3.4 of IEEE 1363.2 [10]) based on [6], [7] in detail cannot resist the dictionary attack and gave a security analysis. We read a lot of paper about chaotic maps [12-15], we found chaotic maps is more suitable for practical applications, and this can resist dictionary attack. For which we put forward a novel client-to-server password-authenticated key retrieval scheme based on chaotic maps. In our solution, we use the chaotic map to solve the problem of dictionary attack in the traditional PKRS scheme.

2. **Chebyshev chaotic maps.** Let n be an integer and let x be a variable with the interval $[-1, 1]$. The Chebyshev polynomial [19] $T_n(x) : [-1, 1] \to [-1, 1]$ is defined as $T_n(x) = \cos(ncos^{-1}(x))$. Chebyshev polynomial map $T_n : R \to R$ of degree n is defined using the following recurrent relation:

$$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x), \tag{1}$$

where $n \geq 2$, $T_0(x) = 1$, and $T_1(x) = x$.

The first few Chebyshev polynomials are:

$T_2(x) = 2x^2 - 1$, $T_3(x) = 4x^3 - 3x$, $T_4(x) = 8x^4 - 8x^2 + 1$, ...

One of the most important properties is that Chebyshev polynomials are the so-called semi-group property which establishes that

$$T_r(T_s(x)) = T_{rs}(x). \tag{2}$$

An immediate consequence of this property is that Chebyshev polynomials commute under composition

$$T_r(T_s(x)) = T_s(T_r(x)). \tag{3}$$

In order to enhance the security, Zhang [20] proved that semi-group property holds for Chebyshev polynomials defined on interval (-,+). The enhanced Chebyshev polynomials are used in the proposed protocol:

$$T_n(x) = (2xT_{n-1}(x) - T_{n-2}(x))(\mathrm{mod}N), \tag{4}$$

where $n \geq 2$, $x \in (-\infty, +\infty)$, and N is a large prime number. Obviously,

$$T_{rs}(x) = T_r(T_s(x)) = T_s(T_r(x)). \tag{5}$$

Definition 2.1. *Semi-group property of Chebyshev polynomials:*

$T_{rs}(x) = T_r(T_s(x)) = \cos(r\cos^{-1}(s\cos^{-1}(x))) = \cos(rs\cos^{-1}(x)) = T_s(T_r(x)) = T_{sr}(x)$.

Definition 2.2. *Given x and y, it is intractable to find the integer s, such that $T_s(x) = y$. It is called the Chaotic Maps-Based Discrete Logarithm problem (CMBDLP or CDL).*

Definition 2.3. *Given x, $T_r(x)$ and $T_s(x)$, it is intractable to find $T_{rs}(x)$. It is called the Chaotic Maps-Based Diffie-Hellman problem (CMBDHP or CDH).*

3. PKRS-1.

3.1. **Key Establishment phase.** In this operation, first, client just need to remembers his/her password pw, server not only need to remember pw, but also has its private key u, which $u \in [1, q-1]$ that corresponds to the password pw. During the key establishment operation, then exchange values between client and server. Client retrieves a static key K derived from both the client's password pw and the server's associated private key u.

Step 1. Client first computes a generator value $R \equiv g_a \cdot g_b^{H(pw)}$, then a random secret number s selected by the client, $s \in [1, q-1]$, and according this secret number s,client computes a blinded password value $W_C \equiv R^s$. Client sends the message (C, W_C) to server.

Step 2. After receiving (C, W_C) from client, server checks if W_C is in $[1, p-1]$. Abort if $(W_C)^q \neq 1$. Otherwise, server uses its private key u computes a permuted blinded password value $W_S \equiv (W_C)^u$. Server sends the first message (S, W_S) to client.

Step 3. After receiving (S, W_S) from server, client checks if W_C is in $[1, p-1]$. Abort if $(W_S)^q \neq 1$. Otherwise, client uses the secrect number computes apermuted password value $Z \equiv (W_S)^{\frac{1}{s}}$. Note that $Z \equiv (W_S)^{\frac{1}{s}} \equiv ((W_C)^u)^{\frac{1}{s}} \equiv R^u \equiv (g_a \cdot g_b^{H(pw)})^u$, and then exports the static key $K = KDF(Z, P)$ from Z, where P is the key derivation parameter.

Step 4. The server S sends the key confirmation value V_S to client C. $S \to C : V_S$

Step 5. After receiving V_S, client C checks correctness of the permuted password value Z by verifying if V_S is equal to $H(Z)$. Note that $Z \equiv R^u \equiv (g_a \cdot g_b^{H(pw)})^u$.

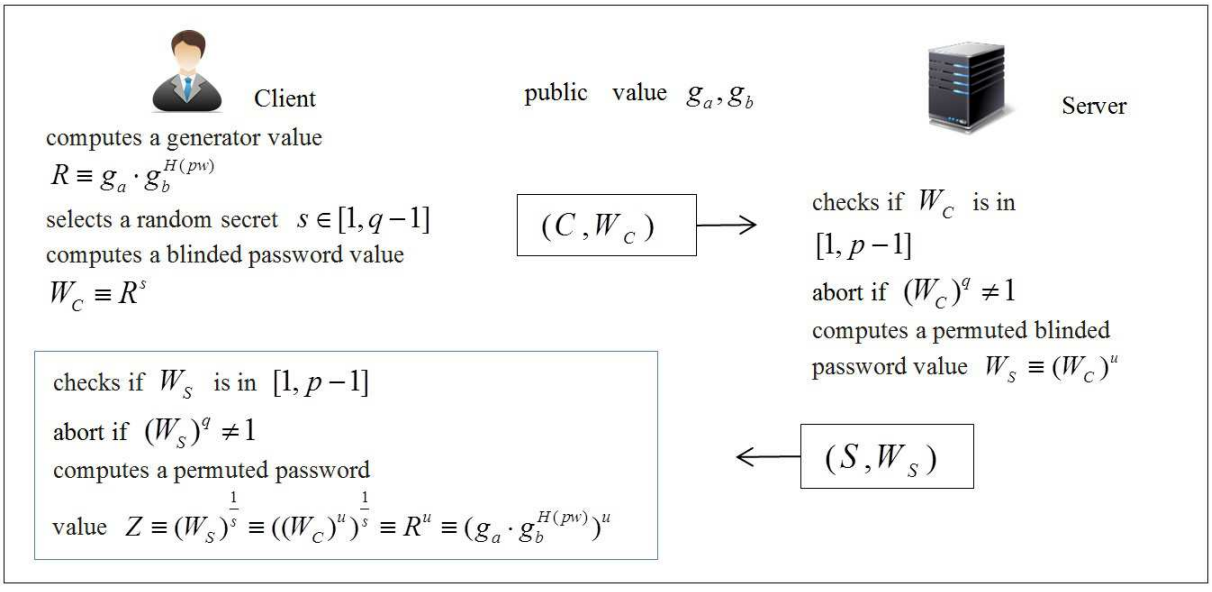

FIGURE 1. Process of PKRS-1

3.2. **An Attack on PKRS-1.** We show an attack on PKRS-1, both client's password pw and the retrieved static key K can be attacked by passive or active attacker use off-line dictionary attacks.

If an attacker A do not know the password pw, so A will impersonates client. The attacker will executes key establishment operation with server.

Step 1'. Attacker A first guess a password pw', then computes a generator value $R' \equiv g_a \cdot g_b^{H(pw')}$. Then a random secret number s selected by the attacker A, $s \in [1, q-1]$, and according this secret number s, client computes a blinded password value $W'_C \equiv (R')^s$. Attacker A sends the message (C, W'_C) to server.

$$A \to S : (C, W'_C)$$

Step 2'. Same as Step 2.

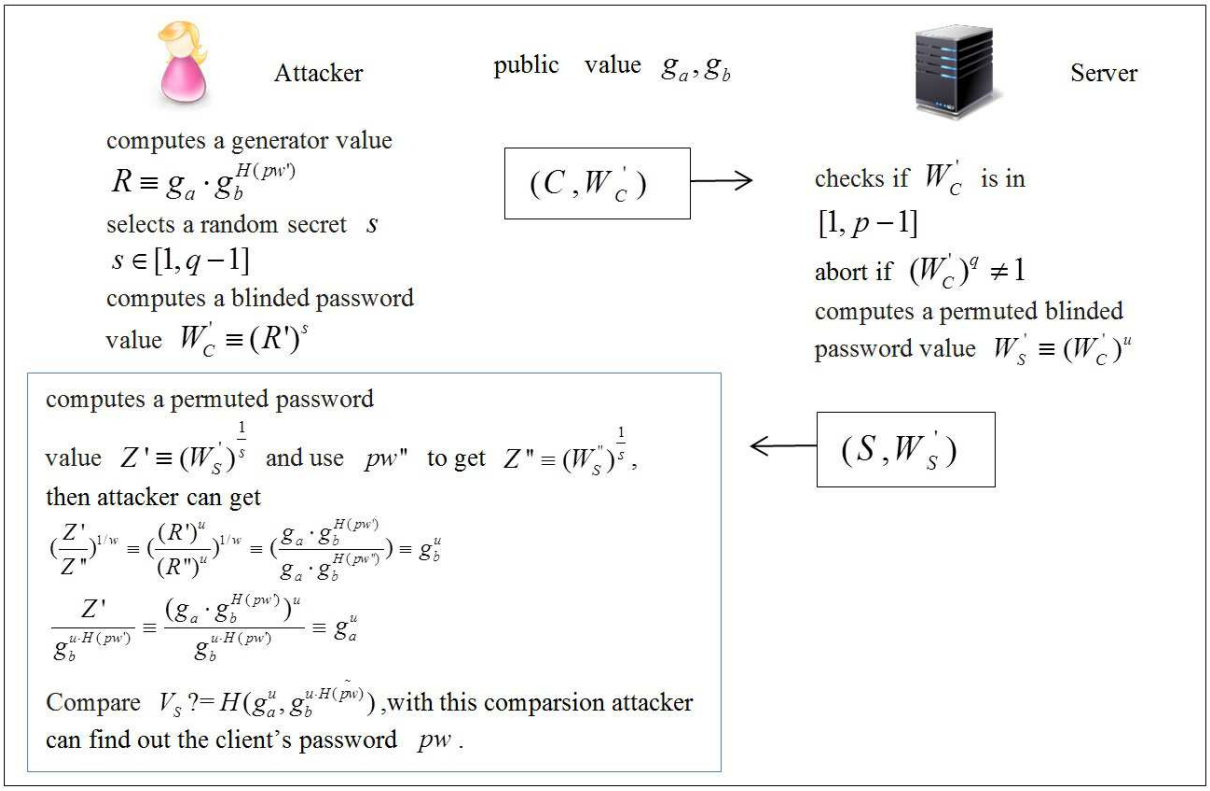

FIGURE 2. security analysis of PKRS-1

Step 3'. After receiving $(S, W'_S \equiv W'_C)^u)$ from server, attacker A uses the secrect number s computes apermuted password value $Z' \equiv (W'_S)^{\frac{1}{s}}$, and then exports the static key $K = KDF(Z, P)$ from Z, where P is the key derivation parameter.

Next the attacker A guess another password value pw'' which is not equal to pw', and repeats the above operation. Then the attacker A can get $Z'' \equiv (W''_S)^{\frac{1}{s}}$ where $W''_S \equiv (W''_C)^u$, $W''_C \equiv (R'')^s$, $R'' \equiv g_a \cdot g_b^{H(pw'')}$.

Let $w \equiv (H(pw') - H(pw'')) \bmod p$. From Z' and Z'', attacker A obtains g_b^u and g_a^u as follows:

$$(\frac{Z'}{Z''})^{1/w} \equiv (\frac{(R')^u}{(R'')^u})^{1/w} \equiv (\frac{g_a \cdot g_b^{H(pw')}}{g_a \cdot g_b^{H(pw'')}}) \equiv g_b^u$$

and

$$\frac{Z'}{g_b^{u \cdot H(pw')}} \equiv \frac{(g_a \cdot g_b^{H(pw')})^u}{g_b^{u \cdot H(pw')}} \equiv g_a^u.$$

After eavesdropping the key confirmation value V_S between client and server, the attacker will do a test to compare if V_S is equal to $H(g_a^u, g_b^{u \cdot H(\widetilde{pw})})$ for all possible password candidates. Through this test, the attacker A can find the client's password from the $pw(= \widetilde{pw})$ to retrieve the static key K is also easy to derive because $K = KDF(Z(\equiv g_a^u, g_b^{u \cdot H(pw)}), P)$.

4. Our Proposed Scheme.

4.1. **Key Establishment phase in our proposed scheme.** In this operation, first, client just need to remembers his/her password pw, server not only need to remember pw, but also has its private key u, which $u \in [1, q-1]$. During the key establishment operation, then exchange values between client and server. Client retrieves a static key K derived from both the client's password pw and the server's associated private key u.

Step 1. Client first computes a generator value $R \equiv T_a T_{H(pw)}(x)$, then a random secret number s selected by the client, $s \in [1, q-1]$, and according this secret number s,client computes a blinded password value $W_C \equiv R^s$. Client sends the message $(T_a(x), C, W_C)$ to server.

Step 2. After receiving $(T_a(x), C, W_C)$ from client, server checks if W_C is in $[1, p-1]$. Abort if $(W_C)^q \neq 1$. Otherwise, server uses its private key u computes a permuted blinded password value $W_S \equiv (W_C)^u$. Server sends the first message (S, W_S) to client.

Step 3. After receiving (S, W_S) from server, client checks if W_C is in $[1, p-1]$. Abort if $(W_S)^q \neq 1$. Otherwise, client uses the secrect number s computes apermuted password value $Z \equiv (W_S)^{\frac{1}{s}}$. Note that $Z \equiv (W_S)^{\frac{1}{s}} \equiv R^u \equiv (T_a T_{H(pw)}(x))^u$, and then exports the static key $K = KDF(Z, P)$ from Z, where P is the key derivation parameter.

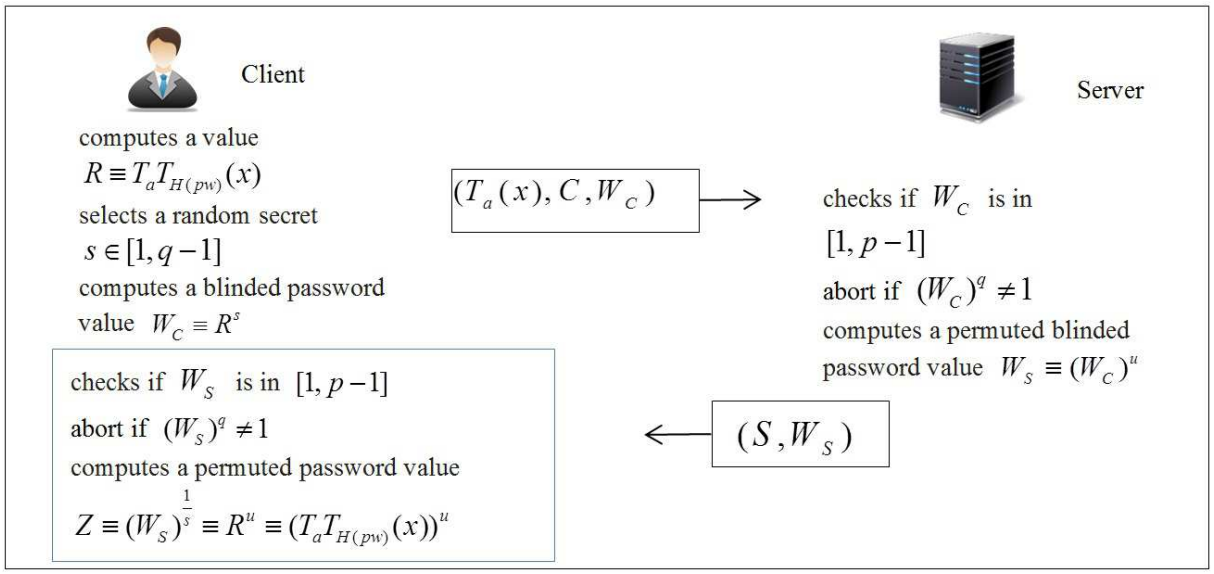

FIGURE 3. Process of our scheme

4.2. **An Attack on our proposed scheme.** We show an attack on PKRS-1, both client's password pw and the retrieved static key K can be attacked by passive or active attacker use off-line dictionary attacks.

If an attacker A do not know the password pw, so A will impersonates client. The attacker will executes key establishment operation with server.

Step 1'. Attacker A first guess a password pw', then computes a value $R' \equiv T_a T_{H(pw')}(x)$. Then a random secret number s selected by the attacker A, $s \in [1, q-1]$, and according this secret number s,client computes a blinded password value $W'_C \equiv (R')^s$. Attacker A sends the message $(T_a(x), C, W'_C)$ to server. $A \to S : (T_a(x), C, W'_C)$

Step 2'. Same as Step 2.

Step 3'. After receiving $(S, W'_S \equiv (W'_C)^u)$ from server, attacker A uses the secrect number s computes apermuted password value $Z' \equiv (W'_S)^{\frac{1}{s}}$. Note that $Z \equiv (W_S)^{\frac{1}{s}} \equiv R^u \equiv (T_a T_{H(pw)}(x))^u$.

Next the attacker A guess another password value pw'' which is not equal to pw' , and repeats the above operation. Then the attacker A can get $Z'' \equiv (W_S'')^{\frac{1}{s}}$ where $W_S'' \equiv (W_C'')^u$, $W_C'' \equiv (R'')^s$, $R'' \equiv T_a T_{H(pw'')}(x)$.

Let $w \equiv (H(pw') - H(pw'')) \bmod p$. From Z' and Z'', attacker A cannot get $T_a^u(x)$ and $T_b^u(x)$ by guess another pw'' , because $T_a T_{H(pw)}(x)$ means $T_a(T_{H(pw)}(x))$, this not a simple mod multiplication relationship, the index can not be eliminated by a simple division. So attacker A cannot attack as above.

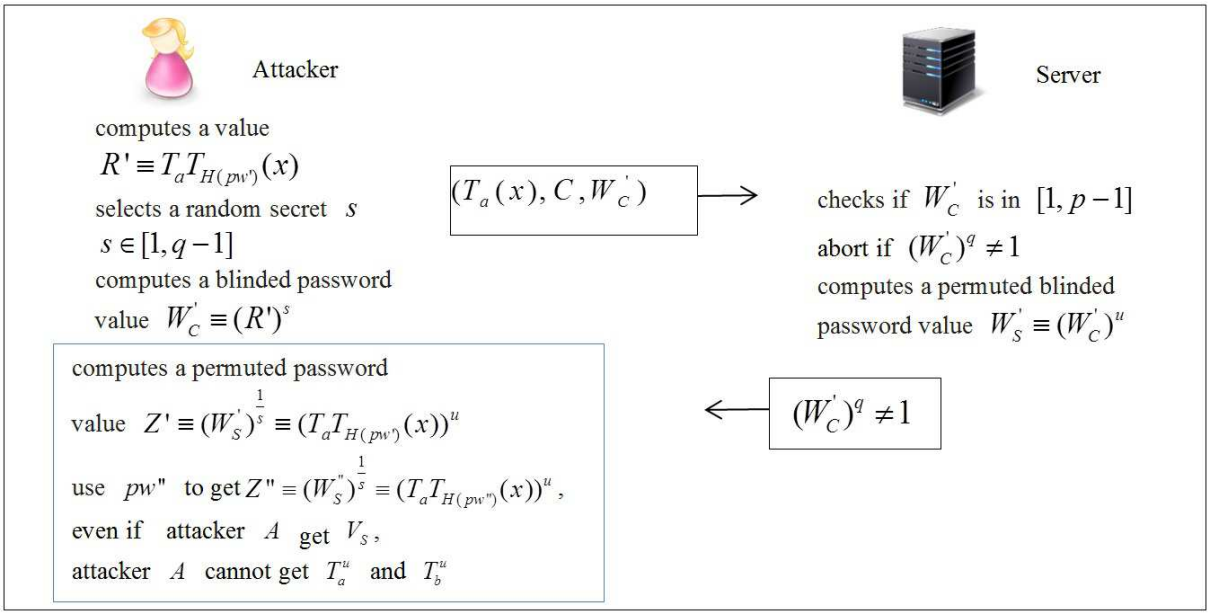

FIGURE 4. security analysis of our scheme

5. Security Analysis.

5.1. Security Model.

The basic descriptions and some queries are shown in Table 1.

5.2. Security Proof. Semantic Security.

Under rational and explicit difficulties, we use the following theorem to prove that the key of the proposed scheme can be safely allocated.

Theorem 1: The parameters in our protocol, we can see in the following, D is the length of the $|D|$ password dictionary. P is the protocol as we can see above. Assuming that the attacker's attack time does not exceed the polynomial time t at runtime, through $SendClient(\Pi_U^i, m)$, $SendServer(\Pi_P^i, m)$, $Execute$, $Reveal$ and $Test$ query, hash random language machine inquiries. Where q_{send}, q_h, q_{exe} is the number of queries, τ_G is the modular exponentiation time we can use these parameters to get the advantage of the attacker A:

$$Adv_P^{ake}(A) \leq \frac{2q_{send}}{|D|} + 2q_h Succ_G^{cdh}(t + (q_{send} + q_{exe} + 1) \cdot \tau_G) + \frac{2q_e^2 + (q_{send} + q_{exe})^2}{(p-1)(q-1)} + \frac{q_h^2}{2^l} + \frac{q_{send}}{2^{l-1}}$$

Proof. We prove this theorem by a series of games from the real agreement, and finally the end of the game with the advantage of attacker A. See Appendix A for details of the proof.

Table 1. Descriptions the model and the queries

Symbol	Definition
Players.	We denote a server S and two client U that can participate in our protocol. Each of them may have several instances, we usually called oracles, involved in distinct, possibly concurrent, executions of P. We denote client instances (resp. server instances) by U_i (resp. S_k), or by Π_P^i refers to ith instance of P when we consider any kind of instance.
Attacker A	A probabilistic Turing machine which controls all communication, with the exception that the adversary cannot inject or modify messages (except for messages from corrupted parties or sessions), and any message may be delivered at most once.
Sessiosns matching	(1) Both Π_U^i and Π_S^j are accepted by the protocol. (2) Both Π_U^i and Π_S^j share the same session identifiers. (3) The partner identifier for Π_U^i is Π_S^j and vice versa. (4) No instance other than Π_U^i and Π_S^j accepts with a partner identifier equal to Π_U^i and Π_S^j. If Π_U^i and Π_S^j in these session are partners mutually, a completed session $\Pi_{U,S}^i$ can be called the matching session.
$Excute(\Pi_U^i, \Pi_S^j)$	The purpose of this query is to simulate passive attacks, where the adversary eavesdrops on honest executions of the protocol among Π_U^i and Π_S^j.
$SendClient(\Pi_U^i, m)$	The purpose of this query is to simulate when A send message to instance Π_U^i, A obtain the message m which the response Π_U^i generates in processing according to the protocol P.In addition, the adversary is allowed to use $SendClient(\Pi_U^i, Start)$ initiate the protocol.
$SendServer(\Pi_P^i, m)$	The purpose of this query is to simulate adversary obtain the message that the server instance Π_P^i would generation receipt of the message m.
$Reveal(\Pi_P^i)$	The purpose of this query is to simulate the misuse of the session key by instance Π_P^i.
$Test(\Pi_{U,S}^i)$	The adversary only can query this form once time, but the adversary A can execute this query at any time. The purpose of this query is to simulate the indistinguishability of the information. To respond to this query by A, a random bit $b \in \{0,1\}$ is selected. If $b=1$, then the information is returned. Otherwise, a uniformly chosen random value is returned. Eventually, A outputs $b' \in \{0,1\}$ as its guess result, according to this result decide what A will get, whether the returned value to the $Test$ query is the information or a random value. If $b=b'$, we say that the adversary A wins the game.
Freshness	Let $\Pi_{U,S}^i$ be a completed session by a party U with some other party S, and $\Pi_{U,S}^j$ be the matching session to $\Pi_{U,S}^i$. We say that the session $\Pi_{U,S}^i$ is fresh if U and S in session $\Pi_{U,S}^i$ and the matching session $\Pi_{U,S}^j$ are honest and the following conditions hold: (a) $\Pi_{U,S}^i$ has accepted the request to establish a session key. (b) $\Pi_{U,S}^i$ has not been revealed. (c) No matching conversation $\Pi_{U,S}^j$ of $\Pi_{U,S}^i$ has been revealed. (d) U, S have not been corrupted. (e) The adversary asks neither $SendClient(\Pi_U^i, m)$ nor $SendClient(\Pi_U^j, m)$ query.

6. **Conclusions.** In this paper, we first proposed a problem of the security of password, and introduced several PAKR scheme. We found the problem that the PAKR scheme cannot resist the dictionary attacks and gave a security analysis, showing that any passive/active attacker can find out the clients password pw and the (long-term) static key K with off-line dictionary attacks. So we have made improvements based on this scheme, put forward a novel client-to-server password-authenticated key retrieval scheme based on chaotic maps, and gave an attack on our proposed scheme to prove an attacker cannot

attack our scheme by the way they used in PKRS-1. Finally, we gave a security analysis to show our scheme is safety.

Acknowledgment. This work is supported by the Liaoning Provincial Natural Science Foundation of China (Grant No. 201602680).

REFERENCES

[1] M. Abdalla, D. Catalano, C. Chevalier, D. Pointcheval, Efficient two-party password-based key exchange protocols in the UC framework, *Lecture Notes in Computer Science*, Springer, Berlin, vol. 4964, pp. 335–351, 2008.

[2] J. Byun, D. Lee, J. Lim, Cryptanalysis of simple three-party key exchange protocol (S-3PAKE), *Information Sciences*, vol. 178, no. 13, pp. 2849–2856, 2008.

[3] R. Phan, W. Yau, B. Goi, A communication-efficient three-party password authenticated key exchange protocol, *Information Sciences*, vol. 181, no. 1, pp. 217–226, 2011.

[4] J. Yang, C. Seo, J. Cho, A three-party authenticated key exchange scheme smartcard using elliptic curve cryptosystem for secure key exchange in wireless sensor network, in: *ISCE*, pp. 1–6, 2007.

[5] S. M. Bellovin and M. Merritt, Augmented encrypted key exchange: A password-based protocol secure against dictionary attacks and password file compromise, in *Proc. 1st ACM Conf. Comput. Commun. Security*, pp. 244–250, 1993.

[6] W. Ford and B. S. Kaliski, Server-assisted generation of a strong secret from a password, in *Proc. 9th IEEE Int.* Workshops Enabling Technol.: Infrastruct. Collaborative Enterprises, pp. 176–180, 2000.

[7] D. P. Jablon, Password authentication using multiple servers, in *Proc. Conf.* Topics Cryptol.: The Cryptographers Track at RSA, pp. pp. 344–360, 2001.

[8] X. Boyen, Hidden credential retrieval from a reusable password, in *Proc.4th Int. Symp. Inf., Comput.*, Commun. Security, pp. 228–238, 2009.

[9] A. Boldyreva, Threshold signatures, multisignatures and blind signatures based on the gap-diffie-hellman-group signature scheme, in Proc. 6th Int. *Workshop Theory Practice Public Key Cryptography*: Public Key Cryptography, pp. 31–46, 2003.

[10] IEEE 1363.2, IEEE Standard Specifications for Password-Based Public-Key Cryptographic Techniques, *IEEE Std* 1363.2TM-2008, IEEE Computer Society, Jan.2009.

[11] S. H. Shin and K. Kobara, Security Analysis of Password-Authenticated Key Retrieval, *IEEE Transactions on dependable and secure computing*, 573-576, 2017.

[12] C. M. Chen, C. T. Li, S. Liu, T. Y. Wu, J. S. Pan, A Provable Secure Private Data Delegation Scheme for Mountaineering Events in Emergency System, *IEEE Access*, pp. 3410-3422, Feb. 2017.

[13] C. M. Chen, W. C. Fang, K. H. Wang, and T. Y. Wu, Comments on An improved secure and efficient password and chaos-based two-party key agreement protocol, *Nonlinear Dynamics*, vol. 87, issue 3, pp. 2073-2075, Feb. 2017.

[14] C. M. Chen, L. L. Xu, T. Y. Wu and C. R. Li, "On the Security of a Chaotic Maps-based Three-Party Authenticated Key Agreement Protocol," *Journal of Network Intelligence*, vol. 1, No 2, 2016.

[15] H. F. Zhu, X. Hao, Y. F. Zhang and M. Jiang, A biometrics-based multi-server key agreement scheme on chaotic maps cryptosystem, *Journal of Information Hiding and Multimedia Signal Processing*, vol. 6, No. 2, pp. 211-224, March 2015.

[16] Y. Kobayashi N. Yanai K. Yoneyama T. Nishide G. Hanaoka, Gateway threshold password-based Authenticated Key Exchange secure against Undetectable On-line Dictionary Attack, *International Joint Conference on E-business & Telecommunications*, 2016 :39-52

[17] S. E. Schechter D. A. Molnar J. R. Lorch B. C. Bond B. J. Parno, Utilization of a protected module to prevent offline dictionary attacks, in *US* , 2016

[18] X. Wang, Intrusion-tolerant password-enabled PKI, in Proc. 2nd Annu.PKI Res. *Workshop*, , pp. 4453, 2003.

[19] X. Wang and J. Zhao, An improved key agreement protocol based on chaos, *Communications in Nonlinear Science and Numerical Simulation*, vol. 15, no. 12, pp. 40524057, Dec. 2010.

[20] L. Zhang, Cryptanalysis of the public key encryption based on multiple chaotic systems, *Chaos, Solitons & Fractals*, vol. 37, no. 3, pp. 669–674, Aug. 2008.

A.Proof of Theorems.

Table 2. *Send*, *Excute*, Re*veal* and *Test* simulation of the query

For a query on $SendClient(\Pi_U^i, Start)$, assume that instance is in the simulated environment, We do the following: select a random number $s \in Z_q^*$, and calculate: $R \equiv T_a T_{H(pw)}(x)$ and $W_C \equiv R^s$, and then query returns $\{T_a(x), C, W_C\}$.
For the query on $Send(T_a(x), C, W_C)$, assuming that instance S is in the simulated environment, we do the following: If the W_C value is invalid, then terminate. If valid, calculate $W_S \equiv (W_C)^u$, if $(W_C)^q \neq 1$, then terminate.
For $Send(S, W_S)$ query, assuming that instance S is in the simulated environment, we do the following: Calculate $Z \equiv (W_S)^{\frac{1}{s}} \equiv R^u \equiv (T_a T_{H(pw)}(x))^u$.
About a query on Re*veal*. If instance P has been accepted, the query will return the key.
About a query on $Test(\Pi_{U,S}^i)$, we will use the query simulator to do the following: Through Re*veal* query to get the key, and then throw a coin b, if $b=1$, return the session key, otherwise, will return a random length of the same array.

$Game_0$: In the random oracle model, a game corresponds to a real attack. $Game_0$ means that the attacker successfully guess $Test$ query in the pre-use of the bit b, you can get:

$$Adv_{P,A}^{ake} = 2pr[S_0] - 1$$

$Game_1$: In this game, we simulate the random prophecy machine h and the encrypted E / decryption D extinction machine, Usually contains a list Λ_h, and an encrypted list Λ_e. We also simulate all instances of the query as well as the participant $SendClient(\Pi_U^i, m)$, $SendServer(\Pi_P^i, m)$, $Excute$, Re*veal* and $Test$. From this simulator, we can easily find this game with the real attack is indistinguishable, in addition to encryption E or decryption D replacement nature does not hold, Thus, the probability difference between $Game_0$ and $Game_1$ is:

$$|\Pr[S_1] - \Pr[S_0]| \leq \frac{q_e^2}{2(p-1)(q-1)}$$

$Game_2$: In this game, we simulate all the oracle in $Game_1$, according to the birthday paradox, the probability of the output of the encryption oracle is $\frac{q_e^2}{2(p-1)(q-1)}$. Hash oracle output has the probability of collision at most $\frac{q_h^2}{2^{i+1}}$. Likewise, the probability of information on the collision is $\frac{(q_{send}+q_{exe})^2}{2(p-1)(q-1)}$, Thus, the probability difference between two games is:

$$|\Pr[S_1] - \Pr[S_0]| \leq \frac{q_e^2}{2(p-1)(q-1)} + \frac{q_h^2}{2^{i+1}} + \frac{(q_{send}+q_{exe})^2}{2(p-1)(q-1)}$$

$Game_3$: In this game, we terminate the attacker successfully guess the key, and its encryption calculation, the encrypted data sent to the server. We do this by modifying the server to perform the inquiry process. We first ask whether $(pw, *, E, Y_i)$ belongs to Λ_e. If the list already exists, the definition is correct and the game is terminated. Likewise, compute $V_{S_1} = A^{x_1}$ and $K_{S_1} = A^{s_1}$. So that we can get $Game_3$ and $Game_2$

in addition to the occurrence of the incident is indistinguishable, so get the probability difference:

$$|\Pr[S_3] - \Pr[S_2]| \leq \Pr[Encrypt_3] = \frac{q_{send}}{|D|}$$

$Game_4$: In this game, we terminate the attacker successfully guess the certification type $auth_i$. That is, $Z' \equiv (W'_S)^{\frac{1}{s}}$, did not by asking the corresponding hash of the message machine will be certified. According to the original agreement, get the accepted certification. There are two cases where the simulator and the attacker successfully decrypted to get a, and inquire about the hash oracle. Therefore, $Game_4$ and $Game_3$ in addition to the attacker did not ask the hash by the whistle machine to guess the certification is indistinguishable. So the probability difference is obtained:

$$|\Pr[S_4] - \Pr[S_3]| \leq \frac{q_{send}}{2^l}$$

$Game_5$: In this game, the simulator defines a private hash oracle h', using the oracle to compute the key so that the value of the key is completely independent of h. Through the query on the simulator *execute*, the return value is $Z \equiv (W_S)^{\frac{1}{s}} \equiv R^u \equiv (T_a T_{H(pw)}(x))^u$. Here we define an event $AskH_5$: the attacker uses the hash function h to calculate $U||h_0^c||S||S^c$ or $U||S||S^c||h_0^{cb}$ of the query, that is, two common values $U||h_0^c||h_0^b||h_0^{cb}$. This also means that the indistinguishability of $Game_5$ and $Game_4$ is the occurrence of the event $AskH_5$ or not. And because only the simulator can access h', and the attacker cannot access, then the attacker *test* query on the value of b is a session with the length of the same length of the random array, the value of the agreement with the session key value is mutually independent. Therefore, the probability difference between $Game_5$ and $Game_4$ is:

$$|\Pr[S_4] - \Pr[S_3]| \leq [AskH_5], \Pr[S_4] = \frac{1}{2}$$

$Game_6$: In this game, we simulate the random execution of the problem from the protocol. Give an example of a CDH knife (A, B). Here we do not need to know the value of ϑ and φ, because we do not need the value it generates to generate the session key. Here we define an event $AskH_6$. An attacker who visits a random oracle h to compute $U||h_0^c||h_0^b||h_0^{cb}$. Through analysis, we can know the probability of occurrence event $AskH_5$ is equal to the probability of occurrence event $AskH_4$, that is $\Pr[AskH_5] \leq \Pr[AskH_6]$. So:

$$|\Pr[AskH_6]| \leq q_h Succ_G^{cdh}(t')$$

In summary:

$$Adv_P^{ake}(A) \leq \frac{2q_{send}}{|D|} + 2q_h Succ_G^{cdh}(t + (q_{send} + q_{exe} + 1) \cdot \tau_G) + \frac{2q_e^2 + (q_{send} + q_{exe})^2}{(p-1)(q-1)} + \frac{q_h^2}{2^l} + \frac{q_{send}}{2^{l-1}}$$