Umudike Journal of Engineering and Technology (UJET); Vol. 8, No. 1, June 2022, pp. 88 – 94; Michael Okpara University of Agriculture, Umudike, Print ISSN: 2536-7404, Electronic ISSN:2545-5257; http://ujetmouau.net; doi: https://doi.org/10.33922/j.ujet_v8i1_8

INVESTIGATING THE METHOD OF AUTHENTICATED KEY EXCHANGE PROTOCOL

Ede C. C.

Department of Computer Engineering, Michael Okpara University of Agriculture, Umudike, Abia State Nigeria.

*Corresponding author: ede.cyril@mouau.edu.ng, cyrisoftnet@gmail.com

ABSTRACT

This research paper focused on investigating the method of authenticated key exchange, a protocol where communicating parties generate and exchange secret session keys for authentication. We examined the Two-Server Password-Only Authenticated Key Exchange by Xun Yi, San Ling, and Huaxiong Wang, a two-server password-only authenticated key exchange protocol. In the protocol, each communicating party generates a shared key such that in the result of their computations, they arrive at the same session key. This analysis was deemed very relevant because failure to arrive at a shared session key is a chief design weakness in any cryptosystem. We employed numerical examples to assist in proving the efficiency of the protocol. In our computations with randomly selected numerical values as suggested by the protocol, it failed to arrive at a common session key. It was discovered that this failure was a result of not considering the congruency of the powers modulo Euler's totient function while selecting parameters at random for the computations. We, therefore, proposed that the parameters whose inverse is involved in the computation should be chosen such that its multiplicative inverse modulo Euler's totient function exists instead of selecting them at random. In another numerical example, we employed this restriction in selecting parameters for computations and it resulted in a more secure and efficient protocol.

***Keywords**: Authenticated key exchange protocol, two-server authentication, secret session keys, password-only authentication*

1. INTRODUCTION

In a communication system, server A communicates with server B such that they do not want a third party, maybe server C to listen in. To ensure server C does not listen in or intercept the content of their communication, they need to communicate securely. The secure way could be achieved either by hiding the content of their communication using encryption or steganography. It could also be realized by hiding the communicating parties (anonymity), or by hiding the fact that communication takes place (security by obscurity). Secure communication ensures that communicating parties establish a shared secret key with which they use to hide the contents of their communication, make themselves anonymous, or obscure their communication.

Password is the most used means to access secure systems such as email servers, computer operating systems, mobile phones, automated teller machines, etc. It does not cost anything for a user to think out a password to enable him or her to access a secure system. This password could be any memorable word or string of characters coined from anything the user can remember easily. However, due to poor remembrance, users choose a password with very low entropy, thus making it susceptible to brute-force dictionary attacks.

In a Password Authenticated Key Exchange (PAKE), the client and server share a password, authenticate each other using the password, and arrive at the same key. As a user inputs his or her password to access a secure system, the hash value of that password transmits through an insecure channel to the server for authentication, thus exposing it to possible adversary activities. The above scenarios are what happens in a typical protocol for a password-based authentication system, where a single server stores the whole password for the client's authentication. This protocol is a weak system because when an adversary compromises the server; the adversary's activities reveal all the stored passwords to the attacker.

Two-server password-based authentication protocol was presented by (Brainard *et al*, 2003), (Yang *et al*, 2005), (Yang *et al*, 2006), (Katz *et al*, 2012) and (Yi *et al*, 2013) to avert the vulnerability issue described above. Two-server password-based authentication is a protocol that allows two servers to collaborate in verifying the identity of a client. Two-server password authentication is designed in a way that does not require a password table on the server side for verification (Ampomah *et al*, 2015). In this two-server architecture, the servers do not need to store or have knowledge of the client's password. The client sends authentication information, based on the chosen password, to the servers. In this system, if the adversary attacks one of the servers, it will not be possible to

fool the other server to be the client. This two-server architecture operates in either asymmetric or symmetric mode. In asymmetric, one server supports the other in the authentication process while, in symmetric, both servers cooperate to authenticate the client.

A practical symmetric solution was offered by Yi *et al* (2013) for a two-server password-only authenticated key exchange. The protocol is such that two servers cooperate to authenticate a client and if one server is down due to the adversary's activity, the adversary cannot fool the other server into being the client. The protocol runs in three phases and some parameters are chosen at random from a list of all invertible elements in a cyclic group of a large prime number. In the first phase, the initialization phase, the two servers choose a cyclic group of large prime *q* with a generator and a secure hash function. These parameters are made public and used in the second phase, the registration phase. During this second phase, the client generates decryption and encryption key pairs and encrypts the chosen password according to the ElGamal (1985) encryption scheme. In the last phase, the authentication and key exchange phase, the parties arrive at the same secret keys at the end of computations.

1.1. Literature Review

The less expensive and mostly used authentication mechanism in security applications is the password. Some authentication mechanisms such as biometrics require additional hardware resources that may be considered too costly for security applications (Anderson, 2001). Due to the low entropy nature of the passwords, they need protection from transmission over insecure channels. The means of protecting these passwords, is by encryption, translating them into unreadable strings such that it makes no sense to any adversary.

In Public Key Infrastructure (PKI), the client shares a password with the server and has the server's public key. The public key of the server is used by the client to encrypt the password, then send it across to the server. The first researchers to present this model are Gong *et al* (1993) and Lomas *et al* (1989). Their protocol concentrated on resisting offline dictionary attacks but lacks security proof for the model. Halevi and Krawczyk (1999) filled this gap, and they became the number one to present thorough proof of security for the setting.

Bellovin and Merritt (1992) proposed the second model. In this model, authentication is based on password-only, and it uses the password to encrypt randomly generated numbers for the goal of key exchange. Their model lacked a security model and Bellare *et al* (2000) and Boyko *et al* (2000) filled this gap. These password-only authenticated protocols were not both practical and secure. Katz *et al* (2001) came up with one that is practical and secure. These protocols assume that a single server stores all the passwords for authentication. For this reason, all the passwords are exposed when an adversary compromises the server. Yi *et al* (2009), Yi *et al* (2011), and Yi *et al* (2012) came up with an identity-based setting relating to the identity-based encryption scheme of Boneh and Franklin (2001) and Boneh and Franklin (2003). In their models, the client only knows the password and the server knows both the password and the private key relating to its identity. The client encrypts the password with the server's identity. This setting is a hybrid of the Public Key Infrastructure (PKI) and password-only model.

In 2013, Yi *et al* proposed a new symmetric two-server password-only authenticated key exchange protocol that enables two-server architecture to compute in parallel. Their protocol claims to be more efficient in practical use than the existing Katz et al (2012) protocol because of its parallelism in computation. The Yi et al (2013) protocol was efficient until it was discovered that choosing parameters at random during computations will at some point lead to communicating parties arriving at different session keys. This failure to arrive at a common session key is caused by not considering the congruency of the exponent's modulo Euler's totient function when choosing parameters during computations. This failure is the reason for this research and the solution we proposed will modify the way parameters are chosen during computations to arrive at a common session key at all points.

To be able to reveal the problem with the Yi *et al* (2013) protocol, we have reviewed its three phases – initialization, registration, and authentication. In each of these phases, the parties perform some computations leading ultimately to establishing the same secret keys. Computations are not explicitly specifying modulo *q* in the protocol, but it is assumed.

1. **Initialization phase:** Two servers S_i (*i = 1, 2*) jointly choose a cyclic group *G* of large prime order *q* with a generator g_1, and a secure hash function $H:\{0,1\}^* \rightarrow \mathbb{Z}_q$. Next, S_i randomly chooses an integer s_i from $\mathbb{Z}_q^*$, *i = 1, 2*. They compute and exchange ${g_1}^{s_1}$ *mod q* and ${g_1}^{s_2}$ *mod q*, and jointly publish public system parameters *G, q*, g_1, g_2, *H* where
$$g_2 = {g_1}^{s_1 s_2} \bmod q. \quad (1)$$
2. **Registration phase:** The client, *C*, registers at S_i (*i = 1, 2*) through secure channels. The client generates decryption and computes the encryption key to arrive at a key pair (x_i, y_i) were
$$y_i = {g_1}^{x_i} \bmod q \quad (2)$$
for server S_i (*i = 1, 2*). It then chooses a password, *pwc*, and

encrypts it using the encryption key y_i, which gives:

$A_i = g_1^{a_i} \bmod q$ (3)

$B_i = g_2^{pwc} y_i^{a_i} \bmod q$ (4)

with a_i randomly chosen from $\mathbb{Z}_q^*$ for *i = 1, 2*, according to ElGamal (1985). The client, C then randomly chooses b_1 from $\mathbb{Z}_q^*$, computes

$b_2 = H(pwc) \oplus b_1$, (5)

and sends authenticators, $Auth_C^{(i)}$, to S_1 and S_2 respectively:

$C \rightarrow S_1: Auth_C^{(1)} = \{x_1, a_1, b_1, (A_2, B_2)\}$ (6)

$C \rightarrow S_2: Auth_C^{(2)} = \{x_2, a_2, b_2, (A_1, B_1)\}$. (7)

3. **Authentication phase:** this phase involves five steps:

Step 1: The client, C, chooses at random *r* from $\mathbb{Z}_q^*$, computes

$R = g_1^r g_2^{-pwc} \bmod q$, (8)

and broadcasts request message in (9) to S_i (*i = 1, 2*)

$C \rightarrow S_i: M_1 = \{C, Req, R\}$. (9)

Step 2: S_1 chooses r_1 at random from $\mathbb{Z}_q^*$, computes

$A_2' = A_2^{r_1} \bmod q$, (10)

$B_2' = (R \cdot B_2)^{r_1} \bmod q$, (11)

and then prepares the message below based on results from (10) and (11)

$M_2 = \{A_2', B_2'\}$. (12)

S_2 chooses r_2 at random from $\mathbb{Z}_q^*$, computes

$A_1' = A_1^{r_2} \bmod q$, (13)

$B_1' = (R \cdot B_1)^{r_2} \bmod q$, (14)

and then prepares the message below based on results from (13) and (14)

$M_3 = \{A_1', B_1'\}$. (15)

S_1 and S_2 exchange messages (12) and (15).

Step 3: S_i (i = 1, 2) chooses r_i' at random from $\mathbb{Z}_q^*$, computes

$R_i = A_i'^{a_i^{-1} r_i'} \bmod q$, (16)

$K_i = (B_i'/A_i'^{x_i})^{r_i'} \bmod q$, (17)

$h_i = H(K_i, 0) \oplus b_i$, (18)

and then replies to the message M_{3+i} to the client *C*

$S_i \rightarrow C: M_{3+i} = \{S_i, R_i, h_i\}$. (19)

Step 4: The client *C* computes the following for *i = 1, 2* after receiving messages (19)

$K_i' = R_i^r \bmod q$, (20)

and checks if

$H(K_1', 0) \oplus H(K_2', 0) \oplus h_1 \oplus h_2 = H(pwc)$. (21)

Servers S_i (*i =1, 2*) are considered to be authentic if equality (21) holds. Then the client, *C,* computes:

$h_i' = H(K_i', 1) \oplus H(K_i', 0) \oplus h_i$, (22)

broadcasts the message, M_6 to S_i (i = 1, 2)

$C \rightarrow S_i: M_6 = \{h_1', h_2'\}$, (23)

and establishes secret session keys with S_i (*i =1, 2*):

$SK_i' = H(K_i', 2)$. (24)

Step 5: S_i (*i = 1, 2*) will check if equation (25) holds after receiving the message in (23) and conclude that the client, *C,* is authentic, otherwise not authentic

$H(K_i, 1) \oplus b_i = h_i'$. (25)

Finally, the servers S_i (*i = 1, 2)*, establish secret session keys with the client, *C*, as in (26).

$SK_i = H(K_i, 2)$. (26)

The left-hand side of (24) is equal to the left-hand side of (26) because the left-hand side of (17) is equal to the left-hand side of (20).

3. METHOD

In the protocol reviewed above, we can see that equations (3) and (4) require the client *C* to choose a_i (*i = 1, 2*) randomly from $\mathbb{Z}_q^*$, "according to ElGamal encryption" [Yi *et al*, (2013), p. 1777, section 4.2.2]. In the proof of their Theorem 1 [Yi *et al*, (2013), p. 1778, right column], it is shown that $K_1 = K_1'$ (see equations (17) and (20)) since, from equations (2) - (4), (8), (13), and (14), we have

$K_1 = g_1^{r r_1' r_2} \bmod q$, (27)

and from (3), (13), and (16), we have

$K_1' = R_1^r = (g_1^{r_1' r_2 a_1 a_1^{-1}})^r = g_1^{r r_1' r_2 a_1 a_1^{-1}} = g_1^{r r_1' r_2} \bmod q$. (28)

But the rightmost-hand side of equation (28) is true only when

$r r_1' r_2 a_1 a_1^{-1} = r r_1' r_2 \bmod (q\text{-}1)$ (29)

As far as a_i is selected and used according to equations (3) and (4) from $\mathbb{Z}_q^*$, its inverse modulo *q* exists and is used in equations (16), (28), and (29). It is known in Stallings (2006) that the inverse of an integer depends on the modulo for which it is considered. In equation (16), it is not specified the modulo for which the inverse of a_i is calculated. In the description of their protocol (see p. 1776-1779 in Yi *et al* (2013)), modulo operations are not shown explicitly, but assumed as modulo *q*. Hence, the inverse of a_i is also to be modulo *q*. Actually, all numbers below *q* are invertible modulo *q* and hence can be selected randomly as it is supposed since they are invertible modulo *q*. In that case, the left-hand side of equation (29) is

$$rr_1'r_2a_1a_1^{-1} = rr_1'r_2(1 + nq) = rr_1'r_2 + nqrr_1'r_2 \quad (30)$$

for some integer *n*, and may not be equal to the right-hand side of equation (29) modulo Euler's totient function $\varphi(q) = q - 1$, for which Stallings (2006) and any *a, k*

$$a^{k\varphi(q)} mod\ q = 1 \quad (31)$$

holds. The source of the problem with this protocol is that parameters used in its exponents are not considered modulo Euler's totient function $\varphi(q) = q - 1$. In the following Tables 1 – 3, we present a numerical example that the communicating parties will arrive at different session keys at the end of the computations due to using congruency of the powers in equation (29) modulo *q* instead of (*q – 1*). In Tables 1 – 3, settings for the Initialization, Registration, and Authentication phases respectively are shown.

Table 1: Numerical example leading to the failure of the Yi et al (2013) protocol – Initialization phase

S/N	Actor	Action	Result
1.	Servers, S_1 and S_2	Choose a cyclic group of large prime *q* with a generator g_1.	*G = {1, 2, … ,12}* *q = 13,* g_1 *= 2*
2.	Server, S_1	Chooses s_1 randomly from $\mathbb{Z}_q^*$	s_1 *= 2*
3.	Server, S_2	Chooses s_2 at random from $\mathbb{Z}_q^*$	s_2 *= 3*
4.	Servers, S_1 and S_2	Exchange messages and arrive at g_2 using equation (1)	$S_1 \rightarrow S_2$: ${g_1}^{s_1} = 4$ $S_2 \rightarrow S_1$: ${g_1}^{s_2} = 8$ $g_2 = 4^3 = 8^2 = 12$

Servers, S_1 and S_2 jointly publish the following public system parameters:

G = {1, 2,…, 12}, q = 13, g_1 *= 2,* g_2 *= 12, H :{0, 1}** $\rightarrow \mathbb{Z}_q$.

Table 2: Numerical example leading to the failure of the Yi et al (2013) protocol – Registration phase

S/N	Actor	Action	Result
1.	Client, *C*	Generates decryption and encryption keys, (x_i, y_i), with y_i (*i = 1, 2*) computed using equation (2).	x_1 *= 2,* x_2 *= 3,* y_1 *= 4,* y_2 *= 8*
2.	Client, *C*	Chooses a password, *pwc*.	*pwc = 3*
3.	Client, *C*	Encrypts the password using equations (3) and (4)) with a_i (*i = 1, 2*) chosen at random from $\mathbb{Z}_q^*$ to obtain A_i and B_i (*i = 1,2*)	a_1*= 6,* A_1*= 12,* B_1*= 12* a_2*= 6,* A_2 *= 12,* B_2 *= 1*
4.	Client, *C*	Chooses b_1 at random from $\mathbb{Z}_q^*$ and compute b_2 according to equation (5)	b_1 *= 5* b_2 *= H(pwc)* $\oplus$ b_1
5.	Client, *C*	Delivers authenticator, $Auth_C^{(i)}$ to S_1 and S_2 according to equations (6) and (7)	$C \rightarrow S_1$: $Auth_C^{(1)}$ $= \{2, 6, 5, (12, 1)\}$ $C \rightarrow S_2$: $Auth_C^{(2)}$ $= \{3, 6, b_2, (12, 12)\}$

Table 3: Numerical example leading to the failure of the Yi et al (2013) protocol – Authentication phase

S/N	Actor	Action	Result
1.	Client, C	Chooses at random *r* from $\mathbb{Z}_q^*$, computes *R* using equation (8) and broadcasts message equation (9).	*r = 5, R = 7* C $\rightarrow$ S_1, S_2: M_1 = {*C, Req, 7*}
2.	Server, S_1	Chooses at	r_1 *= 3*

		random r_1 from $\mathbb{Z}_q^*$ and computes A_2', B_2', prepares message M_2 (see equations (10), (11), and (12); sends the message to S_2.	A_2' = *12* B_2' = *5* $S_1 \rightarrow S_2$: M_2 = {*12, 5*}
3.	Server, S_2	Chooses at random r_2 from $\mathbb{Z}_q^*$ and computes A_1', B_1', prepares message M_3 (see equations (13), (14), (15); sends the message to S_1.	r_2 = 7 A_1' = 12 B_1' = 7 $S_2 \rightarrow S_1$: M_3 = {12, 7}
4.	Server, S_1	Chooses at random r_1' from $\mathbb{Z}_q^*$, computes R_1, K_1, h_1, prepares message M_4 (see equations (16), (17), (18), (19); sends to client, *C*.	r_1' = 3 R_1 = 12 K_1 = 5 $h_1 = H(K_1, 0) \oplus b_1$ $S_1 \rightarrow$ C: M_4= {S_1, 12, h_1}
5.	Server, S_2	Chooses at random r_2' from $\mathbb{Z}_q^*$, computes R_2, K_2, h_2, prepares message M_5 (see equations (16), (17), (18), (19); sends to client, *C*.	r_2' = 6 R_2 = 1 K_2 = 12 $h_2 = H(K_2, 0) \oplus b_2$ $S_2 \rightarrow$ C: M_5= {S_2, 1, h_2}
6.	Client, *C*	Computes K_1' and K_2' using equation (20).	K_1' = 12 K_2' = 1

We cannot continue with the rest of the computations since in Table 3, K_1' in row 6 is not equal to K_1 in row 4, which are meant to be equal to enable parties to arrive at the same secret keys in equations (24) and (26) at the end of the computations. The difference in values of K_1' and K_1 is a result of computations involving a_i (*i = 1, 2*) based on exponents congruent modulo *q*. This shows that choosing a_i (*i = 1, 2*) at random from $\mathbb{Z}_q^*$ with its multiplicative inverse involved in computations, may lead parties to arrive at different secret keys.

4. RESULT AND DISCUSSION

The proof of their theorem 1 [Yi *et al*, (2013), p. 1778, Section 4.2.4], shows that the right-hand sides of equations (17) and (20) are equal, and therefore the secret keys in equations (24) and (26) are the same. From equations (3), (4), (8), (13), and (14) respectively for S_1, we have

$$A_1' = (g_1^{a_1})^{r_2} = g_1^{r_2 a_1} \ (mod\ q), \quad (32)$$

$$B_1' = (g_1^{r} g_2^{-pwc} g_2^{pwc} y_1^{a_1})^{r_2} = g_1^{r r_2} y_1^{r_2 a_1} \ (mod\ q). \quad (33)$$

From equations (32) and (16), we have

$$R_1 = (g_1^{r_2 a_1})^{a_1^{-1} r_1'} = {g_1}^{r_1' r_2}, \quad (34)$$

where $a_i a_i^{-1}$ varnishes in equation (34). Taking $R_1 = (g_1^{r_2 a_1})^{a_1^{-1} r_1'}$ from equation (34) and using the values $q = 13,\ g_1 = 2,\ a_1 = 6,\ r_2 = 7,\ r_1' = 3$ defined in Table 1, row 1, Table 2, row 3, and Table 3, rows 3 and 4, we have $R_1 = (2^{7\cdot 6})^{6^{-1} mod 13 \cdot 3}\ mod\ 13 = (2^{7\cdot 6})^{11\cdot 3}\ mod\ 13 = (2^{4\cdot 10} \cdot 4)^{33}\ mod\ 13 = (3^{10} \cdot 4)^{33}\ mod\ 13 = (3^3 \cdot 3 \cdot 4)^{33}\ mod\ 13 = 12^{33}\ mod\ 13 = 12$, which is same in row 6 of Table 3. But using the right-hand side of equation (34), $R_1 = {g_1}^{r_1' r_2} = 2^{3\cdot 7}\ mod\ 13 = 2^{4\cdot 5} \cdot 2\ mod\ 13 = 3^5 \cdot 2\ mod\ 13 = 9 \cdot 2\ mod\ 13 = 5$ (as in row 4, Table 3), which is not equal to 12, previously obtained. Thus, equation (34) allegedly proved as in Section 4.2.4 of Yi *et al* (2013) is not true, and $R_1 = A_1'^{a_1^{-1} r_1'} = (g_1^{r_2 a_1})^{a_1^{-1} r_1'} \neq {g_1}^{r_1' r_2}$.

This failure of the proof is due to the use of inverse of the exponent a_i (*i = 1, 2*) modulo *q* instead of using multiplicative inverse modulo Euler's totient function $\varphi(x)$ defining the number of numbers less than *x* and relatively prime to *x*, which is for the case under consideration, $\varphi(q) = q - 1$. If we use inverse modulo *q – 1*, we get $R_1 = (g_1^{r_2 a_1})^{a_1^{-1} r_1'} = (2^{7\cdot 6})^{6^{-1} mod 12 \cdot 3} mod\ 13$, which is not defined since $6^{-1}\ mod\ 12$ does not exist. Hence, just finding inverses modulo (*q – 1*) is not sufficient to fix the protocol.

A proposed modification of the protocol is presented such that a_i (i = 1, 2) should be chosen from $\mathbb{Z}_q^*$, the condition of relative primality

$$\gcd(a_i, q - 1) = 1 \quad (35)$$

holds. Hence, in the registration phase of the protocol, instead of writing after (4) "with a_i randomly chosen from $\mathbb{Z}_q^*$", we should write "with relatively prime to ($q - 1$) values a_i meeting equation (35) and randomly chosen from $\mathbb{Z}_q^*$." We see that our choice of values for a_i in Table 2, row 3, violates equation (35), leading to the failure of the protocol. If equation (35) holds, due to equation (31), then the proof of equation (34) and their Theorem 1 in Section 4.2.4 of Yi *et al* (2013) are correct since equation (34) was the only observed problem in the proof. Proving equation (34) using equation (31), we have $R_1 = (g_1^{r_2 a_1})^{a_1^{-1} mod(q-1) r_1'} = g_1^{r_2 a_1 a_1^{-1} mod(q-1) r_1'} = g_1^{r_2(1+k(q-1)) r_1'} = g_1^{r_2 r_1'} g_1^{k(q-1)} = g_1^{r_1' r_2} mod\ q.$

If we use the same settings as previously but with $a_1 = 5$ which is relatively prime to $q - 1 = 12$, we have, $R_1 = (2^{7\cdot5})^{5^{-1} mod 12\cdot3}\ mod\ 13 = (2^{7\cdot5})^{5\cdot3}\ mod\ 13 = 7^{15}\ mod\ 13 = 5$ which is the same as the right-hand side of equation (34) calculated earlier.

5. CONCLUSION

We have presented the password authenticated key exchange in this research work, with much analyses on the efficient two-server password-only authenticated key exchange of Xun Yi, San Ling, and Huaxiong Wang. In our investigation of their protocol, we discovered a challenge that make the protocol fail to arrive at a shared secret session key at some points. This problem is due to the application of inverses in the exponent modulo q, instead of modulo Euler's totient function in the computations that make parties arrive at different secret session keys. The congruency of exponents was not considered modulo Euler's totient function in computations where inverses of the parameter are used. Numerical examples were employed to analyse and reveal the challenge with the protocol. A proposed modification of the protocol was presented by restricting how to choose at random the parameter, a_i from all the elements that are invertible in the cyclic group, since its multiplicative inverse is used in computations. We provided proof and numerical examples to show that the modification will make the protocol more secure and efficient.

REFERENCES

Ampomah M. O., Hayfron-Acquah J. B., Twum F., and Panford J. K. (2015), "Hash-Based Random Salt Password Authentication in Two Servers", International Research Journal of Engineering and Technology (IRJET), Volume: 02 Issue: 05, pp. 853 – 859.

Anderson, R. J. (2001), "*Security Engineering: A Guide to Building Dependable Distributed Systems",* New York: Wiley Pub.

Bellare M., Pointcheval D., and Rogaway P. (2000), "Authenticated Key Exchange Secure against Dictionary Attacks", *Proc. 19th Int'l Conf. Theory and Application of Cryptographic Techniques (Eurocrypt '00),* pp. 139-155.

Bellovin S., and Merritt M. (1992), "Encrypted Key Exchange: Password-Based Protocol Secure against Dictionary Attack", *Proc. IEEE Symp. Research in Security and Privacy,* pp. 72-84.

Boneh D. and Franklin M. (2001), "Identity Based Encryption from the Weil Pairing", *Proc. 21st Ann. Int'l Cryptology Conf. (Crypto '01),* pp. 213-229.

Boneh D. and Franklin M. (2003), "Identity Based Encryption from the Weil Pairing", *SIAM J. Computing,* vol. 32, no. 3, pp. 586-615.

Boyko V., Mackenzie P., and Patel S. (2000), "Provably Secure Password-Authenticated Key Exchange Using Diffie-Hellman", *Proc. 19th Int'l Conf. Theory and Application of Cryptographic Techniques (Eurocrypt '00),* pp. 156-171.

Brainard J., Jueles A., Kaliski B. S., and Szydlo M. (2003), "A New Two-Server Approach for Authentication with Short Secret", *Proc. 12th Conf. USENIX Security Symp,* pp. 201-214.

ElGamal T. (July 1985)., "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms", IEEE Trans. Inf. Theory. [Online]. 31(4), pp. 469-472. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1057074.

Gong L., Lomas T. M., Needham R. M., and Saltzer J. H. (June 1993), "Protecting Poorly-Chosen Secret from Guessing Attacks", *IEEE J. Selected Areas in Comm,* vol. 11, no. 5, pp. 648-656.

Halevi S., and Krawczyk H. (1999), "Public-Key Cryptography and Password Protocols", *ACM Trans. Information and System Security,* vol. 2, no. 3, pp. 230-268.

Katz J., MacKenzie P., Taban G., and Gligor V. (2012), "Two-Server Password-Only Authenticated Key Exchange", *Journal of Computer and System Sciences,* vol. 78, pp. 651-669.

Katz J., Ostrovsky R., and Yung M. (2001), "Efficient Password-Authenticated Key Exchange using Human-Memorable Passwords", *Proc. Int'l Conf. Theory and Application of Cryptographic Techniques: Advances in Cryptology (Eurocrypt '01),* pp. 457-494.

Lomas T. M., Gong L., Saltzer J. H., and Needham R. M. (1989), "Reducing Risks from Poorly-Chosen Keys", *ACM Operating Systems Rev.,* vol. 23, no. 5, pp. 14-18.

Stallings W. (2006), "Introduction to number theory in *Cryptography and Network Security: Principles and Practice", 5*th ed., Boston: Prentice Hall, pp. 243-251.

Yang Y., Bao F., and Deng R. H. (2005), "A New Architecture for Authentication and Key Exchange Using Password for Federated Enterprises", *Proc. 20th IFIP Int'l Information Security Conf. (SEC '05),* pp. 95-111.

Yang, Y., Deng R. H., and Bao F. (Apr.-June 2006), "A Practical Password-Based Two-Server Authentication and Key Exchange System", *IEEE Trans. Dependable and Secure Computing,* vol. 3, no. 2, pp. 105-114.

Yi X., Tso R., and Okamoto E. (2009), "ID-Based Group Password-Authenticated Key Exchange", *Proc. Fourth Int'l Workshop Security: Advances in Information and Computer Security (IWSEC '09)*, pp. 192-211.

Yi X., Tso R., and Okamoto E., (2011), "Three-Party Password-Authenticated Key Exchange without Random Oracles", IEEE *Proc. Int'l Conf. Security and Cryptography (SECRYPT '11)*, pp. 15-24.

Yi X., Tso R., and Okamoto E. (2012), "Identity-Based Password-Authenticated Key Exchange for Client/Server Model", *Proc. Int'l Conf. Security and Cryptography (SECRYPT '12)*, pp. 45-54.

Yi, X., Ling S., and Wang H. (Sept. 2013), "Efficient Two-Server Password-Only Authenticated Key Exchange", *IEEE Transactions on Parallel and Distributed System,* vol. 24, no.9, pp. 1773-1782.

