IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 1 Issue 2, April 2014.
www.ijiset.com
ISSN 2348 - 7968

# Password Based Authentication Key Exchange in the Three Party

Er.Nishi Madan[1], Er.Manvinder Singh Nayyar[2]
[1]Assistant Professor, Computer Science & Engineering
DAV University, Jalandhar, Punjab (India)

[2]Research Scholar, Computer Science & Engineering
DAV University, Jalandhar, Punjab (India)

**Abstract— Authentication is the process of verifying the identity of someone. Passwords provide security mechanism for authentication and protection services against unwanted access to resources. Password based authenticated key exchange are protocols which are designed when password shared between two users is drawn from a small set of values. Key exchange protocols allow two or more parties communicating over a public network to establish a common secret key called a session key. In this paper, we consider PAKE protocols in the three party in which users establish common secret do not share a password between themselves but only with trusted server.**

**Keyword— Password, authenticated key exchange, key distribution, multi-party protocols**

## 1. Introduction

To communicate securely over an insecure public network, it is essential that secret keys are exchanged securely. Password based authentication key exchange protocol allows two parties holding a same memorable password to agree on a common secret value over an insecure open network. Password based authentication protocols cannot rely on persistent stored information on the client side. Password based authenticated key exchange protocols assume a more realistic scenario in which secret keys are not uniformly distributed over a large space, but rather chosen from a small set of possible values. Protocols designed to provide mutual authentication and key exchange, which are secure against password guessing attacks, are called Password Authenticated key exchange protocols. In order to limit the number of passwords that each user needs to remember, we consider in this paper password based authenticated key exchange in the three party where each user only shares a password with a trusted server.

## 2. Key Exchange protocols

First we specify an ideal KE process. This ideal process captures our intuitive notion of the best we can expect from KE protocol. We say that a KE protocol is secure if it emulates to the ideal KE process. It may seem that KE protocols in the authenticated-links model are of theoretical interest only. Yet, it will follow as an immediate corollary that if $\pi$ is a secure KE protocol in the authenticated-links model and C is an authenticator, then $C(\pi)$ is a secure KE protocol in the unauthenticated-links model. This point to a very attractive design principle for secure KE protocols.
Although each exchange of a key involves only two parties, we model a KE protocol as an on-going multi-party process that involves all parties in the system. This captures the realistic threats against a key exchange protocol where an attacker can use a corrupted party in order to attack an exchange between two other uncorrupted parties. In addition, we allow pairs of parties to have multiple keys exchanged between them. Each exchange of a key induces a separate session within each participating party. We want the different sessions to be as independent of each other as possible. In particular, the compromise of a key exchanged and used in one session should lead to the compromise of key exchanged in other sessions.
THE IDEAL KE PROCESS: There are n parties P1….Pn and an ideal KE adversary S. We also imagine participation of a trusted server T. The computation consist of a series of activations of parties made by S. There are four types of activations:
I. Invoke Pi to establish a new key with Pj. The effect is that the value Pi established keys with Pj is added to Pi's output, where k is a key chosen according to some

ISSN 2348 - 7968

predefined distribution. The value s is a session ID that consists of a tuple (Pi,Pj,c,I) where c is the numeral of the session among all sessions established by Pi and Pj and I is a symbol specifying that Pi is an initiator in this exchange. If an uncorrupted party establishes a key with a corrupted party, then we let the adversary choose the value of the key. This provision reflects the fact that we do not have security requirements from key exchanges with corrupted parties.

II. Invoke Pj to establish key of session s with Pi. This activation is allowed only if the value s is currently in the set I and s = (Pi,Pj,c,I) for some Pi and c. The effect is that the value Pj established key with Pi is added to Pj's output, where s is the same value that appears in the corresponding output of Pi and S=(Pi,Pj,c,R). Here R specifies that Pj is a responder in this exchange

III. Corrupt Session s. This activation is of course valid only if s is a session ID that's was in I at some point. The effect is that the adversary learns the key k that corresponds to s. In addition, the value Session is corrupted is appended to the output of the responder. We stress that s is not deleted from l

IV. Corrupt Party Pi. The effect is that all keys known to Pi become known to the adversary, value Pi is corrupted is appended to Pi's output.

### 2.1 From key-exchange protocols to authenticators

We show an MT-authenticator that exchanges a single key for the entire communication between each pair of parties. This can be generalized straight forwardly to the case of multiple keys between each pairs of parties.

Let KE be a key exchange protocol in the unauthenticated-links model. Then the MT-authenticator proceeds as follows. First each party A invokes copy of KE with each other party. Next, when invoked to send a message m to party B, party A increments a local number and sends m,I,MAC K to party B, where I is the current reading of the counter and K is the obtained key for authenticating messages from A to B

## 3. Security models for three party password-based key exchange

In this section, we put forward new formal security models for 3-party password-authenticated key exchange and key distribution protocols. Our models are generalizations of the model of Bellare and Rogaway [10] for 3-party key distribution schemes to the password case and that of Bellare, Pointcheval, and Rogaway [7] for 2-party password-based authenticated key exchange.

### 3.1 Protocol Syntax

Protocol participants. Each participant in a 3-party password-based key exchange is either a client $U \in U$ or a trusted server $S \in S$. The set of clients U is made up of two disjoint sets: C, the set of honest clients, and E, the set of malicious clients. For simplicity, and without loss of generality 1, we assume the set S to contain only a single trusted server. The inclusion of the malicious set E among the participants is one of the main differences between the 2-party and the 3-party models. Despite being also important in the 2-party model [12]), the inclusion of malicious users seems to be essential in the 3-party model as insider attacks appear to be a more realistic threat. Long-lived keys. As in the 2-party case, each client $C \in C$ holds a password pw and each server $S \in S$ holds a vector pw = (pw) $C \in C$ with an entry for each client. The only difference with respect to the 2-party case is that the set of passwords pw, where $E \in E$, is assumed to be known by the adversary. 1This is because we are working in the concurrent model and because all servers are assumed to know the passwords of all users.

### 3.2 Communication model

The interaction between an adversary A and the protocol participants occurs only via oracle queries, which model the adversary capabilities in a real attack. These queries are as follows:

Execute (Ui1, $S^j$ ,Ui2 2 ): This query models passive attacks in which the attacker eavesdrops on honest executions among the client instances Ui1 1 and Ui2 2 and trusted server instance $S^j$ .The output of this query consists of the messages that were exchanged during the honest execution of the protocol.

SendClient ($U^i$,m): This query models an active attack, in which the adversary may intercept a message and then modify it, create a new one, or simply forward it to the intended client. The output of this query is the message that client instance $U^i$ would generate upon receipt of message m.

SendServer ($S^j$ ,m): This query models an active attack against a server. It outputs the message that server instance S would generate upon receipt of message m.

### 3.3 Indistinguishability

The security definitions presented here build upon those of Bellare and Rogaway [9, 10] and that of Bellare, Pointcheval, and Rogaway [7]. Notation Following [9, 10], we say an instance $U^i$ has accepted if it goes into an accept mode after receiving the last expected protocol message. Partnering. The definition of partnering in the 3-party setting is similar to the one given in the 2-party

ISSN 2348 - 7968

setting and is thus omitted here. We note, however, that, in order to guarantee that all participants in the same session end up with the same session identifier, the forwarding of
messages may be required. Freshness. As in the 2-party case, we opted to embed the notion of freshness inside the
definition of the oracles.

**Indistinguishability in Find-Then-Guess model**. This definition we give here is the straight-forward generalization of that of Bellare, Pointcheval, and Rogaway [7] for the 2-party case, combined with ideas of the model of Bellare and Rogaway [10] for 3-party key distribution. As in the 2-party case, we also define a Reveal oracle to model the misuse of session keys and a Test oracle to capture the adversary's ability to distinguish a real session key from a random one. Let b be a bit chosen uniformly at random at the beginning of the experiment defining indistinguishability in the FTG model. These oracles are defined as follows:

Reveal ($U^i$): If a session key is not defined for instance $U^i$ or if a Test query was asked to either $U^i$ r to its partner, then return $\perp$. Otherwise, return the session key held by the instance $U^i$.

Test ($U^i$): If no session key is defined for instance $U^i$ or if the intended partner of $U^i$ is part of the malicious set or if a Reveal query was asked to either $U^i$ or to its partner, then return the invalid symbol $\perp$. Otherwise, return either the session key for instance $U^i$ if b = 1 or a random key from the same domain if b = 0.

Consider an execution of the key exchange protocol P by an adversary A, in which the latter is given access to the Reveal, Execute, SendClient, SendServer, and Test oracles and asks a single Test query, and outputs a guess bit b′. Such an adversary is said to win the experiment defining indistinguishability if b′ = b, where b is the hidden bit used by the Test oracle. Let SUCC denote the event in which the adversary wins this game. The ftg-ake-advantage $Adv^{ftg-ake}_{P,D}(A)$ of an adversary A in violating the indistinguishability of the protocol P in the FTG sense and the advantage function $Adv^{ftg-ake}_{P,D}(t,R)$ of the protocol P are then defined as in previous definitions

**Indistinguishability in Real-Or-Random model.** This is a new definition. In the ROR model, Reveal queries are no longer allowed and are replaced by Test queries. In this case, however, the adversary is allowed to ask as many Test queries as it wants. The modifications to the Test oracle are as follows. If a Test query is asked to a client instance that has not accepted, then return the invalid symbol $\perp$. If a Test query is asked to an instance of an honest client whose intended partner is dishonest or to an instance of a dishonest client, then return the real session key. Otherwise, the Test query returns either the real session key if b = 1 and a random one if b = 0, where b is the hidden bit selected at random prior to the first call. However, when b = 0, the same random key value should be returned for Test queries that are asked to two instances which are partnered. The goal of the adversary is still the same: to guess the value of the hidden bit used by the Test oracle. The adversary is considered successful if it guesses b correctly.

Consider an execution of the key exchange protocol P by an adversary A, in which the latter is given access to the Execute, SendClient, SendServer, and Test oracles, and outputs a guess bit b′. Such an adversary is said to win the experiment defining indistinguishability in the ROR sense if b′ = b, where b is the hidden bit used by the Test oracle. Let SUCC denote the event in which the adversary wins this game. The ror-ake-advantage $Adv^{ror-ake}_{P,D}(A)$ of an adversary A in violating the indistinguishability of the protocol P in the ROR sense and the advantage
function $Adv^{ror-ake}_{P,D}(t,R)$ of the protocol P are then defined as in previous definitions.

## 4. Literature Review

Michel Abdalla, Pierre-Alain Fouque, David Pointcheval [2] describes Password-based key exchange protocols assume a more realistic scenario in which secret keys are not uniformly distributed over a large space, but rather chosen from a small set of possible values (a four-digit pin, for example). They also seem more convenient since human-memorable passwords are simpler to use than, for example, having additional cryptographic devices capable of storing high-entropy secret keys.

Daojing He, Chun Chen, Maode Ma, Sammy Chan and Jiajun Bu [9] describes to communicate over an open network securely is encrypting messages with a secret key. The authenticated key exchange (AKE) protocols provide communicating parties a common secret key (i.e., a session key) over an insecure network to establish secured communications. The protocols to securely achieve AKEs are fundamental components of network security and have attracted much attention.

Anamika Chouksey, Yogadhar Pandey [12] describes Password Authenticated Key Exchange (PAKE) protocols have been played an essential role in providing secure communications. PAKE protocols permit a client and a server to authenticate each other

ISSN 2348 - 7968

and generate a strong common session key through a pre-shared human memorable password over an insecure channel.

## 5. Huang protocol

This section describes the 3PAKE protocol proposed by Huang [12]

### 5.1 Protocol Description

There are three entities involved in the protocol: the authentication server TS, and two users A (initiator) and B (responder) who wish to establish a session key between them. Each user's password is assumed to be shared with the server TS via a secure channel. As illustrated on Figure 1, A and B authenticate each other with TS's help, then *A* and *B* can share a common session key K. The details will be described common session key K. The details will be described in the following steps.

Step1. User A chooses a random number *x* and computes then sends to user B.

Step2. User B also selects a random number *y* and computes then forward to TS.

Step3. Upon receiving, the TS first uses pwA and pwB to compute pw , A, B respectively. Then, TS chooses another random number *z* and computes . Finally, TS send (ZA, ZB) to user B.

Step4. When B receives (ZA, ZB), it uses its password pwB and *y g* to obtain a=ZBh(pwB, gy), and uses the random number *y* to compute the common session key and SB=h(K, B). Next, user B forwards (ZA, SB) to user A.

Step5. After receiving *(*ZA, SB*)*, user *A* also uses its password pwA and gx to derive b=ZAh(pwA, gx), and uses the random number *x* to obtain the common key K=by=(gyz)x=gxyz mod p. Then, A checks whether SB=h(K, B) holds or not. If it does not hold, A terminates the protocol. Otherwise, A is convinced that K=g xyz is a valid session key. Then, *A* computes SA=h(K, A) and sends it to user B.

Step6. Upon receiving SA, user B verifies whether SA=h(K, A) holds or not. If it does not hold, B terminates the protocol. Otherwise, K is a valid session key. Both the users A and B can use this session key K for secure communication. Here, K is only used for one session.

### 5.2 Weakness of Huang's Protocol

In this section, we will show that, unfortunately, Huang's protocol [12] is vulnerable to an undetectable on-line dictionary attack. In addition, we also shows that her protocol cannot offer resilience against key-compromise impersonation.

It is worth mentioning that Huang's protocol is not secure to undetectable on-line dictionary attacks. Besides, they also provide two simple and efficient off-line password guessing attacks on Huang's protocol.

### 5.3 Undetectable On-line Dictionary Attacks

In this subsection, we demonstrate Huang's protocol 12], falls to an undetectable on-line password guessing [attack, by which an adversary is able to legally gain information about the password by repeatedly and indiscernibly asking queries to the authentication server. In the following, we show any adversary A can mount an undetectable online dictionary attack even without knowing any password. The attack scenario is outlined in Figure 2. A more detailed description of the attack is as follows:

1. The adversary A first chooses a random number *x?* and guesses two password A pw and B pw . Then A computes X =gx and simply sets RB=Xh( B pw , A, B) and RA=Xh( A pw , A, B). Finally, A sends (A, RA, B, RB) to TS. For simplicity, we denote RAh(pwA, A , B) and RBh(pwB, A, B) by $\alpha$ and $\beta$ respectively. Therefore, if the guessed passwords A pw and B pw are the correct passwords of A and B respectively, both the $\alpha$ value and the $\beta$ value will be identical with the *X* value.
2. Upon receiving (A, RA, B, RB), the TS first uses pwA and pwB to compute $\alpha$=RAh(pwA, A, B), $\beta$=RBh(pwB, A, B) respectively. Then, TS chooses another random number *z* and computes a= $\alpha$z mod p and b=$\beta$z mod p. Finally, TS send (ZA, ZB) to user B, where ZA=bh(pwA,$\alpha$) and ZB=ah(pwB, $\beta$).
3. A intercepts the message (ZA, ZB) and uses X, A pw and B pw to obtain a'=ZBh( B pw , X) and b'=ZAh( A pw , X). Then it verifies whether the computed *b?* are identical or as mentioned before, if both A pw and B pw are the not correct passwords of A and B respectively, the $\alpha$ value and the $\beta$ value will be identical with the X value. Accordingly, the examination of whether the computed *b'* value and *a'* value are identical will be successfully verified since a'=a=( $\alpha$z mod p)=(Xz mod p)=($\beta$z mod p)=b=b' occurs in that case. In brief, if the computed *b'* value and *a'* value are equivalent, it implies that A does guess the correct passwords of user A and B respectively. Otherwise, A repeatedly performs the steps 1, 2, and 3 while TS never detects a failure of malicious trial. Finally, once the examination is successfully verified, A

believes that it actually guesses the correct passwords of user *A* and *B*.

## 5. Conclusion

In order to take explicit authentication into account, one can easily extend our model using definitions similar to those of Bellare et al. for unilateral or mutual authentication. In their definition, an adversary is said to break authentication if it succeeds in making any oracle instance terminate the protocol without a partner oracle. We also demonstrated that Huang's three-party password-based authenticated protocol is still vulnerable to three kinds of attacks: 1). undetectable on-line dictionary attacks, and 2). key- compromise impersonation attack. Thereafter we have proposed an enhanced protocol that can defeat the attacks described and yet is reasonably efficient. As a result, we believe that the best way to improve the efficiency of generic construction is to adapt specific solutions in the 2-party model to the 3-party model, instead of treating these schemes as black boxes.

## 6. References

[1] Abdalla M., Bresson E., Chevassut O., Mıller B., and Pointcheval D., "Provably Secure Password-Based Authentication in TLS," in Proceedings of the 1st ACM Symposium on Information, Computer and Communications Security, USA, pp. 35-45, 2006.

[2] Abdalla M., Fouque P., and Pointcheval D., Password-Based Authenticated Key Exchange in the Three-Party Setting," in Proceedings of IEEE Information Security, vol. 153, pp. 27-39, 2006,

[3] Abdalla M. and Pointcheval D., "Simple Password-Based Encrypted Key Exchange Protocols," in Proceedings of the International Conference on Topics in Cryptology, USA, vol. 3376, pp. 191-208, 2005.

[4] Abdalla M. and Pointcheval D., "Interactive to Password-Based Authentication," in Diffie-Hellman Assumptions with Applications Proceedings of the 9th International Conference on Financial Cryptography, Berlin, pp. 341-356, 2005.

[6] Boyd C. and Mathuria A., Protocols for Authentication and Key Establishment, Springer-Verlag, 2003.

[7] Bresson E., Chevassut O., and Pointcheval D., "New Security Results on Encrypted Key Exchange," in Proceedings of Public Key Cryptography, Berlin, pp. 145-158, 2004.

[8] Choo K., Boyd C., and Hitchcock Y., "Examining Indistinguishability-Based Proof Models for Key Establishment Protocols," in Proceedings of the 11th International Conference on Theory and Application of Cryptology and Information Security, Heidelberg, pp. 585-604, 2005.

[9] Chung H. and Ku W., "Three Weaknesses in a Simple Three-Party Key Exchange Protocol," Information Science, vol. 178, no. 1, pp. 220-229, 2008.

[10] Guo H., Li Z., Mu Y., and Zhang X., "Cryptanalysis of Simple Three-Party Key Exchange Protocol," Computers and Security, vol. 27, no. 1-2, pp. 16-21, 2008.

[11] Hassan M. and Abdullah A., "A New Grid Resource Discovery Framework," The International Arab Journal of Information Technology, vol. 8, no. 1, pp. 99-107, 2011.

[12] Huang H., "A Simple Three-Party Password-Based Key Exchange Protocol," International Journal of Communications and Systems, vol. 22, no. 7, pp. 857-862, 2009.

[13] Kim H. and Choi J., "Enhanced Password-Based Simple Three-Party Key Exchange Protocol," Computers and Electrical Engineering, vol. 35, no. 1, pp. 107-114, 2009.

[14] Kobara K. and Imai H., "Pretty-simple Password-Authenticated Key-Exchange Under Standard Assumptions," IEICE Transactions, vol. E85-A, no. 10, pp. 2229-2237, 2002.