Analysis of two pairing-based three-party password authenticated key exchange protocols

This item was submitted to Loughborough University's Institutional Repository by the/an author.


Additional Information:

- This is a conference paper [© IEEE]. It is also available at: http://ieeexplore.ieee.org/ Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.

Metadata Record: https://dspace.lboro.ac.uk/2134/5683

Version: Published

Publisher: © IEEE

Please cite the published version.
This item was submitted to Loughborough’s Institutional Repository (https://dspace.lboro.ac.uk/) by the author and is made available under the following Creative Commons Licence conditions.

For the full text of this licence, please go to: http://creativecommons.org/licenses/by-nc-nd/2.5/
Analysis of Two Pairing-based Three-party Password Authenticated Key Exchange Protocols

Raphael C.-W. Phan
Electronic and Electrical Engineering
Loughborough University
United Kingdom
r.phan@lboro.ac.uk

Wei-Chuen Yau, Bok-Min Goi
Faculty of Engineering
Multimedia University
Cyberjaya, Malaysia
wcyau@mmu.edu.my, bmgoi@mmu.edu.my

Abstract—Password-Authenticated Key Exchange (PAKE) protocols allow parties to share secret keys in an authentic manner based on an easily memorizable password. Recently, Nam et al. showed that a provably secure three-party password-based authenticated key exchange protocol using Weil pairing by Wen et al. is vulnerable to a man-in-the-middle attack. In doing so, Nam et al. showed the flaws in the proof of Wen et al. and described how to fix the problem so that their attack no longer works. In this paper, we show that both Wen et al. and Nam et al. variants fall to key compromise impersonation by any adversary. Our results underline the fact that although the provable security approach is necessary to designing PAKEs, gaps still exist between what can be proven and what are really secure in practice.

Keywords—Password-authenticated key exchange; cryptanalysis; attacks; provable security; three-party; key compromise impersonation; Weil pairing

I. INTRODUCTION

A 2-party password-based authenticated key exchange (PAKE) protocol [5] allows two parties to authenticate each other and to establish a common session key for securing a communication session using a low-entropy password. The first known PAKE is due to Bellovin and Merritt [6]. This concept has also been extended to 3 parties, e.g. two clients and a trusted server or key distribution center (KDC) [1], [2], [3], [10], [14], [17], [18], [19], [20], [35], [36], [22].

Recently, Wen et al. proposed a three-party password-based authenticated key exchange protocol using Weil pairing [37]. The protocol was formally proven secure in the random oracle model [4], [5]. Nam et al. [25] however showed that the security proof in [37] is flawed and pointed out that the protocol is vulnerable to unknown key-share attack (UKS) [16], [13], i.e. a client A (resp. B) thinking it is sharing a key with B (resp. A) when it is actually sharing with a malicious adversary C. Nam et al. proposed a way to prevent their attack, basically by including the identities of both clients into the generation of the secret session key shared by each client with the server.

The basic requirements of PAKEs can be found in literature, e.g. [24], [9]. In particular, they are as follows.

- **Dictionary attack resilience:** Originally, a dictionary attack is a password guessing technique in which the adversary attempts to determine a user’s password by successively trying words from a dictionary (a compiled list of likely passwords) in the hope that one of these password guesses will be the user’s actual password. This attack can be performed in online mode (trying successive passwords until a login is successful) or offline mode (hashing or encrypting a dictionary of words and looking for any matches in a copied system file of hashed or encrypted user passwords). Informally, in the scenario of PAKE protocols, we say that a protocol is secure against off-line dictionary attacks if an adversary who obtains all the communication data between the client and the server is unable to carry out the dictionary attack to obtain the client’s password. This can be achieved if and only if there is no verifiable ciphertext based on a human-memorable password in the protocol run.

- **Unknown key-share attack (UKS) resilience:** UKS is an attack where a party A believes that he shares a key with another party B upon completion of a protocol run (this is in fact the case), but B falsely believes that the key is instead shared with a party E ≠ A. A basic PAKE protocol should be resilient to this.

- **Perfect forward secrecy (PFS):** If long-term private keys or secrets of any party is compromised, the secrecy of previously established session keys should not be affected. This is an attempt to still offer some form of security guarantee in spite of the fact that the long-term secret has been leaked.

- **Key-compromise impersonation (KCI) resilience:** The compromise of any party’s (client or server) long-term key or secret should not enable the adversary to impersonate any other parties.

It is important for a security protocol, as is a PAKE protocol, to be secure not only against known types of attacks [14], [13], [15], [16], [28], [29], [30], [33], [34] including those listed above, but also be designed to resist any kind of...
attack by an adversary of some defined adversarial power. Indeed, provably secure protocols take the latter approach and provide a reasonable measure of security in the protocol design. Nevertheless, the fact that a protocol is provably secure does not preclude [11], [12], [26], [27], [31], [32] weaknesses or flaws in proofs due to oversights or subtleties in security model definitions. Thus, it is vital that even provably secure protocols are carefully analyzed to ensure they do not miss capturing resistance against any kinds of attacks.

In this paper, we contribute to this direction by showing that both the Nam et al. variant and provably secure Wen et al. variant are susceptible to key compromise impersonation (KCI) attacks [15], [9].

II. NAM ET AL. AND WEN ET AL. PROTOCOLS

We will use the notations given in Table I. Unless otherwise mentioned, all described operations are done modulo $p$, except operations in the exponents, and all protocols are based on Diffie-Hellman (DH) type assumptions.

| $A, B$ | The clients |
| $ID_i$ | The identity of party $i$ |
| $S$ | The server who stores the identity ($ID_i$) and password ($pw_i$) of client $i$ |
| $s$ | Long-term private key of $S$ |
| $ps_i$ | Equals $aS$; this is the public key of $S$ |
| $pw_i$ | Client $i$‘s human-memorable password shared with $S$ |
| $E_k(\cdot)$ | Symmetric encryption using the secret key, $k$ |
| $\mathcal{D}_k(\cdot)$ | Symmetric decryption using the secret key, $k$ |
| $H$ | Cryptographic hash function |
| $G$ | Hash function $\{0, 1\}^* \rightarrow G_1$ |
| $x \in Z_p^*$ | Randomly choosing an element $x$ of $Z_p^*$ |

Throughout this paper, $(G_1, +)$ and $(G_2, \cdot)$ denote two cyclic groups of prime order $q$. A bilinear map [23], $\hat{e} : G_1 \times G_1 \rightarrow G_2$ satisfies the following properties:

- **Bilinearity:** For all $P, Q \in G_1$ and all $a, b \in Z$, $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$.
- **Non-degeneracy:** There exists a $P \in G_1$ such that $\hat{e}(P, P) \neq 1$.
- **Computability:** There is an efficient algorithm to compute $\hat{e}(P, Q)$ for any $P, Q \in G_1$.

Since the Nam et al. protocol is an improvement of the Wen et al. one, for the rest of this paper our descriptions are based on Nam et al. though all results equally to the Wen et al. variant.

The security of the Nam et al. protocol is based on Weil Diffie-Hellman (WDH) assumption [7], [8]. This assumption states that given groups $G_1, G_2$, a pairing $\hat{e} : G_1 \times G_2 \rightarrow G_2$, and $aP, bP, cP \in G_1$ for random $a, b, c \in Z_q$, it is computationally intractable to compute $\hat{e}(P, D)^{abc} \in G_2$.

1) A selects a random number $a \in R$ $Z_p^*$ and computes $aP, Q = G(ID_S)$, and $k_a = H(ID_A, ID_B, aP, P_S, Q, \hat{e}(P_S, aQ))$. Then, A computes $c_a = \mathcal{E}_{k_a}(PW_A)$ and sends $(ID_A, aP, c_a)$ to $B$.
2) $B$ selects a random number $b \in R$ $Z_p^*$ and computes $bP, Q = G(ID_S)$, $k_b = H(ID_B, ID_A, bP, P_S, Q, \hat{e}(P_S, bQ))$, $R = G(ID_A, ID_B)$, and $K = \hat{e}(aP, bR)$. Then, $B$ computes $c_b = \mathcal{E}_{k_b}(PW_B)$ and $\mu_b = H(ID_B, K)$. $B$ sends $(ID_A, aP, c_a, ID_B, bP, c_b, \mu_b)$ to $S$.
3) Based on $ID_A, ID_B$ from the received message, the server $S$ can retrieve the passwords $PW_A, PW_B$ from the database. $S$ first computes $k_a = H(ID_A, ID_B, aP, P_S, Q, \hat{e}(aP, sQ))$ and $k_b = H(ID_B, ID_A, bP, P_S, Q, \hat{e}(P_S, bQ))$. Then, $S$ checks if $PW_A = \mathcal{D}_{k_a}(c_a)$ and $PW_B = \mathcal{D}_{k_b}(c_b)$ respectively. If not, $S$ stops executing the protocol. Otherwise, $S$ computes $\sigma_a = H(k_b, aP)$ and $\sigma_b = H(k_a, bP)$, and sends $(bP, \mu_b, \sigma_b, \sigma_a)$ to $A$.
4) $A$ first computes $R = G(ID_A, ID_B)$ and $K = \hat{e}(bP, aR)$. Then, $A$ computes and checks if $\sigma_a$ and $\mu_a$ equal to $H(k_a, bP)$ and $H(ID_B, K)$ respectively. If not, $A$ stops executing the protocol. Otherwise, $A$ computes $\mu_a = H(ID_A, K)$ and the session key $SK = H(aP, bP, R, K)$. Finally, $A$ sends $(\mu_a, \sigma_a)$ to $B$.
5) $B$ checks if $\sigma_a = H(k_b, aP)$ and $\mu_a = H(ID_A, K)$. If not, $B$ terminates the protocol. Otherwise, $B$ computes the session key $SK = H(aP, bP, R, K)$.

With this description, then the Wen et al. protocol is similar except in step (1) where $A$ computes $k_a = H(aP, P_S, Q, \hat{e}(P_S, aQ))$; step (2) where $B$ computes $k_b = H(bP, P_S, Q, \hat{e}(P_S, bQ))$; step (3) where $S$ computes $k_a = H(aP, P_S, Q, \hat{e}(aP, sQ))$ and $k_b = H(bP, P_S, Q, \hat{e}(P_S, bQ))$.

III. ON THE SECURITY OF BOTH PROTOCOLS

Recall the definition of key compromise impersonation (KCI) resilience. In more detail, when an adversary learns the long-term key $s$ of the server $S$, obviously then the server can be impersonated trivially. Resilience against KCI attacks is formulated so that some sort of security guarantee can still be afforded even when this long-term key is leaked. In particular, though it is clear that the server can be impersonated, yet KCI resilience offers the guarantee that this is the most an adversary could do, and that the adversary cannot impersonate anyone else to $S$.

We show a key compromise impersonation (KCI) attack on both Nam et al. and Wen et al. protocols that can be mounted by any adversary. In particular, when an adversary learns the long-term key $s$ of the server $S$, the adversary can impersonate anyone else to $S$, thus contradicting the KCI resilience requirement. It works as follows:

1) The message $(ID_A, aP, c_a)$ from $A$ to $B$, and similarly the message $(ID_B, bP, c_b, \mu_b)$ from $B$ to $S$ are
2) Upon compromising the long-term key $s$ of the server $S$, the adversary $C$ is thus able to compute $k_a = H(ID_A, ID_B, aP, Ps, Q, \hat{e}(P_S, aQ))$ and $k_b = H(ID_B, ID_A, bP, Ps, Q, \hat{e}(P_S, bQ))$ for Wen et al. protocol, since the only secret input to the computation of $k_a$ and $k_b$ is $s$.

3) Decrypt $c_a$ and $c_b$ with $k_a$ and $k_b$ respectively. Thus, $C$ can obtain $PW_A$ and $PW_B$.

4) $C$ can now impersonate $A$ (resp. $B$) to $S$ because authentication of $A$ (resp. $B$) to $S$ just depends on $PW_A$ (resp. $PW_B$).

IV. CONCLUSION

Wen et al. proposed three-party password-based authenticated key exchange protocol using Weil pairing [37], with security proof in the random oracle model. Nam et al. [25] showed the insecurity of the Wen et al. protocol, and proposed an improvement to counter their attack. Nevertheless, we have demonstrated that the Nam et al. improvement and the original Wen et al. protocol, both do not provide resilience to key-compromise impersonation (KCI) [9] which is nowadays commonly expected of key exchange protocols.
The problem with both protocols lies in that the client password is encapsulated with a function where the only unknown secret input is the long-term private key of the server. KCI attacks can be prevented for instance by having the password encapsulation (in this case $c_b$ or $c_b$) be a function of not only the long-term private key of the server but also a function of some ephemeral (short-term) unknown variables that are never sent in the clear to another party but instead only used locally within the context of a protocol run.

Nevertheless, we caution against adhocly fixing a protocol without a thorough re-analysis in the provable security model; thus both protocols should not be used in practical applications. Instead, we suggest to use the three-party PAKEs rigorously proven secure in the formal sense, e.g. [1], [3].

**ACKNOWLEDGMENT**

This research was supported by the Malaysia e-Science Fund (01-02-01-SF0048).

**REFERENCES**


