A Chain of Attacks and Countermeasures Applied to a Group Key Transfer Protocol

Ruxandra F. Olimid[1,2]

[1] Department of Computer Science, University of Bucharest, Romania
ruxandra.olimid@fmi.unibuc.ro
[2] Applied Cryptography Group, Orange

Abstract. Yuan et al. have recently introduced a Group Key Transfer (GKT) protocol [12] that permits multiple entities to share a common secret key. Starting from the original version of the protocol, we describe a chain of alternating attacks and countermeasures. First, we present a replay attack and indicate a possible fix, inspired by the analogous work of Nam et al. [5] (applied to the similar protocol of Harn and Lin [1]). Second, we review a successfully insider attack against the improved version that we have revealed in a previous work [6] and introduce a countermeasure that stands against the latter attack. Finally, we mention a password guessing attack inspired by the work of Kim et al. [3] that can be mounted against the original protocol and both the improved versions.

Keywords: group key transfer, insider attack, replay attack, guessing attack, cryptanalysis.

1 Introduction

A *Group Key Transfer* (GKT) protocol permits multiple entities to share a common secret key that they will subsequently use for cryptographic purposes. A privileged party called *Key Generation Center* (KGC) selects a key and securely distributes it to the participants. All parties trust the KGC to select a fresh key (a uniformly random value that has not been used before) and not disclose it to unqualified entities. Only the users within an *authorized* set should be able to recover the key, while it must remain hidden for any other party. The protocol may run for multiple times, called *sessions* and the subset of authorized members may differ for distinct executions. A user is eligible to initiate or take part in protocol sessions if he is a valid member of the group, i.e. he had previously registered to the KGC with whom he shares a long-term secret.

1.1 Related work

In the last years, many papers proposed constructions of GKT protocols based on secret sharing, a cryptographic primitive that splits a secret into multiple

parts such that only authorized sets can reconstruct it. Some examples include the protocols of: Harn and Lin [1], Hsu et al. [2], Sun et al. [11], Yuan et al. [12].

Security models consider the requirements any GKT must satisfy (such as key confidentiality, key freshness, key and entity authentication, key integrity) within a precise environment, specifying the trust assumptions, the relations between participants, the adversarial power, the communication medium and others relevant aspects. The adversary is modeled as a probabilistic polynomial time algorithm with full control over the communication channel (he can modify, delete or insert messages) that interacts with the group members by asking queries (*Execute*, *Send*, *RevealKey*, *RevealState*, *Corrupt*, *Test*). The protocols are proven secure (with respect to a security model) if the adversary wins the security games with only negligible probability. For a survey on group key establishment security models, the reader should refer to [4].

The main drawback of the mentioned constructions is that they lack a security proof, which leads to a high probability of succeeding attacks: Harn and Lin's protocol is vulnerable to replay attacks [5], Hsu et al.'s proposal is susceptible to an insider attack [9], Sun et al.'s construction is vulnerable to insider, known key and guessing attacks [3, 7]. The current work analyzes the security of Yuan et al.'s protocol.

1.2 Our contribution

We review the GKT protocol that Yuan et al. recently introduced [12]. We have mentioned in a previous work [6] its resemblance to Harn and Lin's proposal [1] and highlighted that the attack Nam et al. suggested against their protocol, as well as the proposed countermeasure [5], may also apply to Yuan et al.'s construction. However, we did not explain the analogous vulnerability or the improved version in detail. We accomplish this in the present paper.

In the same article, we have shown that the improved version remains susceptible to an insider attack [6]. Our current work introduces a countermeasure that prevents this vulnerability. We do not claim that this second improved version provides group key confidentiality (as it is not based on a formal security proof), but we only affirm that is makes the proposed insider attack useless.

Finally, we admit that both improved versions maintain the vulnerability of the original protocol against a guessing attack, inspired by the work of Kim et al. [3].

The abstract of a previous version of the current paper is available at [8].

1.3 Outline

The next section gives the preliminaries. Section 3 describes Yuan et al.'s protocol. The following sections represent a chain of alternating attacks and countermeasures: Section 4 presents a replay attack against the original protocol; Section 5 indicates a possible fix; Section 6 reveals an insider attack against the improved version; Section 7 introduces a countermeasure; Section 8 analyzes a possible guessing attack. The last section concludes.

2 Preliminaries

2.1 Notations

Let $\{U_1, \ldots, U_m\}$ be the set of all registered users to the KGC, $\{U_1, \ldots, U_t\}$, $t \leq m$ the subset of authorized participants to a given session with U_1 as initiator (after a possible reordering), (s_j), $j = 1, \ldots, 4$ four particular protocol sessions (we index by (s_j) specific values that correspond to session (s_j)) and h_j, $j = 1, \ldots, 3$ three collision-resistant hash functions.

We denote by $\leftarrow^R X$ a uniformly random choice from a specified set of values X, $\|$ string concatenation, $A \rightarrow B$ a unicast message sent by an entity A to an entity B and $A \rightarrow^*$ a broadcast message originating from A.

2.2 Adversarial Model

Key confidentiality represents one of the main properties any GKT protocol must satisfy. It ensures that it is computationally infeasible for an adversary to compute the shared secret key.

Considering group membership, the adversaries are classified into *outsiders* and *insiders*. An *outsider* has never registered to the KGC and hence he does not own a long-term secret, being unable to initiate or take part in protocol executions; an *insider* has legitimately registered to the KGC and therefore he possesses a long-term secret that gives him the ability to initiate or take part in protocol sessions.

Although many other categorizations exists, we only remind here the *replay attack*, which we will later use in this paper. It is a particular case of impersonation attack that consists in injecting messages that were eavesdropped on previous sessions of the protocol.

For the rest of the paper, we consider U_a to be an insider whose goal is to reveal the long-term password of another user U_i, $i = 1, \ldots, m$, $i \neq a$. This gives the attacker the ability to obtain the session key of all sessions U_i is authorized for (even if U_a is unauthorized for) and therefore break the confidentiality of the protocol.

3 Original Version

Yuan et al. recently introduced a password-based GKT protocol [12] based on Shamir's secret sharing scheme [10]. Fig.1 describes the protocol in detail; since it is self-explaining, we omit any other comments.

4 Replay Attack

Yuan et al.'s construction is very much alike to a protocol that Harn and Lin had been published three years before [1]. We have mentioned in a previous work that this similarity preserves a vulnerability [6]: the protocol is susceptible to

Initialization. The KGC selects 2 large primes p and q and computes $n = pq$;
Users Registration. Each user $U_i, i = 1, \ldots, m$, shares a long-term secret password $pw_i = pw_{ix}||pw_{iy}$ with the KGC;
Round 1. User U_1:
1.1. chooses $k_1 \leftarrow^R \mathbb{Z}_n$;
1.2. computes $K_1 = pw_{1x} + k_1$ and $M_1 = h_1(U_1, \ldots, U_t, k_1)$;
1.3. sends a key generation request:
$U_1 \rightarrow KGC : (U_1, \{U_1, \ldots, U_t\}, K_1, M_1)$
Round 2. The KGC:
2.1. computes $k_1 = K_1 - pw_{1x}$;
2.2. checks if $M_1 = h_1(U_1, \ldots, U_t, k_1)$;
If the equality does not hold, he quits;
2.3. broadcasts:
$KGC \rightarrow^*: \{U_1, \ldots, U_t\}$
Round 3. Each user $U_i, i = 2, \ldots, t$:
3.1. chooses $k_i \leftarrow^R \mathbb{Z}_n$;
3.2. computes $K_i = pw_{ix} + k_i$ and $M_i = h_1(U_1, \ldots, U_t, k_i)$;
3.3. sends:
$U_i \rightarrow KGC : (U_i, \{U_1, \ldots, U_t\}, K_i, M_i)$
Round 4. The KGC:
4.1. computes $k_i = K_i - pw_{ix}$, $i = 2, \ldots, t$;
4.2. checks if $M_i = h_1(U_1, \ldots, U_t, k_i)$, $i = 2, \ldots, t$;
If at least one equality does not hold, he quits;
4.3. selects 2 random numbers x_{ta} and y_{ta} of lengths equal to pw_{ix} and pw_{iy};
4.4. generates the polynomial $f(x)$ of degree t that passes through the $t + 1$ points (x_{ta}, y_{ta}), $(pw_{1x}, pw_{1y} + k_1)$, $\ldots$, $(pw_{tx}, pw_{ty} + k_t)$;
4.5. computes t additional points $P_1, \ldots, P_t$ of $f(x)$;
4.6. computes the verification messages $V_i = h_2(U_1, \ldots, U_t, P_1, \ldots, P_t, k_i)$, $i = 1, \ldots, t$;
4.7. sends, $i = 1, \ldots, t$:
$KGC \rightarrow U_i : (P_1, \ldots, P_t, V_i)$
Key Computation. Each user U_i, $i = 1, \ldots, t$:
5.1. checks if $V_i = h_2(U_1, \ldots, U_t, P_1, \ldots, P_t, k_i)$;
If the equality does not hold, he quits;
5.2. computes the group key $f(0)$ by interpolating the points $P_1, \ldots, P_t$ and $(pw_{ix}, pw_{iy} + k_i)$.

Fig. 1. Original Version of Yuan et al.'s Group Key Transfer Protocol [12]

a replay attack (mounted from inside) analogous to the one that Nam et al. mounted against Harn and Lin's proposal [5]. We now support our claim by explaining the attack in detail in Fig.2.

Step 1. U_a initiates a legitimate session of the protocol (s_1) with U_i;
Step 2. U_a eavesdrops on $(U_i, \{U_i, U_a\}, K_i, M_i)$ in Round 3 of the protocol;
Step 3. U_a initiates another legitimate session of the protocol (s_2) with U_i and uses the same value k_a for both session (s_1) and (s_2);
Step 4. U_a impersonate U_i in session (s_2) by sending in Round 3 the message $(U_i, \{U_i, U_a\}, K_i, M_i)$ he had eavesdropped in Step 2;
Step 5. U_a is an authorized user for both sessions, so he recovers the polynomials:
$$f(x)_{(s_j)} = a_{(s_j)}x^2 + b_{(s_j)}x + c_{(s_j)}, j = 1, 2$$
Step 6. Since $(pw_{ax}, pw_{ay} + k_a)$ and $(pw_{ix}, pw_{iy} + k_i)$ are valid points on $f(x)_{(s_j)}$, $j = 1, 2$, U_a knows that $f(pw_{ax})_{(s_1)} = f(pw_{ax})_{(s_2)} = pw_{ay} + k_a$ and $f(pw_{ix})_{(s_1)} = f(pw_{ix})_{(s_2)} = pw_{iy} + k_i$; therefore both pw_{ax} and pw_{ix} are roots of:
$$(a_{(s_1)} - a_{(s_2)})x^2 + (b_{(s_1)} - b_{(s_2)})x + (c_{(s_1)} - c_{(s_2)}) = 0$$
Step 7. U_a reveals the long-term password of U_i as:
$$pw_{ix} = pw_{ax}^{-1}(a_{(s_1)} - a_{(s_2)})^{-1}(c_{(s_1)} - c_{(s_2)}).$$
$$pw_{iy} = f(pw_{ix})_{(s_j)} - K_{i(s_j)} + pw_{ix}, \text{ for any } j = 1, 2$$

Fig. 2. Replay Attack against the Original Version

5 First Improved Version

The attack revealed in the previous section is possible because the KGC cannot detect replay messages. We give next a countermeasure analogous to the one that Nam et al. proposed against Harn and Lin's protocol [5]. Fig.3 exposes it in detail.

We highlight the main idea: for each session, the KGC selects a uniformly random value k_0, which he broadcasts to the participants (Round 2); then, the principals use it to compute the hash value M_i (Round 3). Since the value k_0 differs for distinct sessions, an eavesdropped value M_i in one session becomes useless for other sessions - the verification in step 4.2 fails and hence the KGC quits.

We mention a slight modification in the protocol definition: Round 1 restricts to the key generation request, while U_1 performs the other steps in Round 3 (i.e. except the initiation request, U_1 behaves similar to the rest of the users). This approach is considered in the improved version to eliminate the replay attack against the initiator, since U_1 uses the nonce k_0 to compute M_i.

6 Insider Attack

Although the first improved version stands against the replay attack mounted from inside, the protocol remains vulnerable to an insider attack [6]. Fig.4 reveals the details.

The proposed attack differs from the replay attack in the sense that U_a does not rely on a previous eavesdropped message originated from U_i; hence, U_i is genuine for all sessions (it is not impersonated anymore). On the other hand,

Initialization. The KGC selects 2 large primes p and q and computes $n = pq$;
Users Registration. Each user $U_i, i = 1, \ldots, m$, shares a long-term secret password $pw_i = pw_{ix}||pw_{iy}$ with the KGC;
Round 1. User U_1:
1.1. sends a key generation request:
$U_1 \rightarrow KGC : \{U_1, \ldots, U_t\}$
Round 2. The KGC:
2.1. chooses $k_0 \leftarrow^R \mathbb{Z}_n$;
2.2. broadcasts:
$KGC \rightarrow^*: (\{U_1, \ldots, U_t\}, k_0)$
Round 3. Each user $U_i, i = 1, \ldots, t$:
3.1. chooses $k_i \leftarrow^R \mathbb{Z}_n$;
3.2. computes $K_i = pw_{ix} + k_i$ and $M_i = h_1(U_1, \ldots, U_t, k_i, k_0)$;
3.3. sends:
$U_i \rightarrow KGC : (U_i, \{U_1, \ldots, U_t\}, K_i, M_i)$
Round 4. The KGC:
4.1. computes $k_i = K_i - pw_{ix}$, $i = 1, \ldots, t$;
4.2. checks if $M_i = h_1(U_1, \ldots, U_t, k_i, k_0)$, $i = 1, \ldots, t$;
If at least one equality does not hold, he quits;
4.3. selects 2 random numbers x_{ta} and y_{ta} of lengths equal to pw_{ix} and pw_{iy};
4.4. generates the polynomial $f(x)$ of degree t that passes through the $t + 1$ points (x_{ta}, y_{ta}), $(pw_{1x}, pw_{1y} + k_1)$, ... , $(pw_{tx}, pw_{ty} + k_t)$;
4.5. computes t additional points $P_1, \ldots, P_t$ of $f(x)$;
4.6. computes the verification messages $V_i = h_2(U_1, \ldots, U_t, P_1, \ldots, P_t, k_i, k_0)$, $i = 1, \ldots, t$;
4.7. sends, $i = 1, \ldots, t$:
$KGC \rightarrow U_i : (P_1, \ldots, P_t, V_i)$
Key Computation. Each user U_i, $i = 1, \ldots, t$:
5.1. checks if $V_i = h_2(U_1, \ldots, U_t, P_1, \ldots, P_t, k_i, k_0)$;
If the equality does not hold, he quits;
5.2. computes the group key $f(0)$ by interpolating the points $P_1, \ldots, P_t$ and $(pw_{ix}, pw_{iy} + k_i)$.

Fig. 3. First Improved Version of Yuan et al.'s Group Key Transfer Protocol

it requires four sessions between the adversary and the victim. It is a natural assumption to consider that the protocol allows multiple sessions between the same parties. However, if it is considered suspicious that a single user initiates the protocol multiple times with the same other participant, a coalition of insiders may mount the attack: each attacker initializes a different session with the victim U_i and finally they cooperate to disclose the long-term key password $pw_{ix}||pw_{iy}$.

We remark that after the long-term secret password is revealed, an impersonation attack is immediate: the adversary U_a uses (pw_{ix}, pw_{iy}) to pretend his identity is P_i.

Step 1. U_a initiates (s_j), $j = 1, \ldots, 4$, four legitimate sessions of the protocol with U_i;
Step 2. U_a is an authorized user for all sessions, so he recovers the polynomials:

$$f(x)_{(s_j)} = a_{(s_j)}x^2 + b_{(s_j)}x + c_{(s_j)}, j = 1, \ldots, 4$$

Step 3. Since $(pw_{ix}, pw_{iy} + k_{i(s_j)})$ are valid points on $f(x)_{(s_j)}$, U_a obtains:

$$pw_{iy} + k_{i(s_j)} = a_{(s_j)}pw_{ix}^2 + b_{(s_j)}pw_{ix} + c_{(s_j)},\ j = 1, \ldots, 4$$

Step 4. U_a eavesdrops on $K_{i(s_j)}$, knows that $k_{i(s_j)} = K_{i(s_j)} - pw_{ix}$ and acquires:

$$pw_{iy} = a_{(s_j)}pw_{ix}^2 + (b_{(s_j)} + 1)pw_{ix} + c_{(s_j)} - K_{i(s_j)}, j = 1, \ldots, 4$$

Step 5. U_a eliminates pw_{iy} from the first two equalities ($j = 1, 2$), respectively from the last two equalities ($j = 3, 4$) and gets:

$$A_{(s_1s_2)}pw_{ix}^2 + B_{(s_1s_2)}pw_{ix} + C_{(s_1s_2)} = 0$$
$$A_{(s_3s_4)}pw_{ix}^2 + B_{(s_3s_4)}pw_{ix} + C_{(s_3s_4)} = 0$$

where:

$A_{(s_1s_2)} = a_{(s_1)} - a_{(s_2)}$ $\quad$ $A_{(s_3s_4)} = a_{(s_3)} - a_{(s_4)}$
$B_{(s_1s_2)} = b_{(s_1)} - b_{(s_2)}$ $\quad$ $B_{(s_3s_4)} = b_{(s_3)} - b_{(s_4)}$
$C_{(s_1s_2)} = c_{(s_1)} - c_{(s_2)} - (K_{i(s_1)} - K_{i(s_2)})$ $\quad$ $C_{(s_3s_4)} = c_{(s_3)} - c_{(s_4)} - (K_{i(s_3)} - K_{i(s_4)})$

Step 6. U_a reveals the long-term password of U_i as:

$$pw_{ix} = (A_{(s_1s_2)}C_{(s_3s_4)} - A_{(s_3s_4)}C_{(s_1s_2)})(A_{(s_3s_4)}B_{(s_1s_2)} - A_{(s_1s_2)}B_{(s_3s_4)})^{-1}$$
$$pw_{iy} = f(pw_{ix})_{(s_j)} - K_{i(s_j)} + pw_{ix}, \text{ for any } j = 1, \ldots, 4$$

Fig. 4. Insider Attack against the First Improved Version [6]

7 Second Improved Version

The insider attack becomes possible because pw_{iy} can be expressed as the value of a polynomial with known coefficients in pw_{ix}. This permits the attacker to replace pw_{iy} and obtain a system of equations with the single unknown pw_{ix}. Fig.5 introduces a countermeasure.

We emphasize the main idea: the KGC generates a polynomial $f(x)$ that passes through $(pw_{ix}, h_3(U_1, \ldots, U_t, pw_{iy}, k_i, k_0))$ instead of $(pw_{ix}, pw_{iy} + k_i)$, $i = 1, \ldots, t$ (Round 4). This leads to the futility of the attack, since the argue fails due to the new form of the equations in Step 3:

$$h_3(U_1, \ldots, U_t, pw_{iy}, k_{i(s_j)}, k_0)) = a_{(s_j)}pw_{ix}^2 + b_{(s_j)}pw_{ix} + c_{(s_j)},\ j = 1, \ldots, 4.$$

8 Guessing Attack

The second improved version maintains a vulnerability of the original protocol: it is susceptible to a guessing attack, similar to the one that Kim et al. [3] introduced for Sun et al.'s protocol [11]. Fig.6 explains the attack in detail.

The attack is successful under the assumption that both the attacker and the victim are authorized group members for at least one session. Unlike the previous attacks, U_a and U_i are not restricted to be the only authorized members; hence the exposure of the adversary decreases: he may use an already existing session with multiple participants.

Initialization. The KGC selects 2 large primes p and q and computes $n = pq$;
Users Registration. Each user $U_i, i = 1, \ldots, m$, shares a long-term secret password $pw_i = pw_{ix}||pw_{iy}$ with the KGC;
Round 1. User U_1:
1.1. sends a key generation request:
$U_1 \rightarrow KGC : \{U_1, \ldots, U_t\}$
Round 2. The KGC:
2.1. chooses $k_0 \leftarrow^R \mathbb{Z}_n$;
2.2. broadcasts:
$KGC \rightarrow^*: (\{U_1, \ldots, U_t\}, k_0)$
Round 3. Each user $U_i, i = 1, \ldots, t$:
3.1. chooses $k_i \leftarrow^R \mathbb{Z}_n$;
3.2. computes $K_i = pw_{ix} + k_i$ and $M_i = h_1(U_1, \ldots, U_t, k_i, k_0)$;
3.3. sends:
$U_i \rightarrow KGC : (U_i, \{U_1, \ldots, U_t\}, K_i, M_i)$
Round 4. The KGC:
4.1. computes $k_i = K_i - pw_{ix}$, $i = 1, \ldots, t$;
4.2. checks if $M_i = h_1(U_1, \ldots, U_t, k_i, k_0)$, $i = 1, \ldots, t$;
If at least one equality does not hold, he quits;
4.3. selects 2 random numbers x_{ta} and y_{ta} of lengths equal to pw_{ix}, respectively h_3 hash values;
4.4. generates the polynomial $f(x)$ of degree t that passes through the $t + 1$ points (x_{ta}, y_{ta}), $(pw_{1x}, h_3(U_1, \ldots, U_t, pw_{1y}, k_1, k_0))$, ... , $(pw_{tx}, h_3(U_1, \ldots, U_t, pw_{ty}, k_t, k_0))$;
4.5. computes t additional points $P_1, \ldots, P_t$ of $f(x)$;
4.6. computes the verification messages $V_i = h_2(U_1, \ldots, U_t, P_1, \ldots, P_t, k_i, k_0)$, $i = 1, \ldots, t$;
4.7. sends, $i = 1, \ldots, t$:
$KGC \rightarrow U_i : (P_1, \ldots, P_t, V_i)$
Key Computation. Each user U_i, $i = 1, \ldots, t$:
5.1. checks if $V_i = h_2(U_1, \ldots, U_t, P_1, \ldots, P_t, k_i, k_0)$;
If the equality does not hold, he quits;
5.2. computes the group key $f(0)$ by interpolating the points $P_1, \ldots, P_t$ and $(pw_{ix}, h_3(U_1, \ldots, U_t, pw_{iy}, k_i, k_0))$.

Fig. 5. Second Improved Version of Yuan et al.'s Group Key Transfer Protocol

The vulnerability is caused by a password guessing attack: U_a eavesdrops (or computes) the hash value, then guesses the unknown bits of the password in the input. More precise: in step 4, U_a knows K_i, M_i, $\{U_1, \ldots, U_t\}$ and k_0 and tries to determine pw_{ix} such that $k_i = K_i - pw_{ix}$ and $M_i = h_1(U_1, \ldots, U_t, k_i, k_0)$; in step 6, U_a knows $f(pw_{iy}) = h_3(U_1, \ldots, U_t, pw_{iy}, k_i, k_0)$ and all inputs except pw_{iy}, which he guesses. The adversary performs the guessing offline (trivially by a dictionary attack) and succeeds if he can reveal the long-term password of the victim before it expires. We highlight that the attack succeeds with high

Step 1. U_a is an authorized participant to the session, so he knows $\{U_1, \ldots, U_t\}$ and k_0 from Round 2;
Step 2. U_a eavesdrops on K_i and M_i in Round 3;
Step 3. U_a eavesdrops on $\{P_1, \ldots, P_t\}$ and V_i in Round 4;
Step 4. U_a obtains pw_{ix} by launching a password guessing attack on pw_{ix}: for all probable values of pw_{ix}, he computes $k_i = K_i - pw_{ix}$ and checks that $M_i = h_1(U_1, \ldots, U_t, k_i, k_0)$ or $V_i = h_2(U_1, \ldots, U_t, P_1, \ldots, P_t, k_i, k_0)$.
Step 5. U_a is an authorized user for the session, so he legitimate computes the coefficients of $f(x)$ and then the value $f(pw_{ix})$;
Step 6. U_a discloses pw_{iy} by launching a guessing attack on $f(pw_{ix}) = h_3(U_1, \ldots, U_t, pw_{iy}, k_i, k_0)$.

Fig. 6. Guessing Attack against the Second Improved Version

probability because users tend to use simple or common passwords that are susceptible to dictionary attacks.

U_a must correctly guess the whole password for the attack to work: the first half pw_{ix} in step 4 and the last half pw_{iy} in step 7. We remark that a similar attack against the original protocol or the first improved version requires only the guessing of pw_{ix} (afterwards, in step 7, U_a can directly compute $pw_{iy} = f(pw_{ix}) - k_i$). However, this does not imply a better security: standing against password guessing is the main feature that a password-based protocol requires; otherwise the security of the established key is upper bounded by the security of the password.

9 Conclusions

The paper reviews Yuan et al. GKT protocol [12] and considers a chain of three attacks and two corresponding improvements, starting from the original version.

We emphasize that Yuan et al.'s original construction and the improved versions skip formal security proofs, which makes them easily susceptible to known attacks. We highlight the necessity of security proofs for practical GKT protocols.

References

1. Harn L., Lin C.: Authenticated Group Key Transfer Protocol based on Secret Sharing. IEEE Trans. Comput. vol.59(6), pp. 842–846 (2010).
2. Hsu,C., Zeng, B., Cheng, Q., Cui, G.: A novel group key transfer protocol. Cryptology ePrint Archive, Report 2012/043, (2012)
3. Kim M., Park N., Won D.: Cryptanalysis of an Authenticated Group Key Transfer Protocol Based on Secret Sharing. Grid and Pervasive Computing, pp. 761–766 (2013).

4. Manulis M.: Survey on security requirements and models for group key exchange. Technical Report 2006/02, Horst-Görtz Institute, Network and Data Security Group (2008).
5. Nam J., Kim M., Paik J, Jeon W., Lee B., Won D.: Cryptanalysis of a Group Key Transfer Protocol based on Secret Sharing. Proceedings of the 3rd International Conference on Future Generation Information Technology, pp. 309–315 (2011).
6. Olimid R.F.: Cryptanalysis of a Password-based Group Key Exchange Protocol Using Secret Sharing. Appl. Math. Inf. Sci. vol.7(4), pp.1585–1590 (2013).
7. Olimid R.F.: On the Security of an Authenticated Group Key Transfer Protocol based on Secret Sharing. Proceedings of ICT-EurAsia, pp.399–408 (2013).
8. Olimid R.F.: A Chain of Attacks and Countermeasures Applied to a Group Key Transfer Protocol (abstract). Pre-Proceedings of WEWoRC, pp.27–28 (2013).
9. Olimid R.F.: On the Vulnerability of a Group Key Transfer Protocol based on Secret Sharing. to appear in Proceeding of IEEE 9th International Symposium on Applied Computational Intelligence and Informatics (2014).
10. Shamir A.: How to Share a Secret. Commun. ACM vol.22(11), pp. 612–613 (1979).
11. Sun Y., Wen Q., Sun H., Li W., Jin Z., Zhang H.: An Authenticated Group Key Transfer Protocol Based on Secret Sharing Procedia Engineering, vol.29(0), pp.403–408 (2012).
12. Yuan W., Hu L., Li H., Chu J.: An Efficient Password-based Group Key Exchange Protocol Using Secret Sharing. Appl. Math. Inf. Sci. vol.7(1), pp.145–150 (2013).