Cryptanalysis of a Group Key Transfer Protocol Based on Secret Sharing: Generalization and Countermeasures

Kallepu Raju [*], Appala Naidu Tentu [†], V. Ch. Venkaiah [‡]

Abstract: Group key distribution protocol is a mechanism in which a group key is generated and distributed by KGC to a set of communicating parties in a group. This group key generally ensures secure communication among communicating parties in an unsecure channel. Harn and Lin protocol is one such. It is based on Shamir's secret sharing scheme. Nam et al. exposed the vulnerability in Harn and Lin protocol through their replay attack and proposed a countermeasure using nonce mechanism. In this paper, we are generalizing the replay attack proposed by Nam et al. and proposing an alternative countermeasure without using nonce mechanism. Novelty of our countermeasure is that KGC is not required to detect replay messages and hence each user doesn't need to compute authentication message as in Nam et al. Proposed countermeasure thereby brings down the computational complexity of the scheme.

1 Introduction

Along with the rapid development of distributed networks and E-commerce, group-oriented applications have become increasingly popular. When group members want to establish a secure communication over an open network, they must run a group key establishment protocol to set up a common session key that encrypts their communication data. Over the last decades, many group key establishment protocols have been proposed. According to [1], group key establishment protocols are classified into two types: group key distribution protocols and group key agreement protocols. In key agreement protocols, all authorized users are involved to determine the session keys. The most commonly used key agreement protocol is Diffie-Hellman(DH) key agreement protocol [2]. In DH protocol, the session key is determined by exchanging public keys of two users. Most

[*]School of Computer and Information Sciences (SCIS), University of Hyderabad, C R Rao Road, Gachibowli, Hyderabad, 500046, India, **E-mail: kallepuraju91@gmail.com**

[†]C R Rao Advanced Institute of Mathematics, Statistics, and Computer Science(AIMSCS), University of Hyderabad Campus, C R Rao Road, Gachibowli, Hyderabad 500046, India, **E-mail: naidunit@gmail.com**

[‡]**corresponding author**: School of Computer and Information Sciences (SCIS), University of Hyderabad, C R Rao Road, Gachibowli, Hyderabad, 500046, India & C R Rao Advanced Institute of Mathematics, Statistics, and Computer Science(AIMSCS), University of Hyderabad Campus, C R Rao Road, Gachibowli, Hyderabad, 500046, India, **E-mail: venkaiah@hotmail.com, vvcs@uohyd.ernet.in**

of the key agreement protocols take natural generalization of DH protocol. There are some other protocols based on non-DH key agreement approach as well. In group key distribution protocols, a trusted third party(KGC) chooses a session key then securely distributes it to all authorized group members. Most often, KGC encrypts session keys under another secret key shared with each entity during registration. A group key distribution protocol is said to be secure when it allows only authorized users to reconstruct a group key in a session.

In 2010, Harn and Lin [4] proposed an authenticated group key transfer protocol based on Shamir's secret sharing scheme [9]. In this protocol KGC shares a random long term secret with each user during registration phase then KGC selects a group key and broadcasts the group key information to all the members in the session. All users in the session reconstruct the group key using group key information and their long term secrets. With Shamir's (t,n) threshold scheme [9], any authorized group member can obtain the group key, and unauthorized users cannot learn anything about the group key. They claim that their protocol can withstand the insider attack i.e any malicious user won't be able to disclose the long term secret of other users. In 2013, Liu et al.[6] proved that Harn and Lin protocol[4] can not resist insider attack and proposed an improved authenticated group key transfer protocol based on secret sharing. They stated that their protocol can resist both insider and outsider attacks. In 2013, Yuan et al.[10] proposed a man-in-the-middle attack against the Harn and Lin protocol [4] and proposed a countermeasure using RSA signatures [8]. They improved the Harn and Lin protocol by incorporating RSA signatures into it.

In 2011, Nam et al.[7] proved that Harn and Lin protocol [4] can not resist insider attack by proposing a replay attack against their protocol. Liu et al's improved protocol [6] and Yuan et al's improved protocol [10] are also can not resist Nam et al's replay attack. Nam et al. proposed a countermeasure to prevent their replay attack. Their countermeasure helps KGC to detect replay messages from users. To ensure this, each user computes an authentication message using his/her long term secret and random integer sent by KGC.

In this paper, we generalize the replay attack proposed by Nam et al. and propose an alternative countermeasure without using nonce mechanism. Novelty of our countermeasure is that the KGC is not required to detect replay messages and hence each user doesn't need to compute authentication message; thereby bringing down the computational complexity of the scheme.

In 2012, Guo and Chang [3] proposed a group key distribution protocol similar to Harn and Lin protocol. Instead of Lagrange's interpolation polynomial they used generalized Chinese remainder theorem to reduce the communication cost. In 2013, Liu et al.[5] exposed the security problems in Guo and chang group key distribution protocol and proposed a simpler authenticated group key distribution protocol based on Chinese remainder theorem.

Organization of the paper is as follows: Section 2 describes the Harn and Lin's key transfer protocol, Section 3 describes the Replay attack and its countermeasure, Section

4 describes the proposed generalization of the replay attack, Section 5 describes the proposed countermeasure and improved Harn and Lin protocol and Conclusions are given in Section 6.

2 Harn and Lin's Group Key Transfer Protocol

The Harn and Lin protocol [4] consists of three phases: initialization of KGC, user registration and, group key generation and distribution phases.

Initialization of KGC. The KGC randomly selects two large safe primes p and q (i.e $p' = \frac{p-1}{2}$ and $q' = \frac{q-1}{2}$ are also primes) and computes n=pq. n is made publicly known.

User Registration. Each user is required to register at KGC to subscribe to the key distribution service. The KGC keeps track of all the registered users and removes any unsubscribed users. During the registration, KGC shares a long-term secret (x_i, y_i) with each user U_i, $1 \leq i \leq m$, where $(x_i, y_i) \in Z_n^* \times Z_n^*$.

Group Key Generation and Distribution. After receiving a group key generation request from any user, KGC randomly chooses a group key and distributes this group key to all the group members in a secure and authenticated way as follows: Let a session be consisting of t members, $\{U_1, \cdots, U_t\}$ whose long term secrets shared with the KGC are (x_i, y_i), $1 \leq i \leq t$. The key generation and distribution process for this session consists of the following five steps:
Step 1. The initiator sends a key generation request to KGC with a list of group members as $\{U_1, \cdots, U_t\}$.

Step 2. KGC broadcasts the participants list $\{U_1, \cdots, U_t\}$ to all U_i, $1 \leq i \leq t$, as a response to the request.

Step 3. Each user U_i, $1 \leq i \leq t$, sends a random challenge $R_i \in Z_n^*$ to KGC.

Step 4. KGC randomly selects a session key k and constructs a t^{th} degree interpolation polynomial f(x) passing through (t + 1) points: $\{(0, k), (x_1, y_1 \oplus R_1), \cdots, (x_t, y_t \oplus R_t)\}$. Next, KGC selects t additional points $P_1, \cdots, P_t$ that lie on the polynomial f(x). KGC then computes the authentication message Auth=$h(k, U_1, \cdots, U_t, R_1, \cdots, R_t, P_1, \cdots, P_t)$, where h is a one-way hash function, and broadcasts $(R_1, \cdots, R_t, P_1, \cdots, P_t, Auth)$ to all users participated in session. All computations with respect to f(x) are performed modulo n.

Step 5. Each U_i, $1 \leq i \leq t$, constructs the polynomial f(x) from the (t+1) points: $P_1, \cdots, P_t$ and $(x_i, y_i \oplus R_i)$. Then U_i recovers the session key k = f(0) and checks the correctness by calculating $h(k, U_1, \cdots, U_t, R_1, \cdots, R_t, P_1, \cdots, P_t)$. U_i aborts if the equality does not hold.

3 Replay Attack and Its Countermeausre

In replay attack, adversary's main goal is to get the long term secret of a targeted user. If so, he can reconstruct the group key of any session in which the targeted user is authorized and the adversary is unauthorized. Harn and Lin considered this attack but stated that the scheme is resistant to this attack saying that solving for the long term secret is equivalent to factoring the n, which is chosen as a product of two safe primes. Nam et al. proved that their claim is wrong by slightly modifying the attack. This modified attack is as follows :

Attack-1.
Suppose U_a is an adversary whose main goal is to get the long-term secret (x_i, y_i) of a targeted user U_i, where $1 \leq i \leq m$, $1 \leq a \leq m$ (where m is the number of registered users at registration phase) and $i \neq a$. U_a can achieve his goal by mounting the following attack against the Harn and Lin protocol.

Round 1. The adversary U_a initiates two legitimate concurrent sessions S_1 and S_2 of the protocol and sends $\{U_a,\ U_i\}$ list as group key generation request to KGC in both sessions. But U_i doesn't know that he is one of the participants in both these sessions.

Round 2. The KGC broadcasts the $\{U_a,\ U_i\}$ list as response to all the participated users in both sessions but U_a prevents it from reaching U_i.

Round 3.In this round U_a plays dual role as himself and targeted user U_i.
In S_1, U_a selects two random integers $R_a \in Z_n^*$ and $R_i \in Z_n^*$, then U_a sends R_a to KGC as its own challenge and sends R_i to KGC as U_i's challenge.

In S_2, U_a replays the challenges R_a and R_i i.e. U_a sends R_a to KGC as its own challenge and sends R_i as U_i's challenge.

Round 4.
In S_1, after receiving R_a and R_i messages, the KGC randomly selects a random session key $k \in Z_n^*$ and constructs a 2nd degree interpolation polynomial $f(x) = a_2x^2 + a_1x + a_0$ passing through the 3 points: $(0, k)$,$(x_a, y_a \oplus R_a)$,$(x_i, y_i \oplus R_i)$. Next, KGC selects 2 additional points P_1, P_2 that lie on the polynomial f(x). KGC then computes the authentication message $Auth = h(k, U_a, U_i, R_a, R_i, P_1, P_2)$, and broadcasts the group key information $(R_a, R_i, P_1, P_2, Auth)$ to users U_a,U_i but U_a prevents it from reaching U_i .

In S_2, KGC follows the procedure same as in S_1. It constructs a 2nd degree polynomial $f'(x) = a'_2x^2 + a'_1x + a'_0$ passing through the 3 points: $(0, k')$,$(x_a, y_a \oplus R_a)$,$(x_i, y_i \oplus R_i)$. Then KGC broadcasts $(R_a, R_i, P'_1, P'_2, Auth')$ to users U_a,U_i but U_a prevents it from reaching U_i, where $Auth'$ is $h(k', U_a, U_i, R_a, R_i, P'_1, P'_2)$.

Round 5.
In S_1, After receiving the group key information in Round 4, adversary U_a constructs a polynomial $f(x) = a_2x^2 + a_1x + a_0$ passing through 3 points: P_1, P_2 and $(x_a, y_a \oplus R_a)$. To recover the session key k, U_a substitutes 0 in f(x), then polynomial becomes
$f(x) = a_2x^2 + a_1x + k$, where $k = a_0$.

In S_2, U_a follows the procedure same as in S_1 and constructs the polynomial
$f'(x) = a'_2x^2 + a'_1x + a'_0$ passing through the 3 points: P'_1, P'_2 and $(x_a, y_a \oplus R_a)$. After substituting 0 in polynomial $f'(x)$, it becomes $a'_2x^2 + a'_1x + k'$, where $k' = a'_0$.

In order to find the x_i of targeted user U_i, the adversary U_a forms the polynomial $g(x)$ as

$$\begin{aligned} g(x) &= f(x) - f'(x) \\ &= (a_2 - a'_2)x^2 + (a_1 - a'_1)x + (k - k') \end{aligned}$$

Note that x_a and x_i are the roots of $g(x)$. This is because $f(x_a) = f'(x_a) = y_a \oplus R_a$ and $f(x_i) = f'(x_i) = y_i \oplus R_i$.

Therefore,

$$x_i = {x_a}^{-1}(k - k')(a_2 - a'_2)^{-1}$$

Once adversary knows the x_i , he can easily compute the y_i of U_i by substituting x_i value in $f(x)$ or $f'(x)$.

$$f(x_i) = y_i \oplus R_i$$

The value of y_i varies and depends on whether $y_i \oplus R_i < n$ or $y_i \oplus R_i \geqslant n$.

$$y_i = f(x_i) \oplus R_i \qquad \text{if } y_i \oplus R_i < n \; ;$$

$$y_i = (f(x_i) \text{ mod n}) \oplus R_i \qquad \text{otherwise.}$$

Attack-2.
In case if the KGC does not allow the two party key establishment in Harn and Lin protocol, then the above Attack 1 is not possible. But, Nam et al. proposed that there is another possibility of replay attack. In this attack, adversary U_a can collude with another malicious user U_j to know the long term secret of targeted user U_i. U_a initiates two concurrent sessions as in the Round 1 of Attack-1, whereas the participants of both sessions are $\{U_a, U_i, U_j\}$. If U_a and U_j collude together and run the two sessions as in the above Attack1, they can construct a cubic polynomial

$$g(x) = (a_3 - a'_3)x^3 + (a_2 - a'_2)x^2 + (a_1 - a'_1)x + (k - k')$$

such that $g(x_a) = g(x_i) = g(x_j) = 0$ which implies x_a, x_i and x_j are the roots of the cubic equation $(a_3 - a'_3)x^3 + (a_2 - a'_2)x^2 + (a_1 - a'_1)x + (k - k') = 0$. Adversary already knows the x_a and x_j , he can compute x_i as

$$x_i = (-1)(x_ax_j)^{-1}(k - k')(a_3 - a'_3)^{-1}$$

Once adversary knows the x_i, he can compute y_i as in Attack 1.

Countermeasure.
Countermeasure-1. The above replay attack is possible because the KGC can not detect the replay messages. That is, in Round 3 of Attack-1, though the adversary U_a is sending the same challenges R_a, R_i to KGC in both the sessions, KGC was unable to detect those reply messages. This made it possible for attacker to compute a polynomial g(x) such that x_a,x_i are roots to g(x). In order to prevent this replay attack Nam et al. proposed a countermeasure so that KGC can detect the replay messages. They modified step 2 and step 3 of Harn and Lin protocol to prevent the replay messages. The modified steps are as follows:

Step 2. The KGC selects a random integer R_0 and broadcasts it to all participants along with the participants list $\{U_1, \cdots, U_t\}$.

Step 3. Each user U_i, for $1 \leq i \leq t$, selects a random integer $R_i \in Z_n^*$ and computes $Auth_i$=$h(x_i, y_i, R_0, R_i, U_1, \cdots, U_t)$ and sends $(Auth_i, R_i)$ to KGC.

Then in Step 4 of protocol, the KGC checks the correctness of $Auth_i$ by computing $h(x_i, y_i, R_0, R_i, U_1, \cdots, U_t)$. KGC aborts if the equality of atleast one user does not hold. This counters the replay attack as $Auth_i$ will be different for different sessions.

Here, we generalize the above Attack-2 to t number of users where any $t-1$ malicious users can collude to know the longterm secret of the targeted user. The possible attack is explained below:

4 Proposed Generalization of the Replay Attack

Suppose $\{U_1, \cdots, U_t\}$ are the t users participating in a session S. In S, U_1 is the adversary(insider), he wants to know the long term secret of a targeted user U_t so he colludes with all other malicious users in that session except U_t.

Round 1. The adversary U_1 initiates two legitimate concurrent sessions S_1 and S_2 of the protocol and sends $\{U_1, U_2, \cdots, U_t\}$ list as group key generation request to KGC in both sessions. But U_t doesn't know that he is one of the participant in both these sessions.

Round 2. The KGC broadcasts the $\{U_1, U_2, \cdots, U_t\}$ list as response to all the participated users in both sessions but U_1 prevents it from reaching U_t.

Round 3. In this round U_1 plays dual role as himself and targeted user U_t.
In S_1, U_1 selects two random integers $R_1 \in Z_n^*$ and $R_t \in Z_n^*$, then U_1 sends R_1 to KGC as its own challenge and sends R_t to KGC as U_t's challenge. For $2 \leq i \leq t-1$, each user U_i selects random challenge $R_i \in Z_n^*$ and sends it to KGC

In S_2, U_1 replays the challenges R_1 and R_t i.e. U_1 sends the R_1 to KGC as its own challenge and sends R_t as U_t's challenge. Remaining all users U_i, $2 \leq i \leq t-1$ sends the previously selected challenge R_i in S_1 to KGC as their challenge.

Round 4.
In S_1, after receiving R_1,R_2,$\cdots$,R_t challenges, the KGC randomly selects a session key k and constructs a t^{th} degree interpolation polynomial

$$f(x) = a_t x^t + a_{t-1}x^{t-1} + a_{t-2}x^{t-2} + \cdots + a_2x^2 + a_1x^1 + a_0$$

passing through the t points: $(0, k), (x_1, y_1 \oplus R_1), (x_2, y_2 \oplus R_2), \cdots , (x_t, y_t \oplus R_t)$. Next, KGC selects t additional points $P_1, P_2, \cdots , P_t$ that lie on the polynomial f(x). KGC then computes the authentication message $Auth = h(k, U_1, \cdots , U_t, R_1, \cdots , R_t, P_1, \cdots , P_t)$, and broadcasts the group key information $(R_1, \cdots , R_t, P_1, \cdots , P_t, Auth)$ to all users but U_1 prevents it from reaching U_t .

In S_2, KGC follows the procedure same as in S_1. It constructs a t^{th} degree polynomial $f'(x) = a'_t x^t + a'_{t-1}x^{t-1} + a'_{t-2}x^{t-2} + \cdots + a'_2x^2 + a'_1x^1 + a'_0$ passing through the t points: $(0, k'), (x_1, y_1 \oplus R_1), (x_2, y_2 \oplus R_2), \cdots , (x_t, y_t \oplus R_t)$. Then KGC broadcasts $(R_1, \cdots , R_t, P'_1, \cdots , P'_t, Auth')$ to all participated users ,where $Auth'$ is $h(k', U_1, \cdots , U_t, R_1 , \cdots , R_t, P'_1, \cdots , P'_t)$. But U_1 prevents the group key information reaching U_t

Round 5.
In S_1, after receiving the group key information in Round 4, adversary U_1 constructs a polynomial $f(x) = a_t x^t + a_{t-1}x^{t-1} + a_{t-2}x^{t-2} + \cdots + a_2x^2 + a_1x^1 + a_0$ passing through t+1 points: $P_1, P_2, \cdots , P_t$ and $(x_1, y_1 \oplus R_1)$. To recover the session key k, U_1 substitutes 0 in computed f(x), then polynomial becomes

$$f(x) = a_t x^t + a_{t-1}x^{t-1} + a_{t-2}x^{t-2} + \cdots + a_2x^2 + a_1x^1 + k$$

, where $k = a_0$.

In S_2, U_1 follows the procedure same as in S_1 and constructs the polynomial

$$f'(x) = a'_t x^t + a'_{t-1}x^{t-1} + a'_{t-2}x^{t-2} + \cdots + a'_2x^2 + a'_1x^1 + a'_0$$

passing through the $t + 1$ points: $P'_1, P'_2, \cdots , P'_t$ and $(x_1, y_1 \oplus R_1)$. After substituting 0 in polynomial $f'(x)$, it becomes $a'_t x^t + a'_{t-1}x^{t-1} + a'_{t-2}x^{t-2} + \cdots + a'_2x^2 + a'_1x^1 + k'$, where $k' = a'_0$.

In order to find the x_t of targeted user U_t, The adversary U_1 forms the polynomial g(x) such that

$$\begin{aligned} g(x) &= f(x) - f'(x) \\ &= (a_t - a'_t)x^t + (a_{t-1} - a'_{t-1})x^{t-1} + \cdots + (a_2 - a'_2)x^2 + (a_1 - a'_1)x + (k - k') \end{aligned}$$

The roots of this polynomial are x_1,x_2,$\cdots$,x_t. This is because $f(x_i) = f'(x_i) = y_i \oplus R_i$, for $1 \leq i \leq t$.

Therefore,

$$x_t = (x_1 x_2 x_3 \cdots x_{t-1})^{-1}(k - k')(a_{t-1} - a'_{t-1})^{-1} \text{ ..... (1).}$$

Once adversary knows the x_t , he can easily compute the y_t of victim U_t as in the Attack-1.

Note: To compute x_t using $Eq.(1)$, the adversary needs x_i value of each user U_i, $2 \leq i \leq t-1$. But $\{U_2, \cdots, U_{t-1}\}$ may not be willing to share their long term secret x_i, as this allows the adversary to compromise the security of the participants. This problem can be overcome by the adversary U_1 as follows: Initially adversary U_1 multiplies his x_1 with some random integer $R \in Z_n^*$ then forwards the result to U_2. Then U_2 multiplies it with his x_2 and forwards it to the next user. Similarly all the users in the session except victim multiplies their x_i with the value sent by its previous user. The last user then sends it to the adversary who divides the resultant value with R. This way adversary can get the $x_1 x_2 x_3 \cdots x_{t-1}$ and substitutes it in the above $Eq.(1)$.

In countermeasure-1, Nam et al. prevented the replay attack by making the KGC to detect replay messages in Step 4 of the Harn and Lin protocol. To do that, each user has to compute a authentication message i.e. $Auth_i$=$h(x_i, y_i, R_0, R_i, U_1, \cdots, U_t)$ in Step 3 of protocol and sends $(Auth_i, R_i)$ to KGC. But computing authentication message by each user will increase the computational complexity of the scheme and if the number of participants increase, computational cost will increase more. The following countermeasure that we hereby propose is an alternative to Countermeasure-1 and which will reduce the computational complexity of the scheme. In our countermeasure, KGC is not required to detect the replay messages to prevent the replay attack and hence each user doesn't need to compute authentication message.

5 Proposed Countermeasure

Countermeasure-2. The above replay attack is possible because adversary was able to compute a t^{th} degree polynomial g(x) by sending replay messages. The roots of g(x) are x_i of user U_i, for $1 \leq i \leq t$. As the degree of the polynomial is equal to the number of roots, adversary can easily compute x_i of targeted user using the product of the roots of a polynomial. To prevent this attack, we made following modifications to the step 4 and step 5 of original Harn and Lin protocol.

Step 4. The KGC randomly selects a session key k and constructs a $(t+1)^{th}$ degree interpolation polynomial f(x) passing through the (t + 2) points: $(0, k)$,$(x_1, y_1 \oplus R_1)$,$\cdots$,$(x_t, y_t \oplus R_t)$ and a random coordinate $(x_r, y_r \oplus R_r)$, where $(x_r, y_r) \in Z_n^* \times Z_n^*$ and $R_r \in Z_n^*$. Next, KGC selects (t+1) additional points $P_1, \cdots, P_t, P_{t+1}$ that lie on the polynomial f(x). KGC then computes the authentication message Auth=$h(k, U_1, \cdots, U_t, R_1, \cdots, R_t, P_1, \cdots, P_t, P_{t+1})$, where h is a one-way hash function and broadcasts $(R_1, \cdots, R_t, P_1, \cdots, P_t, P_{t+1}, Auth)$ to all users participating in the session. All computations with respect to f(x) are performed modulo n.

Step 5. Each U_i, for $1 \leq i \leq t$, computes the group key k = f(0) by interpolating the points $P_1, \cdots, P_t, P_{t+1}$ and $(x_i, y_i \oplus R_i)$, Then checks the correctness of Auth by calculating $h(k, U_1, \cdots, U_t, R_1, \cdots, R_t, P_1, \cdots, P_t, P_{t+1})$. U_i aborts if the equality does not hold.

Based on step 4 and step 5, the adversary computes two polynomials $f(x)$ and $f'(x)$ of degree t+1 in sessions S_1 and S_2 respectively using two $t+2$ points in Round 5 of replay attack. Then the adversary computes a new polynomial g(x) of $t+1$ degree whose roots are x_i value of user U_i, for $1 \leq i \leq t$ and x_r, which is randomly selected by the KGC. To find the x_i of targeted user, the adversary needs t roots but he is equipped with only $t-1$ roots and hence fails to compute the x_i value of targeted user using products of the roots as he did in the earlier attacks.

As explained above, our countermeasure prevents the generalized replay attack. Now we will explain how it can be customized to prevent the simple replay attack i.e. Attack-1.
In Attack-1, adversary U_a computes long term secret of a targeted user U_i beacuse he was able to form a 2^{nd} degree polynomial $g(x)$ using $f(x)$ and $f'(x)$ in the Round 5 of Attack-1. The roots of g(x) are x_a and x_i values of users U_a and U_i. As adversary already knows his x_a value, he computes x_i of targeted user U_i using product of the roots of a polynomial. Now based on modified steps Step 4 and Step 5, adversary computes a 3^{rd} degree polynomial $g(x) = (a_3 - a'_3)x^3 + (a_2 - a'_2)x^2 + (a_1 - a'_1)x + (k - k')$ using polynomials $f(x) = a_3x^3 + a_2x^2 + a_1x + a_0$ and $f(x) = a'_3x^3 + a'_2x^2 + a'_1x + a'_0$. To find the x_i of targeted user U_i using product of the roots of a polynomial, the adversary needs two known roots but he is equipped with only one root i.e. his own x_a value and thereby fails to compute the x_i of the targeted user U_i.

After incorporating the above mentioned countermeasure, we have the following improved protocol that can resist the replay attacks.

Initialization of KGC. The KGC randomly selects two large safe primes p and q (i.e $p' = \frac{p-1}{2}$ and $q' = \frac{q-1}{2}$ are also primes) and computes n=pq. n is made publicly known.

User Registration. Each user is required to register at KGC to subscribe the key distribution service. The KGC keeps track of all the registered users and removes any unsubscribed users. During the registration, KGC shares a long-term secret (x_i, y_i) with each user U_i, $1 \leq i \leq m$, where $(x_i, y_i) \in Z_n^* \times Z_n^*$.

Group Key Generation and Distribution. After receiving the group key generation request from any user, KGC randomly chooses a group key. Then KGC distributes this group key to all the group members in a secure and authenticated way. Let us consider a group consisting of t members, $\{U_1, \cdots, U_t\}$, whose long term secrets shared with the KGC are (x_i, y_i), $1 \leq i \leq t$. The key generation and distribution process for the above session consists of the following five steps.

Step 1. The initiator sends a key generation request to KGC with a list of group members as $\{U_1, \cdots, U_t\}$.

Step 2. KGC broadcasts the participants list $\{U_1, \cdots, U_t\}$ to all U_i, $1 \leq i \leq t$, as a response to the request.

Step 3. Each user U_i, $1 \leq i \leq t$, chooses a random challenge $R_i \in Z_n^*$ and sends it to KGC.

Step 4. The KGC randomly selects a session key k and constructs a $t+1^{th}$ degree interpolation polynomial f(x) passing through (t + 2) points: $(0, k), (x_1, y_1 \oplus R_1), \cdots, (x_t, y_t \oplus R_t)$ and a random coordinate $(x_r, y_r \oplus R_r)$, where $(x_r, y_r) \in Z_n^* \times Z_n^*$ and $R_r \in Z_n^*$. Next, KGC selects (t+1) additional points $P_1, \cdots, P_t, P_{t+1}$ that lie on the polynomial f(x). KGC then computes the authentication message Auth=$h(k, U_1, \cdots, U_t, R_1, \cdots, R_t, P_1, \cdots, P_t, P_{t+1})$, where h is a one-way hash function, and broadcasts $(R_1, \cdots, R_t, P_1, \cdots, P_t, P_{t+1}, Auth)$ to all users participated in session.

Step 5. Each U_i, $1 \leq i \leq t$, computes the group key k = f(0) by interpolating the points $P_1, \cdots, P_t, P_{t+1}$ and $(x_i, y_i \oplus R_i)$, then checks the correctness of Auth by calculating $h(k, U_1, \cdots, U_t, R_1, \cdots, R_t, P_1, \cdots, P_t, P_{t+1})$. U_i aborts if the equality does not hold.

6 Conclusion

Nam et al. proposed a replay attack against the Harn and Lin's group key distribution protocol [4]. They also proposed a counter measure to resist this attack using a nonce mechanism [7]. However, this countermeasure increases the computational cost of the scheme. So, this paper proposes an alternative counter measure that does away with the computation of authentication by each user. Also studied in this paper is a generalization of the Nam et al.'s replay attack.

References

[1] Colin Boyd. On key agreement and conference key agreement. In *Information Security and Privacy, Second Australasian Conference, ACISP'97, Sydney, NSW, Australia, July 7-9, 1997, Proceedings*, pages 294–302, 1997.

[2] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Trans. Information Theory*, 22(6):644–654, 1976.

[3] Cheng Guo and Chin-Chen Chang. An authenticated group key distribution protocol based on the generalized chinese remainder theorem. *Int. J. Communication Systems*, 27(1):126–134, 2014.

[4] Lein Harn and Changlu Lin. Authenticated group key transfer protocol based on secret sharing. *IEEE Trans. Computers*, 59(6):842–846, 2010.

[5] Yanjun Liu, Lein Harn, and Chin-Chen Chang. An authenticated group key distribution mechanism using theory of numbers. *Int. J. Communication Systems*, 27(11):3502–3512, 2014.

[6] Yining Liu, Chi Cheng, Jianyu Cao, and Tao Jiang. An improved authenticated group key transfer protocol based on secret sharing. *IEEE Trans. Computers*, 62(11):2335–2336, 2013.

[7] Junghyun Nam, Moonseong Kim, Juryon Paik, Woongryul Jeon, Byunghee Lee, and Dongho Won. Cryptanalysis of a group key transfer protocol based on secret sharing. In *Future Generation Information Technology - Third International Conference, FGIT 2011 in Conjunction with GDC 2011, Jeju Island, Korea, December 8-10, 2011. Proceedings*, pages 309–315, 2011.

[8] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

[9] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.

[10] Wei Yuan, Liang Hu, Hongtu Li, and Jianfeng Chu. Security and improvement of an authenticated group key transfer protocol based on secret sharing. In *Appl. Math.Inf.Sci.7,No.5, 1943-1949*, pages 309–315, 2013.