Hindawi
Security and Communication Networks
Volume 2017, Article ID 3109624, 7 pages
https://doi.org/10.1155/2017/3109624

Research Article

Group Authentication with Multiple Trials and Multiple Authentications

Hung-Yu Chien

Department of Information Management, National Chi Nan University, 470 University Road, Puli, Taiwan

Correspondence should be addressed to Hung-Yu Chien; hychien@ncnu.edu.tw

Received 20 January 2017; Revised 22 April 2017; Accepted 27 April 2017; Published 28 May 2017

Academic Editor: Ángel Martín Del Rey

Group authentication aims at facilitating efficient authentication of a group of provers by a group of verifiers. A new group authentication scheme is proposed to improve the security of existent asynchronous group authentication schemes and to achieve better computational performance. The new scheme allows any groups of legitimate members to execute multiple authentication trials even under the participation of active attackers.

1. Introduction

Authentication is a must for securing computer and network applications. Conventional authentications, either user authentication or device authentication, all focus on the one-to-one scenario where one verifier aims at verifying the legitimacy of one prover at one time. As more and more Internet-of-Things (IoTs) [1, 2] applications and many social networking applications require the authentication of a group of participants efficiently, these many-to-many authentication scenarios call for new kinds of group authentications in which many verifiers would like to verify the legitimacy of many provers at one time to save cost and increase efficiency.

Based on Shamir's (t, n) secret sharing [3], Harn [4] proposed three (t, m, n) group authentication schemes, where t represents the minimum threshold of participants, m denotes the number of participants in one trial, and n denotes the total number of members of the group. As long as the number $m \geq t$ and all these m participants are legitimate, the group authentication succeeds; otherwise, it fails. These group authentication schemes can efficiently authenticate a group of legitimate entities or act as a preprocess to detect the existence of any illegitimate participants. One of Harn's group authentication schemes is synchronous (t, m, n) group authentication in which all participants are required to release their secret tokens simultaneously; otherwise, an illegitimate participant might forge valid tokens, using the released tokens of others. The other two schemes are the asynchronous (t, m, n) group authentication and the asynchronous (t, m, n) group authentication with multiple authentications; we, respectively, call them Harn's asynchronous GAS1 and Harn's asynchronous GAS2 in the rest of this paper. The two schemes all allow the participants release their tokens asynchronously; Harn's asynchronous GAS2 further provides the group to execute multiple authentications (to recover multiple system secrets) using the same set of predistributed tokens.

This paper would focus on the asynchronous schemes because the synchronous case is impractical. We find that Harn's two asynchronous schemes could not support legitimate entities execute multiple trials even if the specific secret is not yet recovered. This weakness has two implications; if the groups of entities try several times to recover a specific secret (for group authentications), then an attacker might derive entities' tokens and further derive the system secret; if the system only allows at most one trial for any specific secret (corresponds to a specific group authentication), then an attacker can easily paralyze the system by simply releasing invalid tokens. In Harn's publication [4], it only emphasizes that once a secret is recovered, then the corresponding group authentication is no longer valid; however, the security of the cases that the members try several times for the not-yet-recovered secret has been neglected.

This paper will show the weaknesses of Harn's asynchronous schemes and propose a new scheme to conquer the weaknesses and improve the efficiency. This rest of this paper is organized as follows. Section 2 reviews Harn's

asynchronous schemes. Section 3 shows the weaknesses. Section 4 proposes our new scheme, and Section 5 analyzes its securities and evaluates its performance. Section 6 states our conclusions.

2. Review of Harn's Asynchronous Group Authentication Schemes (GAS)

The schemes consist of two phases: the initialization phase and the group authentication phase. The group manager (GM) initializes the system parameters and assigns each registered entity some secret tokens in the initialization phase. Then, any groups of m legitimate entities with $m \geq t$ can execute the group authentication to verify the legitimacy of the participating entities.

2.1. (t, m, n) *Asynchronous GAS-Harn's Asynchronous GAS1.* Initially, the group manager (GM) selects k (where $kt > n-1$) random polynomials with degree $t-1$, $f_l(x) = a_{l,0} + a_{l,1}x + \cdots + a_{l,t-1}x^{t-1} \bmod p$, $l = 1, \ldots, k$, where p is a prime and $a_{l,i} \in_R Z_p$. He also generates and assigns secret tokens $f_l(x_i)$, $l = 1, \ldots, k$, to each entity $U_i \in U = \{U_1, \ldots, U_n\}$, where x_i is U_i's public identity. For any secret s, the GM finds integers d_j, w_j, $j = 1 \sim k$, in GF(p), such that $s = \sum_{j=1\sim k} d_j f_j(w_j) \bmod p$, where $w_i \neq w_j$ for every pair of i and j. The GM publishes these parameters d_j, w_j, $j = 1 \sim k$, and $h(s)$, where $h()$ is a secure cryptographic hash function.

When m $(m \geq t)$ entities $\widetilde{P} = \{P_1, \ldots, P_m\}$ would like to authenticate each other, each P_i computes and releases $c_i = \sum_{j=1}^{k} d_j f_j(x_i) \prod_{r=1, r\neq i}^{m} ((w_j - x_r)/(x_i - x_r)) \bmod p$. After gathering all the released values, the participants compute $s' = \sum_{i=1}^{m} c_i \bmod p$ and verify whether the equation $h(s) \stackrel{?}{=} h(s')$ holds. If the verification succeeds, then the group authentication succeeds; otherwise, it fails. This scheme only allows one valid group authentication.

2.2. (t, m, n) *Asynchronous GAS with Multiple Authentications-Harn's Asynchronous GAS2.* The (t, m, n) asynchronous GAS with multiple authentications allows the tokens to be reused for multiple authentications (for multiple secrets).

Initially, the GM selects two large primes p and q, such that q divides $p-1$, GF(q) is a subgroup of GF(p), and every g_i is a generator for the subgroup GF(q). The GM selects two random polynomials, $f_l(x)$, $l = 1, 2$, having degree $t-1$ each with coefficients in GF(p). The GM generates tokens, $f_l(x)$, $l = 1, 2$, for each registered member U_i. The GM selects multiple secrets s_is. For each secret s_i, the GM selects $w_{i,j}, d_{i,j}$, $j = 1, 2$, in GF(q), where $w_{i,1} \neq w_{i,2}$. The secret s_i is determined as $s_i = g_i^{\sum_{j=1}^{2} d_{i,j} f(w_{i,j}) \bmod q} \bmod p$. The GM publishes these numbers $w_{i,j}, d_{i,j}$, $j = 1, 2$, and $\{(g_i, h(s_i))\}$.

When m $(m \geq t)$ entities $\widetilde{P} = \{P_1, \ldots, P_m\}$ would like to perform the group authentication corresponding to the reconstruction of the secret s_i, each participant $P_v \in \widetilde{P}$ computes $c_v = \sum_{j=1}^{2} d_j f_j(x_v) \prod_{r=1, r\neq v}^{m} ((w_j - x_r)/(x_v - x_r)) \bmod q$ and $e_v = g_i^{c_v} \bmod p$. Each P_v releases e_v. After collecting all e_vs, $v = 1, \ldots, m$, the participating entities compute $s_i' = \prod_{v=1}^{m} e_v$ and check whether $h(s_i') \stackrel{?}{=} h(s_i)$ holds. If it holds, then the group authentication succeeds; otherwise, it fails.

3. The Weaknesses of Harn's Asynchronous Schemes

We find that both Harn's asynchronous GAS1 and Harn's asynchronous GAS2 share one critical weakness. The schemes perform group authentication by recovering and verifying the sealed secret. If the schemes allow users to launch several trials before the secret is recovered, then an attacker would recover both the system secrets and the users' secret tokens by joining the process several times. On the other hand, if each secret only allows one trial of authentication no matter whether the specific secret is recovered or not, then the system is vulnerable to Denial of Service (DOS) attacks by simply releasing a false value to spoil the authentication instance and the group authentication function of the system. After releasing a fake data, any groups of valid members can no longer perform any group authentications.

The key idea of our attack on Harn's asynchronous GAS1 is introduced in the following phases.

Phase 1. Even though the secret tokens $f_j(x_i)$s are well protected in the released value $c_i = \sum_{j=1}^{k} d_j f_j(x_i) \prod_{r=1, r\neq i}^{m} ((w_j - x_r)/(x_i - x_r)) \bmod p$, one could solve these unknown variables $f_j(x_i)$s as long as he gets k distinct $\{c_i\}$, where each c_i corresponds to the value released by a specific user U_i in an authentication instance and there is at least one member different in any pair of groups in these authentication instances; in such cases, the attacker will have k independent equations with k unknown variables $f_j(x_i)$s and he can solve the equations. Let $F_i \equiv \{f_j(x_i) \mid 1 \leq i \leq k\} = \{f_1(x_i), \ldots, f_k(x_i)\}$ denote the set of secret tokens owned by the user U_i; after the above attack, the attacker can acquire F_i. Now the attacker continues the next phase to acquire the secret polynomials.

Phase 2. The attacker repeatedly involves the authentication instances and acquires the secret tokens until he gets the secret tokens of more than t users. Denote these secret token set as $F \equiv \{F_1, \ldots, F_i, \ldots, F_t\} = \{\{f_1(x_1), \ldots, f_k(x_1)\}, \ldots, \{f_1(x_i), \ldots, f_k(x_i)\}, \ldots, \{f_1(x_t), \ldots, f_k(x_t)\}\}$. At this point, he organizes these secret tokens as $F^1 = \{f_1(x_1), f_1(x_2), \ldots, f_1(x_t)\}, F^2 = \{f_2(x_1), f_2(x_2), \ldots, f_2(x_t)\}, \ldots, F^k = \{f_k(x_1), f_k(x_2), \ldots, f_k(x_t)\}$. Based on F^i, $1 \leq i \leq k$, the attacker applies the Lagrange polynomial equation to reconstruct the polynomials $f_i(x)$, $1 \leq i \leq k$. The attacker then continues the next phase to derive the system secrets and the secret tokens of other remaining members.

Phase 3. Using the polynomials $f_i(x)$, $1 \leq i \leq k$, the attacker compute $s = \sum_{j=1\sim k} d_j f_j(w_j) \bmod p$ for the system secret. For any user $U_r \in U = \{U_1, \ldots, U_n\}$ and the secret tokens of U_r that have not yet been disclosed, the attacker computes $\{f_1(x_r), \ldots, f_k(x_r)\}$. At this point, the attacker has derived all

the system secrets and all the secret tokens of all users. The minimum number of runs that the attacker should participate in is k.

The above attack can be easily extended to plot on Harn's asynchronous GAS2. Attackers can acquire the secret values $\{(g_i^{f_1(x_v)}, g_i^{f_2(x_v)}) \mid \forall \text{ valid } i \text{ and } 1 \le v \le n\}$ corresponding to the secret tokens $\{f_1(x_v), f_2(x_v) \mid 1 \le v \le n\}$ and the system secrets $s_i = g_i^{\sum_{j=1}^{2} d_{i,j} f(w_{i,j}) \bmod q} \bmod p$.

Example 1. Now we take one example to demonstrate the attack process.

System Initialization. Let $p = 11$, $t = 3$, $n = 6$, $k = 2$, ($d_1 = 1$, $w_1 = 6$), and ($d_2 = 2, w_1 = 7$) be the system parameters. $f_1(x) = 1 + 2x + 2x^2 \bmod 11$ and $f_2(x) = 2 + 2x + 1x^2 \bmod 11$ are the two secret polynomials, and the system secret is $s = 1 \cdot f_1(6) + 2 \cdot f_2(7) \bmod 11 = 8 + 20 \bmod 11 = 6$. The group of users, U, is $\{U_i$ with identity $x_i = i,\ 1 \le i \le 6\}$. U_1 gets the secret tokens ($f_1(1) = 5$, $f_2(1) = 5$), U_2 gets the secret tokens ($f_1(2) = 2$, $f_2(2) = 10$), U_3 gets the secret tokens ($f_1(3) = 3$, $f_2(3) = 6$), U_4 gets the secret tokens ($f_1(4) = 8$, $f_2(4) = 4$), U_5 gets the secret tokens ($f_1(5) = 6$, $f_2(5) = 4$), and U_6 gets the secret tokens ($f_1(6) = 8$, $f_2(6) = 6$).

Now we show the attack.

Attack Phase 1. Assume that the attacker **A** participates in two runs of authentications with $\{U_1, U_2, U_3\}$ and, respectively, impersonates U_4, U_5 in these runs.

In run 1, **A** will get

$$\left\{ c_i = \sum_{j=1}^{2} d_j f_j(x_i) \prod_{r=1, r\neq i}^{4} \frac{w_j - x_r}{x_i - x_r} \bmod p \mid 1 \le i \le 3 \right\}. \quad (1)$$

We list the calculations as follows:

$$\begin{aligned}
c_1 &= \sum_{j=1}^{2} d_j f_j(1) \prod_{r=1, r\neq 1}^{4} \frac{w_j - r}{1 - r} \\
&= 1 \cdot f_1(1) \frac{(6-2)(6-3)(6-4)}{(1-2)(1-3)(1-4)} + 2 \\
&\quad \cdot f_2(1) \frac{(7-2)(7-3)(7-4)}{(1-2)(1-3)(1-4)} \\
&= 1 \cdot 5 \cdot \frac{4 \cdot 3 \cdot 2}{-1 \cdot -2 \cdot -3} + 2 \cdot 5 \cdot \frac{5 \cdot 4 \cdot 3}{-1 \cdot -2 \cdot -3} \\
&= 1 \cdot 5 \cdot 7 + 2 \cdot 5 \cdot 1 = 1 \\
c_2 &= \sum_{j=1}^{2} d_j f_j(2) \prod_{r=1, r\neq 2}^{4} \frac{w_j - r}{2 - r} \\
&= 1 \cdot f_1(2) \frac{(6-1)(6-3)(6-4)}{(2-1)(2-3)(2-4)} + 2 \\
&\quad \cdot f_2(2) \frac{(7-1)(7-3)(7-4)}{(2-1)(2-3)(2-4)} \\
&= 1 \cdot 2 \cdot \frac{5 \cdot 3 \cdot 2}{1 \cdot -1 \cdot -2} + 2 \cdot 10 \cdot \frac{6 \cdot 4 \cdot 3}{1 \cdot -1 \cdot -2} \\
&= 1 \cdot 2 \cdot 4 + 2 \cdot 10 \cdot 3 = 2 \\
c_3 &= \sum_{j=1}^{2} d_j f_j(3) \prod_{r=1, r\neq 3}^{4} \frac{w_j - r}{3 - r} \\
&= 1 \cdot f_1(3) \frac{(6-1)(6-2)(6-4)}{(3-1)(3-2)(3-4)} + 2 \\
&\quad \cdot f_2(3) \frac{(7-1)(7-2)(7-4)}{(3-1)(3-2)(3-4)} \\
&= 1 \cdot 3 \cdot \frac{5 \cdot 4 \cdot 2}{2 \cdot 1 \cdot -1} + 2 \cdot 6 \cdot \frac{6 \cdot 5 \cdot 3}{2 \cdot 1 \cdot -1} \\
&= 1 \cdot 3 \cdot 2 + 2 \cdot 6 \cdot 10 = 5.
\end{aligned} \quad (2)$$

In run 2, **A** will get

$$\left\{ c_i = \sum_{j=1}^{2} d_j f_j(x_i) \prod_{r\in\{1,2,3,5\}, r\neq i} \frac{w_j - x_r}{x_i - x_r} \bmod p \mid 1 \le i \le 3 \right\} \quad (3)$$

as follows:

$$\begin{aligned}
c_1 &= \sum_{j=1}^{2} d_j f_j(1) \prod_{r=2,3,5} \frac{w_j - r}{1 - r} \\
&= 1 \cdot f_1(1) \frac{(6-2)(6-3)(6-5)}{(1-2)(1-3)(1-5)} + 2 \\
&\quad \cdot f_2(1) \frac{(7-2)(7-3)(7-5)}{(1-2)(1-3)(1-5)} \\
&= 1 \cdot 5 \cdot \frac{4 \cdot 3 \cdot 1}{-1 \cdot -2 \cdot -4} + 2 \cdot 5 \cdot \frac{5 \cdot 4 \cdot 2}{-1 \cdot -2 \cdot -4} \\
&= 1 \cdot 5 \cdot 4 + 2 \cdot 5 \cdot 6 = 3 \\
c_2 &= \sum_{j=1}^{2} d_j f_j(2) \prod_{r=1,3,5} \frac{w_j - r}{2 - r} \\
&= 1 \cdot f_1(2) \frac{(6-1)(6-3)(6-5)}{(2-1)(2-3)(2-5)} + 2 \\
&\quad \cdot f_2(2) \frac{(7-1)(7-3)(7-5)}{(2-1)(2-3)(2-5)} \\
&= 1 \cdot 2 \cdot \frac{5 \cdot 3 \cdot 1}{1 \cdot -1 \cdot -3} + 2 \cdot 10 \cdot \frac{6 \cdot 4 \cdot 2}{1 \cdot -1 \cdot -3} \\
&= 1 \cdot 2 \cdot 5 + 2 \cdot 10 \cdot 5 = 0 \\
c_3 &= \sum_{j=1}^{2} d_j f_j(3) \prod_{r=1,2,5} \frac{w_j - r}{3 - r}
\end{aligned}$$

$$= 1 \cdot f_1(3) \frac{(6-1)(6-2)(6-5)}{(3-1)(3-2)(3-5)} + 2 \cdot f_2(3) \frac{(7-1)(7-2)(7-5)}{(3-1)(3-2)(3-5)} = 1 \cdot 3 \cdot \frac{5 \cdot 4 \cdot 1}{2 \cdot 1 \cdot -2} + 2 \cdot 6 \cdot \frac{6 \cdot 5 \cdot 2}{2 \cdot 1 \cdot -2} = 3. \quad (4)$$

So now **A** has the following independent equations in (5a), (5b), and (5c). He then solves the equations and gets $f_1(1) = 5$, $f_2(1) = 5$, $f_1(2) = 2$, $f_2(2) = 10$, $f_1(3) = 3$, $f_2(3) = 6$.

$$7f_1(1) + 2f_2(1) = 1 \bmod 11, \quad 4f_1(1) + f_2(1) = 3 \bmod 11 \quad (5a)$$

$$4f_1(2) + 6f_2(2) = 2 \bmod 11, \quad 5f_1(2) + 10f_2(2) = 0 \bmod 11 \quad (5b)$$

$$2f_1(3) + 9f_2(3) = 5 \bmod 11, \quad 6f_1(3) + 3f_2(3) = 3 \bmod 11. \quad (5c)$$

A applies the Lagrange polynomial formula on ($f_1(1) = 5, f_1(2) = 2, f_1(3) = 3$) and derives the polynomial $f_1(x) = 1 + 2x + 2x^2 \bmod 11$, applies the formula on ($f_2(1) = 5$, $f_2(2) = 10$, $f_2(3) = 6$), and derives $f_2(x) = 2 + 2x + 1x^2 \bmod 11$. Finally, he computes $s = 1 \cdot f_1(6) + 2 \cdot f_2(7) \bmod 11 = 8 + 20 \bmod 11 = 6$. He can further computes the secret tokens of other remaining members $\{(f_1(i), f_2(i)) \mid 4 \le i \le 6\}$.

4. An Improved Scheme That Enables Multiple Trials and Multiple Authentications

Now we will propose an improved scheme that not only conquers the weaknesses of Harn's (t, m, n) asynchronous schemes but also improves the system performance. The GM in our scheme only publishes simple public data and the members can execute group authentication with multiple authentications and multiple trials.

4.1. Preliminaries. We shall propose our scheme, based on elliptic curve cryptography and bilinear pairing. We now briefly review them as follows.

Elliptic curves over $GF(p)$ [6]: a nonsupersingular elliptic curve $E(Fp)$ is the set of points $P = (x, y)$, for $x, y \in Z_P$ satisfying the equation $y^2 \equiv x^3 + ax + b \pmod p$, where $a, b \in Z_p$ are constants such that $4a^3 + 27b^2 \neq 0 \bmod p$, together with the point O called the point at infinity. Two points $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ on the elliptic curve E can be added together using the following rule: if $x_2 = x_1$ and $y_2 = -y_1$, then $P + Q = O$; otherwise, $P + Q = (x_3, y_3)$ where $x_3 = \lambda^2 - x_1 - x_2 \bmod p$, $y_3 = \lambda(x_1 - x_3) - y_1 \bmod p$, and $\lambda = (y_2 - y_1)/(x_2 - x_1)$ if $P \neq Q$ or $\lambda = (3{x_1}^2 + a)/(2y_1)$ if $P = Q$.

Definition 2 (nondegenerate, bilinear, computable map [7]). Let G_1, G_2, and G_3 be cyclic groups of prime order q, where G_1 and G_2 are additive groups on elliptic curves and G_3 is multiplicative. Let e: $G_1 \times G_2 \to G_3$ be a map with the following properties:

(1) Nondegenerate: there exists $X, Y \in G_1, G_2$ such that $e(X, Y) \neq 1$.

(2) Bilinear: $e(X_1 + X_2, Y) = e(X_1, Y) \cdot e(X_2, Y)$ and $e(X, Y_1 + Y_2) = e(X, Y_1) \cdot e(X, Y_2)$.

(3) Computability: there exist efficient algorithms to compute $e(P, Q)$ for all $P, Q \in G_1, G_2$.

Definition 3. The elliptic curve discrete logarithm problem (ECDLP) [6] is as follows: given an elliptic curve over a finite field F_p and two points $P, Q \in E(F_p)$, find a number k such that $Q = kP$.

Definition 4 (the Bilinear Pairing Inversion (BPI) problem [7]). Given $e(P, Q) \in G_3$ and $Q \in G_2$, find P.

Definition 5. The computational elliptic curve Diffie-Hellman problem (ECDHP) [6] is as follows: given an elliptic curve over a finite field F_p, a point $P \in E(F_p)$ of order q, and points $A = aP$, $B = bP \in \langle P \rangle$, find the point $C = abP$.

It is believed that the ECDHP, the ECDLP, and the BPI are hard problems for proper parameter setting.

4.2. The Proposed Scheme

4.2.1. The System Model. Here we describe the model for one group, and it is easy to extend this model for several groups. In the system, there are two kinds of participants: the GM and a group of registered members. The GM is responsible for setting up/updating the system parameters. After initialization, the participants in each session would like to verify whether all the participants belong to the same group; this verification is achieved by the validation of the aggregated released-shares. The GM is trusted, and registered members might be compromised and disclose their secrets. Unless being compromised, a registered member always behaves honestly.

The GM publishes a predetermined parameter t. The scheme can verify whether all participants of one session with m participants ($m \ge t$) belong to the same group. The scheme is secure if it can withstand the collusion of up to $t-1$ insiders (registered members).

4.2.2. The Scheme Facilitating Multiple Trials and Multiple Authentications without Server's Active Participation. Like Harn's asynchronous (t, m, n) schemes, our scheme also follows the same (t, m, n) notation, the asynchronous communication, and multiple authentications. Additionally, our scheme allows multiple trials. The GM only needs to publish some simple data no matter how many authentications and trails these members would like to perform. The scheme consists of two phases: the initialization phase and the group authentication phase.

Initialization. The GM sets up three cyclic groups G_1, G_2, and G_3 with order q, where G_1 and G_2 are additive groups on elliptic curves and G_3 is multiplicative. P is a generator for G_2. It chooses a secret random polynomial with degree $t-1$, $f(x) = a_0 + a_1 x + \cdots + a_{t-1}x^{t-1} \bmod q$ with $a_0 = s$, and a master secret s. It computes $Q = sP$ and publishes Q as the system-wise public key. For each registered member $U_i \in U = \{U_1, \ldots, U_n\}$ with identity x_i, it assigns $f(x_i)$ as U_i's secret token.

Group Authentication. When m ($m \geq t$) entities $\widetilde{P} = \{P_1, \ldots, P_m\}$ would like to authenticate each other in the vth authentication instance, the group of users agree on a random point R_v (we discuss two options of implementing the generation of random points R_vs in Section 4.3). Each $P_i \in \widetilde{P}$ computes $c_i = f(x_i)\prod_{r=1,r\neq i}^{m}(-x_r/(x_i - x_r))$ and $c_i R_v$ and releases $c_i R_v$. After all users release their values, they compute $\sum_{i=1}^{m} c_i R_v$ and verify whether the equation $e(\sum_{i=1}^{m} c_i R_v, P) \stackrel{?}{=} e(R_v, Q)$ holds. If it holds, they satisfy the group authentication; otherwise, they fail.

4.3. Implementation Options of Choosing R_v. The generation and selection of the random points R_v play a crucial factor affecting the security. Here, we discuss two possible options. The two options mainly tackle the possible threat that an adversary might manipulate the selection of R_v.

The first one is that the GM periodically updates a list of authenticated random points, and the entities choose one from the list of unused ones. The entities refuse to apply any points that they have used.

The second approach is applying a one-way hash function that maps any strings to a random point—$H_1 : \{0,1\}^* \to G_1$. Boneh and Franklin's MaptoPoint function is one of such functions [7]. In this approach, each participant in the group calculates $R_v = H_1(\text{date} \parallel \text{time} \parallel x_1 \parallel x_2 \parallel \cdots \parallel x_m)$, where date and time are the current timestamp and x_is are the participants' identities.

4.4. Comparison of Our Improvements with One Possible Extension of Harn's GAS2. In addition to our proposed scheme, one another possible improvement is by extending Harn's GAS2. The system might require that each participant never tries to recover one specific secret twice; that is, whenever an authentication fails, he should only try another authentication corresponding to other secrets. This arrangement could prevent the attacks in Section 3.

However, we would like to discuss the differences between the above extension with our scheme. The extension, even though it could reduce the treats of our attacks, is still not absolutely immune to DOS attacks. The GM has to preselect lots of possible secrets and publishes these numbers $w_{i,j}, d_{i,j}$, $j = 1, 2$, and $\{(g_i, h(s_i))\}$. If the list is not large enough, then the successive releasing of false shares could quickly deplete the list. If the GM tries to prepare a very long list, it causes it lots of overhead.

On the contrary, our implementation Option 2 is much more simple, efficient, and withstand heavy DOS attacks.

5. Security Analysis and Performance Evaluation

5.1. The Security

Lemma 6. *Given a random point R_v and a group of members $\widetilde{P} = \{U_1, \ldots, U_m\}$ with $m \geq t$, the only condition that the group P can reconstruct the value sR_v in the proposed scheme is that all the participating members are valid and the scheme can resist up to $t-1$ colluded insiders.*

Proof. Since the secret tokens are generated using a secret $t-1$-degree polynomial, the scheme can resist the collusion of up to $t-1$ insiders. Also, any single invalid contribution from any invalid participants would ruin the computation of sR_v. □

Lemma 7. *Given a valid released value $c_i R_v$, where $c_i = f(x_i)\prod_{r=1,r\neq i}^{m}(-x_r/(x_i - x_r))$, one cannot derive the value c_i and the corresponding $f(x_i)$ as long as the ECDLP is hard.*

Lemma 8. *Given a valid released $c_i R_v$ and another point $\widetilde{R}$, one cannot derive the value $c_i\widetilde{R}$ as long as the ECDHP is hard.*

Lemma 9. *Given the values $e(R_v, Q)$ and Q, one cannot derive the value $\sum_{i=1}^{m} c_i R_v$ which satisfies $e(\sum_{i=1}^{m} c_i R_v, P) \stackrel{?}{=} e(R_v, Q)$ as long as the BPI is hard.*

Based on the above lemmas, we have the following theorem.

Theorem 10. *The proposed scheme satisfies the security requirements of the asynchronous (t, m, n) group authentication.*

5.2. The Performance. We first compare the computational complexities of the three schemes: ours, Harn's GAS1, and Harn's GAS2.

Let T_h denote the time complexity for one hash operation, T_{EM} denote that of one elliptic curve point multiplication, T_{EA} denote that for one elliptic curve point addition, $T_{\text{mul},q}$ denote that for one multiplication in field q (where q corresponds to the order of G_1, G_2, G_3 in our scheme), $T_{\text{inv},q}$ denote that for one inverse operation in field q, $T_{\text{mul},p}$ denote that for one multiplication in field p (where p corresponds to the modular field in Harn's schemes), $T_{\text{inv},p}$ denote that for one multiplication in field p, $T_{\text{exp},p}/T_{\text{exp},q}$, respectively, denote that for one exponentiation in field p/q, and T_{pair} denote that for one pairing.

Each user U_i in our scheme needs to compute one Lagrange component $c_i = f(x_i)\prod_{r=1,r\neq i}^{m}(-x_r/(x_i - x_r))$, one $c_i R$, $m-1$ point additions in G_1, and the verification $e(\sum_{i=1}^{m} c_i R, P) \stackrel{?}{=} e(R, Q)$. The verification costs two pairing operations ($2T_{\text{pair},G_1}$). The addition in G_1 costs mT_{EA}. The computation of $c_i R$ costs T_{EM}, and the computation of c_i costs $(2(m-1)+2)T_{\text{mul},q} + T_{\text{inv},q}$. So totally it takes $(2(m-1)+2)T_{\text{mul},q} + T_{\text{inv},q} + T_{\text{EM}} + mT_{\text{EA}} + 2T_{\text{pair}}$.

Each participant in Harn's GAS1 takes $k[(2(m-2)+3)T_{\text{mul},p} + T_{\text{inv},p}]$ for computing c_i, takes $(m-1)T_{\text{add},p}$ for

Table 1: Summaries of performance of group authentication schemes.

	Ours	Harn's GAS1	Harn's GAS2
Securities under multiple trials for each secret	Allow multiple authentications with multiple trials	Attackers can disclose system's secrets and users' secret tokens	Attackers can disclose system's secrets and users' secret tokens
Each user's computational cost for one group authentication (detailed)	$(2(m-1)+2)T_{\text{mul},q} + T_{\text{inv},q} + T_{\text{EM}} + mT_{\text{EA}} + 2T_{\text{pair}}$	$k[(2(m-2)+3)T_{\text{mul},p} + T_{\text{inv},p}] + (m-1)T_{\text{add},p} + 1T_h$	$2[(2(m-2)+3)T_{\text{mul},q} + T_{\text{inv},q}] + T_{\text{exp},p} + (m-1)T_{\text{mul},p} + 1T_h$
Approximated complexities (based on the figures from [5])	$(7m+1429)T_{\text{mul},q} + 2T_{\text{pair}} \cong (7m+6785)T_{\text{mul},q}$	$41k(2m+239)T_{\text{mul},q} + (m-1)T_{\text{add},p} + 1T_h \cong 41k(2m+239)T_{\text{mul},q} + (m-1)T_{\text{add},p}$	$(45m+1418)T_{\text{mul},q}$

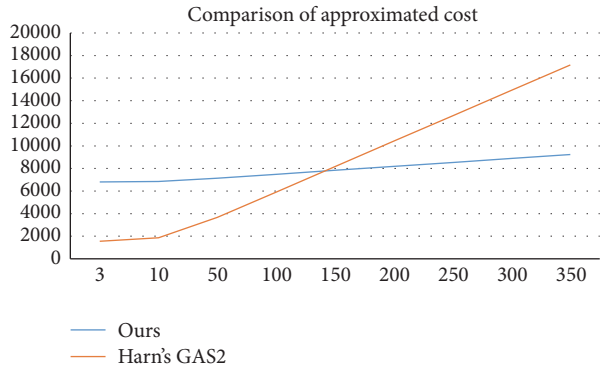

Figure 1: The comparison of computational cost versus the number of participants. The unit of the cost is $T_{\text{mul},q}$.

computing s', and takes $1T_h$ for verifying $h(s) \stackrel{?}{=} h(s')$. Each participant totally takes $k[(2(m-2)+3)T_{\text{mul},p} + T_{\text{inv},p}] + (m-1)T_{\text{add},p} + 1T_h$.

Each participant in Harn's GAS2 takes $[(2(m-2)+3)T_{\text{mul},q} + T_{\text{inv},q}]$ for computing c_v, takes $T_{\text{exp},p}$ for computing e_v, takes $(m-1)T_{\text{mul},p}$ for computing s_i', and takes $1T_h$ for verifying $h(s_i) \stackrel{?}{=} h(s_i')$. Each participant totally takes $[(2(m-2)+3)T_{\text{mul},q} + T_{\text{inv},q}] + T_{\text{exp},p} + (m-1)T_{\text{mul},p} + 1T_h$.

Table 1 summarizes the performance. Row 2 lists the security properties. Only our scheme can resist an attacker from deriving the secrets when the schemes allow multiple trials. Row 3 lists the detailed computational complexity. Based on Row 3, it is still difficult to get an insight of the complexities since they involve quite different operations. We, therefore, further evaluate the computational cost under the practical setting from NSA [8] and the algebra equations of elliptic curve operations [6]. The security of ECC with 160-bit key is roughly equivalent to that of RSA with 1024-bit key or D-H algorithm with 1024-bit key. So let us assume that the q (the order of G_1 and G_2) in our scheme is 160 bits, and p in Harn's schemes is 1024 bits. Under the above setting and approximations, $T_{\text{mul},p}$ (the time complexity of a field multiplication in Z_p, where p is 1024 bits) is 41 times $T_{\text{mul},q}$ (the time complexity of field multiplication in Z_q, where q is 160 bits), $T_{\text{EM}} \cong 29T_{\text{mul},p}$, $T_{\text{EA}} \cong 0.12T_{\text{mul},p}$, $T_{\text{exp},p} \cong 240T_{\text{mul},p} \cong 8T_{\text{EM}}$, and $T_{\text{EM}} \cong 241T_{\text{EA}}$, where $\cong$ means "roughly equal."

Using the above setting and approximations and neglecting the minor computations like hashing and field addition, the complexities in Row 3 can be further simplified as follows. The complexity of our scheme is as follows: $(2(m-1)+2)T_{\text{mul},q} + T_{\text{inv},q} + T_{\text{EM}} + mT_{\text{EA}} + 2T_{\text{pair}} \cong (2(m-1)+2)T_{\text{mul},q} + T_{\text{inv},q} + 29T_{\text{mul},p} + 0.12mT_{\text{mul},p} + 2T_{\text{pair}} \cong 2mT_{\text{mul},q} + 240T_{\text{mul},q} + 29 * 41T_{\text{mul},q} + 0.12m * 41T_{\text{mul},p} + 2T_{\text{pair}} \cong (7m+1429)T_{\text{mul},q} + 2T_{\text{pair}}$. The complexity of Harn's GAS1 is as follows: $k[(2(m-2)+3)T_{\text{mul},p} + T_{\text{inv},p}] + (m-1)T_{\text{add},p} + 1T_h \cong 41k(2m+239)T_{\text{mul},q} + (m-1)T_{\text{add},p}$. The complexity of Harn's GAS2 is as follows: $2[(2(m-2)+3)T_{\text{mul},q} + T_{\text{inv},q}] + T_{\text{exp},p} + (m-1)T_{\text{mul},p} + 1T_h \cong 2[(2(m-2)+239)T_{\text{mul},q}] + 240 * 41T_{\text{mul},q} + 41(m-1)T_{\text{mul},q} + 1T_h \cong (45m+1418)T_{\text{mul},q}$.

To further simplify the complexity approximation, we refer to an efficient paring implementation [5]. Based on the figures there [5], we roughly approximate one pairing operation as $5356T_{\text{mul},q}$. We approximate $(7m+1429)T_{\text{mul},q} + 2T_{\text{pair}} \cong (7m+6785)T_{\text{mul},q}$ in Row 4. From Row 4, we can tell that Harn's schemes have lower computational cost than ours when the number of participants is small; but the costs of Harn's schemes grow faster than ours when the number of participants increases. All the costs of the three schemes increase as the number of participants increases, but the cost of Harn's GAS1 also depends on the value k. Because the parameter k in Harn's GAS1 should satisfy $kt > n-1$, we only compare our scheme with Harn's GAS2 in Figure 1 to give us an insight of the performances of the three schemes. From Figure 1, we can see that the cost of Harn's GAS2 increases much faster than ours when the number of participants increases. When the number is around 141, the cost of Harn's GAS2 overpasses ours and increases very fast. The comparison shows that our scheme not only owns better security but also provides better computational performance when the number of participants is large.

6. Conclusions

In this paper, we have shown the weaknesses of Harn's asynchronous group authentication schemes. An attacker can derive the system secrets and the members' secret tokens if the schemes allow multiple trials before the corresponding secret is recovered, or an attacker can easily disable the functions of the schemes by simply releasing invalid shares if the schemes do not allow multiple trials. We have proposed an improved scheme that allows multiple trials for each system secret. The analysis shows that our scheme even has better computational performance when the number of participants is greater than 141.

Conflicts of Interest

The author declares that he has no conflicts of interest.

Acknowledgments

This project is partially supported by the Ministry of Science and Technology, Taiwan, under Grant no. MOST 105-2221-E-260-014. The author would like to express the gratitude to Kun-Bo Chen for his efforts of collecting literature.

References

[1] S. Kim, J.-Y. Choi, and J. Jeong, "On authentication signaling costs in hierarchical LTE networks," in *Proceedings of 7th International Conference on <italic>Ubi-Media Computing and Workshops</italic>, U-MEDIA 2014*, pp. 11–16, Ulaanbaatar, Mongolia, July 2014.

[2] J. Li, M. Wen, and T. Zhang, "Group-based authentication and key agreement with dynamic policy updating for MTC in LTE-A networks," *IEEE Internet of Things Journal*, vol. 3, no. 3, pp. 408–417, 2016.

[3] A. Shamir, "How to share a secret," *Communications of the Association for Computing Machinery*, vol. 22, no. 11, pp. 612-613, 1979.

[4] L. Harn, "Group authentication," *IEEE Transactions on Computers*, vol. 62, no. 9, pp. 1893–1898, 2013.

[5] J.-L. Beuchat, J. E. G. Díaz, S. Mitsunari, E. Okamoto, F. Rodríguez-Henríquez, and T. Teruya, High-Speed Software Implementation of the Optimal Ate Pairing over Barreto-Naehrig Curves," *IACR Eprint*, 2010/354, https://eprint.iacr.org/2010/354.pdf.

[6] A. Jurisic and A. J. Menezes, *Elliptic Curves and Cryptography*, Certicom Whitepaper, 1997.

[7] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," in *Proceeding of the Advances in Cryptology—CRYPTO 2001*, vol. 2139 of *Lecture Notes in Computer Science*, pp. 213–229, Springer-Verlag, Santa Barbara, Calif, USA, 2001.

[8] National Security Agency, the US, "*The Case for Elliptic Curve Cryptography*", https://www.nsa.gov/business/programs/elliptic_curve.shtml. Accessed 25 Dec. 2014.