Brief Contributions

Group Authentication

Lein Harn

Abstract—A new type of authentication, call *group authentication*, which authenticates all users belonging to the same group is proposed in this paper. The group authentication is specially designed for group-oriented applications. The group authentication is no longer a one-to-one type of authentication as most conventional user authentication schemes which have one prover and one verifier; but, it is a many-to-many type of authentication which has multiple provers and multiple verifiers. We propose a basic t-secure m-user n-group authentication scheme ((t, m, n) GAS), where t is the threshold of the proposed scheme, m is the number of users participated in the group authentication, and n is the number of members of the group, which is based on Shamir's (t, n) secret sharing (SS) scheme. The basic scheme can only work properly in synchronous communications. We also propose asynchronous (t, m, n) GASs, one is a GAS with one-time authentication and the other is a GAS with multiple authentications. The (t, m, n) GAS is very efficient since it is sufficient to authenticate all users at once if all users are group members; however, if there are nonmembers, it can be used as a preprocess before applying conventional user authentication to identify nonmembers.

Index Terms—Authentication, group communication, secret sharing, ad hoc network, group-oriented applications

1 Introduction

User authentication is one of the most important security services in computer and communication application. Knowledge-based authentication (e.g., password) [9], [17] and key-based authentication (e.g., public/private key) [6], [13] are the two most popular approaches. Knowledge-based authentication has some security flaws. Most users like to use simple and short passwords. Internet hackers can easily crack simple passwords. To circumvent these issues, public-key cryptography has been utilized to provide user authentication [7], [11]. Public-key-based authentication needs a certificate authority (CA) to provide the authenticity of public keys. In addition, public-key computations involve large integers. Computational overhead is one of the main concerns for public-key based authentication.

All user authentication schemes [5], [10] are one-to-one type of authentications in which the prover interacts with the verifier to verify the identity of the prover. For example, the RSA digital signature [14] can be used to authenticate the signer of the signature. In this approach, the verifier sends a random challenge to the prover. Then, the prover digitally signs the random challenge and returns the digital signature of the challenge to the verifier. After successfully verifying the digital signature, the verifier is convinced that the prover is the one with the identity of the public-key digital certificate used to verify the digital signature.

Network applications are no longer just one-to-one communication, but involve multiple users (>2). Group communication [1], [15] implies a many-to-many communication and it goes beyond both one-to-one communication (i.e., unicast) and one-to-many communication (i.e., multicast). In a group-oriented application, there are multiple members who want to form a private network and to exchange messages among themselves. In order to establish such a network, every user participated in the application needs to authenticate other users belonging to the same group. There are two popular models to provide group authentication services in such application. The first model involves a centralized authentication server (AS) [2], [12] and the second model has no AS [3], [4]. In the first model, AS manages the access rights of the network. For example, Bhakti et al. [2] proposed to adopt extensible authentication protocol (EAP) in the IEEE 802.1x Standard [6] for wireless ad hoc network. This approach requires setting up the AS and mobile users have to access the AS for the authentication service. However, for some applications, mobile users have no access to the AS. For example, in a wireless ad hoc network, each user needs to take in charge to authenticate other users. If there are n users participated in such application, each user can use the conventional authentication scheme for $n-1$ times to authenticate other users. The complexity of this approach is $O(n)$. This complexity may become the bottleneck of a group-oriented application.

In this paper, we propose a new type of authentication, called group authentication, which authenticates all users at once. The group authentication is specially designed to support group-oriented applications. In the group authentication, the group manager (GM) is responsible to register all group members initially. During registration, the GM uses Shamir's secret sharing (SS) scheme [16] to issue a private token to each group member. Later, all users participated in the group authentication work together without the assistance of the GM to authenticate each other. We propose a noninteractive basic t-secure m-user n-group authentication scheme ((t, m, n) GAS), where t is the threshold of the proposed scheme, m is the number of users participated in a group-oriented application, and n is the total number of group members. This basic scheme only works for synchronous communications. Then, we propose an asynchronous (t, m, n) GAS. The proposed (t, m, n) GAS can determine whether all users participated in a group communication belong to the same group. The complexity of a (t, m, n) GAS is $O(1)$ which is more efficient than the complexity using a conventional user authentication scheme to authenticate multiple users. The proposed (t, m, n) GAS is sufficient if all users are group members; however, if there are nonmembers, it can be used as a preprocess before applying conventional user authentication to identify nonmembers.

The rest of this paper is organized as follows: in Section 2, we review of Shamir's (t, n) SS scheme. In Section 3, we introduce the model of our proposed group authentication including adversaries and security requirements. We present a noninteractive basic (t, m, n) GAS in Section 4. Then, we propose an asynchronous (t, m, n) GAS in Section 5, and an asynchronous (t, m, n) GAS with multiple authentications in Section 6. We conclude in Section 7.

2 Review of Shamir's (t, n) SS Scheme [16]

In this section, we review Shamir's (t, n) SS scheme based on a linear polynomial. There are n shareholders, $U = \{U_1, U_2, \ldots, U_n\}$ and a dealer D. The scheme consists of two algorithms as illustrated in Fig. 1.

Shamir's (t, n) SS scheme satisfies security requirements of the SS scheme, that are, 1) with knowledge of any t or more than t shares can reconstruct the master secret s, and 2) with knowledge of fewer than t shares cannot get any information about the master secret s. Shamir's scheme is unconditionally secure since the scheme satisfies these two requirements without making any computational assumption. For more information on this scheme, readers can refer to the original paper [16].

• *The author is with the Department of Computer Science Electrical Engineering, School of Computing and Engineering, University of Missouri, 5100 Rockhill Road, Kansas City, MO 64110. E-mail: harnl@umkc.edu.*

Manuscript received 25 Feb. 2012; revised 16 Aug. 2012; accepted 4 Oct. 2012; published online 12 Oct. 2012.
Recommended for acceptance by C. Nita-Rotaru.
For information on obtaining reprints of this article, please send e-mail to: tc@computer.org, and reference IEEECS Log Number TC-2012-02-0155.
Digital Object Identifier no. 10.1109/TC.2012.251.

Share generation

Dealer D picks a random polynomial $f(x)$ of degree t-1: $f(x) = a_0 + a_1 x + ... + a_{t-1} x^{t-1} \bmod p$, such that the secret is $s = f(0) = a_0$, and all coefficients, a_i, $i = 0,1,...,t$-1, are in the finite field $GF(p)$ with $p > s$. D computes n shares, $y_i = f(x_i)$, $i = 1,2,...,n$, where x_i is the public information associated with shareholder U_i. Then, dealer distributes each share y_i to corresponding shareholder U_i secretly.

Secret reconstruction

Assume that t shareholders, $\{U_1, U_2, ..., U_t\}$, want to recover the secret s. Shareholders release their shares and use the Lagrange interpolating formula, $s = f(0) = \sum_{i=1}^{t} f(x_i) \prod_{r=1, r \neq i}^{t} \frac{-x_r}{x_i - x_r} \bmod p$, to recover the secret.

Fig. 1. Shamir's (t, n) SS scheme.

3 Model of Group Authentication

3.1 Group Authentication

We assume that there are m users, P_i, $i = 1, 2, \ldots, m$, participated in a group-oriented application. These users want to make sure whether they all belong to the same group with n group members, $U_i \in U$, $i = 1, 2, \ldots, n$, at the beginning of the application. The group authentication can be used to determine whether all users belong to the same group. The group authentication is no longer a one-to-one type of authentication as most conventional user authentication schemes which have one prover and one verifier; but it is a many-to-many type of authentication which has multiple provers and multiple verifiers. In fact, each user in the group authentication acts both roles of the prover and the verifier. There are only two possible outcomes of the group authentication, that are, either all users belong to the same group or there are nonmembers. Thus, the group authentication is sufficient if all users are group members; however, if there are nonmembers, it can be used as a preprocess before applying conventional user authentication to identify nonmembers.

3.2 Adversaries in the Group Authentication

We consider two types of adversaries in the group authentication: outside attacker and inside attacker. The outside attacker is an adversary who does not own any valid token generated by GM during system set up. The outside attacker may try to impersonate to be a group member participated in the authentication. In our proposed solution, each user needs to release a value based on his token obtained from GM initially. Group authentication is based on all released values from users. Since values are released asynchronously, the outside attacker can wait to release a "good" value last after knowing all released values from others. There is another type of attackers. The inside attacker is a group member who owns a valid token obtained from GM. The inside attacker may try to generate "good" tokens from his own token or to collude with other group members to recover the secret of GM. In this paper, we propose group authentication schemes which can resist up to $t - 1$ (i.e., t is the *threshold*) colluded inside adversaries.

3.3 An Informal Model of Proposed Scheme

In a conventional user authentication, the prover P_i wants to prove to a verifier that he is a particular user with identity, U_i (i.e., $P_i = U_i$). In most user authentication schemes, the prover needs to interact with the verifier in the following way. First, the verifier sends a random challenge r to the prover. The prover uses his secret key to compute a response and sends the response to the verifier. The verifier can authenticate the prover to be a particular user based on the response. We can use the following notations to represent the user authentication. The prover P_i uses his secret key to compute and releases a value c_i using his secret key and a random challenge sent by the verifier as inputs. In the user authentication, there is an algorithm, UA, which allows the verifier to verify that the prover is a particular user, i.e.,

$$UA\{c_i | i \in P\} = \begin{cases} 0 \rightarrow P_i \neq U_i, \\ 1 \rightarrow P_i = U_i. \end{cases}$$

In this paper, we propose a new notion, called t-secure m-user n-group authentication scheme ((t, m, n) GAS.

Definition 1: t-Secure m-User n-Group Authentication Scheme ((t, m, n) GAS). *Let t, m, n, be positive integers with $t \leq m \leq n$. A t-secure m-user n-group authentication scheme has the following properties: 1) the scheme can resist up to $t - 1$ colluded group members, and 2) for m users, the scheme can determine whether these users belong to the same group with n members.*

In a (t, m, n) GAS, the GM selects the secret s and computes tokens, $s_i, i, = 1, 2, \ldots, n$, for the group with n members during system set up. The GM sends each token s_i to each group member $U_i \in U$ secretly and makes $H(s)$ publicly known, where $H(s)$ is the one-way function of the secret. In a (t, m, n) GAS, there are m users, P_i, $i, = 1, 2, \ldots, m$, and each user uses his token to compute and release c_i. There is an algorithm, *GA*, which allows users to verify that all released values are valid, where F is a public function. That is,

$$GA\{H(s) \underline{?} H(F(c_1, c_2, \ldots, c_m))\} = \begin{cases} 0 \rightarrow P \not\subset U, \\ 1 \rightarrow P \subseteq U. \end{cases}$$

The (t, m, n) GAS can only detect the existence of nonmembers, but it cannot identify nonmembers. One unique feature of a (t, m, n) GAS is that all users are authenticated at once, but only one prover is authenticated by one verifier in a conventional user authentication.

The (t, m, n) GAS is described by the following scheme:

Initialization. All system parameters are generated and published by the GM in initialization.

Distribution. The GM generates and distributes token s_i for each group member U_i, secretly, $i, = 1, 2, \ldots, n$.

Authentication. Each user computes and releases a value, c_i, using his token. After receiving all c_i, $i = 1, 2, \ldots, m$, (i.e., $t \leq m \leq n$), users verify whether these values are released by members of the group. If the verification fails, additional user authentication is needed to identify nonmembers.

3.4 Security Model of Proposed (t, m, n) Group Authentication Scheme

We will propose a noninteractive basic (t, m, n) GAS using Shamir's (t, n) SS scheme. However, this basic scheme works properly if all values are released simultaneously. We will modify the basic scheme to a noninteractive asynchronous (t, m, n) GAS. Finally, we propose a noninteractive asynchronous (t, m, n) GAS for multiple authentications with the following properties:

Correctness. The outcome of this scheme is positive if all users are group members; otherwise, there are nonmembers.

Efficiency. If the outcome of the proposed scheme is negative, the proposed (t, m, n) GAS can be used as a preprocess of conventional user authentication scheme to identify nonmembers. Thus, the proposed (t, m, n) GAS must be efficient. In addition, in one of our proposed schemes, the same tokens generated by the GM initially can be reused for multiple authentications. This arrangement can improve the efficiency of token distribution.

Security. The scheme must be able to resist up to $t-1$ colluded inside adversaries. In addition, since values are released asynchronously, any outside adversary cannot impersonate to be a member by forging a valid value after knowing at most $n-1$ values from other members. For group authentication with multiple authentications, there are multiple secrets to be recovered sequentially. The scheme must be able to protect uncovered secrets when some secrets have been recovered.

Flexibility. The scheme should work properly for various size m (i.e., $t \le m \le n$) of users participated in the authentication. In addition, the scheme must work properly for releasing values synchronously and asynchronously.

4 Basic (t, m, n) Group Authentication Scheme

In the following discussion, we assume that there are n group members, $U_i \in U$, $i = 1, 2, \ldots, n$, registered at the GM to form a group. During registration, GM selects a random $(t-1)$th (i.e., $t \le n$) degree polynomial $f(x)$ with $f(0) = s$, and computes secret tokens of members as $y_i = f(x_i)$, $i = 1, 2, \ldots, n$, where x_i is the public information associated with member U_i. GM sends each token y_i to member U_i secretly. GM makes *H(s)* publicly known, where H is a one-way function.

Remark 1. The threshold t is an important security parameter that affects the security of the group authentication. Using a (t, n) SS scheme to issue tokens in the registration can prevent up to $t-1$ colluded inside attackers to derive the secret polynomial $f(x)$ selected by the GM and to forge valid tokens. Furthermore, since (t, n) SS scheme has been used to issue tokens, the GM only needs to issue new token to any new member who just joins the group. On the other hand, when any member leaves the group, it assumes that one token has been compromised. The GM needs to make this information available to all remaining members. The GM keeps on counting the number of leaving members. When this number reaches the threshold, t, GM needs to issue new tokens to all remaining group members.

From now on, we assume that there are m (i.e., $t \le m \le n$) users, P_i, $i = 1, 2, \ldots, m$, with their tokens $\{f(x_1), f(x_2), \ldots, f(x_m)\}$ participated in the group authentication. The basic idea of this scheme is that each user releases the token obtained from the GM during registration. If all released tokens are valid, the interpolation of the released tokens can reconstruct the polynomial $f(x)$ and obtain the secret s. The published one-way value of the secret, $H(s)$, is used to compare with the one-way value of the reconstructed secret. We outline the scheme in Fig. 2.

Theorem 1. *Scheme 1 has the properties of the t-secure m-user n-group authentication scheme as we described in Section 3.4.*

Correctness. It is obvious that the secret can be successfully reconstructed in Scheme 1 if users are all members and act honestly to release their Lagrange components.

Efficiency. The communication overhead is very limited. Every participant only needs to release a value to all other participants. This can be accomplished by sending a broadcast message. Every participants only needs to compute polynomial operations and, therefore, can be efficiently implemented on various platforms. In addition, not like the most conventional user authentication schemes which authenticate one user each time, this proposed group authentication scheme authenticates all users at once.

Token generation

GM selects a random polynomial *f(x)* of degree t-1: $f(x) = a_0 + a_1 x + \ldots + a_{t-1} x^{t-1} \bmod p$, such that the secret is $s = f(0) = a_0$, and all coefficients, a_i, $i = 0,1,\ldots,t$-1, are in the finite field $GF(p)$ with $p>s$. GM computes n tokens, $f(x_i)$, $i = 1,2,\ldots,n$, where x_i is the public information associated with group member U_i. Then, GM distributes each token $f(x_i)$ to corresponding group member U_i secretly.

Group authentication

Assume that m users, $\{P_1, P_2, \ldots, P_m\}$, participated in the group authentication.

1. Each user P_i releases his token $f(x_i)$ to all other users.
2. After knowing all tokens, $f(x_i)$, $i = 1,2,\ldots,m$, following Lagrange interpolating formula, each user computes $s' = \sum_{i=1}^{m} f(x_i) \prod_{r=1, r\neq i}^{m} \frac{-x_r}{x_i - x_r} \bmod p$. If $H(s') = H(s)$, all users have been authenticated successfully; otherwise, there are non-members.

Fig. 2. Basic (t, m, n) group authentication-Scheme 1.

Security. Shamir's secret reconstruction scheme can be generalized to take more than t shares as

$$s = f(0) = \sum_{i=1}^{m} f(x_i) \prod_{r=1, r\neq i}^{m} \frac{-x_r}{x_i - x_r} \bmod p,$$

when there are m (i.e., $t \le m \le n$) shareholders with their shares, $\{f(x_1), f(x_2), \ldots, f(x_m)\}$, in the secret reconstruction. In Scheme 1, if every released token $f(x_i)$ is a valid token obtained from the GM initially, the one-way value of the recovered secret must satisfy $H(s') = H(s)$. However, if there is any nonmember who does not own a valid token on the polynomial $f(x)$, the reconstructed secret will be different from the secret s. Furthermore, the nonmember cannot forge a valid token after knowing other released tokens since all tokens are released simultaneously.

Flexibility. The authentication scheme works properly for m users with $t \le m \le n$. However, Scheme 1 is secure only when all tokens are released synchronously.

Remark 2. Scheme 1 cannot prevent outside adversaries to impersonate to be group members when there are more than t users and all tokens are released asynchronously. This is because if there are more than t users, the adversary needs only t valid tokens to recover the secret polynomial $f(x)$ and to forge a valid token. The adversary can be the last one to release his token. In the next section, we modify Scheme 1 to propose an asynchronous (t, m, n) GAS. The modified scheme needs to prevent the adversary to obtain other members' tokens from released values.

Token generation

GM selects k (i.e., $kt > n$-1) random polynomials, $f_l(x)$, l=1,2,...,k, having degree t-1 each and generates tokens, $f_l(x_i)$, l=1, 2,..., k, for each group member U_i. For any secret, s, GM finds integers, w_j, d_j, j=1,2,...,k, in $GF(p)$, such that $s = \sum_{j=1}^{k} d_j f_j(w_j)$, where $w_i \neq w_j$, for every pair of i and j. The dealer makes these integers, w_j, d_j, j=1,2,...,k, and $H(s)$ publicly known.

Group authentication

1. Each user P_i uses his tokens, $f_l(x_i)$, l=1,2,...,k, to compute and release one Lagrange component, $c_i = \sum_{j=1}^{k} d_j f_j(x_i) \prod_{r=1, r\neq i}^{m} \frac{w_j - x_r}{x_i - x_r} \bmod p$.
2. After knowing, c_r, r=1,2,...,m, each user computes $s' == \sum_{r=1}^{m} c_r \bmod p$. If $H(s') = H(s)$, all users have been authenticated successfully; otherwise, there are non-members.

Fig. 3. Asynchronous (t, m, n) group authentication-Scheme 2.

5 Asynchronous (t, m, n) Group Authentication Scheme

In this section, we propose a (t, m, n) GAS which allows m (i.e., $t \leq m \leq n$) users to release their values asynchronously in the group authentication. The basic idea is that the GM needs to select k (i.e., $kt > n-1$, we will use this condition in Theorem 2) random polynomials, $f_l(x)$, $l = 1, 2, \ldots, k$, having degree $t-1$ each, and generates tokens, $f_l(x_i)$, $l = 1, 2, \ldots, k$, for each group member U_i. For any secret, s, GM can always find integers, w_j, d_j, $j = 1, 2, \ldots, k$, in $GF(p)$, such that $s = \sum_{j=1}^{k} d_j f_j(w_j)$, where $w_i \neq w_j$, for every pair of i and j. GM makes these integers, w_j, d_j, $j = 1, 2, \ldots, k$, and $H(s)$ publicly known.

In the group authentication, each user P_i uses his tokens, $f_l(x_i)$, $l = 1, 2, \ldots, k$, to compute and release one Lagrange component, $c_i = \sum_{j=1}^{k} d_j f_j(x_i) \prod_{r=1, r\neq i}^{m} \frac{w_j - x_r}{x_i - x_r} \bmod p$. Thus, after knowing c_i, $i = 1, 2, \ldots, m$, each user can recover the secret as $s\prime = \sum_{i=1}^{m} c_i \bmod p$. The authentication is based on the comparison between the one-way value of the reconstructed secret and the published one-way value of the secret. However, it is computationally impossible to derive tokens from c_i. We outline this scheme in Fig. 3.

Theorem 2. *Scheme 2 has the properties of the t-secure m-user n-group authentication scheme as we described in Section 3.4 if $kt > n-1$.*

Correctness. It is obvious that the secret can be successfully reconstructed in Scheme 2 if users are all members and act honestly to release their Lagrange components. If there is any nonmember who does not own any valid token, the nonmember cannot release a valid Lagrange component. Thus, the recovered secret must be different from the secret s.

Efficiency. The communication overhead is very limited. Every participant only needs to release a value to all other participants. The most time-consuming operation for each user is to compute the Lagrange component as

$$c_r = \sum_{l=1}^{k} d_{i,l} f_l(x_r) \prod_{v=1, v\neq r}^{m} \frac{w_l - x_v}{x_r - x_v} \bmod p,$$

in Step 1. This polynomial evaluation becomes the main computation in our proposed scheme. However, the modulus p in our polynomial computation is much smaller than the modulus (e.g., 1,024 bits) used in most public-key cryptosystems, such as RSA cryptosystem [14]. In addition, not like most conventional user authentication schemes which authenticate one user each time, this proposed group authentication scheme authenticates all users at once. Thus, the proposed scheme is very efficient in comparing with most user authentication schemes.

Security. Since the tokens are generated by polynomials, $f_l(x)$, $l = 1, 2, \ldots, k$, having degree $t-1$ each, Scheme 2 can resist up to $t-1$ colluded inside adversaries trying to recover the polynomials selected by the GM initially.

For any outside adversary participated in the group authentication, the secret tokens, $f_l(x_i)$, $l = 1, 2, \ldots, k$, are protected unconditionally in the released Lagrange component $c_i = \sum_{j=1}^{k} d_j f_j(x_i) \prod_{r=1, r\neq i}^{m} \frac{w_j - x_r}{x_i - x_r} \bmod p$. The outside adversary cannot derive any private token from each released Lagrange component. On the other hand, each released Lagrange component is a linear function of kt coefficients of polynomials, $f_l(x)$, $l = 1, 2, \ldots, k$, with each polynomial having degree $t-1$. In the following discussion, we consider the scenario that gives the adversary the most information to recover private tokens. If there are n users participated in the group authentication, the adversary can obtain at most $n-1$ Lagrange components if the adversary is the last one to release his value. Since $kt > n-1$, this condition prevents the outside adversary to solve the secret polynomials, $f_l(x)$, $l = 1, 2, \ldots, k$. Thus, the outside adversary cannot forge any valid Lagrange component when values are released asynchronously.

Flexibility. The authentication scheme works properly for m users with $t \leq m \leq n$. In addition, when values are released asynchronously, the proposed scheme can still detect any outside adversary even the adversary releases his Lagrange component last. This is because the proposed scheme requires every user to contribute a valid Lagrange component. Since the adversary does not own any token generated by the GM initially, the adversary cannot contribute a valid Lagrange component.

Remark 3. For any secret, s, the GM needs to select $w_i \neq w_j$, for every pair of i and j and the secret is $s = \sum_{j=1}^{k} d_j f_j(w_j)$. If $w = w_i = w_j$, for every pair of i and j, the outside adversary can still forge valid tokens. This is because in this case the secret, $s = \sum_{j=1}^{k} d_j f_j(w)$, is a share of the additive sum of polynomials as $\sum_{j=1}^{k} d_j f_j(x)$ having degree $t-1$. Each P_i needs to use his tokens to compute and release the Lagrange component, $c_i = \sum_{j=1}^{k} d_j f_j(x_i) \{\prod_{r=1, r\neq i}^{m} \frac{w - x_r}{x_i - x_r}\} \bmod p$. The adversary can recover the additive sum of tokens, $\sum_{j=1}^{k} d_j f_j(x_i)$, from each released Lagrange component c_i. Thus, the adversary can recover the additive sum of polynomials, $\sum_{j=1}^{k} d_j f_j(x)$, and forge a valid Lagrange component from t additive tokens. Scheme 2 protects the secrecy of tokens, but, it is a one-time authentication since the secret is no longer a secret once it has been recovered. In the next section, we extend the design used in Scheme 2 to propose an asynchronous (t, m, n) GAS with multiple authentications. In the proposed scheme, tokens generated by the GM initially can be reused to recover multiple secrets. In addition, any recovered secrets will not compromise the secrecy of uncovered secrets.

6 Asynchronous (t, m, n) Group Authentication Scheme with Multiple Authentications

In this section, we propose an asynchronous (t, m, n) GAS in which tokens obtained from the GM initially can be reused for multiple authentications. The basic idea of Scheme 3 is that the GM needs to select two large public primes, p and q, such that q divides $p-1$, $GF(q)$ is a unique subgroup of $GF(p)$ with order q, and every g_i is a generator of $GF(q)$. GM select two random polynomials, $f_l(x)$, $l = 1, 2$, having degree $t-1$ each with coefficients in $GF(q)$ and generates tokens, $f_l(x_i)$, $l = 1, 2$, for each group member U_i. There are multiple secrets selected by the GM. For any secret, s_i, GM first selects random integers, $w_{i,j}, d_{i,j}$, $j = 1, 2$, in $GF(q)$, where $w_{i,1} \neq w_{i,2}$, and the secret, s_i, is determined as

$$s_i = g_i^{\sum_{j=1}^{2} d_{i,j} f_j(w_{i,j}) \bmod q} \bmod p.$$

GM makes these integers, $w_{i,j}, d_{i,j}$, $j = 1, 2$, and $(g_i, H(s_i))$ publicly known.

In the group authentication to reconstruct the secret, s_i, each user P_r uses his tokens, $f_l(x_r)$, $l = 1, 2$, to compute one Lagrange component, $c_r = \sum_{l=1}^{2} d_{i,l} f_l(x_r) \prod_{v=1, v\neq r}^{m} \frac{w_{i,l} - x_v}{x_r - x_v} \bmod q$. Then, P_r computes and releases $e_r = g_i^{c_i} \bmod p$. Thus, after knowing e_r, $r = 1, 2, \ldots, m$, each user can recover the secret as $s_i = \prod_{r=1}^{m} e_r \bmod p$. The authentication is based on the comparison between the one-way value of the reconstructed secret and the published one-way value of the secret. However, it is computationally impossible to derive tokens from e_r. We outline this scheme in Fig. 4.

Theorem 3. *Scheme 3 has the properties of the t-secure m-user n-group authentication scheme as we described in Section 3.4.*

Correctness. If each user P_r is a group member and acts honestly in the group authentication, then in Step 2, the recovered value is equivalent to

$$s_i' = \prod_{r=1}^{m} e_r \bmod p = g_i^{\sum_{r=1}^{m} c_r \bmod q} \bmod p$$

$$= g_i^{\sum_{l=1}^{2} d_{i,l} f_l(x_r) \prod_{v=1, v\neq r}^{m} \frac{w_{i,l} - x_v}{x_r - x_v} \bmod q} \bmod p$$

$$= g_i^{\sum_{j=1}^{2} d_{i,j} f_j(w_{i,j}) \bmod q} \bmod p = s_i.$$

□

Efficiency. The communication overhead is very limited. Every participant only needs to release a value to all other participants. The most time-consuming operation for each user is to compute the modular exponentiation in Step 1. This computation is the same as most public-key computations, such as RSA cryptosystem [14]. Not like most conventional user authentication schemes which authenticate one user each time, this proposed group authentication scheme authenticates all users at once.

In our proposed scheme, finding each Lagrange component, $c_r = \sum_{l=1}^{k} d_{i,l} f_l(x_r) \prod_{v=1, v\neq r}^{m-} \frac{w_{i,l} - x_v}{x_r - x_v} \bmod p$, from each released value, $e_r = g_i^{c_i} \bmod p$, needs to solve the discrete logarithm. It is computationally infeasible to derive any private token from the released values. Thus, same tokens generated by the GM initially can be reused for multiple authentications. This arrangement can improve the efficiency of token distribution.

Security. Since the tokens are generated by a polynomial having degree $t-1$, this scheme can resist up to $t-1$ colluded inside adversaries trying to recover the polynomial $f_l(x)$ selected by the GM initially.

For any outside adversary participated in the group authentication, to derive each Lagrange component, c_r, from each released value, $e_r = g_i^{c_i} \bmod p$, needs to solve the discrete logarithm and it is computationally infeasible. In other words, private tokens are protected from the released values. Thus, the outside adversary cannot forge a valid value since he does not have any valid token.

Token generation

GM selects two large public primes, p and q, such that q divides p-1, $GF(q)$ is a unique subgroup of $GF(p)$ with order q, and every g_i is a generator of $GF(q)$. Then, GM selects two random polynomials, $f_l(x)$, $l = 1,2$, having degree t-1 each with coefficients in $GF(q)$ and generates tokens, $f_l(x_i)$, for $l = 1,2$, for each group member U_i. There are multiple secrets selected by the GM. For each secret, s_i, GM first selects random integers, $w_{i,j}, d_{i,j}$, $j = 1,2$, in $GF(q)$, where $w_{i,1} \neq w_{i,2}$, and the secret, s_i, is determined as $s_i = g_i^{\sum_{j=1}^{2} d_{i,j} f_j(w_{i,j}) \bmod q} \bmod p$. The dealer makes these integers, $w_{i,j}, d_{i,j}$, $j = 1,2$, and $(g_i, H(s_i))$ publicly known.

Group authentication We consider the group authentication to reconstruct the secret s_i.

1. Each user P_r uses his tokens, $f_l(x_r)$, $l = 1,2$, to compute one Lagrange component, $c_r = \sum_{l=1}^{2} d_{i,l} f_l(x_r) \prod_{v=1, v\neq r}^{m} \frac{w_{i,l} - x_v}{x_r - x_v} \bmod q$. Then, P_r computes and releases $e_r = g_i^{c_i} \bmod p$.
2. After knowing, e_r, $r = 1,2,...,m$, each user computes $s_i' = \prod_{r=1}^{m} e_r \bmod p$. If $H(s_i') = H(s_i)$, all users have been authenticated successfully; otherwise, there are non-members.

Fig. 4. Asynchronous (t, m, n) group authentication scheme for multiple authentications scheme.

To derive any exponent, $\sum_{j=1}^{2} d_{i,j} f_j(w_{i,j}) \bmod q$, from each recovered secret,

$$s_i = g_i^{\sum_{j=1}^{2} d_{i,j} f_j(w_{i,j}) \bmod q} \bmod p,$$

also needs to solve the discrete logarithm and it is computationally infeasible. In other words, private tokens are protected from any recovered secret. Since each secret uses random integers, $w_{i,j}, d_{i,j}$, $j = 1, 2$, the secrecy of uncovered secrets is protected from the recovered secrets.

In the following, we explain why the GM needs to select random integers, $w_{i,j}, d_{i,j}$, $j = 1, 2$, with the condition $w_{i,1} \neq w_{i,2}$ initially. If $w = w_{i,1} = w_{i,2}$, the outside adversary can still forge valid value after knowing t released values, e_r. This is because, if $w = w_{i,1} = w_{i,2}$, the exponent of the secret,

$$s_i = g_i^{\sum_{j=1}^{2} d_{i,j} f_j(x_r)} \bmod p,$$

is a share of the additive sum of polynomials, $\sum_{j=1}^{2} d_{i,j} f_j(x)$ having degree $t-1$. The adversary can recover the modular exponentiation of the additive tokens, $\sum_{j=1}^{2} d_{i,j} f_j(x_r)$, from each released value, e_r, as

$$h_r = g_r^{\sum_{l=1}^{2} d_{i,l} f_l(x_r)} = (e_r)^{\left(\prod_{v=1, v\neq r}^{m} \frac{w-x_v}{x_r-x_v} \bmod q\right)^{-1}} \bmod p.$$

After knowing h_r, $r = 1, 2, \ldots, t$, the adversary can forge a valid value e_r without being detected.

Flexibility. The authentication scheme works properly for m users with $t \leq m \leq n$. In addition, when values are released asynchronously, the proposed scheme can still detect any outside adversary even the adversary releases his Lagrange component last.

7 Conclusions

We propose a special type of authentication, called group authentication, which is specially designed for group-oriented applications. The proposed group authentication is no longer a one-to-one type of authentication and it is a many-to-many type of authentication. Group authentication can authenticate multiple users at once. Our proposed (t, m, n) group authentication schemes, Schemes 1 and 2, are very efficient since the schemes are based on Shamir's (t, n) SS scheme and the computations involve only polynomial operations. Scheme 3 allows tokens obtained from the GM initially to be reused for multiple authentications. Group authentication opens a new research direction for the SS.

References

[1] B. Bruhadeshwar and S.S. Kulkarni, "Balancing Revocation and Storage Trade-Offs in Secure Group Communication," *IEEE Trans. Dependable and Secure Computing*, vol. 8, no. 1, pp. 58-73, Jan./Feb. 2011.

[2] M.A. Catur Bhakti, A. Abdullah, and L.T. Jung, "EAP-Based Authentication for Ad Hoc Network," *Proc. Seminar Nasional Aplikasi Teknologi Informasi (SNATI) Conf.*, pp. C-133-C-137, 2007.

[3] P. Caballero-Gil and C. Hernández-Goya, "Self-Organized Authentication in Mobile Ad-Hoc Networks," *J. Comm. and Networks*, vol. 11, no. 5, pp. 509-517, 2009.

[4] S. Capkun, L. Buttyan, and J.P. Hubaux, "Self-Organized Public-Key Management for Mobile Ad Hoc Networks," *IEEE Trans. Mobile Computing*, vol. 2, no. 1, pp. 52-64, Jan.-Mar. 2003.

[5] M.L. Das, "Two-Factor User Authentication in Wireless Sensor Networks," *IEEE Trans. Wireless Comm.*, vol. 8, no. 3, pp. 1086-1090, Mar. 2009.

[6] I. Downnard, "Public-Key Cryptography Extensions into Kerberos," *IEEE Potentials*, vol. 21, no. 5, pp. 30-34, 2003.

[7] L. Harn and J. Ren, "Generalized Digital Certificate for User Authentication and Key Establishment for Secure Communications," *IEEE Trans. Wireless Comm.*, vol. 10, no. 7, pp. 2372-2379, July 2011.

[8] IEEE CS, "802.1X, IEEE Standard for Local and Metropolitan Area Networks, Port-Based Network Access Control," *The Inst. of Electrical and Electronics Engineers*, Inc., 2004.

[9] W.C. Ku, "Weaknesses and Drawbacks of a Password Authentication Scheme Using Neural Networks for Multiserver Architecture," *IEEE Trans. Neural Networks*, vol. 16, no. 4, pp. 1002-1005, July 2005.

[10] R. Oppliger, R. Hauser, and D. Basin, "SSL/TLS Session-Aware User Authentication," *Computer*, vol. 41, no. 3, pp. 59-65, 2008.

[11] H.A. Park, J.W. Hong, J.H. Park, J. Zhan, and D.H. Lee, "Combined Authentication-Based Multilevel Access Control in Mobile Application for DailyLifeService," *IEEE Trans. Mobile Computing*, vol. 9, no. 6, pp. 824-837, June 2010.

[12] A.A. Pirzada and C. McDonald, "Kerberos Assisted Authentication in Mobile Ad-Hoc Networks," *Proc. 27th Australasian Computer Science Conf. (ACSC)*, vol. 26, no. 1, pp. 41-46, 2004.

[13] K. Ren, S. Yu, W. Lou, and Y. Zhang, "Multi-User Broadcast Authentication in Wireless Sensor Networks," *IEEE Trans. Vehicular Technology*, vol. 58, no. 8, pp. 4554-4564, Oct. 2009.

[14] R. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Comm. ACM*, vol. 21, no. 2, pp. 120-126, 1978.

[15] P. Sakarindr and N. Ansari, "Survey of Security Services on Group Communications," *IET Information Security* vol. 4, no. 4, pp. 258-272, Dec. 2010.

[16] A. Shamir, "How to Share a Secret," *Comm. ACM*, vol. 22, no. 11, pp. 612-613, 1979.

[17] J. Yan, A. Blackwell, R. Anderson, and A. Grant, "Password Memorability and Security: Empirical Results," *IEEE Security and Privacy Magazine*, vol. 2, no. 5, pp. 25-31, Sept./Oct. 2004.