Asian Journal of Information Technology, 5 (3) : 237-241, 2006

A New Group Key Exchange Protocol Based on Zero-Knowledge Set

Sun Haibo, Lin Dongdai and Xue Rui
The State Key Laboratory of Information Security, Institute of Software,
Chinese Academy of Sciences, Beijing, 100080, China
The Graduate School of Academy of Sciences, Beijing 100039, China

Abstract: Zero-knowledge set is a set that has zero-knowledge quality. The structure of the set makes that a prover can commit to any arbitrary finite set of strings and for any string, reveal with a proof whether a given element is in the set without revealing any knowledge beyond the verity of these membership assertions. In this study, we advance a new Group Key Distribution protocol based on zero-knowledge set and polynomial computation so that the identity and number of the group members can be concealed and realize key distribution at the same time. The protocol not only provides a dynamic distribution on a group key, but also guarantees nobody can get additional information about other members except the group key. Compared with previous work, our protocol can provide more security and is suitable for some special network application, such as military action and anonymous e-commerce meeting.

Key word: Zero-knowledge set, pederson commitment, group key distribution protocol

INTRODUCTION

The notion of zero-knowledge quality had been proposed for about twenty years, but the past research didn't deal with the zero-knowledge quality of set. Till 2003, Silvio Micali, Michael Rabin and Joe Kilian[1,2] advanced the notion of zero-knowledge set for the first time. The main idea of zero-knowledge set focus on that for any arbitrary finite set, the prover can construct a commitment so that for any element, the prover can provide a proof to prove whether this element belongs to the set and at the same time don't leak any information about the set, such as the size of the set. In[1], the authors proposed one scheme of constructing zero-knowledge set based on Pederson commitment scheme[3]. In his scheme, for the computation of commitment the authors adopt structure of binal-tree and non-interactive proof[4] during the verified process. Because the process of commitment is similar to the tree-based group key exchange process, in this paper we will attempt to combine these two process to propose a new group key exchange scheme so that in our scheme, the session key can be securely distribute to group members and at the same time, this scheme have favorable zero-knowledge quality. In our group key distribution scheme, we can directly adopt Pederson commitment method to realize the zero-knowledge quality, but by this method, the main operation is exponential computation that has relative heavy overhead. So in order to reduce the overhead of computation and communication, in our scheme, we will adopt polynomial computation to realize the zero-knowledge quality.

The generic group key exchange protocols mainly include group key agreement protocols and group key distribution protocols. Currently correlative researches mainly don't focus on zero-knowledge quality of group but on the method of key agreement or distribution, or the research of other security property. Now mostly group key exchange protocols are based on authentication, each member of group must complete mutual authentication during key exchange. In another word, in the process of key exchange, each member knows the object of communication clearly. But in some special realm, the group members need to communicate anonymously such as military action or anonymous network meeting, these traditional group key exchange scheme are not adaptive. So in this study, we attempt to propose a new scheme to realize this purpose.

The organization of this paper is as follow. The first, we introduce the basic definition, then we propose our scheme and the analysis of the security properties of this scheme. In the end, the conclusion and the future directions.

THE BASIC DEFINITIONS AND RELATIVE KNOWLEDGE

Basic definitions

Definition 1: A set S has zero-knowledge quality if S satisfies:

- The number of elements of S is finite.
- For S, the prover can compute a commitment C_s.
- For any element x_i, the prover can give a non-

interactive proof to prove $x_i \in S$ or $x_i \notin S$ and doesn't leak any information about S.

In former zero-knowledge proof, it is imperfect to prove $x_i \notin S$. In general, given set $S=\{x_1, x_2 \ldots x_n\}$ and any arbitrary element x, one can prove $x \neq x_1 \ldots x \neq x_n$. But in the proof, the prover leaks some information about the set such as the number of elements.

Definition 2: If for any positive polynomial, P(x), $\lim P(x)\varepsilon((x) = 0$, then we call $\varepsilon(\bullet)$ negligible functions.

Definition 3: If for any polynomial-time arithmetic A, polynomial-time computable function H has $\Pr\{\forall x,y.\ x \neq y,\ H(x) = H(y)\} = \varepsilon(k)$ and the reversed computation of H is computationally infeasible, we call H collision-free hash function[5,6].
We prescribe:

- $x \xleftarrow{R} S$ denotes the act of choosing an element x at random according to S, here S can be probability space or finite set.
- $P_1[x_1 \xleftarrow{R} S_1; x_2 \xleftarrow{R} S_2; \ldots : p(x_1, x_2 \ldots)]$ denotes the probability that $p(x_1, x_2 \ldots)$ will be true after the ordered execution of the probabilistic assignments $x_1 \xleftarrow{R} S_1; x_2 \xleftarrow{R} S_2; \ldots$

The zero-knowledge proof system include two processes: prover provide proof and verifier validate commitment. In our scheme, for convenience, we define two functions as follow:

- $\{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^* \times \{0,1\}^*$: Commit $(\sigma, x) = (c, r)$, here s is a public nonce as system parameter, x is the object of commitment, (c, r) is the commitment of x.
- $\{0,1\}^* \times \{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^*$: Verify $(s, c, r) = D(x)$, this function denote the verification of x.

Based on these definitions and marks, in this study, we will discuss three properties of our scheme: completeness, soundness and zero-knowledge.

Completeness:

$$\forall x \in \{0,1\}^*.\ P_r\{(c,r) \xleftarrow{R} \text{Commit}(\sigma,x):\ \} = 1.$$
$$\text{Verify}(\sigma,c,r) = D(x)$$

Completeness denote that the prover can make commitment of a set and for any element x, verifier can verify the validity of the commitment.

Soundness: for any given function COMMIT,

$$\Pr\{(c,r),(c',r') \xleftarrow{R} \text{COMMIT}(\sigma,x):$$
$$\text{VERIFY}(\sigma,c,r) \neq \text{VERIFY}(\sigma,c',r') \neq \perp\} = 1.$$

Soundness denotes for one x, the prover can't provide feigned commitment to cheat verifier.

Zero-Knowledge:

$$\forall m_1, m_2, \in \{0,1\}^*.,\ \ C(\sigma, m_1) = C(\sigma, m_2) \text{ here.}$$

$$C(\sigma,m) = \{(c,r) \xleftarrow{R} \text{COMMIT}(\sigma,m) : c\}$$

Zero-knowledge denote that the verifier can only get the value of D (x) according to the commitment and proof provided by the prover.

Hereinbefore we introduce the basic definition and property of zero-knowledge set. Our group key exchange scheme adopts binal-tree structure, the elements of set are distributed in the leaf nodes and each leaf node need to store some values, so it is necessary to introduce the definition of Merkle tree.

Merkle tree[1], In our scheme, the distribution of session key, the origination of commitment and the process of verification are all based on binal-tree structure. We use T_k to denote one complete binal-tree that has 2^k leaf nodes. The depth of one node is the distance from the node to root. For an inner node v, we denote its left-child node and right-child node v_0 and v_1 respectively, Its parent node parent (v). For any leaf node v, we denote the set of nodes in the path from v to root S_v. Obviously for one node v whose depth is k, the number of elements in S_v is k. Also we define one set $FS_v = \{u |\ u \notin S_v$ and parent $(u) \in S_v\}$, the elements in this set is the brother node of each element in S_v.

Merkle tree is a kind of special binal-tree. In Merkle tree, each node needs to store some values. Every leaf node v store one random value V_v and for every inner node v, the value needed to store is computed by $V_v = H(V_{v0}, V_{v1})$. Here the function H is a collision-free hash function introduced above. In another word, the value stored in inner node depends on the value stored in its children node. In our scheme, the computation of the commitment of set is based on Merkle tree and the verification of node v is also depended on the value stored in the element node of S_v and FS_v

Based on the structure of Merkle tree, we can store the elements of zero-knowledge set to the leaf nodes. Then the commitments from inner nodes to root depend on some polynomial computation and the commitment of

root will be the final commitment of the set. The material scheme will be introduced in next section.

GROUP KEY DISTRIBUTION SCHEME BASED ON ZERO-KNOWLEDGE SET

In this section, we will introduce our Group Key Distribution scheme based on zero-knowledge set and analyze the security of our scheme. We suppose that there are n users participate in the group key exchange and there is one believable commitment server. We denote the n users $M_1, M_2 \ldots M_n$, the believable server Prover. We also suppose the Prover has established secure communication channel with every user before the run of our scheme and these secure communication channel provide the authentication and data integrality between these users and Prover. First every user M_i chooses one nonce r_i and transmits it to the server (Prover) via secure communication channel. According to some criterion, the server decides m users who have the qualification to join the group session (m<n). We suppose these m users as $M_1 \ldots M_m$. Then the Prover randomly picks 2^k-degree polynomials h(x) and f(x), here $(f(r_i) \neq 0)$ and construct 2^k-degree polynomial g (x) so that $g(r_i)_{I=m+1,\ldots,m} \neq 0$. Prover compute h $(r_i)_{i=1,\ldots,n}$ and, $W(x) = g(x) \bullet K + h(x)$ then Prover transmits $h(r_i)_{i=1,\ldots,n}$ to M_i via the secure communication channel and broadcast the W (x) and g (x).

After these operations, the Prover constructs a Merkle tree T_k so that $(n << 2^k)$. Among the 2^k leaf nodes, the Prover picks n nodes $(v_1 \ldots v_n)$ to denote $M_1, M_2 \ldots M_n$. By the definition of Merkle tree, in our scheme, every node v needs to store two values m_v and C_v. The computation of these values is as follow:

- For a leaf node, if the node v denote one of the n users (suppose M_i), m_v=, otherwise. (The node that store 0 denote empty nodes those don't denote users).

We call the m leaf nodes that denote chosen users (group members) true nodes. Also for one inner node, if at least one of its children nodes is true node, it is also a true node. We call all other nodes vain nodes. Now according to the construct of W (x) and g (x), for one true leaf node v, $m_v = W(r_i)$ and for one vain leaf node v, if it denote one of these users, $m_v = h(r_i)$, otherwise $m_v = 0$. Then the Prover constructs another polynomial $Z(x) = x \bullet \prod M_{i \text{ is unchosen user}} (x - h(r_i)) \bullet t(x) + R$, here R is one random number, t (x) satisfies that $t(m_v) \neq 0$, where v is true node and the degree of Z(x) is 2^k. Then for every leaf node v, the prover computes $C_v = Z(m_v)$. Obviously, for any true leaf node v, $C_v = m_v \bullet \prod M_{i \text{ is unchosen user}} (m_v - h(r_i)) \bullet t(m_v) + R \neq R$, but for any vain node $C_v = R$. Now the values stored in leaf nodes (m_v, C_v) have been computed and the C_v is the commitment of leaf node v.

- For a inner node u, it also need to store two values m_u and C_u. The computation of the commitment of inner nodes is similar to[7] from leaf nodes to root nodes. The two values can be computed as follow:

We suppose the children nodes of u is u_0 and u_1 respectively, $m_u = H(C_{u0}, C_{u1})$, here H (a,b) is one collision-free hash function $(H(a,a) \neq a)$ and C_{u0}, C_{u1} denote the commitment of nodes u_0 and u_1 respectively. Then the Prover picks randomly polynomial T(x) and $C_u = T(m_u)$. C_u is the commitment of node u. Then the Prover computes every inner node from leaf nodes to root e and the C_e is the public commitment to all users.

From the description above, the computation of commitment is very fast. For every leaf nodes, it just needs two polynomial computations. And for any inner nodes, it just needs one hush computation and one polynomial computation. Obviously, all these computations are polynomial-time computable.

Now we give the verification method of users: for any user, the order of verification is same to the order of the origination of commitment. Here we denote the path from leaf node v to root P (v).

- For any chosen group member M_i, after transmitting the $h(r_i)$ and broadcasting the W(x), g(x), the Prover transmits the Z(x), T(x), every C_w stored in node w $(w \in S_v)$ in the path P (v) except root and the C_l stored in brother node l $(l \in FS_v)$ of every w to user M_i via secure communication channel. When received all these information, M_i first verify $g(r_i) \neq 0$ and $W(r_i) \neq h(r_i)$. From above description, the transmission of r_i and $h(r_i)$ is via secure communication channel, so other users can't verify these two inequality. If both inequality are satisfied, M_i can primary know he has been chosen to be one member of the group. Now according information received from the Prover, M_i can verify the commitment of every inner node u in P(v) step by step:
- $m_u = h(C_{u0}, C_{u1})$, here C_{u0}, C_{u1} denote the commitment of children nodes of u;
- $C_u = T(m_u)$;

By recursively verify the commitment of every inner node in P (v) till root e; M_i can compute the final commitment C_e. If this value equals to the commitment published by the Prover. The verification complete, the user can confirm that the Prover had given correct commitment and he had been chosen to be a member

of group. Now the user can compute the session key by $K = (W(r_i)-h(r_i))/g(r_i)$

- For any unchosen user M_i, the verification is similar to the chosen user. After receiving the verification information from the Prover, he first verify the value stored in leaf node v that denote himself $g(r_i) = 0$ and $W(r_i) = h(r_i)$. Now M_i knows that he hadn't been chosen to be the member of group, but in order to guarantee that the Prover had made righteous filtration and provided correct commitment, he must verify sequentially. The verification of commitment of inner nodes and root is the same to the verification of chosen user. If the commitment computed by himself equals to the value published by the Prover, the verification complete and this means the Prover hadn't cheated users. Otherwise this group key exchange is invalid.

Hhere according to the choice of g (x) and the construct of W (x), for any true node, K is computable. But for any vain node $g(r_i)=0$, so if a user hadn't been chosen to be one member of the group, he can't compute the session key.

Now we introduce the dynamic instances when new user wants to join the group or some user leaves. When some new user M_{n+1} wants to join the group, he first chooses his nonce r_{n+1} and transmits to the Prover and the Prover decide whether the new user can join the group according the criterion he used in the initial choice. Now the Merkle tree has 2^k-n empty leaf nodes, the Prover can pick randomly one v to denote the new user. If the user doesn't accord with the criterion, v will be a vain node. Now because the polynomial h (x) is unknown by all users, the Prover can choose one root of the polynomial t (x) as $h(r_{n+1})$, now he can construct one new polynomial h`(x) according to the n+1 pairs value $((r_1,h`(r_1)),(r_2)),...,(r_{n+1},h`(r_{n+1})))$ so that $h(r_i) = h`(r_i)$,i= 1,...n. Also the Prove must reconstruct polynomial g`(x) so that $g`(r_i) = g(r_i)_{i = 1,...,n}$ and $g`(r_{n+1}) = 0$ and transmit the $h`(r_{n+1})$ and other verification information to the new user. Because the degree of and h'(x) is both,so this operation is reasonable and feasible. Although after the new use joins, the value m_v stored in v has changed from 0 to, from the construct of Z (x) and, the value C_v is unchangeable, so the commitment of every node in path P (v) is unchangeable and all values stored in other nodes of this Merkle tree are not need to be changed. It is unaffected to the computations of other nodes and the verifications of all users.

Similarly, when some unchosen user leaves, the operation of the Prover is similar to above, he just need to change the value m_v stored in node v that denote the left user from $h(r_i)$ to 0. By the construct of polynomial Z(x), the commitment of this node is unchangeable and it is unaffected to group key exchange.

But when some chosen user (group member) leaves or some new user pass through the Prover's filtration to join the group, the Prover must re-run the scheme and compute the new commitment because now the operation act on some true node in the Merkle tree. These operations will change the commitment of some nodes, so it is necessary to re-run the scheme. Furthermore, If some new user's join makes the number of users beyond 2^k, the Merkle tree don't have empty node, the scheme must also be re-run. In general, we choose $2k\rangle\rangle n$, so this instance will seldom happen.

The degree of h (x) and Z (x) are both 2^k, this insure our scheme doesn't have the threshold problem, because if any attacker (include users) wants to know the session key, he must confirm h (x) and compute $K = (W(x)-h(x))/g(x)$, when the number of users beyond 2^k, the scheme will re-run, so it is impossible to confirm h (x) by attacker. Also the construct of Z (x) reduces the overhead of computations and communication under dynamic instances. If we adopt Peterson commitment scheme to realize group key exchange, the computations mainly focus on exponential computation and in the process of verification the Prover must transmit more parameter to each user in every step. But in our scheme all the computations are polynomial computation and the Z (x), g (x), W (x), T (x) can be broadcasted to all users, so the overhead of computation and communication is relative smaller.

ANALYSIS OF SECURITY

In this section, we will simply analyze the security of our scheme:

Secrecy: For any unchosen user M_i, because $g(r_i) = 0$, he can't compute the session key from $K = (W(r_i)-h(r_i))/g(r_i)$. Also, for any attacker, the degree of h (x) is 2^k, if the number of users beyond 2^k, the scheme will be re-run, so he can't confirm h (x) and computer K. The secrecy of our scheme can be guaranteed.

Authentication: based on our assumptions, the authentication between users and the Prover can be completed via secure communication channel.

Completeness: From the description of our commitment and verification scheme, it is easy for each user to verify

whether he had been chosen to be the member of group and for any chosen member, he can be distributed the session key securely and effectively. So the completeness can be guaranteed.

Soundness: If the Prover wants to cheat users, he must make $g(r_i) = 0, W(r_i) \neq h(r_i)$ or $g(r_i) \neq 0, W(r_i) = h(r_i)$ or he can find one collision of the hash function H (). From the form of W (x), it is impossible to make $g(r_i) = 0, W(r_i) \neq h(r_i)$ or $g(r_i) \neq 0, W(r_i) = h(r_i)$. Also, from our assumption, the H () is one collision-free hash function. So it is impossible for the Prover to cheat users. So soundness can be guaranteed.

Zero-knowledge: In our scheme, any user can verify whether he had been chosen to be the member of group, but he can't get any information about other users and the size of the group. During the process of verification, for any leaf node v which denote one of the users, the only information he can get from P(v) is the $C_u(u \in FS_v)$. Because the parameter R is secret to all users and from our assumption, $H(a,a) \neq a$, he can't judge the node u is a true node or a vain node. So he can't estimate how many group members are there in the brother embranchment. So the zero-knowledge can be guaranteed.

CONCLUSIONS

In this study, we proposed a new group key distribution scheme that based on zero-knowledge set. Compare to common group key exchange scheme, our scheme can not only complete the group key distribution,but also realize the zero-knowledge quality to group members. In another word, in our scheme the entity and number of group members can be conceal effectively. This property may be very important in some special areas. Furthermore, the overhead of computation and communication in our scheme is relative small. How to make the session key is also secret to the Prover is our further work.

REFERENCES

1. Micali, S., M. Rabin and J. Kilian, 2003. Zero-Knowledge sets. In: 44th Annual IEEE Symposium on Foundations of Computer Science(FOCS'03).
2. Liu, D., P. Ning and K. Sun, 2003. Efficient Self-Healing Group Key Distribution with Revocation Capability. In 10th ACM Conference on Computer and Communications Security, Washington. DC, USA.
3. Pedersen, T., 1991. Noninteractive and information-theoretic secure verifiable secret sharing. Leture Notes in Comput. Sci., 576: 129-140.
4. Blum, M., A. De Santis, S. Micali and G. Persiano, 1991. Noninteractive zero-knowledge. SIAM J. Comput., 20: 1084-1118.
5. Halevi, S. and S. Micali 1996. Practical and provably-secure commitment schemes from collision-free hashing. In Proc. 16th international Advances in Cryptology Conference-Crypto'96, pp: 201-215.
6. National institute of Standards and Technology, 1993. FIPS PUB 180-1: Secure Hash Standard. National Institute for Standards and Technology, Gaithersburg, MD, USA, April 1995. Supersedes FIPS PUB 180.
7. Kim, Y., A. Perrig and G. Tsudik, 2000. Simple and fault-tolerant key agreement for dynamic collaborative groups. In: S. Jajodia, editor, 7th ACM Conference on Computer and Communications Security, Athens, Greece, pp: 235-244. 37807 8390