Supporting Efficient Authorization in Delegation with Supervision [*]

Richard W.C. Lui Sherman S.M. Chow Lucas C.K.Hui S.M. Yiu
Department of Computer Science, University of Hong Kong, Pokfulam, Hong Kong
email: {wclui,smchow, hui,smyiu}@cs.hku.hk

Abstract

Delegation is commonly used in organizations to transfer some permission by one user to another user. However, most existing delegation schemes do not support supervision, which allows the delegators to retain control over how the delegated permission can be exercised. In this paper, we will describe how to support efficient authorization in delegation with supervision using proxy signature techniques.

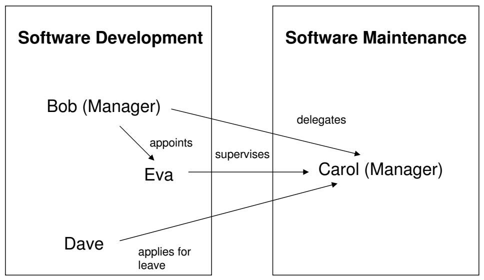

Figure 1. Leave Application Scenario

1. Introduction

Supervision in delegation. Delegation is commonly used in organizations to transfer some permission by one user to another user to achieve organization goals [8]. Consider a leave application scenario [5]. To apply for a leave, the employee should obtain the agreement from his/her manager and the HR department. Although the manager is authorized to signify acceptance for any leave application, the permission should not be discharged in an arbitrary manner. For instance, before accepting the application for an annual leave, the employee's manager should determine that during the period of absence, the critical tasks which are handled by the employee can be safely delegated to his/her colleagues.

Suppose the manager of the software development group (Bob) is planning for a vacation himself and he intends to delegate the authorization to handle the leave application of the employees in the group to the manager of the software maintenance group (Carol) (See Figure 1). After that, when an employee in the software development group (say Dave) applies for an annual leave, Carol will take over the responsibility of Bob to evaluate the Dave's application according to the organization policy. However, Carol and Dave are working in different groups and so Carol may have no knowledge of the nature of the tasks being handled by Dave. Therefore, Bob may appoint a third party (Eva) in the software development group to perform supervision on Carol. In order for Carol to process a leave application, she should request approval from Eva (who determines the criticality of the tasks currently handled by Dave). Carol can only accept an application if the approval from Eva is acquired. In this example, the delegate (Carol) is not completely trusted by the delegator (Bob). Therefore, Bob should appoint a third party (Eva) to perform supervision on Carol.

The above example motivates the need for supporting supervision in delegation. In [5], SPKI [2] was extended to support supervision in delegation. The delegators in a delegation chain may appoint supervising agents (SA). Approval from the agents is required in order for the delegate to exercise the delegated permission.

Contribution. In this paper, we propose an improved delegation scheme, which is based on the proxy signature scheme in [6], to support efficient authorization in delegation with supervision. One important feature of the proposed delegation scheme is that the proxy signature generated without supervision is iden-

* This research is supported in part by the Areas of Excellence Scheme established under the University Grants Committee of the Hong Kong Special Administrative Region, China (Project No. AoE/E-01/99), two grants from the Research Grants Council of the HKSAR, China (Project No. HKU/7144/03E and HKU/7136/04E), and two grants from the Innovation and Technology Commission of the HKSAR, China (Project No. ITS/170/01 and UIM/145).

tical to the proxy signature generated with supervision. As a result, there is no need for the verifier to be aware of how/whether supervision is performed by the delegators. In this way, the verifier and the end-user (with the associated delegators) can interact even without knowledge of the internal workflow of the other parties (e.g. there is no need for the verifier to know whether supervision is adopted in the end-user's organization).

Organization. In Section 2, we will describe how to support efficient authorization with supervision in delegation chains. In Section 3, the security and efficiency of the scheme will be analyzed. Finally, in Section 4, the summary and future research directions will be discussed.

2. Delegation With Supervision

In this section, we first perform a review on the related proxy signature schemes. Next, we describe how supervision can be carried out for a delegation chain with two users. After that, we introduce the general protocol for supporting supervision in a delegation chain.

2.1. Proxy Signature Scheme by Kim et al.

We first outline Kim et al.'s scheme [4]. Let p and q be large primes such that q divides $p-1$. Let g be a generator of a multiplicative subgroup of Z_p^* with order q, $h()$ denotes a collision resistant cryptographic hash function with range Z_q, $(x_u \in_R \; Z_q^*, y_u = g^{x_u} \; (mod\, p))$ be the private and public key for user u respectively.

Suppose user c_1 intends to delegates to user c_2. User c_1 randomly generates an ephemeral key pair $(k_{c_1} \in_R \; Z_q^*, r_A = g^{k_{c_1}} \; (mod \; p))$ and computes the proxy $s_{c_1} = x_{c_1} \; h(w_{c_1}, r_{c_1}) + k_{c_1} \; (mod \; q)$ where w_{c_1} is the delegation warrant which specifies the public key of Alice, Bob, and the restrictions on the use of this delegation.

To sign on behalf of A, user c_2 generate the proxy private key $p_{c_2} = s_{c_1} + x_{c_2} \; h(w_{c_1}, r_{c_1}) \; (mod \; q)$, randomly generates an ephemeral key pair $(k \in_R \; Z_q^*, r = g^k \; (mod \; p))$ and uses the Schnorr signature scheme [10][1] to sign the message using the proxy private key p_{c_2}. By following the verification procedure of Schnorr signature using the proxy public key $t_{c_2} = (y_{c_1} \; y_{c_2})^{h(w_{c_1}, r_{c_1})} r_{c_1} (mod \; p)$, the verifier can check the validity of the signature.

1 One can use other discrete logarithm based signature schemes like ElGamal [1] to replace Schnorr signature scheme.

2.2. Proxy signature scheme for chained delegation

In this section, we will describe a scheme, which is proposed in [6], to extend Kim et al.'s proxy signature scheme [4] to handle a chain of delegation. Let $C =< c_1, c_2, ..., c_n >$ be a delegation chain for $n > 1$ where user c_i delegates its signing right to user c_{i+1} for $i = 1, 2, \ldots, n-1$. We denote x_{c_i} and y_{c_i} to be the private and public key of the user c_i respectively. Also, we denote w_{c_i} to be the delegation warrant [2] and s_{c_i} to be the proxy issued by the user respectively. In addition, we denote $(k_{c_i} \in_R \; Z_q^*, r_{c_i} = g^{k_{c_i}} \; (mod \; p))$ to be a randomly generated ephemeral key pair for user c_i, $A_{c_i} = h(w_{c_i}, r_{c_i})$ and p_{c_i} to be the proxy private key (which is used by the user c_i to discharge the delegated rights). The values for s_{c_1}, p_{c_1} and t_{c_1} can be computed as described in the previous section. In general, user c_i, where $1 < i < n$, receives the proxy $s_{c_{i-1}}$ from user c_{i-1} and he generates the proxy s_{c_i} for user c_{i+1}. User c_{i+1} then generates the proxy private key $p_{c_{i+1}}$ and the proxy public key $t_{c_{i+1}}$. This process is carried out in a similar manner along the whole chain. The values of s_{c_i} $(1 \leq i < n-1)$, p_{c_j} and t_{c_j} where $1 < j \leq n$ in the scheme are calculated as follows.

Lemma 1 *The proxy s_{c_i}, the proxy private key $p_{c_{i+1}}$ and the proxy public key $t_{c_{i+1}}$ for user c_{i+1} can be expressed by the following equations.*

$$\begin{aligned}
s_{c_i} &= x_{c_1}(A_{c_1} A_{c_2} \ldots A_{c_i}) + \textstyle\sum_{\sigma=2}^{i} (x_{c_\sigma} \prod_{j=\sigma-1}^{i} A_{c_j}) \\
&\quad + \textstyle\sum_{\sigma=1}^{i-1} (k_{c_\sigma} \prod_{j=\sigma+1}^{i} A_{c_j}) + k_{c_i} \; (mod\, q) \\
p_{c_{i+1}} &= x_{c_1}(A_{c_1} A_{c_2} \ldots A_{c_i}) + \textstyle\sum_{\sigma=2}^{i} (x_{c_\sigma} \prod_{j=\sigma-1}^{i} A_{c_j}) \\
&\quad + x_{c_{i+1}} A_{c_i} + \textstyle\sum_{\sigma=1}^{i-1} (k_{c_\sigma} \prod_{j=\sigma+1}^{i} A_{c_j}) \\
&\quad + k_{c_i} \; (mod\, q) \\
t_{c_{i+1}} &= y_{c_1}^{A_{c_1} \ldots A_{c_i}} \textstyle\prod_{\sigma=2}^{i} y_{c_\sigma}^{A_{c_{\sigma-1}} \ldots A_{c_i}} (y_{c_{i+1}})^{A_{c_i}} \\
&\quad \textstyle\prod_{\sigma=1}^{i-1} r_{c_\sigma}^{A_{c_{\sigma+1}} \ldots A_{c_i}} r_{c_i} \; (mod\, p)
\end{aligned}$$

Given a proxy signature which is signed by the end-user c_n using the proxy private key p_{c_n}, the signature can be verified with the proxy public key t_n using the verification equation of the Schnorr signature scheme [10] [3].

2.3. Supporting Supervision in Delegation

In this section, we will describe how user c_1 may delegate to user c_2 such that supervision is possible.

2 For each delegation warrant w_{c_i} where $1 < i < n$, the public key of the delegator and delegate, the delegated authorization, the cryptographic hash of $w_{c_{i-1}}$ and the validity period of the delegation should be included.

3 For the proof of correctness of the Lemma and the security analysis of the scheme, please refer to the original paper [6]

First, user c_1 generates s_{c_1} as described in the previous section. The delegation warrant w_{c_i} should specify the permission to be delegated (which may be general in nature, e.g. reading all the files in a web server). If the delegator intends to appoint an SA, the proxy should be distributed to the SA of user c_1 in a secure manner (such that it can be kept secret from user c_2). In this way, user c_2 cannot exercise the delegated permission directly.

To exercise the delegated permission, user c_2 should prepare an access specification M (which includes the details of a particular access, such as the URL of the file to be read) and send it to c_1 (or his/her SA) for approval. If approved, c_1 cooperates with the delegate (user c_2) to sign M using some discrete-logarithm based multi-signature schemes (e.g. 2Schnorr signing protocol [9]).

2.4. The Proposed Delegation Protocol

Now, we generalize the delegation protocol to handle a delegation chain. In the following discussion, we consider the delegation chain $C =< c_1, c_2, ..., c_n >$ for $n > 1$ where user c_1 performs delegation to user c_2, who further delegates until delegation is performed to user c_n (the end-user). We first introduce the notion of partial proxy. For each user c_i, where $1 \leq i < n$, we define the partial proxy for user c_i to be

$$s'_{c_i} = \begin{cases} s_{c_1} = x_{c_1} A_{c_1} + k_{c_1} \ (mod\ q) & \text{for } i = 1 \\ x_{c_i} A_{c_{i-1}} A_{c_i} + k_{c_i} \ (mod\ q) & \text{for } 1 < i < n \end{cases}$$

For $1 < j < n$, s_{c_j} can be computed from $\{s'_{c_1}, s'_{c_2}, ..., s'_{c_j}\}$ as follows.

Lemma 2 $s_{c_j} = \sum_{m=1}^{j-1} (s'_{c_m} \prod_{l=m+1}^{j} A_{c_l}) + s'_{c_j}\ (mod\, q)$.

The Lemma can be proved by induction but the details are skipped due to space limitation.

The delegation process for C is as follows. User c_1 first generates the partial proxy s'_{c_1}. The partial proxy should be transferred to user c_2 (if SA is not appointed) or his/her SA (if SA is appointed). An encrypted channel is necessary only when an SA is appointed. In both cases, the delegate or the SA should verify the partial proxy by checking if $g^{s'_{c_1}} = y_{c_1}^{A_{c_1}} r_{c_1}\ (mod\ p)$.

For the other users c_i $(1 < i < n)$ to re-delegate, c_i should first compute s'_{c_i}. Suppose an SA is not appointed, c_i transfers s'_{c_i} and all the partial proxy he/she receives from the delegator to the delegate (user c_{i+1}) (without a secure channel). On the other hand, if an SA is appointed, c_i should transfer s'_{c_i} to the SA using an encrypted channel. In addition, he/she should transfer the partial proxy he/she receives from the delegators in the delegation chain to the delegate (user c_{i+1}). In both cases, for each partial proxy s'_{c_j}, the delegate or the SA should verify the partial proxy by checking if $g^{s'_{c_j}} = y_{c_j}^{A_{c_j}} r_{c_j}\ (mod\ p)$ (if $j = 1$) or $g^{s'_{c_j}} = y_{c_j}^{A_{c_{j-1}} A_{c_j}} r_{c_j}\ (mod\ p)$ (if $j > 1$) .

To describe how user c_n may request access, we define $o(c_i)$ to be the set of partial proxy owned by the user c_i, $D(C)$ to be the set of all the delegators $\{c_1, c_2, ..., c_{n-1}\}$ in C, $S(C) \subseteq D(C)$ to be the set of the delegators who appoint SA in C, $U(C) = S(C) \cup \{c_n\}$ and $N(C) = D(C) \setminus S(C)$. For each user c_i, o_{c_i} can be computed where

$$o(c_i) = \begin{cases} \phi & c_i \in N(C) \\ \{s'_{c_i}\} & c_i \in S(C) \\ \bigcup_{c_j \in N(C)} s_{c'_j} & c_i = c_n \end{cases}$$

As an example, consider a delegation chain $C =< c_1, c_2, c_3, c_4, c_5 >$ where user c_1 and user c_4 appoint their own SA. User c_1 first generates and transfers the partial proxy s'_{c_1} to the appointed SA with an encrypted channel. After that, user c_2 re-delegates by generating and transferring the partial proxy s'_{c_2} to user c_3. Afterwards, user c_3 generates the partial proxy s'_{c_3}. He/she transfers s'_{c_3}, and also the partial proxy s'_{c_2} received from user c_2 to user c_4. For user c_4, since he/she appoints an SA, the partial proxy s'_{c_4} should be transferred to his/her agent with an encrypted channel. In addition, he/she should transfer the partial proxy received from the delegators (which includes s'_{c_2} and s'_{c_3}) to user c_5. In this example, we have $D(C) = \{c_1, c_2, c_3, c_4\}$ and $S(C) = \{c_1, c_4\}$. Also, we have $o(c_i) = \phi$ for $i = \{2, 3\}$, $o(c_j) = s'_{c_j}$ for $j = \{1, 4\}$ and $o(c_5) = \{s'_{c_2}, s'_{c_3}\}$

Suppose user c_n intends to exercise the delegated permission, he should cooperate with the SA appointed by the users in $S(C)$ to sign the access specification M [4]. Each user $c_i \in U(C)$ (or the appointed SA) generates the signing key

$$x'_{c_i} = \begin{cases} \sum_{s'_{c_m} \in o(c_i)} s'_{c_m} v_m + x_{c_n} A_{c_{n-1}} & \text{if } i = n \\ \sum_{s'_{c_m} \in o(c_i)} s'_{c_m} v_m & \text{otherwise} \end{cases}$$

where $v_m = \prod_{l=m+1}^{n-1} A_{c_l}$ (if $n > 2$ and $m < n - 1$) or $v_m = 1$ (if $m = n - 1$) using the partial proxy he/she owns in the delegation chain. In addition, each user $c_i \in U(C)$ (or the appointed SA) generates an ephemeral

4 For the sake of illustration, we will make use of Schnorr signature scheme for generating the signature. However, other discrete logarithm based digital signature schemes may also be used.

key pair ($k(c_i) \in_R Z_q^*, r(c_i) = g^{k(c_i)} \ (mod\ p)$) and they agree on the value of $r = \prod_{c_i \in U(C)} r(c_i)$. For the simplicity of discussion, we assume there is a secure broadcast channel shared by all the signers in the delegation chain to agree on the value of r in a signing operation [5] .

Each user $c_i \in U(C)$ (or the appointed SA) generates a partial signature $s(c_i) = x'_{c_i} h(M, r) + k(c_i) \ (mod\ q)$ and they form (r, s) where $k = \sum_{c_i \in U(C)} k(c_i)(\text{mod q})$, $r = g^k (mod\ p)$ and $s = \sum_{c_i \in U(C)} s(c_i) \ (mod\ q)$

Theorem 1 *The tuple (r, s) is a Schnorr signature with the proxy private key p_{c_n}.*

Proof We prove by showing $s = p_n h(M, r) + k \ (mod\ q)$. For clarity of presentation, we assume the following operations are performed modulo q.

$$\begin{aligned}
s &= \textstyle\sum_{c_i \in U(C)} s(c_i) \\
&= \textstyle\sum_{c_i \in U(C)} (x'_{c_i} h(M, r) + k(c_i)) \\
&= (\textstyle\sum_{c_i \in S(C)} x'_{c_i} + x'_{c_n}) h(M, r) + k \\
&= (\textstyle\sum_{c_i \in S(C)} (\sum_{s'_{c_m} \in o(c_i)} s'_{c_m} v_m) \\
&\quad + \textstyle\sum_{s'_{c_m} \in o(c_n)} s'_{c_m} v_m + x_{c_n} A_{c_{n-1}}) h(M, r) + k \\
&= (\textstyle\sum_{c_i \in S(C)} s'_{c_i} v_i + \sum_{s'_{c_m} \in \bigcup_{c_j \in N(C)} s_{c'_j}} s'_{c_m} v_m \\
&\quad + x_{c_n} A_{c_{n-1}}) h(M, r) + k \qquad (\text{Definition of } o(c_i)) \\
&= (\textstyle\sum_{c_i \in S(C)} s'_{c_i} v_i + \sum_{c_j \in N(C)} s'_{c_j} v_j \\
&\quad + x_{c_n} A_{c_{n-1}}) h(M, r) + k \\
&= (\textstyle\sum_{c_i \in D(C)} s'_{c_i} v_i + x_{c_n} A_{c_{n-1}}) h(M, r) + k \\
&= (\textstyle\sum_{s'_{c_m} \in \{s'_{c_1}, \ldots, s_{c'_{n-1}}\}} s'_{c_m} v_m \\
&\quad + x_{c_n} A_{c_{n-1}}) h(M, r) + k \\
&= (\textstyle\sum_{i=1}^{n-1} s'_{c_i} v_i + x_{c_n} A_{c_{n-1}}) h(M, r) + k
\end{aligned}$$

Here, we have two cases. Case 1: $n = 2$

$$\begin{aligned}
s &= (s'_{c_1} v_1 + x_{c_2} A_{c_1}) h(M, r) + k \\
&= p_{c_2} h(M, r) + k
\end{aligned}$$

Case 2: $n > 2$

$$\begin{aligned}
s &= (\textstyle\sum_{i=1}^{n-2} s'_{c_i} v_i + s'_{c_{n-1}} v_{n-1} + x_{c_n} A_{c_{n-1}}) h(M, r) \\
&\quad + k \\
&= (s_{c_{n-1}} + x_{c_n} A_{c_{n-1}}) h(M, r) + k \qquad (\text{Lemma 2}) \\
&= (p_{c_n}) h(M, r) + k \qquad (\text{Lemma 1})
\end{aligned}$$

From Theorem 1, (r, s) is a proxy signature for the delegation chain C and so it can be verified with the proxy public key t_{c_n} using the same procedure as described in Section 2.2.

The partial proxy can be managed in a number of ways. For instance, the delegator may share his/her partial proxy with some secret sharing schemes (e.g. [11]) among a group of SA such that m out of n users in the group is required to signify acceptance. Also, an appointed SA may re-delegate the responsibility for performing supervision to another agent by further distributing the partial proxy using an encrypted channel. Therefore, the proposed protocol provides a flexible way to support supervision in delegation.

5 Alternatively, multi-signature protocols such as [7] can be adopted in the signing protocol.

3. Discussion and Security Analysis

The proposed delegation scheme is more efficient in terms of the verification of authorization when compared with the scheme in [5], which is based on SPKI. Given a proxy public key which signifies that a certain authorization is propagated along a certain delegation chain and a signed message which can be verified by the corresponding proxy public key, it can be shown that the message is signed by an authorized user (which knows the corresponding public key). Thus, verification of the authorization certificates along the delegation chain is not required. Also, the delegate will not be able to generate the proxy signature by himself/herself if one or more SA are appointed. Therefore, the ability to sign an access specification means that approval from the SA has been acquired. As a consequence, there is no need to verify the approval signature of the SA and efficient access control can be supported.

The security of the protocol relies on the security of the underlying delegation scheme [6] and the multi-signature scheme (if one is adopted). In this paper, we mainly focus on the security of the use of partial proxy. Consider a delegation chain $C =< c_1, c_2, ..., c_i >$. We first consider the case where user c_i does not appoint an SA. For the case where $i = 1$, $s'_{c_i} = s_{c_i}$ and it has been shown that the proxy can be transferred without a secure channel [3]. For the case where $i > 1$, $s'_{c_i} = x_{c_i} A_{c_{i-1}} A_{c_i} + k_{c_i} = (x_{c_{i-1}} A_{c_{i-1}}) h(w_{c_i}, r_{c_i}) + k_{c_i}$, which is essentially the signature of w_{c_i} using the Schnorr signature scheme with $x_{c_i} A_{c_{i-1}}$ as the secret signing key. If the underlying Schnorr signature scheme is secure, an attacker cannot alter w_{c_i} (the message to be signed). Given a partial proxy for a certain delegation chain (a signature of a delegation warrant), it is infeasible to determine the partial proxy for another delegation chain (a signature for another delegation warrant) if the underlying signature scheme is secure. Also, although $A_{c_{i-1}}$ may be publicly known, the attacker cannot determine the secret signing key and so he/she cannot compromise the private key of user c_i even with the partial proxy s'_{c_i}. In addition, the partial proxy cannot be misused by the attacker. Since the public key of the delegate (user c_{i+1}) is included in w_{c_i}, only user c_{i+1} can make use of the proxy to create proxy signature or perform further delegation. Therefore, the transfer of partial proxy does not require a secure channel to protect its confidentiality.

Now, suppose user c_i appoints an SA when performing delegation to user c_{i+1}. The partial proxy s'_{c_i} is given to the SA securely and it should be kept secret from the delegate. As discussed, given the partial proxy s'_{c_i}, potential attackers (which include the SA) cannot determine the private key of the delegator c_i. In addition, the SA themselves cannot exercise the permission associated with the partial proxy because the public key of the delegate is included in the delegation warrant of the partial proxy. Only the delegate (user c_{i+1}) will be able to exercise the permission associated with the delegation chain. Therefore, the SA, who do not know the private key of user c_{i+1}, cannot exercise the permission even though they know the partial proxy. On the other hand, since c_i does not know s'_{c_i}, the delegate will not be able to create a valid proxy signature without the cooperation of the SA.

4. Summary and Future Research Directions

A new and efficient delegation protocol to support supervision is proposed in this paper. The delegation protocol inherits the advantage of the proxy signature, which supports efficient verification of authorization. One of the novel features of this protocol is that the verification of the approval signature by SA is not required. The ability to sign an access specification using the proxy for a certain delegation chain implicitly means that the end-user is authorized by the delegators and the approval from the SA has been acquired. By performing one signature verification using the proxy public key for a certain delegation chain, the authorization of the delegate and the approval of the SA can be verified at the same time. Therefore, efficient access control can be supported. In addition, in the proposed delegation scheme, the proxy signature generated without supervision is identical to the proxy signature generated with supervision. As a result, there is no need for the verifier to be aware of how/whether supervision is performed by the delegators. In this way, the verifier and the end-user (with the associated delegators) can interact even without knowledge of the internal workflow of the other parties.

The proposed delegation protocol requires the partial proxy to be kept and managed by different users in the organization. Therefore, one possible future work is to provide an infrastructure to allow the partial proxy to be managed in a decentralized manner. Also, other issues such as the revocation of delegation should also be studied.

References

[1] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Crypto '84, LNCS 196, pp. 1018*, 1984.

[2] C. Ellison, Frantz, B. Lampson, R. Rivest, B. Thomas, and T. Ylonen. SPKI certificate theory. *IETF, RFC 2693*, September 1999.

[3] Seungjoo Kim Jung-Yeun Lee, Jung Hee Cheon. An analysis of proxy signatures: Is a secure channel necessary? In *CT-RSA'03, LNCS 2612, pp. 68-79, 2003. Berlin: Springer-Verlag*, 2003.

[4] S. Kim, S. Park, and D. Won. Proxy signatures, revisited. In *Information and Communications Security (ICICS'97), LNCS 1334, pp. 223-232, 1997. Berlin: Springer-Verlag*, 1997.

[5] Richard W.C. Lui, Lucas C.K. Hui, S.M.Yiu, and Y.Woo. A model to support fine-grained delegation of authorization. In *the 2005 International Conference on Security and Management, Las Vegas (SAM'05), Nevada, USA, June 20-23*, 2005. to appear.

[6] Richard W.C. Lui, S.M. Yiu, and Lucas C.K. Hui. Efficient authorization in delegation chains with strong non-repudiation. submitted.

[7] Silvio Micali, Kazuo Ohta, and Leonid Reyzin. Accountable-subgroup multisignatures: extended abstract. In *ACM Conference on Computer and Communications Security*, pages 245–254, 2001.

[8] M.Mambo, K.Usuda, and E.Okamoto. Proxy signatures for delegating signing operations. In *3rd ACM Conf. on Computer and Communication Security, pp. 48–57*, 1996.

[9] A. Nicolosi, M. Krohn, Y. Dodis, and D. eres. Proactive two-party signatures for user authentication. In *Proceedings of the 10th Annual Network and Distributed System Security Symposium, pages 233– 24 , February*, 2003.

[10] C.P. Schnorr. Efficient identification and signatures for smart cards. *Advances in Cryptology – CRYPTO '89, Lecture Notes in Computer Science, Vol. 435, pp. 239-252, Berlin: SpringerVerlag*, 1990.

[11] Adi Shamir. How to share a secret. In *Communications of the ACM, Volume 22 , Issue 11*, November 1979.