Providing Efficient Certification Services Against Active Attacks in Ad Hoc Networks

Bo Zhu*†, Guilin Wang†, Zhiguo Wan*†, Mohan S. Kankanhalli*, Feng Bao†, Robert H. Deng‡,

*School of Computing
National University of Singapore
Singapore 117543
†Institute for Infocomm Research
21 Heng Mui Keng Terrace
Singapore 119613
‡School of Information Systems
Singapore Management University
Singapore 259756

***Abstract*— Most of previous research work in key management can only resist passive attacks, such as dropping the certificate request, and are vulnerable under active attacks, such as returning a fake reply to the node requesting the certification service. In this paper, we propose two algorithms to address both security and efficiency issues of certification services in ad hoc networks. Both of the algorithms can resist active attacks. In addition, simulation results show that, compared to the previous works, our second algorithm is not only much faster in a friendly environment, but it also works well in a hostile environment in which existing schemes work poorly. Furthermore, the process of generating partial certificates in our second algorithm is extremely fast. Such advantage is critical in ad hoc networks where by nature the less help a node requests from its neighbors, the higher is the chance of obtaining the help. Consequently, using our second algorithm, a node can easily find enough neighboring nodes which provide the certification service.**

I. Introduction

Among all the security issues in ad hoc networks, key management is the most crucial one, because it is the essential assumption of many other security services. For instance, many secure routing protocols, such as ARAN [1] and SRP [2], assume that a pair of private and public keys and a certificate signed by a Trusted Third Party (TTP) have been assigned to nodes. One core problem of key management in ad hoc networks is how to provide certification services securely. Most of previous research work mainly focused on passive attacks, such as dropping the certificate request, and are inefficient under cases that malicious nodes launch active attacks, such as returning a fake reply to the node requesting the certification service. However, active attacks may result in great risks, especially in the military field. Due to the poor physical security in ad hoc networks, adversaries may take over some nodes in the network. By launching active attacks, adversaries can disable the certification service of the whole network without being caught, with only a very small portion of nodes.

To overcome these challenges, we propose two algorithms based on threshold cryptography and Verifiable Secret Sharing (VSS) to address both security and efficiency issues of certification services in ad hoc networks. Both of the two algorithms can resist active attacks on certification services in ad hoc networks. Simulation results show that, compared to the previous works [3], [4], our second algorithm is not only much faster in a friendly environment, but it also works well in a hostile environment in which existing schemes work poorly.

The rest of the paper is organized as follows. Section II studies the related work in the literature. After introducing some cryptographical primitives in Section III, in Section IV, we present the system model of our algorithms. Certification services and secret share updates are presented in Section V and Section VI, respectively. Simulation results on the security and efficiency of our scheme are given in Section VII. In Section VIII, we draw the conclusions.

II. Related Work

In [5], Zhou and Haas focused on how to establish a secure key management service in an ad hoc networking environment. They proposed to use threshold cryptography [6], [7] to distribute trust among a set of servers. The focus of their work is to maximize the security of the shared secret in the presence of possible compromises of the secret share holders. It assumes a small group of servers with rich connectivity. Therefore, it is not suitable for purely ad hoc environments.

In [3], [4], Kong et al. also took advantage of threshold secret sharing to distribute the functions of the Certificate

Authority (CA) but extended it to normal nodes. In other words, each node holds a secret share and multiple nodes in a local neighborhood jointly provide complete services. It minimizes the effort and complexity for mobile clients to locate and contact the service providers. One major weakness of this scheme is that, due to the inability of distinguishing adversaries who provide invalid partial certificates from honest nodes, both of their algorithms for certification renewals are vulnerable to active attacks, such as sending invalid partial certificates which result in the failure of renewing or assigning a certificate. In [8], Lehane et al. presented a similar scheme based on shared RSA key generation, and thus share the pros and cons of [3], [4]. In addition, according to their empirical results, the efficiency of their protocol is not good.

One recent work by Narasimha et. al. [9] pointed out that Kong's algorithm [3], [4] is vulnerable to active attacks, and proposed a Threshold DSA signature scheme. Their scheme is in fact similar to our first algorithm, in the sense that both of them are standard VSS processes based on *Discrete Logarithm Problem* (**DLP**), and thus are more costly, compared to our second algorithm. In addition, the standard VSS process requires almost twice as many as neighbors involved in the certification service within one round. More specifically, given that, there are $k-1$ adversaries in the network, the Threshold DSA signature scheme [9] requires helps from $2k-1$ neighbors, while our second algorithm needs helps from k neighbors instead.

Both [5] and [3], [4] require a trusted authority, though the latter needs the authority only at the start-up phase. In [10], a more extreme case, where there is no central authority at all, was considered. Each user is her own authority domain and issues public-key certificates to other users. When user u wants to verify user v, they merge their local certificate repositories and find an appropriate certificate chain from u to v in the merged repository. This method has a few weaknesses. Firstly, the initialization phase (i.e. build the local certificate repository) is relatively expensive. In addition, to achieve better assurance about the user-key binding, authentication metrics are used. However, how to find the most appropriate metric for ad hoc networks remains an open issue. Even if such a metric exists, the dynamic property[1] of ad hoc networks results in very high computation costs in reconstructing local certificate repositories of all the nodes. Furthermore, the metric's confidence in someone's honesty can be easily cheated, as any user can create an arbitrary number of public keys and issue many false certificates.

[1]The dynamic property here does not mean the join/drop operations of nodes but revocations of certificates due to compromises or other reasons.

III. Cryptographic Primitives

In this section, we briefly describe various cryptographic techniques underlying our approach.

A. Secret Sharing

The first secret sharing scheme was proposed by Shamir in 1979 [11]. This scheme is also called a *(n, k)-threshold scheme*, since it has the following properties: (1) any k or more users can reconstruct the secret from their shares; (2) for $k-1$ or fewer users, it is impossible to reconstruct the secret. The integer k is called the *threshold*.

B. Proactive Security

A proactive threshold cryptography scheme [12]–[14] uses share refreshing, which enables users to compute new shares from old ones in collaboration without disclosing the service private key, i.e. the shared secret, to any user. The new shares constitute a new (n, k) sharing of the service private key. After refreshing, users remove the old shares and only keep the new ones. Because the new shares are independent of the old ones, the adversary cannot combine old shares with new shares to recover the private key of the service. Thus, the adversary is challenged to compromise k users between periodic refreshings.

C. Verifiable Secret Sharing Schemes

There are two weaknesses in the Shamir secret sharing scheme: (1)If the dealer distributes erroneous subshares to some or all the users, how can users verify whether the subshares received are correct? (2)During the share recovery process, if some compromised users provide false subshares, how can other users detect them?

Share refreshing must tolerate missing subshares and erroneous subshares from compromised users. A compromised user may not send any subshares. However, as long as correct users agree on the set of subshares to use, they can generate new shares using only subshares generated from k users in a (n, k) secret sharing scheme. To detect incorrect subshares, a few VSS schemes were proposed in [12], [15]. A VSS scheme generates extra public information for each (sub)share using a one-way function. The public information can testify the correctness of the corresponding (sub)shares without disclosing them.

IV. System Model

Before providing the certification service, the secret key of the whole ad hoc network has been securely shared by all the members in the network using a (n, k) secret sharing scheme. Namely, only k or more nodes can jointly recover the secret key. And besides a share of the secret key, each node has a personal PKI key pair. The key pair can be generated by either a TTP or even the node itself.

A. Notation and Assumptions

In the rest of this paper, we use the following notations:

- $[SK, PK]$: the secret and public key pair of the whole ad hoc network. PK is known by all the nodes.
- SK_i: a secret share of SK, which is held by node "*i*".
- $[sk_i, pk_i]$: the personal secret and public key pair of node "*i*".
- ID_i: the identifier of node "*i*".
- $cert_i$: a statement consisting of the association between node i and its public key and the expired time.
- $CERT_i$: the certificate of node "*i*". It is a signature on the statement $cert_i$ using SK.

We make the following assumptions in this paper:

1) Each node i has a personal PKI key pair [pk_i, sk_i].
2) Given the public key of any node, it is computationally infeasible to obtain the corresponding secret key.
3) Between any two consecutive secret share updates, the number of adversaries that hold secret shares originating from SK is less than k. Adversaries may still stay in the network, or have already left it.

B. Types of Attacks on Certification Services

In this paper, we consider two types of possible attacks on certification services in ad hoc networks: passive attacks and active attacks. In passive attacks, adversaries simply drop and refuse to forward other nodes' requests of assigning or renewing certificates. In active attacks, in contrast, adversaries may return a fake reply (namely, an invalid partial certificate) to the node requesting certification services. In [3], [4], Kong et al. proposed two algorithms to prevent passive attacks. However, their algorithms are vulnerable to active attacks. Specifically, if any partial certificate returned to the requesting node is invalid, although the requesting node can detect that the combined certificate is invalid, it cannot figure out which partial certificates are invalid. Consequently, the current process fails, and all the computation done are wasted. The node has to restart the process by selecting k partial certificates again. Simulation results in Section VII shows that the success rate of completing the certification service in their algorithms drops very fast when the threshold of the network increases. Accordingly, the number of rounds for completing the certification service increase rapidly when the threshold of the network increases. Here, one round is defined to be the whole procedure that begins from a node requesting to be assigned a new certificate or renew its certificate to the combination and validation process of certificates assigned or renewed.

V. Certificate Initialization and Renewal

There are two ways to issue a new certificate or renew a certificate. A node may be issued an initial certificate by an online or offline TA, after the authority verifies the authenticity through external means (e.g., in-person ID). However, it is both costly for the TA to maintain certificates of all the nodes and is inconvenient for new nodes to request their certificates from the TA. An alternative approach is to use any coalition of k networking nodes to issue an initial certificate via collaborative admission control for this new node.

In [3], [4], Kong et al. employed the RSA scheme to provide the certification services. In our algorithms, we design them based on the difficulty of DLP. Variants of the Schnorr signature scheme [16] and the signature scheme proposed by Park and Kurosawa [17] are designed to fulfill the task. Here we only show two algorithms based on the Schnorr signature scheme. In the former, nodes i selects a group of $2k-1$ nodes, and these nodes cooperate to assign node i a certificate. It requires only one round to assign a certificate. In the latter, node i selects a group of k nodes, and these nodes cooperate to assign node i a certificate. It may need more than one round to assign a certificate, when there are malicious nodes launching active attacks.

The first algorithm has been proposed by Stinson and Strobl [18], which is provably secure, i.e., one can break this threshold signature iff one can break Schnorr's signature. The second algorithm is modified from [18] by us, which is also provably secure. It is very efficient but not fault-tolerant. These two algorithms, especially the second one, are more efficient and secure than algorithms proposed in [3], [4]. Firstly, their algorithms cannot handle active attacks. It can only verify whether the combined certificate is valid, but fails to find out the invalid partial certificates. Consequently, as long as there is one fake partial certificate among the k partial certificates chosen to generate the certificate, all the work done by other honest nodes are useless. Even worsely, adversaries can perform the attack without being caught. To prevent such attack, in our scheme nodes can verify each partial certificate to detect those malicious nodes. Secondly, our algorithms are based on DLP, which is faster than RSA on which [3], [4] is based. Thirdly, the time for generating a partial certificate in our second algorithm is 7 to 150 times shorter than [3], [4]. Such advantage is critical in ad hoc networks where by nature the less help a node requests from its neighbors, the higher is the chance of obtaining the help. Furthermore, their solution requires a k-bounded offsetting to recover the real certificate, while we can generate the real certificate directly.

A. Assigning certificates based on $2k-1$ nodes

In this algorithm, the VSS scheme proposed in [12] is employed to find out adversaries who launch active attacks. Since Pedersen's VSS scheme [15] requires a dealer, it is not suitable for ad hoc environments. Instead, we follow the distributed way proposed in Stinson's scheme [18] to achieve verifiable secure sharing. This algorithm requires cooperations from $2k-1$ nodes[2], and it ensures that the whole process of assigning a certificate can be finished within one round, in spite of active attacks launched by malicious nodes, since there are at most $k-1$ adversaries.

Let p and q be two large primes such that q divides $p-1$, and let G_q is the unique multiplicative subgroup of $\mathbb{Z}_p$ with order q. Let m be the statement claiming that a new node i's public key is pk_i, let $h(\cdot)$ be a one-way hash function: $\{0,\ 1\}^* \rightarrow \mathbb{Z}_q$.

Firstly, node i chooses a group of $2k-1$ nodes from its neighbors. Without loss of generality, let the group be $G = \{1, \cdots, 2k-1\}$. Then node i broadcasts the request m together with the IDs of the $2k-1$ nodes among the group G. To achieve VSS, we need to generate a random shared secret denoted as r within group G. Details is shown as follows.

For each node $j \in G$ receiving the request and deciding to serve the request, it chooses $r_j, r'_j \in \mathbb{Z}_q$ at random, and verifiably shares them among G acting as the dealer according to Pedersen's VSS scheme. Let the sharing polynomials be $f_j(u) = \sum_{t=0}^{k-1} a_{jt}u^t$, $f'_j(u) = \sum_{t=0}^{k-1} a'_{jt}u^t$, where $a_{j0} = r_j$, $a'_{j0} = r'_j$. Let the public commitments be $C_{jt} = g^{a_{jt}} \cdot h^{a'_{jt}} \ (mod\ p)$ for $t \in \{0, \cdots, k-1\}$, where g and h are two generators of G_q and no one knows $\log_g h$. Let H_0 be the set of nodes which are not detected to be cheating. Then the shared secret r is defined as $r = \sum_{j \in H_0} r_j$, and node j sets his share of the secret as $e_j = \sum_{t \in H_0} f_t(ID_j) \ (mod\ q)$, and the value $e'_j = \sum_{t \in H_0} f'_t(ID_j) \ (mod\ q)$.

Next, each node j ($\in H_0$) broadcasts $A_{jk} = g^{a_{jt}}$ for $t \in \{0, \cdots, k-1\}$, and each node s in H_0 can verify the values broadcasted by other nodes in H_0 by checking if

$$g^{f_j(ID_s)} = \prod_{t=0}^{k-1} (A_{jt})^{ID_s^t} \quad (mod\ p) \tag{1}$$

If the check fails for an index j, node s complains against node j by broadcasting the values $f_j(ID_s)$, $f'_j(ID_s)$ that satisfy

$$g^{f_j(ID_s)} \cdot h^{f'_j(ID_s)} = \prod_{t=0}^{k-1} (C_{jt})^{{ID_s}^t} \quad (mod\ p) \tag{2}$$

[2]On consideration of the dynamic property of ad hoc networks, a slightly higher number of nodes may need to be involved in this algorithm.

but do not satisfy Eq. (1). For node i which received at least one valid complaint, other nodes run the reconstruction phase of Pedersen's VSS scheme to compute r_j, $f_j(\cdot)$, A_{jt} for $t = 0, \cdots, k-1$. Therefore, all the players in H_0 set $X_j = g^{r_j}$.

After performing these steps, the following equations hold:

$$X = \prod_{j \in H_0} X_j = g^r (mod\ p)$$

$$f(u) = r + a_1 u + \cdots + a_{k-1}u^{k-1}$$

where $a_t = \sum_{j \in H_0} a_{jt}$, for $t \in 1, \cdots, k-1$, and $f(j) = e_j \ (mod\ q)$ for $j \in H_0$.

In [19], the above scheme is proved to be robust under the assumption that $k \leq \frac{n}{2}$. In ad hoc networks, such assumption is reasonable, since normally the threshold is set to be much less than the number of of nodes in the whole network.

Then for each node $j \in H_0$, reveals its partial certificate:

$$\gamma_j = e_j + h(m||X) \cdot SK_j \quad (mod\ q),$$

and γ_j can be verified by:

$$g^{\gamma_j} = X \cdot \prod_{t=1}^{k-1} C_t^{{ID_j}^t} \cdot PK_j^{h(m||X)}$$

where $C_t = g^{a_t}$, for all $j \in H_0$.

Let H_1 denote the set of nodes not detected to be cheating in the above step. After verifying the partial certificates, node i selects an arbitrary subset $H_2 \subseteq H_1$ with $|H_2| = k$. Without loss of generality, we denote $H_2 = \{1, \cdots, k\}$, and compute:

$$\sigma = \sum_{j \in H_2} \gamma_j l'_j(0) \qquad (mod\ q)$$

where $l'_j(0) = \prod_{r \neq j,\ j \in H_2}^{k} \frac{ID_r}{ID_r - ID_j} \ (mod\ q)$. Then node i's signature for m (i.e. $CERT_i$) is the pair (X, σ). Since $\sigma = e + h(m||X)SK \ (mod\ q)$, other nodes can verify the certificate by:

$$g^\sigma = X \cdot PK^{h(m||X)} \quad (mod\ p)$$

B. Assigning certificates based on k nodes

Although the algorithm presented Section V-A can complete the certification service within one round, the computation overhead for achieving VSS is heavy for nodes in ad hoc networks. To solve this problem, in this section we propose another algorithm which requires lightweight computation. This algorithm needs cooperations from only k nodes, but it may take more than one round to assign a certificate, when malicious nodes are launching active attacks. Although this algorithm is not fault-tolerant, we can distinguish honest and malicious

nodes, and those honest nodes are selected directly as members of group G of the next round. Simulation results show that more than 96% of certification renewals can be finished within two rounds, even if malicious nodes launch active attacks. It is much higher than algorithms proposed in [3], [4], which declines very fast when the threshold increases.

Let p and q be two large primes such that q divides $p-1$, let G_q is the unique multiplicative subgroup of $\mathbb{Z}_p$ with order q, and let g be a generator of G_q. Let m be the statement claiming that a new node i's public key is pk_i, let $h(\cdot)$ be a one-way hash function: $\{0,\ 1\}^* \rightarrow \mathbb{Z}_q$.

Details of the algorithm for generating a threshold Schnorr signature are shown as follows. Firstly, node i chooses a group of k nodes from its neighbors. Without loss of generality, let the group be $G = \{1, \cdots, k\}$. Then node i broadcasts the request m together with the IDs of the k nodes in group G.

Once a node $j \in G$ receives the request and decides to serve the request, it first chooses a random integer $e_j \in \mathbb{Z}_q$ and broadcasts $x_j = g^{e_j}\ (mod\ p)$ and $PK_j = g^{SK_j}\ (mod\ p)$ within the group G. Then node j calculates its partial certificate γ_j that is specific to group G

$$\gamma_j = e_j + h(m||X) \cdot SK_j \cdot l_j(0) \quad (mod\ q)\ ,$$

where $l_j(0) = \prod_{r=1, r\neq j}^{k} \frac{ID_r}{ID_r - ID_j}\ (mod\ q)$ and $X = \prod_{j=1}^{k} x_j\ (mod\ p)$, and returns it to node i. Node i can verify the partial certificate as follow:

$$g^{\gamma_j} = x_j \cdot PK_j^{h(m||X) l_j(0)} \quad (mod\ p) \tag{3}$$

If the k partial certificates are valid, node i calculates $\sigma = \sum_{j=1}^{k} \gamma_j$, and its signature for m (i.e. $CERT_i$) is the pair $(X,\ \sigma)$. Other nodes and node i can verify the certificate by:

$$g^{\sigma} = X \cdot PK^{h(m||X)} \quad (mod\ p) \tag{4}$$

In practice, node i can first verify $(X,\ \sigma)$ using Eq. (4). If it is valid, the task is completed. Otherwise, node i then verifies the partial certificates using Eq. (3). Those adversaries providing invalid partial certificates are detected at this stage.

VI. Scalable Share Updates

Although adversaries are not able to compromise k or more nodes at the same time, the number of nodes compromised may increase gradually. Therefore, to prevent the secret key of the network from being disclosed, we need to update the shares of the secret key periodically. To ensure the security of share updates, we employ proactive secret sharing schemes [12]–[14], [20] to adapt the configuration of the secret sharing to variations in highly dynamic environments, such as ad hoc networks. The proactive threshold cryptography scheme enables nodes in the network to compute new shares from old ones in collaboration without disclosing SK. It relies on the homomorphic property.

We notice that it is unnecessary to require all the nodes involved in the share refreshing process. Instead, the task can be done by only k nodes, since we assume that, between any two consecutive secret share updates, the number of adversaries which hold secret shares originating from the same secret key is less than k. To detect those incorrect subshares, VSS scheme [18] is employed.

Details are shown as follows. To renew the secret shares of all the n nodes, firstly, k nodes are chosen from the network. Without lose of generality, we denote them as $(ID_1, \cdots, ID_k)$. Then each of the k nodes, denoted as node i, randomly generates a $(n,\ k)$ sharing of 0, denoted as $(SK_{i1}, SK_{i2}, \cdots, SK_{in})$. After receiving all the subshares generated by the k nodes, each node in the network, denoted as node j, can compute a new share from them and its old share ($SK'_j = SK_j + \sum_{i=1}^{k} SK_{ij}$). The new shares constitute a new $(n,\ k)$ sharing of the service secret key. After refreshing, nodes remove the old shares and use the new ones to generate partial signatures. Because the new shares are independent of the old ones, the adversary cannot combine old shares with new shares to recover the secret key of the service. Thus, the adversary is challenged to compromise k nodes between periodic refreshing.

VII. Simulations on Security and Efficiency

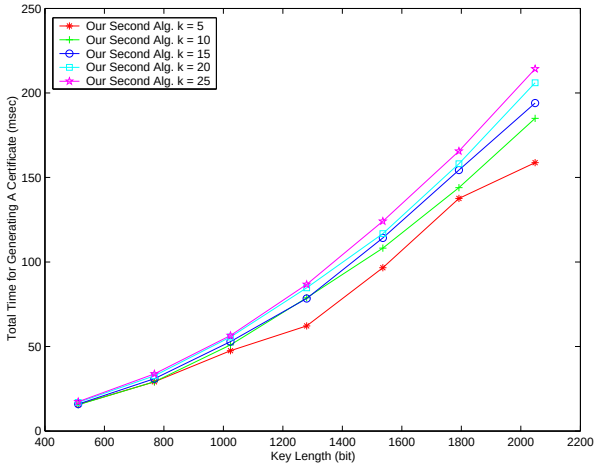
Current on-demand routing protocols, such as AODV [21] and OLSR [22], handle well when the size of an ad hoc network is around 100 to 250 nodes. Thus, we set the network size within this range in our simulation. Let p_n be the probability of a node being compromised, p_r be the probability of SK being compromised, and n be the size of the network. Table I shows the settings on the threshold under different conditions. For example, when p_n is not less than 0.01, to ensure p_r lower than 10^{-4}, the threshold should be set to at least 7 and 11 for an ad hoc network with 100 and 250 nodes, respectively.

TABLE I
SETTINGS ON THE THRESHOLD OF A REGION

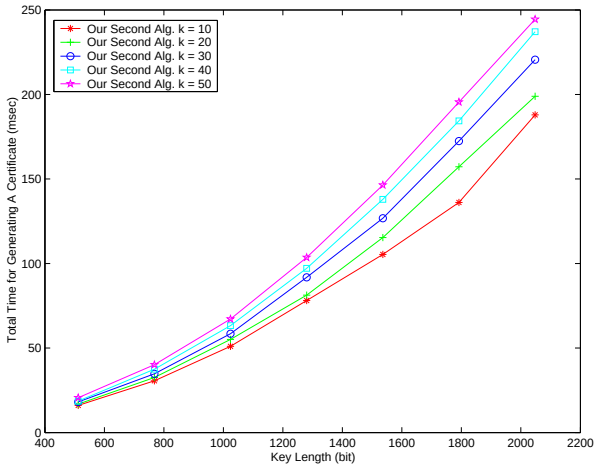
	n = 100		n=250	
p_n	$p_r < 10^{-4}$	$p_r < 10^{-5}$	$p_r < 10^{-4}$	$p_r < 10^{-5}$
0.01	7	8	11	13
0.05	16	17	28	30
0.1	24	26	45	48

A. Computational Costs of Certification Services

We measure and compare the performance of Kong's algorithm and our second algorithm on two Windows

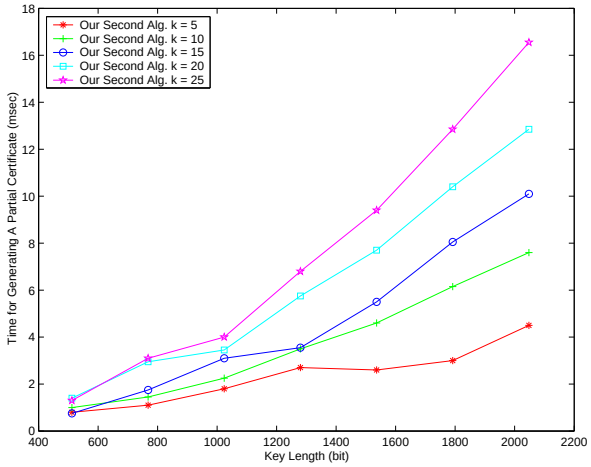


(a) n = 100

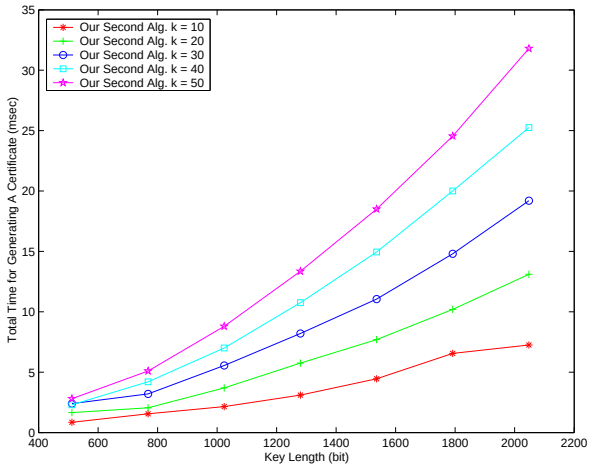


(b) n = 250

Fig. 1. Total Time for Generating or Renewing A Certificate in Our Second Algorithm



(a) n = 100



(b) n = 250

Fig. 2. Time for Generating A Partial Certificate in Our Second Algorithm

2000 machines. One is a Pentium IV 2.2G desktop, and the other is a Pentium III 800 laptop. Due to the limited space, we only show the implementation results on the Pentium III 800 laptop. Both of algorithms are implemented in Java. We run the experiments under different settings, e.g. different key lengths, thresholds, and region sizes. For each setting, we run the two algorithms for 20 times respectively and then calculate the average values.

As shown in Figure 1, the total time for generating or renewing a certificate in our second algorithm varies from 20 to 250 milliseconds in the experiments, depending on the setting on the key length, threshold, and region size. In particular, when the key length is 1024 bits, it takes our second algorithm around 40 to 70 milliseconds to generate or renew a certificate. We also find that, in our second algorithm, the process that a neighbor generates a partial certificate for the new node is very fast. As shown in Figure 2, such process takes less than 32 milliseconds under all the settings tested in the experiments. In particular, when the key length is 1024 bits, it takes less than 9 milliseconds to generate a partial certificate.

To compare with previous work [3], [4], [23], in Figure 3 and Figure 4, we show the ratio of the total time for generating or renewing a certificate in Kong's algorithm to that of our second algorithm and the ratio of the time for generating a partial certificate in Kong's algorithm to that of our second algorithm, respectively. As shown in Figure 3, our second algorithm is more efficient, when we consider the total time for assigning a new certificate. For instance, when the key length is 1024 bits, our second algorithm is around six to eight times faster

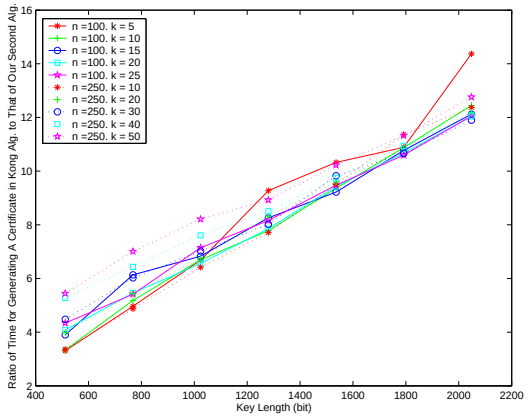


Fig. 3. Ratio of Time for Generating or Renewing A Certificate in Kong Algorithm to That of Our Second Algorithm

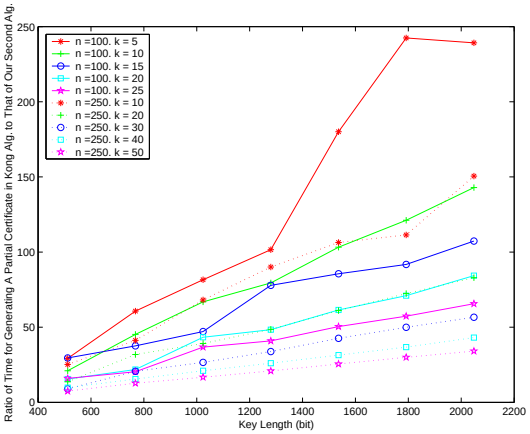
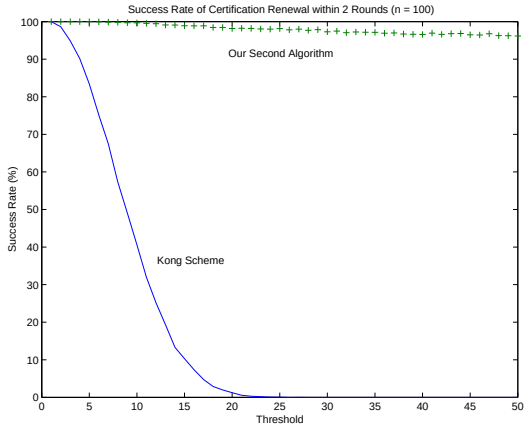
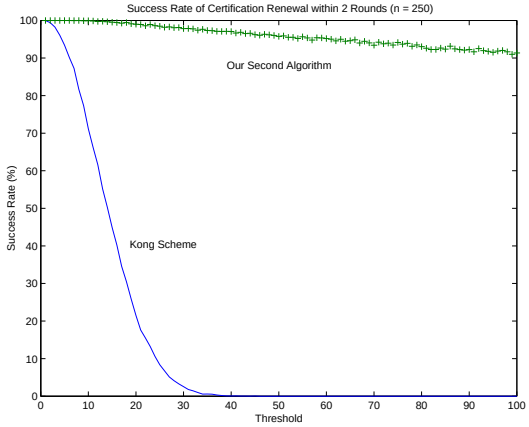


Fig. 4. Ratio of Time for Generating A Partial Certificate in Kong Algorithm to That of Our Second Algorithm



(a) n = 100



(b) n = 250

Fig. 5. Success Rate of Certification Renewals in 2 Rounds

than Kong's algorithm. As to the process of generating a partial certificate, the efficiency is greatly improved in our second algorithm. For example, when the key length is 1024 bits, our second algorithm is around 20 to 80 times faster. Consequently, using our second algorithm, a node can easily find enough neighboring nodes to provide the certification service, since very little effort is involved.

From empirical results, we notice that the performance of our algorithm is tightly related to the key length and the threshold. The larger the key length is, the more time we need to complete the certification service. However, compared to Kong et al.'s scheme, our second scheme is less sensitive to this parameter, and thus is more efficient. Similarly, the larger the threshold is, the more time we need to complete the certification service.

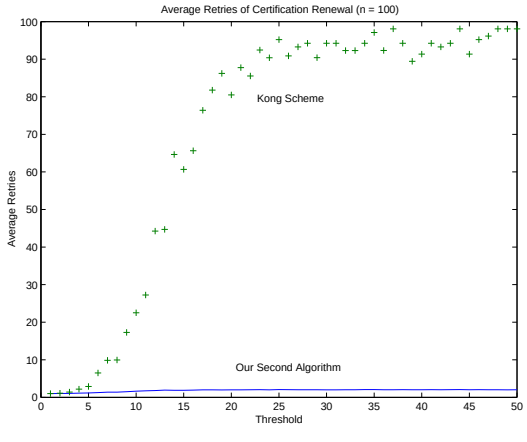
B. Certification Services Under Active Attacks

To compare the efficiency of certification services under active attacks, we use the success rate of certification renewals within two rounds and the average rounds of retries before successfully assigning or renewing a certificate as the evaluation metrics.

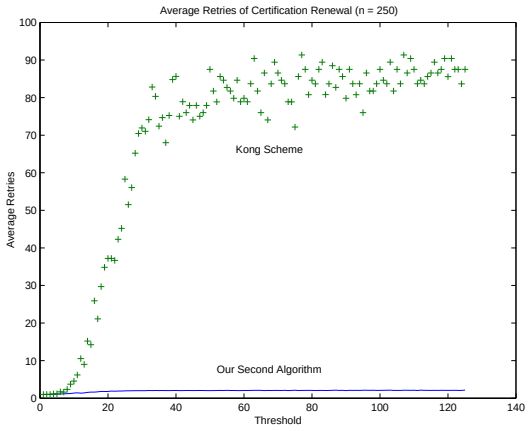
We run the simulations in a 600m X 600m network with 100 or 250 nodes, and the speed of nodes ranges from 1 m/s to 20 m/s. We apply the random way-point model to emulate node mobility pattern.

As shown in Figure 5, the success rate of [3], [4] declines very quickly when the threshold increases. For example, for a network with size 100 and p_n is 0.01, if the threshold is set to be 5, the success rate is 83.41%. However, if the threshold increases to 10, the success rate drops to 40.59%.

Figure 6 show the average retries of certification renewals in [3], [4] and our scheme. In the former, the number of average retries is very sensitive to the increment of the threshold. In contrast, in our scheme, the average number of retries is very stable and only around 2.



(a) n = 100



(b) n = 250

Fig. 6. Average Retries of Certification Renewals

VIII. Conclusion

In this paper, we proposed two algorithms based on threshold cryptography and VSS to address both security and efficiency issues of certification services in ad hoc networks. Both of the two algorithms can resist active attacks. In addition, simulation results show that, compared to the previous works, our second algorithm is not only much faster in a friendly environment, but it also works well in a hostile environment in which existing schemes work poorly. Besides that, using our second algorithm, a node can easily find enough neighboring nodes which provide the certification service.

References

[1] K. Sanzgiri, B. Dahill, B. N. Levine, C. Shields, and E. M. Belding-Royer., "A secure routing protocol for ad hoc networks," in *Proceedings of the 10th IEEE International Conference on Network Protocols (ICNP)*, 2002.

[2] Panagiotis Papadimitratos and Zygmunt J. Haas, "Secure routing for mobile ad hoc networks," in *SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2002)*, January 2002.

[3] Jiejun Kong, Petros Zerfos, Haiyun Luo, Songwu Lu, and Lixia Zhang, "Providing robust and ubiquitous security support for mobile ad-hoc networks," in *IEEE 9th International Conference on Network Protocols (ICNP'01)*, 2001.

[4] Haiyun Luo, Jiejun Kong, Petros Zerfos, Songwu Lu, and Lixia Zhang, "URSA: Ubiquitous and robust access control for mobile ad hoc networks," *IEEE/ACM Transactions on Networking*, 2004.

[5] Lidong Zhou and Zygmunt J. Haas, "Securing ad hoc networks," *IEEE Network Magazine, Special Issue on Network Security*, vol. 13, no. 6, 1999.

[6] Yvo Desmedt and Yair Frankel, "Threshold cryptosystems," in *Proceeding of Advances in Cryptology - CRYPTO '89, LNCS 0435*, 1990, pp. 307–315.

[7] Y. Desmedt, "Threshold cryptography," *European Transactions on Telecommunications*, vol. 5, no. 4, pp. 449–457, 1994.

[8] Brian Lehane, Linda Doyle, and Donal O'Mahony, "Shared RSA key generation in a mobile ad hoc network," in *Proceedings of IEEE Military Communications Conference (MILCOM 2003)*, 2003.

[9] Maithili Narasimha, Gene Tsudik, and Jeong Hyun Yi, "On the utility of distributed cryptography in P2P and MANETs: The case of membership control," in *Proceedings of the 11th IEEE International Conference on Network Protocols (ICNP'03)*, Nov. 2003, pp. 336–345.

[10] Srdjan Čapkuny, Levente Buttyán, and Jean-Pierre Hubaux, "Self-organized public-key management for mobile ad hoc networks," Tech. Rep. 2002/34, EPFL/IC, May 2002.

[11] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, November 1979.

[12] P. Feldman, "A practical scheme for non-interactive verifiable secret sharing," in *Proceedings of the 28th IEEE Symposium on the Foundations of Computer Science*, 1987, pp. 427–437.

[13] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung, "Proactive secret sharing or: How to cope with perpetual leakage," in *Advances in Cryptology – Crypto '95, LNCS 963*, 1995, pp. 457–469.

[14] Y. Frankel, P. Gemmell, P. MacKenzie, and M. Yung, "Proactive RSA," in *Advances in Cryptology – Crypto '97, LNCS 1294*, 1997, pp. 440–454.

[15] T. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Advances in Cryptology – Crypto'91, LNCS 576*, 1992, pp. 129–140.

[16] C. P. Schnorr, "Efficient signature generation for smart cards," *Journal of Cryptology*, vol. 4, no. 3, pp. 239–252, 1991.

[17] Choonsik Park and Kaoru Kurosawa, "New ElGamal type threshold digital signature scheme," *IEICE TRANS Fundamentals Special Section on Cryptography and Information Security*, vol. E79-A, no. 1, pp. 86–93, 1996.

[18] Douglas R. Stinson and Reto Strobl, "Provably secure distributed schnorr signatures and a (t, n) threshold scheme for implicit certificates," in *ACISP 2001, LNCS 2119*, 2001, pp. 417–434.

[19] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Secure distributed key generation for discrete-log based cryptosystem," in *Eurocrypt '99*, 1999, pp. 295–310.

[20] Y. Frankel and Y. G. Desmedt, "Parallel reliable threshold multi-signature," Tech. Rep. TR-92-04-02, Dept. of EECS, University of Wisconsin-Milwaukee, 1992.

[21] Charles E. Perkins and Elizabeth M. Royer, "Ad-hoc on-demand distance vector routing," in *WMCSA'99*, 1999.

[22] Thomas Clausen, Philippe Jacquet, Anis Laouiti, Pascale Minet, Paul Muhlethaler, Amir Qayyum, Laurent Viennot, and INRIA Rocquencourt, "Optimized link state routing protocol," Sept. 2001, Internet draft, draft-ietf-manet-olsr-06.txt.

[23] Haiyun Luo, Jiejun Kong, Petros Zerfos, Songwu Lu, and Lixia Zhang, "Self-securing ad hoc wireless networks," in *Seventh IEEE Symposium on Computers and Communications (ISCC'02)*, 2002.