Self-Managed Heterogeneous Certification in Mobile Ad Hoc Networks

Weihong Wang, Ying Zhu, Baochun Li

***Abstract*— As mobile ad hoc networks grow into a pervasive computing infrastructure, it is commonplace for wireless nodes owned by different entities to collaborate and communicate with one another. However, in cases where identity authentication is required to secure the communications, a new problem will be raised. On one hand, certificates of different nodes are possibly issued by different *Certificate Authorities* (CAs), thus, nodes may not be able to authenticate the identity of each other if they do not trust the CAs associated with their communicating parties; on the other hand, as networks scale up and the variety of CAs increases, it will become increasingly difficult to decide the trustworthiness of different CAs through human intervention. In this paper, we propose a self-managed heterogeneous certification scheme, in which multiple distributed nodes cooperatively carry out the functionality of each CA. Nodes may trust a different CA, if there exist sufficiently many nodes which are trustworthy to them and which also constitute that CA. The scheme eliminates the necessity of maintaining any dedicated CA nodes in mobile ad hoc networks, and trust of nodes in heterogeneous CAs can be managed securely and automatically by mobile nodes themselves. Our simulation results have shown that the proposed mechanism can evidently enhance the success ratio of identity authentication between communicating nodes.**

I. Introduction

In almost all the prevalent encryption systems, asymmetric key encryptions are regarded as a secure way for two communicating nodes to deliver their symmetric encryption key that protects their end-to-end communication. To prevent forgery of identities in this procedure, *digital certificates* issued by trustworthy *certificate authorities* (CAs) are commonly employed as a hard bind between the identity of a node and its public encryption key. Thus, a node A believes the correspondence between a node B's ID and its claimed public key, only if B holds a digital certificate signed by a CA that A trusts. For this purpose, it is highly demanded to employ dedicated nodes as CAs, which makes the certification service more *reliable*.

However, in mobile ad hoc networks, unpredictable mobility patterns, as well as the freedom for nodes to join or leave the network on the fly, make it infeasible to maintain any dedicated certificate authorities. Instead, the certification schemes must be designed to be *robust*, *ubiquitous* and *scalable*, besides being reliable. In this context, implementations that distribute the functionality of one CA into multiple mobile nodes, based on *threshold secret sharing*, have proved to be an effective solution [1], [2], [3].

Nevertheless, an important issue has been left out by previous research work. Since mobile ad hoc networks are highly susceptible to security attacks, the distributed certificate authorities and the certificates issued by these CAs need to be updated periodically. Failures in performing these updates make the associated CAs appear to be untrustworthy, even if these failures are due to non-security reasons, for example, node mobility and power depletion. Further, existing research tends to assume that there exits only one CA in the network, which may not hold when nodes from different administrative domains constitute a network. As mobile nodes move and route, there would inevitably appear some cases that two communicating nodes are not able to authenticate the identity of each other. The reason is that nodes may be associated with heterogeneous CAs, and there might not exist sufficient reason for them to trust CAs that are off their own lists of trusted ones. The problem will be especially evident when the mobile ad hoc network involves into a pervasive computing environment, where various types of devices produced by different manufactures may be present. In those cases, to decide the trustworthiness of other CAs, human intervention would become indispensable in adjusting relevant configurations on the spot. Clearly, to manage such trust relationship through human intervention is increasingly inconvenient or hard, as the range of the network scales up and the variety of CAs increases. Therefore, we are in need of a certification and authentication scheme, in which trust in heterogeneous CAs can be decided and managed by wireless nodes themselves.

In this paper, we propose a novel distributed certification mechanism, which *automates* the management of trust relationship within the heterogeneous CA environment. Under this mechanism, we regard distributed CAs constructed on the basis of K-threshold secret sharing as the primary form of certificate authorities. Thus, the conditions for a node to perceive a functioning CA are equivalent to the conditions for the node to trust the authenticity of that CA. If two communicating nodes temporarily do not have any trusted CAs in common, both of them would seek to find a trustworthy CA system that is also trusted by the other. In this way, the authenticator attempts to verify the identity of the other node by accepting a new CA, while the node to be authenticated attempts to obtain a new certificate that is verifiable by the authenticator. In this procedure, nodes rely on *transitive authentications* between themselves and members of the new CA system, to extend their trust in the heterogeneous CA environment.

The remainder of the paper is organized as follows. Sec. II addresses the K-threshold certification mechanism; Sec. III presents the self-managed heterogeneous certification scheme; Sec. IV describes the simulation results regarding the perfor-

The authors are affiliated with the Department of Electrical and Computer Engineering, University of Toronto. Their email addresses are {*wwang, yz, bli*}*@eecg.toronto.edu*.

mance of our scheme, and Sec. VI concludes the paper.

II. Distributed Certificate Authority

To eliminate human intervention as much as possible in the procedure of trusting new CAs and obtaining certificates, it is most advantageous to resort to certificate authorities that are self-managed by the independent nodes themselves. In this aspect, Kong *et al.* [1] have introduced a distributed certificate authority system based on K-threshold secret sharing and the RSA algorithm.

In their work, the entire network is assumed to trust a homogeneous CA, which corresponds to a (PK, SK) pair. PK is the public key that is publicly known to all the nodes in order for them to parse digital certificates signed by that CA; SK is the secrete key that is only known to the CA and is used to sign certificates. During bootstrapping of the CA , the (PK, SK) pair is decided by an external authority, where $SK = \langle d, n\rangle$ is the RSA secret key. A polynomial $f(x)$ of degree $(K-1)$ is randomly selected, so that $f(0) = d$. Any arbitrary node A_i, once becomes part of the CA, will be allocated a *secret share* of $P_{A_i} \equiv (f(A_i) \bmod n)$, and is thus referred to as a *share holder* of the CA.

Based on K-threshold sharing, the functionality of a CA is distributed to K separate nodes, and K is referred to as the *threshold* of secret sharing. Knowing the IDs of $(K-1)$ other share holders, any share holder A_j can calculate its *share of secret key* SK_{A_j} by Lagrange interpolation:

$$SK_{A_j} = P_{A_j} \cdot l_{A_j}(0)$$

where $l_{A_j}(x)$ is the Lagrange coefficient of A_j:

$$l_{A_j}(x) = \frac{\prod_{m=1, m\neq j}^{K} (x - A_m)}{\prod_{m=1, m\neq j}^{K} (A_j - A_m)}$$

Further, given K share holders, the secret key SK may be collectively recovered by Lagrange interpolation:

$$d \equiv \sum_{j=1}^{K} (P_{A_j} \cdot l_{A_j}(0) \bmod \text{n}) \equiv \sum_{j=1}^{K} (SK_{A_j} \bmod \text{n})$$

In this way, the secret key SK is shared among the K share holders, and the K nodes jointly form a *distributed CA system*, with each of them accomplishing part of the encryption procedure. For example, given a message m, the K secret share holders can cooperatively sign it as

$$m^{SK_{A_1}} \cdot m^{SK_{A_2}} \ldots m^{SK_{A_K}} = m^{SK_{A_1} + SK_{A_2} + \ldots + SK_{A_K}}$$

In the previous work, nodes request for certificates from such a CA system by searching for share holders in their one-hop neighborhood. If K eligible share holders are available within a node A_j's one-hop proximity, a certificate $\langle X_{A_j}, X_{A_j}^{(SK_{A_1} + \ldots + SK_{A_K})}\rangle$ can be issued to A_j by the distributed CA they have formed. Here, X_{A_j} stands for $\langle A_j, \overline{pk}_{A_j}, T_{sign}, T_{expire}\rangle$; $\overline{pk}_{A_j}$ is the acclaimed public key of A_j; T_{sign} and T_{expire} indicate the valid period of the certificate. Each component $X^{SK_{A_i}}$ is referred to as a *partial certificate* of node A_j.

Since the secret key SK is distributed to K nodes, adversaries must compromise K share holders before starting to crack SK. If node A_j has found K share holders which hold verifiable certificates regarding the same CA, it is justifiable for the node to trust the authenticity of the CA system formed by these nodes. Moreover, once certified by the CA system, A_j can spontaneously obtain a secret share P_{A_j}:

$$P_{A_j} \equiv \sum_{m=1}^{K} (P_{A_m} \cdot l_{A_m}(A_j) \bmod \text{n})$$

and gains the ability to issue partial certificates for other nodes. Therefore, the functionality of the distributed CA system can be fulfilled by any K valid share holders in the network.

Although the network is tolerant to up to $(k-1)$ compromised nodes, to further improve system resistance to attacks, nodes are demanded to update their certificates and secret shares periodically, following the same procedure of obtaining new certificates. Failure in collecting enough share holders would make their past certificates expire and make the CA untrustworthy.

Clearly, the distributed CA system can be automatically maintained by the participating nodes, once it is initialized. Assume none of the distributed CA systems in the mobile ad hoc network are created to be malicious, then having K valid share holders would manifest the trustworthiness of the certificate authority. Based on this feature, we develop the self-managed heterogeneous certification mechanism for mobile ad hoc networks.

III. Trust Migration in the Heterogeneous CA Environment

We assume that every node keeps a physically unforgeable identification (*e.g.*, an ID recorded in a smart card) as its fundamental proof for receiving an original certificate; besides, an end-to-end *mutual* authentication procedure is required before any critical session begins, upon the success of which a symmetric encryption key can be created and shared between the two nodes during the session.

Based on the distributed CA systems, we extend previous work in three aspects. First of all, we allow the coexistence of heterogeneous CA systems in mobile ad hoc networks. Due to periodic certificate updates and communication activities, nodes may drop and obtain certificates depending on their physical locations and network dynamics. Dynamically, nodes each maintain a list of CAs that they currently trust, *i.e.*, the CAs that have signed the certificates their are currently holding. Secondly, during the procedures of obtaining certificates or updating secret shares, share holders are searched for in the multi-hop neighborhood of the node considered, rather than the one-hop proximity. Thus, the probability of a CA system appearing untrustworthy is reduced. Thirdly, before any critical end-to-end session begins, both nodes attempt to find a CA system (in a procedure we would refer to as *Distributed Multi-hop Certificate Request* (DMCR) procedure) that they both trust in their own locality. The goal is to have either

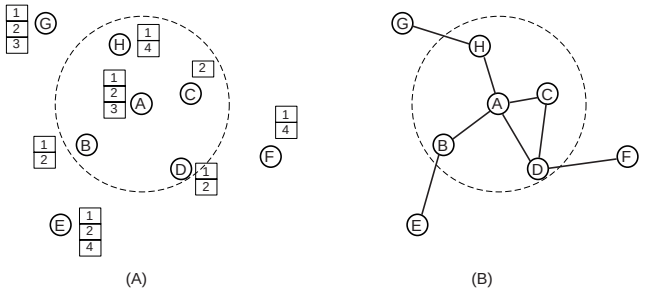

Fig. 1. An illustration of trust graph. (A) The network topology. Every node records a list of CAs it currently trusts. CAs are labeled with numbers. (B) The trust graph associated with the network (assume all the nodes hold authentic certificates).

side certified by a CA that it decides to be trustworthy and is trusted by the other side, so that the identities of both nodes can be verified, based on a commonly trusted CA.

A. Trust Graph

In dealing with valid share holders of a specific CA, we demand that the certificate requester always receive authentic messages, even if messages come through multiple hops. Thus, relevant messages must always be exchanged between nodes that are able to authentic each other. For this purpose, we define a *trust graph* in the network, which illustrates the trust relationship among different nodes. Here, we refer to *trust* of node A in node B as the fact that node B can be verified as authentic based on B's digital certificate signed by a CA that A currently trusts.

Fig. 1 (A) demonstrates a simple network, where every node has its own list of trusted CAs. Due to the intersections of their lists, some nodes can verify the identities of one another. In the graph in Fig. 1 (B), nodes are interconnected if they are within one-hop distance from each other and trust each other. Note that the topology of the trust graph may change over time, since certificates may expire or be obtained.

When searching for valid share holders, the goal of the node is to find enough candidates that indeed come from a CA that is not yet cracked. When share holders have to be found outside its one-hop neighborhood, the node must be sure of the identities of related relaying nodes. Therefore, over every hop of message forwarding, mutual authentications are always performed, so that only those nodes that are reachable along the edges of the trust graph can be taken as trustworthy share holders. We are now at the stage to present the mechanism that deals with heterogeneous CAs.

B. Heterogeneous Certification

Suppose node I is to start a critical session with node A (shown in Fig. 2). Before performing mutual authentication, the two nodes independently carry out DMCR procedures, which proceed in three phases. In the description that follows, we set the physical range of searching share holders to two hops, and take the behavior of A for instance, since I and A perform symmetrically.

Phase 1: I sends a certificate inquiry to A, which contains CAlist_I, the list of CAs that I currently trusts. Each member of CAlist_I is a unique identifier of a CA system, which may be either an ID or a public key. In return, A sends back CAlist_A.

Phase 2: A compares CAlist_I with CAlist_A. If there exists a common CA that both nodes trust, A then proceeds to send I its certificate signed by this CA, and in return, receives I's corresponding certificate. If multiple common CAs exist, the one with the smallest value is chosen for consistency on both sides. Otherwise, A would attempt to find a CA that it may deem trustworthy and is trusted by I, and seek to gain the ability to authenticate the identity of I. The search is done in two steps:

(1) A finds the set of nodes $N_A^{1*} \subset N_A^1$, where N_A^1 consists of the one-hop neighbors of node A in the trust graph (thus are mutually trustworthy with A); among them, nodes that also share common CA(s) with node I form N_A^{1*}. In the example shown in Fig. 2 (*Phase 2*), N_A^1 corresponds to $\{B, C, D, H\}$, and N_A^{1*} corresponds to $\{H\}$.

(2) A finds the set of nodes $N_A^{2*} \subset N_A^2$, where all members of N_A^2 are two-hop neighbors of node A in the trust graph (thus are mutually trustworthy with members of N_A^1); among these nodes, those which also share common CA(s) with node I constitute N_A^{2*}. In Fig. 2, nodes E, F, and G form the set N_A^2, while E and F constitute N_A^{2*}. Members of N_A^{1*} and N_A^{2*} send to A their own IDs and the lists of commonly trusted CAs they share with node I.

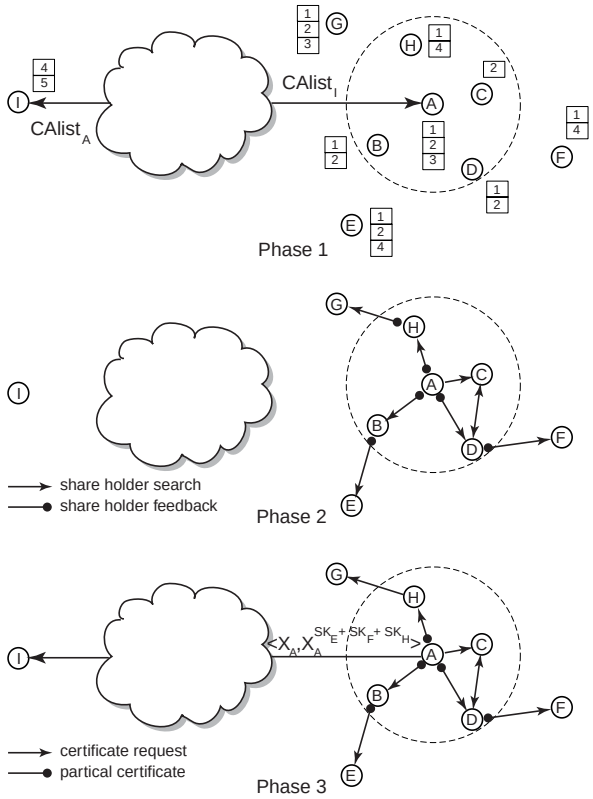

Fig. 2. An illustration of the DMCR procedure carried out at node A.

Phase 3: A chooses a set of nodes ($\{E, F, H\}$ in the example of Fig. 2) from $N_A^{1*} \bigcup N_A^{2*}$, which shares a common CA with I, and the size of the set is the largest for all the CA(s) shared by I and nodes in $N_A^{1*} \bigcup N_A^{2*}$. If the size reaches K, the corresponding CA is deemed to be trustworthy to A, since all the candidate share holders are guaranteed to be authentic through the identity authentications performed on the edges of the trust graph. Hence, node A is now ready to authenticate the

identity of node I. At the same time, A select K share holders from the set and request for a certificate from them, in order to have its identity verifiable by node I. If A cannot find enough share holders for any CA trusted by I, the DMCR procedure performed by A fails, The trust relationship between I and A is only verifiable if the DMCR procedure performed by node I results in a certificate verifiable to node A issued. Note that members of N_A^{1*} and N_A^{2*} are taken as intermediate *references* between nodes (*e.g.*, I and A) that originally trusted different CAs to extend their trust to other CAs.

Certificate renewals and secret share updates can be carried out through steps similar to *phase 2* and *phase 3*. When considering the self-management of nodes' certificates from heterogeneous CAs, we have been focused on distributed CA systems that are based on K-threshold secret sharing; however, in choosing intermediate references, those nodes that can be authenticated through other CA schemes are also deemed to be trustworthy.

The distributed algorithms executed by the node (again, we take node A as an example) searching for a trustworthy CA and its neighboring nodes are formulated in Table I and Table II.

TABLE I

PROCEDURE OF SEARCHING FOR VALID SHARE HOLDERS

Node A, upon receiving CAlist_I:
  if $\text{CAlist}_A \cap \text{CAlist}_I \neq \varnothing$
    sends I its certificate signed by the common CA with the smallest ID
  else
    sets a timer
    broadcasts CAlist_A to one-hop neighbors
    if K valid share holders of the same CA are found before the timer expires
      proceeds to get a certificate from the CA
    else
      the procedure fails

Node B (an arbitrary neighbor of node A), upon receiving CAlist_A:
  if $\text{CAlist}_B \cap \text{CAlist}_A \neq \varnothing$
    if $\text{CAlist}_B \cap \text{CAlist}_I \neq \varnothing$
      unicasts its ID, $\text{CAlist}_A \cap \text{CAlist}_B$ and $\text{CAlist}_B \cap \text{CAlist}_I$ to A
    if CAlist_A is received from A
      broadcasts CAlist_A to one-hop neighbors

TABLE II

PROCEDURE OF OBTAINING A CERTIFICATE FROM THE NEW CA SYSTEM

Node A, upon collecting K IDs of valid share holders of the same CA:
  broadcasts X_A with the ID list to one-hop neighbors
  set a timer
  if K partial certificates are received before the time expires
    computes P_A and SK_A
    proceeds to authenticate the certificate of node I and send its new certificate to I
  else
    the procedure fails

Node B (an arbitrary neighbor of node A), upon receiving X_A:
  if B is in the list
    computes $l_B(0)$, SK_B and $X_A^{SK_B}$
    unicasts the partial certificate $X_A^{SK_B}$ to A
  if X_A is received from A
    broadcasts X_A and the ID list of K share holders to one-hop neighbors

IV. SIMULATION

We evaluate the performance of our *heterogeneous CA* mechanism and compare it with that of the *homogeneous CA* mechanism [1] through simulations (assuming all CAs are based on K-threshold secret sharing). In particular, we are focused on one major advantage of the heterogeneous certification mechanism: significantly improving the *success ratio* of mutual authentication between communicating parties when multiple CA systems exist in the network.

A. Simulation Settings

We randomly place $N = 100$ nodes in a 1000×1000 unit^2 square region, and assign to them random velocities which are uniformly distributed in $[-10, 10]$ unit/time step in x and y directions. The transmission range of each node is set to be 100 units. We define the maximum number of CA systems in the entire network to be $MAX\_CA = 8$, and randomly allocate to each node a list of trusted CAs, at the beginning of the simulation. Within each initial list, CAs are numbered between 1 and MAX_CA, and the number of CAs in the list is uniformly distributed between 1 and C, which is adjustable in the simulation. The threshold K is chosen as 3. The searching distance for share holders is restricted to two hops.

We determined the success ratio of mutual authentication as follows. At any time instant, assume all pairs of nodes are equally likely to start sessions that require identity authentication. Under the heterogeneous CA mechanism, both sides of these sessions will attempt to authenticate their communicating parties through DMCR procedure. Once either side succeeds in its DMCR procedure, the pair can proceed to perform authentication between each other. Under the homogeneous CA mechanism, however, nodes do not seek to join or obtain certificates from other CA systems, thus only deal with authentications of nodes that originally share commonly trusted CAs with them. In both mechanisms, the success ratio is computed as the number of node pairs that can build verifiable trust relationship between each other divided by the total number of attempts to start sessions.

Further, at every time step, a pair of nodes is randomly chosen to *really* start a new session, which will last for a random number of time steps. Under the heterogeneous CA mechanism, if nodes succeed in their DMCR procedures, their lists of trusted CAs will be extended accordingly. In both situations, nodes periodically attempt to renew their certificates from associated CA systems. When nodes fail to find enough trustworthy share holders in their two-hop neighborhood, the corresponding certificates are dropped.

B. Simulation Results

We have carried out two groups of experiments to compare the performance of the two mechanisms. Fig. 3 shows the comparison of success ratios in the initial stage of the simulation. Under the heterogeneous CA mechanism, the possibility for nodes to obtain new certificates evidently improves the success ratio of possible mutual authentications. As implied by the results, the more CAs a node trusts, the more probable it can set up a verifiable trust relationship with other nodes. Trusting at most one CA system reduces the heterogeneous

CA mechanism to a homogeneous CA system, since there no long exist any intermediate references that can verify the authenticity of share holders from a different CA system. Between the two trivial cases of $C = 1$ and $C = 8$, the heterogeneous CA mechanism always renders higher success ratios than the homogeneous CA mechanism.

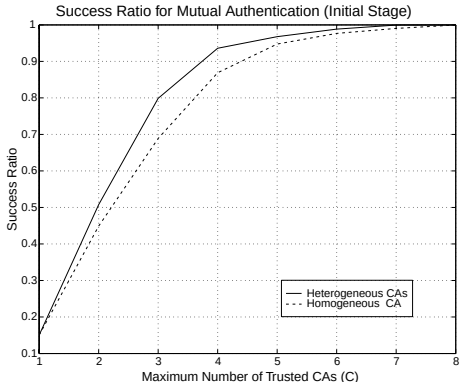

Fig. 3. Comparison of success ratios for mutual authentication at the initial stage.

Fig. 4 shows the results obtained in a dynamic procedure during which nodes are constantly moving. Under the heterogeneous CA mechanism, nodes can extend their lists of trusted CAs, if they obtain new certificates when communicating with nodes that used not to share common CAs with them. Especially, as nodes move around in the network, heterogeneous CAs tend to *merge* together when each node joins more CA systems. Although certificates may sometimes be dropped due to renewal failures, they might be re-issued later on as nodes get involved in future sessions, therefore, the success ratios can be significantly enhanced. On the contrary, under the homogeneous CA mechanism, once a certificate expires, the node cannot re-gain it without human intervention, thus, the success ratio keeps going down. In the simulations, certificates are assumed to expire after 20 time steps.

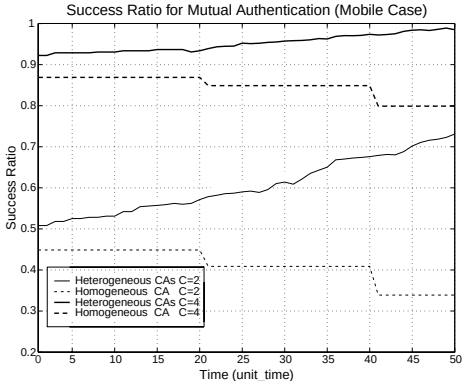

Fig. 4. Comparison of success ratios for mutual authentication in the mobile case.

Our simulations results have demonstrated that the self-managed heterogeneous certification mechanism can substantially increase the probability of successful mutual authentication between nodes that may be associated with different CA systems. The mechanism incurs little human intervention, and only imposes moderate requirements on storage and computing capabilities. Each node only needs to store a list of PKs, secret shares, and identifiers of CAs, and to compute intersection of two CA lists for at most K times in each DMCR procedure. The number of messages transmitted in the DMCR procedure is $O(d^2)$, where d is the average degree of a node in the topology graph.

V. Related Work

In previous work, Gokhale *et al.* [2] and Venkatraman *et al* [4] have introduced distributed authentication schemes for mobile ad hoc networks, which allow for *transitive trust* or *certificate chains* to assist in setting up trust relationship between different nodes. Although their strategies are logically similar to ours, they have not explored the problem of dynamically managing their trust in heterogeneous CAs based on the capabilities of the nodes themselves. Besides, their approaches tend to rely on designated controllers for different groups of mutually trusted nodes, thus incur overhead and complexity in controller hierarchy maintenance, which are non-trivial in mobile ad hoc networks. Kong *et al.*[1] have provided a fully distrusted certificate authority mechanism, which serves as a good foundation for our work.

VI. Conclusion

It is highly demanded in mobile ad hoc networks to develop fully distributed certification scheme. Based on previous work [1], we propose an enhanced system, by allowing for secure interactions between nodes associated with heterogeneous CAs. In the new scheme, a different CA is deemed trustworthy if mutually trusted reference nodes constitute a connection from the node to the CA. The scheme enables certificates, as well as the functionality of distributed CAs to automatically permeate their original geographic boundaries through self-management among mobile ad hoc nodes, and still achieves reliability, ubiquity, robustness and scalability, with low communication and computation costs.

References

[1] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang, "Providing Robust and Ubiquitous Security Support for Mobile Ad-Hoc Networks," in *Proceedings of International Conference on Network Protocols*, 2001.

[2] S. Gokhale and P. Dasgupta, "Distributed Authentication for Peer-to-Peer Networks," in *Proceedings of IEEE Workshop on Security and Assurance in Ad hoc Networks*, 2003.

[3] A. Khalili, J. Katz, and W. Arbaugh, "Toward Secure Key Distribution in Truly Ad-Hoc Networks," in *Proceedings of IEEE Workshop on Security and Assurance in Ad hoc Networks*, 2003.

[4] L. Venkatraman and D. Agrawal, "A Novel Authentication Scheme for Ad Hoc Networks," in *Proceedings of Wireless Communications and Networking Confernce*, 2000.