INTERNATIONAL JOURNAL OF COMMUNICATION SYSTEMS
Int. J. Commun. Syst. 2009; **22**:1177–1197
Published online 29 April 2009 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/dac.1015

A robust certification service for highly dynamic MANET in emergency tasks

M. Ge[1], K. Y. Lam[1,*,†], D. Gollmann[2], S. L. Chung[3], C. C. Chang[4] and J. B. Li[5]

[1]*Key Laboratory for Information System Security, Ministry of Education, Tsinghua National Laboratory for Information Science and Technology, School of Software, Tsinghua University, Beijing, People's Republic of China*
[2]*TU-Hamburg-Hargurg, Germany*
[3]*The Open University of Hong Kong, Hong Kong*
[4]*Feng Chia University, Taiwan*
[5]*State Administration of Taxation, People's Republic of China*

SUMMARY

In emergency tasks, cross-agency operations being carried out in disaster-hit areas require some supporting communication system for command and control. Mobile Ad hoc Network (MANET) is a very suitable way to meet such communication requirements since it can function without any pre-installed communication infrastructure. Owing to potential threats in the field environment and the unique features of MANET (e.g. the open nature of wireless links and the absence of security infrastructure), security of communications over MANET is a serious issue that is typically addressed by asymmetric cryptographic mechanisms. In this paper, we tackle issues critical to asymmetric key management in MANET, which almost invariably serves as a basis of security services in a network environment. To address the deficiencies of existing key management schemes, we propose the concept of mission-specific certificate to manage public keys in our scenario. For issuance and/or revocation of mission-specific certificate, a Mission-specific Certificate Authority (MCA), which consists of a collection of server nodes to operate the threshold cryptographic scheme, is proposed. Furthermore, to cater for the occurrence of network partitioning, which is common in highly dynamic MANET, we propose a partition-tolerant mechanism for MCA by introducing the notion of auxiliary server nodes. We discuss the security and performance of our scheme and show that our approach is a secure and partition-tolerant mechanism can effectively improve availability of the MCA. Copyright © 2009 John Wiley & Sons, Ltd.

Received 30 June 2008; Revised 31 January 2009; Accepted 21 February 2009

KEY WORDS: MANET; security; key management; network partitioning

*Correspondence to: K. Y. Lam, School of Software, Tsinghua University, Beijing 100084, People's Republic of China.
†E-mail: lamky@mail.tsinghua.edu.cn

1. INTRODUCTION

In emergency tasks, cross-agency operations are usually carried out in disaster-hit areas (e.g. in the aftermath of earthquake, tsunami, hurricane, etc.). The operation execution requires command and control during all the time; thus, some supporting communication system is necessary. However, building such a system is challenging since there might be no communication infrastructure or the existing infrastructure might have been destroyed. Mobile ad hoc network (MANET) that typically consists of heterogeneous mobile nodes connected by wireless links in an ad hoc manner is suitable for providing communication services in such cases since it can function without any pre-installed communication infrastructure [1].

In a typical disaster-hit area, irrelevant or even malicious parties may exist; thus, the communication system has to be properly protected. Security services such as schemes proposed in [2, 3] could be used.

As the basis of almost all security services, key management is essential for securing communication systems. To communicate securely with other participants of the emergency task, one has to first obtain the associated cryptographic keys of those participants. Owing to the open nature of wireless links and the absence of security infrastructure, security of communications over MANET is typically addressed by asymmetric cryptographic mechanisms. For traditional public key infrastructures (PKIs) [4], it is reasonable to assume that an internal PKI exists in each participating organization. For example, an Internal Certificate Authority (ICA) could be set up to provide a certificate service for entities within an organization. A certificate generated in this manner assures that the entity belongs to this organization and that the public key is associated with the entity. However, internal PKIs are not appropriate in cross-agency operations, where cross-agency certificates verifiable by all participants are required. The provision of such a certificate service is still technically challenging.

On the other hand, though the use of cross-certificates is another possible option widely adopted to support cross-organization transactions in electronic commerce systems, it is not suitable in our case for at least two reasons. First, emergency missions by their very nature are assembled ad hoc; hence, it is not realistic to expect in advance installation of verification keys or cross-certificates for all participants on all devices to be used in the field. Second, it is obvious that not all members of an organization are assigned, hence authorized, to participate in the emergency mission. Furthermore, as a good security practice, long-term cross-certificates are better not be used in short-term emergency missions. While it might be desirable to tear down the 'trust relationships' established during the missions at the end, e.g. to make sure that all cross-certificates are revoked, revocation of cross-certificates is very difficult to enforce effectively.

An alternative solution more suitable for our case is the use of short-term and cross-agency verifiable mission-specific certificates. The mission-specific certificates could be issued on-demand for participants on the ground and used to certify their role and their public keys. Since the whole life cycle of the mission-specific certificate only exists during the mission operations, it avoids the problems experienced by cross-certificates. In order to manage mission-specific certificates that are desirable in our case, we propose the notion of a Mission-specific Certificate Authority (MCA) and cryptographic mechanisms to implement the MCA in an MANET environment. The MCA helps to establish trust relationships among mission participants from different organizations based on the trust relationship between the MCA and the ICAs. MCA needs to be highly available in order to support the operations needed for issuing, updating and revoking mission-specific certificates. Inspired by the approaches of Distributed Certificate Authority (DCA) [5–10], we further distribute the authoritative power of the MCA (i.e. the power to issue/update/revoke certificates) to more nodes in order to achieve better partition tolerance.

Our proposed scheme works as follows. The MCA scheme is implemented by a group of distributed server nodes. The MCA service is used by other nodes (the clients) to generate mission-specific certificates. When there are enough servers, some client nodes are selected and converted by the servers into auxiliary servers, which do not have the privilege to sign until they are activated by the servers at a later stage. When a mission-specific certificate needs to be issued in an isolated segment of a partitioned MANET, where there are not enough servers, some of the auxiliary servers are activated by the remaining servers. Once activated, the auxiliary nodes behave like a normal server in terms of ability to participate in MCA operations. The activated auxiliary nodes as well as the remaining servers collaborate to issue mission-specific certificates. When the service of an activated auxiliary server is no longer needed, it is deactivated by the normal servers, thus losing its ability to participate in MCA operations.

The rest of the paper is organized as follows. In Section 2, we briefly describe the idea of threshold-based DCA, which provides a basis for the MCA. In Section 3, we propose the system model of the MCA. In Section 4, the details of the proposed scheme are presented. In Section 5, we discuss the proposed approach in aspects of security and effectiveness, respectively. Finally, we present the related work and conclude our work in Section 6.

2. THRESHOLD-BASED DCA

Threshold-based DCA was proposed in [5–10] for providing certificate services in MANET. The basic components of a threshold-based DCA consist of a dealer and a number of server nodes. The dealer is a trusted entity that takes charge of computing shares from the CA's private signature key based on the threshold scheme, and securely distributing them to the server nodes. The server nodes collaborate to issue certificates as long as there is at least a threshold number of them available.

Before being deployed, the DCA has to be initialized. In this process, a trusted dealer is usually assumed to operate the initialization. The goal of initialization is to distribute the shares of the CA's private signature key to n servers based on an (n,t) threshold scheme. Here Shamir's secret sharing [11] is usually adopted.

Shamir's secret sharing. Let q be a prime number, $s \in Z_q$ be the private signature key to be distributed and $f(x)$ denotes a $t-1$ degree polynomial with random coefficients over Z_q subject to the condition $f(0)=s$. For each server S_i, let x_i refer to a unique integer identifier in Z_q. The shares of the key s could be evaluated as

$$s_i = f(x_i) = s + a_1 x_i + a_2 x_i^2 + \cdots + a_{t-1} x_i^{t-1} \bmod q \quad (i = 1, 2, \ldots, n)$$

Any t of n shares could reconstruct the secret later as

$$s = \sum_{i=1}^{t} f(x_i) l_i(0) \bmod q$$

where $l_i(0) = \prod_{j=1, j\neq i}^{t} x_j/(x_j - x_i) \bmod q$.

After generating the shares, the dealer then securely distributes each share s_i to the server S_i, respectively, in an out-of-band way. Meanwhile, the public key of the DCA is also given to all the nodes for verifying DCA certificates later.

After successful completion of initialization, the DCA works as follows. When one requests the DCA for a new or updated certificate, the request is first sent to a proxy server, which is one of the n servers chosen by the requestor. The proxy server then forwards the request to other servers. Upon receiving the request, each server independently verifies the binding of the entity with its public key. If the verification is successful, the server generates a partial certificate (signature) that is computed using its share and sends it to the proxy server. A valid certificate will be generated by the proxy server by combining at least t partial certificates. Since no shares are disclosed in this process, the combination could be done by the requestor as well, if it has enough computing resource. While the DCA mechanisms are independent of any specific threshold signature scheme, such as [12], an example of a threshold RSA signature algorithm proposed in [10] is briefly sketched as follows for illustration. It could be introduced as four basic processes.

P1: *Initializing* (*generating public key*, *private key and private key shares*). Let $N = PQ$, where P and Q are two primes. Select the public key v, satisfying $\gcd(\varphi(N), v) = 1$, $1 < v < \varphi(N)$ and evaluate the private signature key $s \equiv v^{-1} (\bmod\, \varphi(N))$. The private key is shared as $s_i = f(x_i)$ $(i = 1, 2, \ldots, n)$, where $f(x)$ denotes a $t-1$ degree polynomial with random coefficients over Z_N subject to the condition $f(0) = s$.

P2: *Generating partial signatures.* With share s_i, a partial signature on message M could be generated as $M^{s_i l_i(0)(\bmod N)} \bmod N$, where $l_i(0) = \prod_{j=1, j\neq i}^{t} x_j/(x_j - x_i) \bmod N$.

P3: *Combining*. The t (or $>t$) partial signatures could be combined as

$$M^{\sum_{i=1}^{t} (s_i l_i(0) \bmod N)} = \prod_{i=1}^{t} M^{s_i l_i(0)(\bmod N)} \bmod N$$

Since the equation $\sum_{i=1}^{t} (s_i l_i(0) \bmod N) = kN + s$ holds for certain k, one has to further compute the valid signature M^s from M^{kN+s}. To get the valid signature, one could set $Y = M^{kN+s}$ and repeatedly compute $Y \leftarrow Y \cdot M^{-N} \bmod N$ until $M \equiv Y^v (\bmod N)$, then Y is the valid signature.

P4: *Verifying*. The signature $Y = M^s$ is valid if and only if $M \equiv Y^v (\bmod N)$.

DCA schemes presented in [5–9] suffer from the availability problem when operating in a highly dynamic MANET where network partitioning occurs frequently. In such case, t or more servers might only be available intermittently in some (or even all) segments, so the availability of the DCA could be interrupted.

3. THE MCA SYSTEM MODEL

The underlying system model on which the MCA is based consists of different types of network nodes that are connected by a MANET. The MANET is subject to frequent partitioning due to the dynamic nature of its topology resulting from the movement of mobile nodes. In this section, the various types of nodes and their respective behaviorial changes in face of network partitioning are explained. An overview of behavior and operations of the MCA is also provided.

3.1. Classification of nodes

From the perspective of MCA operations, previous works classified MANET nodes into two types: server nodes and client nodes. In this paper, we make use of the heterogeneity of MANET nodes

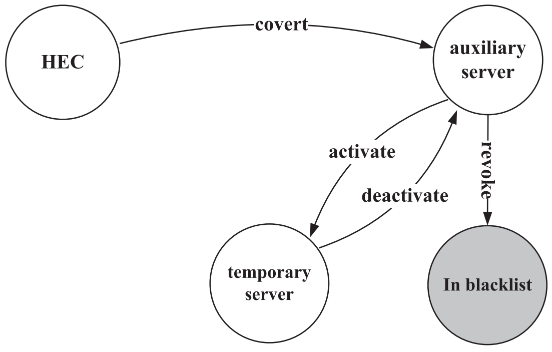

Figure 1. Status transition diagram of an HEC.

and differentiate them into three types according to their different functions: servers, high-end Clients (HECs) and low-end clients (LECs).

Server. Servers are usually relatively stable nodes with best security protection and richest in computing resources in the system. They are the basic components of the MCA, and the only nodes that have to be pre-configured with longer-term security parameters before each mission. The term *server group* denotes the set of MCA servers connected by the MANET or the segments of a partitioned MANET. Long-term security parameters stored in the servers enable secure communication channels to be established among servers.

HEC. HECs are client nodes with relatively good security protection and moderate computing resources. An HEC needs to request for a mission-specific certificate from the MCA when it joins MANET. In the MCA, they could be converted to auxiliary servers by the servers. Each auxiliary server has a *coordinating server*, which initiated the conversion. If needed, for example when there is not enough server nodes in a segment, the auxiliary servers in the segment are activated to be temporary servers. The temporary servers are deactivated after being used and the temporary servers resume their state of being auxiliary servers after deactivation. If some auxiliary servers are detected to exhibit malicious behaviors, they could be revoked by the servers and put into the blacklist on each server (see Figure 1).

LEC. LECs are client nodes other than HECs. Before joining the MANET, an LEC needs to request its mission-specific certificate through a server, which is called a *proxy server*. The proxy server is selected by each client node independently according to some local policies, such as the nearest server or the most trusted server.

Let the MANET consists of m nodes, which are divided into three categories: servers denoted by S_i $(i=1,2,\ldots,n)$, HECs denoted by C_i $(i=1,2,\ldots,l)$ and LECs denoted by c_i $(i=1,2,\ldots,m-n-l)$. The roles of the nodes, i.e. Server, HEC and LEC, are decided and stored on servers from each organization before deployment. Such information is shared among all servers once they start the MCA operation.

3.2. Network partitioning

As the network topology of a MANET may change dynamically, network partitioning may occur in an unpredictable manner. A MANET is said to be partitioned if it is divided into several isolated segments, which are disconnected from one another. The topology of a segment will last for some time, and then the segments may be either merged (re-connected) or further partitioned. We make the assumption that the duration of the segment topology is long compared with the execution time of our procedures. Thus, in most cases, the segment will not be partitioned during the execution of our procedures. This is a reasonable assumption as observed from our simulation results.

Note that, in our system, partitioning of the MCA is detected by a client only when its request for mission-specific certificates fails due to insufficient servers.

3.3. Initializing the MCA for missions

In each organization, there exists an administrator that handles security administration within the organization. In particular, an ICA in each organization is assumed to issue internal certificates to its members for internal business functions.

Before the deployment of the MCA, pre-configuration for each mission is required. In the initializing procedure, a trusted dealer is assumed as in other threshold-based DCA schemes. For practical consideration, this procedure should be as simple as possible. Although the MCA consists of servers and auxiliary servers converted from HECs, only a small number of the nodes, i.e. the servers, need to be pre-configured for each mission.

First, the dealer computes shares from the private signature key of the MCA and securely distributes them to the servers. Second, the public key of the MCA is given to each server. Third, the public keys of the other ICAs are also given to each server; hence, the action of verifying entities from other organizations is based on these internal certificates. Thus, any two servers could get public keys of each other from internal certificates and generate the session keys to establish secure communication channels. Besides, the servers are equipped with mission-specific policies to operate the MCA for the mission.

After the initialization, an MCA based on (n,t) threshold scheme (where $n \ll m$) is available for providing a global certificate service for all the nodes in the MANET. Since $n \ll m$, pre-configuration before each mission is time-efficient, hence could expedite the deployment significantly.

3.4. The MCA operations

There are five basic procedures when the MCA is working: *selecting and converting procedure* (server select/convert HEC), *requesting procedure* (LEC/HEC request server), *activating procedure* (server activate auxiliary server), *deactivating procedure* (server deactivate temporary server) and *revoking procedure* (server revoke auxiliary server).

- Selecting and converting procedure: According to the network status and task requirements, a coordinating server could launch the *selecting and converting procedure* to convert an HEC to an auxiliary server by generating an auxiliary share. The auxiliary share is generated by the auxiliary server with the help of the servers in the current server group.
- Requesting procedure: An HEC or LEC has to launch the *requesting procedure* to acquire a valid mission-specific certificate when it joins the network or when its existing certificate expired. The request is first sent to a proxy server, which then forwards the request to other servers. If there are t or more servers in the server group, the certificate could be generated by combining at least t partial certificates that are generated by servers.

- Activating procedure: The *activating procedure* could be launched by the proxy server to activate auxiliary servers when there are less than t servers in the current segment. The basic idea of this procedure is to find auxiliary servers that could be connected and activated by remaining servers in the current segment. After being activated, the auxiliary servers could be used along with the remaining servers to issue certificates.
- Deactivating procedure: As the reverse procedure of activating procedure, the *deactivating procedure* is launched at time out or on demand. The basic idea of deactivating is that remaining servers discard the temporary shares generated in the activating procedure. After the deactivating procedure, the deactivated temporary servers that lose their power to generate valid partial certificates resume their status as auxiliary servers, which may be activated again when needed.
- Revoking procedure: The *revoking procedure* could be launched by any server when an auxiliary server exhibits malicious behaviors. After the revoking procedure, the auxiliary server cannot be activated again.

4. THE PARTITION-TOLERANT MCA SCHEME

4.1. Selecting and converting

An HEC can only be converted when there are at least t servers available. The server that initializes the selecting and converting procedure for an auxiliary share is called the *coordinating server* of that auxiliary server. Note that more than one auxiliary server could be generated each time. For simplicity, we consider the situation in which only one node is converted each time.

Suppose the server group G in the current segment consists of n servers S_i $(i=1,2,\ldots,n)$. Our goal is to generate an auxiliary share for the selected HEC. Let S_{C_k} denote the coordinating server of C_k in G. This procedure consists of *three phases*: selecting an HEC, disguising all servers' shares and generating an auxiliary share. Note that we disguise the servers' shares for two purposes: protect each server's share from being disclosed and make the newly generated auxiliary share appear to be a 'random number' before it is activated. Some of our steps are inspired by the approach proposed in [13]. To guard against wiretapping and tampering, all communications among servers are transmitted via secure communication channels. These channels could be protected by symmetric cryptography established through symmetric key distribution enabled by the long-term security parameters.

Phase 1. Selecting an HEC.

- S_{C_k} chooses an HEC C_k nearby according to network conditions, such as the possibility of network partitioning and application-specific policies including conflict-of-interest rules.
- If C_k agrees to be converted, it applies to servers in G for auxiliary share. The request is signed with C_k's private key and forwarded by S_{C_k}.
- After receiving the request, each S_i verifies the signature and decides whether the request complies with security policies. If verification is successful, S_i responds to S_{C_k}.

The selecting phase is illustrated in Figure 2. The chosen HEC C_1 applies for an auxiliary share through its coordinating server S_1. Four servers (including S_1) that received the request pass the

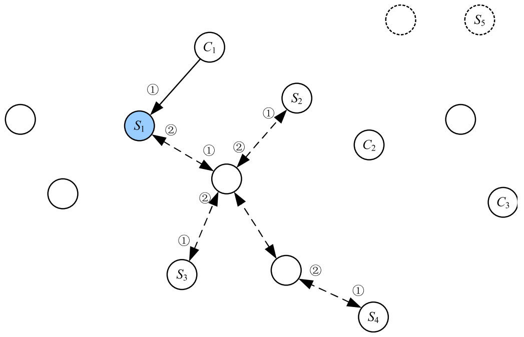

Figure 2. Selecting an HEC.

verifications and respond. In figures of this section, nodes with dashed circle are assumed to be out of the current segment. Suppose that there exist t' responding servers (including S_{C_k}) denoted by S_i $(i=1,2,\ldots,t')$. If $t'\geqslant t$, S_{C_k} will trigger the next phase on each S_i.

Phase 2. Computing disguised shares by each server.

- Each S_i independently generates a new random polynomial $g_i(x)$ with degree $(t-1)$ in Z_q.
- For each $S_j\in\{S_1,S_2,\ldots,S_{t'}\}$, S_i evaluates $g_i(x)$ at position x_j and sends $g_i(x_j)$ to S_j through the symmetric key-protected secure communication channel. If $S_j=S_{C_k}$, S_i evaluates $g_i(x)$ at position y_k and sends $g_i(y_k)$ to S_{C_k} through secure communication channel. Here x_j and $y_k\in Z_q$ are unique integer identifiers representing S_j and C_k, respectively.
- If all $g_i(x_j)$ $(i=1,2,\ldots,t')$ are obtained, S_j computes its disguised share $s'_j=s_j+\sum_{i=1}^{t'} g_i(x_j) \bmod q$.
- Each s'_i from S_i is encrypted with C_k's public key and sent to C_k through S_{C_k}.

Phase 3. Generating an auxiliary share.

- If t or more of the s'_i are received, C_k computes auxiliary share $h_k=\sum_{i=1}^{t} s'_i l_{x_i}(y_k) \bmod q$ by the Lagrange interpolation, where $l_{x_i}(y_k)=\prod_{j=1,j\neq i}^{t}(y_k-x_j)/(x_i-x_j) \bmod q$.
- C_k notifies S_{C_k} that the auxiliary share h_k is computed.
- After being notified, S_{C_k} computes $\Delta_k=\sum_{i=1}^{t'} g_i(y_k) \bmod q$, which is needed to activate C_k. Δ_k is stored by S_{C_k} and is used when it needs to activate C_k.
- S_{C_k} notifies the servers in G that the C_k has been converted successfully.
- The polynomial $g_i(x)$, together with its derived values, and s'_i are then deleted.

There exists a relationship between h_k held by C_k and Δ_k held by S_{C_k} i.e. $h_k=f(y_k)+\Delta_k \bmod q$. The selecting and converting procedure could be illustrated as Figure 3.

4.2. *Requesting and activating*

If there are enough servers in the server group of the current segment, the requesting procedure is similar to other DCA schemes (as described in Section 2). However, if there are not enough servers, the MCA can still provide certification services by taking advantage of the activated auxiliary servers.

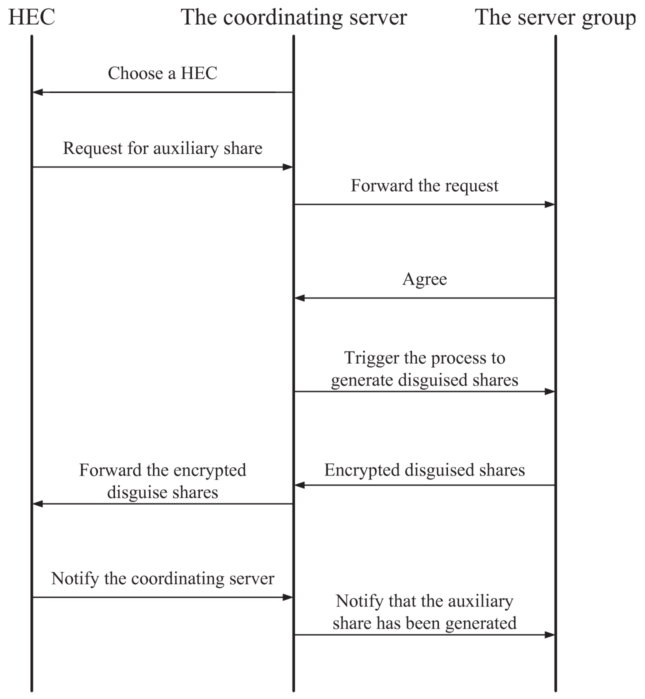
HEC
The coordinating server
The server group
Choose a HEC
Request for auxiliary share
Forward the request
Agree
Trigger the process to generate disguised shares
Forward the encrypted disguise shares
Encrypted disguised shares
Notify the coordinating server
Notify that the auxiliary share has been generated

Figure 3. The selecting and converting procedure.

Suppose only n' $(n'<t)$ servers S_i $(i=1,2,\ldots,n')$ exist in the current server group. The goal of activating procedure is to turn l' auxiliary servers C_k $(k=1,2,\ldots,l')$ into temporary servers by activating l' auxiliary shares h_k $(k=1,2,\ldots,l')$ where $l'=t-n'$.

Suppose a client is applying to the MCA for a mission-specific certificate but there exist insufficient servers (see Figure 4), its proxy server S_p (such as 'S_1' in Figure 4) will launch the activating procedure.

- S_p informs S_i $(i=1,2,\ldots,n')$ in the server group to search for connected auxiliary servers in the current segment.
- After being informed, each S_i searches for auxiliary servers whose coordinating server is S_i and responds to S_p with the search result (see Figure 5, where three HECs (C_1, C_2, C_3) are connectable and two of them (C_1, C_2) have been converted to auxiliary servers by S_1, S_2, respectively. C_1 and C_2 are available to be activated).
- If the number of connected auxiliary servers is no less than l', S_p selects l' connected auxiliary servers C_k $(k=1,2,\ldots,l')$ and triggers the activating procedure on each S_i. The trigger message includes a list of servers in the current server group and a list of selected auxiliary servers.
- After receiving the trigger message, each S_i independently generates a new random polynomial $g_i^*(x)$ with degree $(t-1)$ in Z_q satisfying $g_i^*(0)=0$.
- For each $S_j \in \{S_1, S_2, \ldots, S_{n'}\}$, S_i evaluates $g_i^*(x_j)$ and sends $g_i^*(x_j)$ to S_j through a secure communication channel.

- For each $C_k \in \{C_1, C_2, \ldots, C_{l'}\}$, S_i evaluates $g_i^*(y_k)$ and sends $g_i^*(y_k)$ to S_{C_k} through a secure communication channel.
- With the received values of $g_j^*(x_i)$ $(j=1,2,\ldots,n')$ (including $g_i^*(x_i)$) each S_i computes the temporary share $s_i^* = s_i + \sum_{j=1}^{n'} g_j^*(x_i) \bmod q$. The share s_i is retained.
- Similarly, each S_{C_k} evaluates $\Delta_k^* = \sum_{j=1}^{n'} g_j^*(y_k) - \Delta_k \bmod q$ with $g_j^*(y_k)$ $(j=1,2,\ldots,n')$ (including $g_{C_k}^*(y_k)$). The original activating material Δ_k is retained.
- After being encrypted with C_k's public key, Δ_k^* is sent to auxiliary server C_k by S_{C_k}.
- With Δ_k^*, C_k activates its auxiliary share by evaluating $h_k^* = h_k + \Delta_k^* \bmod q$.
- C_k then notifies S_p so that h_k has been activated successfully. The original auxiliary share h_k is retained.
- The polynomial $g_i^*(x)$, including the derived values, and Δ_k^* are deleted after being used.

The whole activating procedure could be illustrated as Figure 6. After the above steps, the temporary shares h_k^* $(k=1,2,\ldots,l')$ and shares of new version s_i^* $(i=1,2,\ldots,n')$ could be used together to generate a valid signature of the MCA until C_k $(k=1,2,\ldots,l')$ are deactivated. The procedure for generating certificate with the help of temporary servers is illustrated as Figure 7. In this figure, S_1 collects the partial certificates from servers S_1, S_2 and temporary servers C_1, C_2 that have just been activated. Upon receiving t or more partial certificates, S_1 could generate the DCA certificate for R.

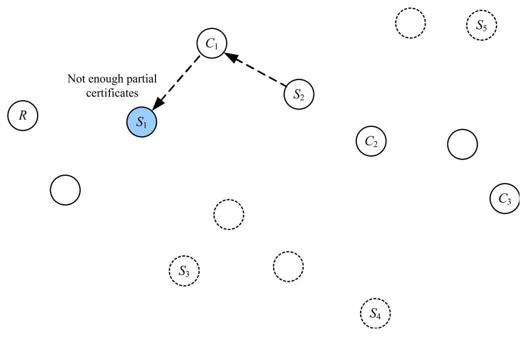

Figure 4. The request fails due to insufficient servers.

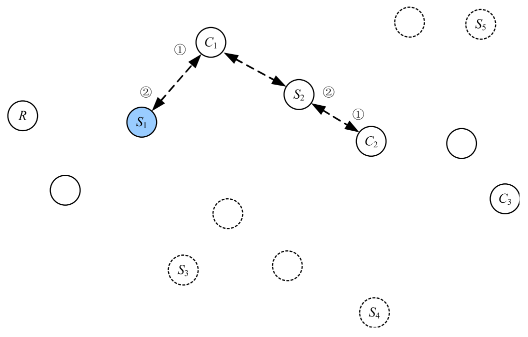

Figure 5. Searching for auxiliary servers available to be activated.

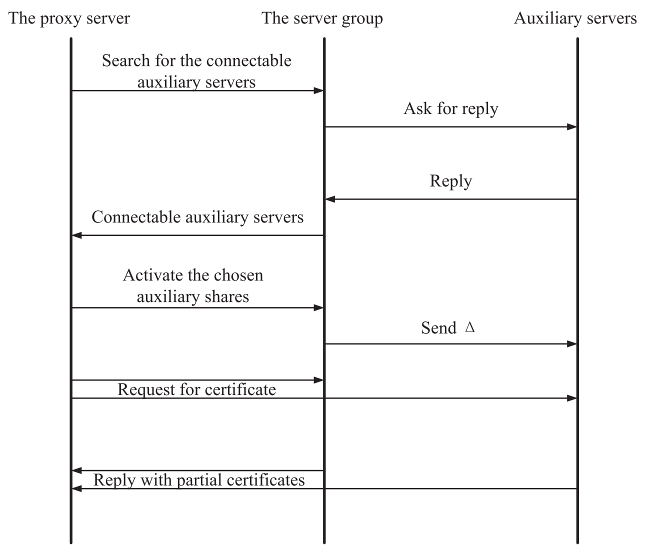
The proxy server
The server group
Auxiliary servers
Search for the connectable auxiliary servers
Ask for reply
Reply
Connectable auxiliary servers
Activate the chosen auxiliary shares
Send Δ
Request for certificate
Reply with partial certificates

Figure 6. The activating procedure.

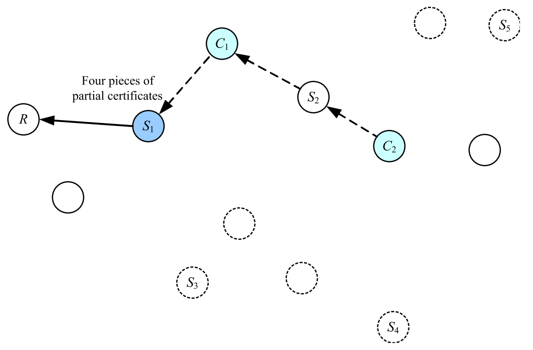

Figure 7. Generating the DCA certificate with the help of temporary servers.

We now demonstrate that one can generate a valid MCA signature from the shares held by n' servers and $t-n'$ temporary servers, whose auxiliary shares have been activated by these n' servers.

Lemma
Given h_k^* is the activated auxiliary share corresponding to h_k and $f^*(x) \leftarrow f(x)+\sum_{i=1}^{n'} g_i^*(x)$ mod q, then $f^*(y_k)=h_k^*$.

Proof
For y_k, we have $f^*(y_k)=f(y_k)+\sum_{i=1}^{n'} g_i^*(y_k)$ mod q. Since h_k is C_k's auxiliary share, hence $h_k=f(y_k)+\Delta_k$ mod q, and $f^*(y_k)=h_k-\Delta_k+\sum_{i=1}^{n'} g_i^*(y_k)$ mod q. Now that $h_k^*=h_k+\Delta_k^*$

mod q, $f^*(y_k)=h_k^* - \Delta_k^* - \Delta_k + \sum_{i=1}^{n'} g_i^*(y_k)$ mod q. Since $\Delta_k^* = \sum_{i=1}^{n'} g_i^*(y_k) - \Delta_k$ mod q, $f^*(y_k)=h_k^*$. □

Theorem
Given s_i^* $(i=1,2,\ldots,n')$ held by n' servers and h_k^* $(k=1,2,\ldots,t-n')$ held by $t-n'$ temporary servers, the private MCA signature key s could be recovered.

Proof
Since $h_k^*=f^*(y_k)$ and $s_i^*=f(x_i)+\sum_{j=1}^{n'} g_j^*(x_i)$ mod $q=f^*(x_i)$, we have t points of the polynomial $f^*(x)$ where $f^*(0)=f(0)=s$. Thus, the private signature key s could be reconstructed via Lagrange interpolation as

$$s=\sum_{i=1}^{n'} f^*(x_i)l_{x_i}(0)+\sum_{k=1}^{t-n'} f^*(y_k)l_{y_k}(0) \bmod q$$

where $l_{x_i}(0)=\prod_{j=1,j\neq i}^{n'} x_j/(x_j-x_i)\prod_{j=1}^{t-n'} y_j/(y_j-x_i)$ mod q and $l_{y_k}(0)=\prod_{j=1}^{n'} x_j/(x_j-y_k)$ $\prod_{j=1,j\neq k}^{t-n'} y_j/(y_j-y_k)$ mod q. □

4.3. Deactivating

To deactivate temporary servers, we make use of a soft state mechanism, i.e. the temporary shares are deactivated after time out. After generating the temporary shares in activating procedure, each server/temporary server starts a timer. When the timer runs out, the temporary share h_k^* or share s_i^* is deleted by each node involved in the activating procedure. The proxy server which launched the activating procedure is also allowed to deactivate temporary servers by informing the servers and temporary servers involved in the activating procedure to delete h_k^* $(k=1,2,\ldots,l')$ and s_i^* $(i=1,2,\ldots,n')$.

4.4. Revoking

If one server detects a malicious node, it will launch the revoking procedure by broadcasting a revocation notification including the evidence. Upon receiving the notification, the servers will put the revoked node into the local blacklists. As a result, the revoked node cannot be activated or converted again. The coordinating server would delete Δ corresponding to the malicious nodes. Since the notification might only reach the servers in one server group, servers should exchange the blacklist periodically. To guard against slandering and tampering attack, the revocation notification should include the evidence and be signed by the server which initializes it.

5. DISCUSSION

5.1. Security

The main security concern of the MCA is the possibility of fabricating mission-specific certificates by attackers. Like other threshold-based DCA approaches, an attacker has to obtain at least t shares in order to compute the private signature key s and hence fabricate a certificate. Thus, the security of the MCA mainly depends on how difficult it is for an attacker to obtain t or more shares. As a partition-tolerant mechanism, it is also necessary to ensure that an attacker will not be able to fabricate certificates simply by collecting auxiliary shares. These security issues are analyzed in this section.

When a node is *compromised*, any resource of the node might be under the control of the attacker, including the public key pair, share and auxiliary share. Furthermore, it is possible for the attacker to compromise more than one node. We thus use N^{c} to denote the set of compromised nodes and N^{s} to denote the set of nodes that are not compromised. Note that the main objective of threshold schemes is to make the system more robust and secure because it is believed to be more difficult to compromise t or more nodes compared with the centralized approach. If there have been t or more compromised servers, the attacker could fabricate the certificates anyway. Thus we will not consider this case in the following discussion.

In the MCA, the attacker might compromise some auxiliary servers besides the compromised servers. However, only the shares/temporary shares can be used to reconstruct the private signature key or fabricate a certificate. Thus, we present the following security properties that are related to the protection of secret shares/temporary shares.

Security Property I: Let C_k denotes the auxiliary server to be converted in the selecting and converting procedure, h_k denotes the auxiliary share to be generated, $f(y_k)$ denotes the corresponding secret share of h_k and V^{c} denotes the set of values generated in the selecting and converting procedure but compromised. The attacker cannot compute $f(y_k)$ from V^{c}, unless $C_k \in N^{\mathrm{c}}, S_{C_k} \in N^{\mathrm{c}}$.

Suppose that less than t servers are compromised, for any $S_i \in N^{\mathrm{s}}$ in the selecting and converting procedure, it is possible for the following values of S_i to be obtained by the attacker:

$$g_i(x), \{g_i(x_1), g_i(x_2), \ldots, g_i(x_{t'})\}, g_i(y_k), s_i' = s_i + \sum_{j=1}^{t'} g_j(x_i) \bmod q$$

where $g_i(x)$ is for local use and should be deleted after being used, $\{g_i(x_1), g_i(x_2), \ldots, g_i(x_{t'})\}$ are sent securely to the corresponding servers, $g_i(y_k)$ is sent securely to S_{C_k}, and $s_i' = s_i + \sum_{j=1}^{t'} g_j(x_i) \bmod q$ is sent securely to C_k.

Under the condition that $S_{C_k} \in N^{\mathrm{s}}$, the attacker cannot obtain t or more values of $g_{C_k}(x)$ to interpolate $g_{C_k}(x)$; thus, $g_{C_k}(x) \notin V^{\mathrm{c}}$, and since $g_{C_k}(x_i)$ will only be sent to S_i securely, the attacker can neither obtain $g_{C_k}(x_i)$ from S_{C_k} directly nor compute $g_{C_k}(x_i)$ by interpolating $g_{C_k}(x)$, i.e. $g_{C_k}(x_i) \notin V^{\mathrm{c}}$. Since there exists $g_{C_k}(x_i) \in \{g_j(x_i)\}(j = 1, 2, \ldots, t')$ that is not available, the attacker cannot compute s_i from $s_i' = s_i + \sum_{j=1}^{t'} g_j(x_i) \bmod q$, i.e. in this case, the values in V^{c} will not help to interpolate $f(x)$. In particular, since $g_{C_k}(y_k)$ is only kept by S_{C_k}, we have $g_{C_k}(y_k) \notin V^{\mathrm{c}}$. Since there exists $g_{C_k}(y_k) \in \{g_j(y_k)\}(j = 1, 2, \ldots, t')$ that is not available, the attacker cannot compute $\Delta_k = \sum_{j=1}^{t'} g_j(y_k) \bmod q$. Thus, $f(y_k) = h_k - \Delta_k \bmod q$ cannot be computed from V^{c}.

Under the condition that $C_k \in N^{\mathrm{s}}$, for any $S_i \in N^{\mathrm{s}}, s_i' \notin V^{\mathrm{c}}$. Suppose that there are less than t servers in N^{s}, we have $h_k \notin V^{\mathrm{c}}$. Note that $g_i(x)$ is just a random polynomial, although it is possible for some $g_i(x)$ to be computed by interpolation, if $s_i' \notin V^{\mathrm{c}}$, $s_i \notin V^{\mathrm{c}}$. In this case, the values in V^{c} will not help to interpolate $f(x)$ as well. If $h_k \notin V^{\mathrm{c}}$, $f(y_k)$ cannot be computed either by the relationship $f(y_k) = h_k - \Delta_k \bmod q$ or by interpolating $f(x)$.

After the selecting and converting protocol, all temporary values for generating h_k are deleted except for Δ_k, which becomes the only trapdoor for computing $f(y_k)$ from h_k. Thus, a compromised auxiliary server will not threaten the secrecy of the corresponding share as long as the coordinating server is not compromised as well.

Security Property II: As a basic property of threshold schemes, for any subset of l shares $\{s_1, s_2, \ldots, s_l\}$ of s, where $l<t$, $\forall k \notin [1, 2, \ldots, l]$, s_k cannot be computed from $\{s_1, s_2, \ldots, s_l\}$.

Security Property III: For any C_k being activated in the activating procedure, let $f(y_k)$ denotes the corresponding secret share, and V^{c} denotes the set of values generated in the activating procedure but compromised. The attacker cannot compute $f(y_k)$ from V^{c}, unless $S_{C_k} \in N^{\mathrm{c}}$.

Suppose C_k's coordinating server $S_{C_k} \in N^{\mathrm{s}}$, it is possible for the attacker to obtain the following values of S_{C_k} by compromising other nodes:

$$\{g^*_{C_k}(x_i) | S_i \in N^{\mathrm{c}}\}, \{g^*_{C_k}(y_j) | S_{C_j} \in N^{\mathrm{c}}\}, \Delta^*_k = \sum_{j=1}^{n'} g^*_j(y_k) - \Delta_k \bmod q.$$

Note that $g^*_{C_k}(x)$ is just a new random polynomial generated independently, the values of $f(x)$ cannot be computed from nothing more than the values of $\{g^*_{C_k}(x_i) | S_i \in N^{\mathrm{c}}\}$ and $\{g^*_{C_k}(y_j) | S_{C_j} \in N^{\mathrm{c}}\}$. The only possible risk is that if the attacker obtains Δ^*_k, he might compute Δ_k by collecting all the values of $\{g^*_j(y_k)\}$ $(j = 1, 2, \ldots, n')$. In this case, the attacker could easily compute $f(y_k)$.

According to the activating procedure, $g^*_{C_k}(y_k)$ is only kept by S_{C_k}. Since $S_{C_k} \in N^{\mathrm{s}}$ and $n' + l' = t$ (as stated in Section 4.2), $|\{g^*_{C_k}(x_i) | S_i \in N^{\mathrm{c}}\}| + |\{g^*_{C_k}(y_j) | S_{C_j} \in N^{\mathrm{c}}\}| < t$. Thus, the attacker can neither obtain $g^*_{C_k}(y_k)$ from S_{C_k} directly nor compute $g^*_{C_k}(y_k)$ by interpolating $g^*_{C_k}(x)$. Since $\exists g_{C_k}(y_k) \in \{g^*_j(y_k)\} (j = 1, 2, \ldots, n')$, which is not available to the attacker, Δ_k cannot be computed. Thus $f(y_k)$ cannot be computed from V^{c}.

After being deactivated, the shares/temporary shares generated in the activating procedure should be deleted. As long as one of these shares/temporary shares is deleted, the remaining less than t ones cannot be used to compute the private signature key s. We also know from Security Property III that with the remaining shares/temporary shares, the attacker cannot compute a long-term valid secret share, i.e., the value of polynomial $f(x)$, for further usage. Since for each execution of the activating procedure, the generated shares/temporary shares are from different polynomials, shares/temporary shares of one execution cannot work with shares/temporary shares of another to compute the private signature key s.

Security properties I and III state that the attacker cannot obtain any information about shares of MCA private signature key from a compromised auxiliary server unless its coordinating server is compromised as well. The property II states that certificate could only be issued with sufficient shares. Thus, the attacker cannot fabricate certificate by compromising auxiliary servers as long as the servers are well protected.

5.2. Partition tolerance

In this section, we illustrate the effectiveness of the proposed scheme through a series of simulation results, in particular, how often the main operation of the MCA (i.e. issuing/updating certificates) fails is measured in scenarios with different radio ranges and number of HECs.

The MCA was implemented as an agent (written in C++) in the ns-2 network simulator [14]. To simulate a highly dynamic situation in which network partitioning occurs frequently, a series of 1 km by 1 km scenario files with 50 nodes were generated based on random way-point model [15].

The random way-point model is one of the most popular mobility models for generating scenario files that depict the mobility of the nodes. In each scenario file, there are a certain number of nodes moving at random speeds from one random way-point to another. A node will wait for a random pause time before starting to move toward next way-point. In our simulation, the maximum node speed is set to 10 m/s and the pause time is set to 10 s.

The generated scenario files were loaded by the simulation script that is written in TCL language. We could adjust most of the parameters in the script. In particular, the $(10, 4)$ threshold scheme was applied in our experiment, i.e. there are ten servers and any four of them could provide certificate service. The radio range of nodes was also set in the script: one option is 125 m and the other one is 150 m. In the scenario where radio range of nodes is set to be 125 m, network partitioning is more likely to occur. In the other scenario, the nodes are better connected.

Another important parameter has to be set is l, which denotes the number of HECs in the system. l was set ranging from 0 to 30 in the simulation script. As explained before, an HEC is selected and converted through the selecting and converting procedure before it becomes an auxiliary server and starts to become operational. For HEC selection, we adopted a specific policy in which the nearest HEC is selected in the selecting and converting procedure.

As for auxiliary shares generation, we consider another important system parameter T_{sc}, which denotes the times of executing the selecting and converting procedure in each scenario. Larger T_{sc} indicates that more auxiliary shares are generated. Note that it is possible for some executions of the procedure to fail since (1) it is possible for one server to select the same HEC (i.e. the nearest one) in two consecutive executions; and (2) the scenarios are partition-prone. In the simulation script, T_{sc} was set to be $5l$, with which about l and $2l$ auxiliary shares were generated for radio range = 125 m scenarios and radio range = 150 m scenarios, respectively.

After a relatively short period of time, there will be a certain number of auxiliary shares held by auxiliary servers. We then started to measure the availability of the MCA. In practice, the HECs and LECs request the MCA for either issuing new certificates or updating certificates. However, we do not strictly differentiate the two kinds of operations in our simulation because their basic procedures are similar.

When one requests for a certificate, the MCA is either available or unavailable. If the MCA is unavailable, the request will fail. Supposing that the MCA servers are always turned on, the reasons that the MCA is unavailable are due to either network congestion or partitioning. By avoiding unrelated heavy overloads, the failure ratio may approximately show the impact of network partitioning on the certificate service. We thus illustrate the effectiveness of our scheme by measuring the failure ratio. To get more reasonable results, the failure ratio we used is the average of failure ratios from 20 different scenarios, as in Figures 8 and 9.

Owing to the random way-point model, nodes are assumed to exhibit 'random' movement in the simulation. Network partitioning occurs as a result of such random movements i.e. when some nodes move to a distance that exceeds the radio range of a partition.

Figure 8 shows the average failure ratio in scenarios where radiorange = 125 m. When $l = 0$, it amounts to the DCA without the partition-tolerant mechanism. The average failure ratio of the requests is about 67.3% in this case. It means that about 67.3% requests fail due to the network partitioning. As l increases, the average failure ratio gradually decreases. In scenarios where $l = 30$, the average failure ratio is reduced to 47.4% i.e. the average failure ratio is reduced by 20%. Figure 9 shows the average failure ratio of scenarios where radio range = 150 m. Since partitioning occurs relatively infrequently in this case, the total failure ratio is much lower than the scenarios

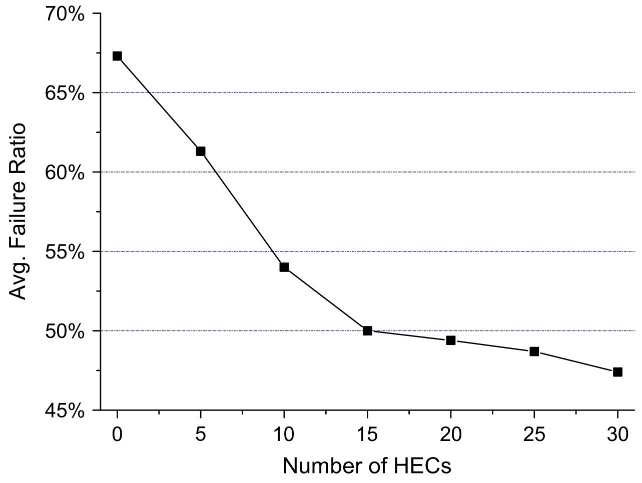

Figure 8. Average failure ratio of requesting DCA certificate in scenarios with radiorange = 125 m.

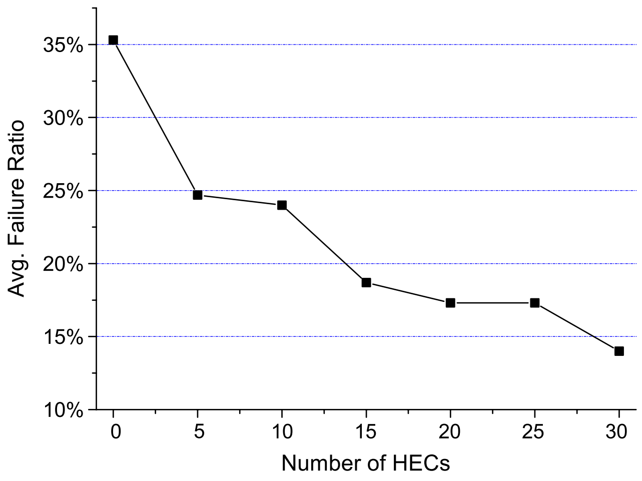

Figure 9. Average failure ratio of requesting DCA certificate in scenarios with radiorange = 150 m.

where radio range = 125 m. As l increases, the average failure ratio is reduced from 35.3% when $l = 0$ to 14% when $l = 30$ i.e. the average failure ratio is reduced by 21%. We conclude from these results that the partition-tolerant mechanism of MCA can effectively alleviate the negative impact of network partitioning on the certificate services.

6. CONCLUSION

In general, applications running in a MANET environment have non-trivial security requirements due to the open nature of wireless links, physical vulnerability of mobile nodes and lack of centralized security administration. As a basis of security services in a network environment, key management is essential for the security protection of MANET applications. In a traditional wired network system, the PKI that mainly consists of a Centralized Certificate Authority and Registration Authority is employed. However, key management schemes for conventional wired networks are not suitable for MANET since a centralized authority cannot be ensured in a typical ad hoc network due to the issue of single point of failure. Addressing the key management issues in MANET has gained attention of the research community in the past few years. Broadly speaking, key management schemes for MANET may be categorized into two groups: peer-to-peer key management [16] that is used for secure communication between any two participants and group key management [17, 18] that is used for secure group communication. In our application scenario, we considered the peer-to-peer key management issue.

In order to meet the security requirements of communication systems for emergency tasks, a threshold-based MCA scheme was proposed in this paper. The MCA issues short-term mission-specific certificates for all participants of the emergency task. Compared with other approaches, it has following advantages:

- High availability/partition-tolerance. The availability of the MCA is reasonably enhanced by introducing auxiliary servers, in particular when network partitioning happens frequently. The performance is shown in the simulation.
- Security. As stated in the analysis section, the auxiliary servers have limited power to participate in issuing certificates. Their abilities are controlled by servers. Besides, only short-term certificates are issued by the MCA and this does not influence the certificate used in each organization.
- Simple pre-configuration/fast deployment. Although the MCA consists of servers and typically more HECs, only servers are needed to be initialized for each mission. This largely expedites the deployment of the communication system.
- Cross-agency. The MCA is designed with cross-agency applications in mind. With internal PKIs, the MCA could effectively establish trust relationship among participants and different organizations, thus providing security support to security services.

These features make the MCA very suitable for supporting communication system of emergency tasks, which are typically carried out by several organizations assembled in an ad hoc manner.

In some of the peer-to-peer key management schemes, such as the Distributed Certificate Authority based approaches proposed by [5–9], there exists a security authority for key management service. As stated in Section 2, the authoritative power of CA is first distributed to servers based on a threshold scheme. A quorum of servers could collaborate to issue certificates. It could efficiently address the problem of single point of failure and achieve relatively high availability and security. However, all these DCA schemes do not consider the availability issue caused by network partitioning, which occurs frequently in highly dynamic MANET. As shown in our experimental results, the DCA schemes without partition-tolerant mechanism (0 HECs) suffer from this problem. Kong *et al.* [10] proposed a scheme that could improve the availability of the DCA by means of 'local servers', i.e. shares are held by the neighbors of requesting node. This fully distributed method actually makes each well-behaving node a server, thus improving the availability and efficiency of the service. The main problem, however, is that it is vulnerable to the Sybil attack [19] (i.e. creating multiple fake identities), by which the attacker could learn the private signature key of DCA by injecting any t fake identities. Furthermore, this approach does not consider the fact that MANET typically consists of heterogeneous nodes, especially with respect to the security protection. By compromising any t weakly protected nodes, an attacker could easily break the system. Budakoglu and Gulliver [20] proposed a multiple threshold scheme with different threshold values at the same time in order to provide key management services in different security levels. When decreasing the threshold value, the availability will increase. It thus provides flexibility of tradeoff between security and availability. The important distinction between this work and our approach is that Budakoglu and Gulliver [20] uses multiple threshold values, DCA private signature keys and sets of shares of private signature key, while our approach needs only one private signature key and utilizes the auxiliary shares to provide higher availability.

The fully self-organized approaches, i.e. certificate chain (CC)-based schemes, were proposed in [21–23]. In the CC scheme, each node could create its own public key pair and issue certificates signed by its own private key to others. The issued certificates will be further sent to other nodes. Nodes and certificates in the system could be modeled as a directed graph, where the vertices represent the public keys of nodes with edges representing the certificates. When one has to verify public key of another node, at least one path between the two nodes has to be found in the graph. The certificates along the path constitute a CC. To use the CC, each certificate along the chain has to be verified. The CC schemes adopt the self-organizing nature of ad hoc networks in which they could be initialized spontaneously. However, it comes with the following disadvantages compared with the DCA schemes:

- The overhead problem—more than one certificate is issued for each node. It incurs heavier communication and storage overheads, which is undesirable for mobile devices.
- The verification efficiency problem—to verify public key via CC, more than one certificate has to been verified, which requires more computational resources.
- The security problem—as the length of the CC increases, the trustworthiness of the public key obtained through the chain might be decreased; hence, the system might become vulnerable to attacks.

There are some hybrid approaches proposed for combining these two schemes together. An approach proposed in [24] shows how the CC scheme could be used in parallel with DCA. The availability is improved since both the DCA and CC could be utilized to verify a public key, but it suffers from the security and overhead problems as the CC schemes. Van der Merwe [16] presented a complete survey of peer-to-peer key management schemes for MANET.

REFERENCES

1. Perkins C. *Ad hoc Networking*. Addison-Wesley: Reading, MA, 2001.
2. Sakarindr P, Ansari N. Security services in group communications over wireless infrastructure, mobile ad hoc, and wireless sensor networks. *IEEE Wireless Communications* 2007; **14**(5):8–20.
3. Lee JS, Chang CC. Secure communications for cluster-based ad hoc networks using node identities. *Journal of Network and Computer Applications* 2007; **30**(4):1377–1396.
4. Rivest RL, Shamir A, Adleman L. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM* 1978; **21**(2):120–126.
5. Zhou LD, Zygmunt JH. Securing ad hoc networks. *IEEE Network* 1999; **13**(6):24–30.

6. Yi S, Kravets R. Moca, Mobile certificate authority for wireless ad hoc networks. *The Second Annual PKI Research Workshop* (*PKI 03*), Gaithersburg, MD, U.S.A., 2003.
7. Wu B, Wu J, Fernandez EB, Ilyas M, Magliveras S. Secure and efficient key management in mobile ad hoc networks. *Journal of Network and Computer Applications* 2007; **30**(3):937–954.
8. Bechler M, Hof HJ, Kraft D, Pahlke F, Wolf L. A cluster-based security architecture for ad hoc networks. *IEEE INFOCOM 2004—Conference on Computer Communications*, Hongkong, China, vols 1–4, 2004; 2393–2403.
9. Luo J, Hubaux JP, Eugster PT. Dictate: distributed certification authority with probabilistic freshness for ad hoc networks. *IEEE Transactions on Dependable and Secure Computing* 2005; **2**(4):311–323.
10. Kong J, Zerfos P, Luo H, Lu S, Zhang L. Providing robust and ubiquitous security support for mobile ad-hoc networks. *IEEE Ninth International Conference on Network Protocols*, Riverside, CA, 2001; 251–260.
11. Shamir A. How to share a secret. *Communications of the ACM* 1979; **22**(11):612–613.
12. Gennaro R, Jarecki S, Krawczyk H, Rabin T. Robust threshold dss signatures. *Advances in Cryptology—Eurocrypt '96*, Saragossa, Spain. Lecture Notes in Computer Science, vol. 1070. Springer: Berlin, 1996; 354–371.
13. Herzberg A, Jarecki S, Krawczyk H, Yung M. Proactive secret sharing or: how to cope with perpetual leakage. *Advances in Cryptology—CRYPTO '95*. Lecture Notes in Computer Science, vol. 963. Springer: Santa Barbara, CA, 1995; 339–352.
14. ns-2 simulator. WWW page. http://nsnam.isi.edu/nsnam/index.php/Main_Page (12 February 2008).
15. Camp T, Boleng J, Davies V. A survey of mobility models for ad hoc network research. *Wireless Communications and Mobile Computing* 2002; **2**(5):483–502.
16. Van der Merwe J, Dawoud D, McDonald S. A survey on peer-to-peer key management for mobile ad hoc networks. *ACM Computing Surverys* 2007; **39**(1):3–45.
17. Rafaeli S, Hutchison D. A survey of key management for secure group communication. *ACM Computing Surveys* (*CSUR*) 2003; **35**(3):309–329.
18. Wu B, Wu J, Dong Y. An efficient group key management scheme for mobile ad hoc networks. *International Journal of Security and Networks* 2009; **4**:125–134.
19. Douceur JR. The sybil attack. *Proceedings of the First International Workshop on Peer-to-Peer Systems* (*IPTPS*), Boston, MA, U.S.A., 2002; 251–260.
20. Budakoglu C, Gulliver T. Hierarchical key management for mobile ad hoc networks. *Vehicular Technology Conference* (*VTC2004*), Los Angeles, U.S.A., 2004.
21. Capkun S, Buttyan L, Hubaux J. Self organized public-key management for mobile ad hoc networks. *IEEE Transactions on Mobile Computing* 2003; **2**(1):52–64.
22. Li RD, Li J, Liu P, Chen HH. On-demand public-key management for mobile ad hoc networks. *Wireless Communications and Mobile Computing* 2006; **6**(3):295–306.
23. Capkun S, Hubaux JP, Buttyan L. Mobility helps peer-to-peer security. *IEEE Transactions on Mobile Computing* 2006; **5**(1):43–51.
24. Yi S, Kravets R. Composite key management for ad hoc networks. *Proceedings of MOBIQUITOUS 2004—First Annual International Conference on Mobile and Ubiquitous Systems*: *Networking and Services*, Boston, MA, U.S.A., 2004; 52–61.

AUTHORS' BIOGRAPHIES

Meng Ge received his BS degree in Computer Science from the Shandong University, China, 2004. He is currently working toward the PhD degree in the key laboratory for Information System Security, Ministry of Education, Tsinghua University, Beijing, China. His research interests include security in mobile ad hoc networks, social networks and security management. He is a student member of the IEICE.

Kwok-Yan Lam is the Director of Key Laboratory for Information System Security, Ministry of Education, PR China. He has been a Professor at the School of Software, Tsinghua University, PR China since 2002. Prior to joining the Tsinghua University, Professor Lam has been a faculty member of the National University of Singapore and the University of London since 1990. His main research interests include distributed systems, information security and tamper-resistant design. Professor Lam was a visiting scientist at the Isaac Newton Institute of the Cambridge University and a visiting professor at the European Institute for Systems Security. He has been a chief security architect for a number of electronic banking and electronic government systems in Singapore and Hong Kong. Professor Lam received his BSc (First Class Honours) from the University of London in 1987 and his PhD from the University of Cambridge in 1990.

Prof. Dieter Gollmann received his Dipl.-Ing. in Engineering Mathematics (1979) and Dr.tech. (1984) from the University of Linz, Austria, where he was a research assistant in the Department of System Science.

He was a Lecturer in Computer Science at Royal Holloway, University of London, and later a scientific assistant at the University of Karlsruhe, Germany, where he was awarded the 'venia legendi' for Computer Science in 1991. He rejoined Royal Holloway in 1990, where he was the first Course Director of the MSc in Information Security. He was a Visiting Professor at the Technical University of Graz in 1991, an Adjunct Professor at the Information Security Research Centre, QUT, Brisbane, in 1995, and has acted as a consultant for HP Laboratories Bristol. He joined Microsoft Research in Cambridge in 1998. In 2003, he took the chair for Security in Distributed Applications at Hamburg University of Technology, Germany. He is a Visiting Professor with the Information Security Group at Royal Holloway, a Visiting Professor with the School of Software at Tsinghua University, Beijing, and an Adjunct Professor at the Technical University of Denmark.

Dieter Gollmann is one of the editors-in-chief of the International Journal of Information Security and an associate editor of the IEEE Security & Privacy Magazine.

Siu-Leung Chung is an Associate Professor at the School of Business and Administration, The Open University of Hong Kong. Prior to joining The Open University, he has been a faculty member of the National University of Singapore and the University of Toledo, Ohio, USA since 1991. His main research interests are information and system security, e-commerce security and software project management modeling. Dr Chung received his BSc (Engineering) from the University of Hong Kong in 1977 and his PhD in Computer Science from the University of Illinois in 1991.

Chin-Chen Chang received his BS degree in applied mathematics in 1977 and the MS degree in computer and decision sciences in 1979, both from the National Tsing Hua University, Hsinchu, Taiwan. He received his PhD in Computer Engineering in 1982 from the National Chiao Tung University, Hsinchu, Taiwan. During the years 1980–1983, he was on the faculty of the Department of Computer Engineering at the National Chiao Tung University. From 1983-1989, he was on the faculty of the Institute of Applied Mathematics, National Chung Hsing University, Taichung, Taiwan. From August 1989 to July 1992, he was the head of, and a professor in, the Institute of Computer Science and Information Engineering at the National Chung Cheng University, Chiayi, Taiwan. From August 1992 to July 1995, he was the dean of the college of Engineering at the same university. From August 1995 to October 1997, he was the provost at the National

Chung Cheng University. From September 1996 to October 1997, Dr Chang was the Acting President at the National Chung Cheng University. From July 1998 to June 2000, he was the director of Advisory Office of the Ministry of Education of the R.O.C. From 2002 to 2005, he was a Chair Professor of National Chung Cheng University. Since February 2005, he has been a Chair Professor of Feng Chia University. In addition, he has served as a consultant to several research institutes and government departments. His current research interests include database design, computer cryptography, image compression and data structures.

Jianbin Li is a Senior Engineer at the Information Centre, State Administration of Taxation, P.R. China. He is a Visiting Professor of the School of Software, Tsinghua University; and a Visiting Researcher of the Software Institute, Academia Sinica. He specializes in design and planning of very large scale networks and information security management. His research interests include information security management, security risk analysis and disaster recovery planning.