Resilient and Robust Key Management for Mobile Ad Hoc Networks

Hua-Yi Lin and Yueh-Min Huang
Information Management Department of Yung Ta Institute of Technology
Department of Engineering Science National Cheng Kung University

Abstract

In contrast with traditional networks, with the characteristics of mobile wireless devices that can dynamically form a network without any infrastructure and wired line, mobile ad hoc networks usually do not provide on-line access to trusted authorities or to centralized servers. Furthermore, they frequently exhibit partition due to link or node failures or node mobility. For these reasons, if we apply traditional PKI (Public Key Infrastructure) security architecture to mobile ad hoc networks, it will appear secure blind sides especially in large-scale ad hoc networks. For this point of view, we propose a new scalable and robust cluster-organized key management scheme and distribution of trust to an aggregation of nodes by taking advantage of threshold scheme faculty to provide mobile ad hoc networks with robust key management. Furthermore, our approach provides CA (Certificate Authority) with fault tolerance mechanism to keep off single point of failure or single point of compromise, and saves CA large repository maintaining members' certificates that make our approach more suitable for many mobile devices. In addition, we enhance the routing performance and non-repudiation and propose a mathematical model to prove our cluster-based communication performance that is better than node-based approach.

Keyword: PKI, CA, Cluster

I. INTRODUCTION

Ad hoc network is a class of wireless networks without fixed infrastructure. Unlike traditional AP (Access Point) based wireless networks, they have no a fixed server to coordinate the activities of mobile hosts. Each node acts as a router transmitting messages from one node to another, contacts nodes out of your transmission range, passes through multi-hops to reach destinations, and routes dynamically because of node movement. These nodes are heterogeneous devices with power and CPU constrained and limited physical security, they also need to perform all other functions involved in any network. Once mobile hosts make the topology changes frequently, the lack of a centralized control server makes it very challenging to incorporate various network layers into ad hoc networks. Owing to the aforementioned characteristics, ad hoc networks are vulnerable and facile attacks, such as eavesdropping, traffic analysis, DoS attacks on routing, and so on. The intruders can successfully partition a network or introduce excessive traffic load and nodes themselves can be compromised, but detection of compromised nodes is difficult since they can generate valid signatures.

So far, we have seen that are many secure networks studies including secure routing, key management, distribution, and so on. However, they are not suitable for ad hoc networks because they usually rely on a CA (Central Authority) that is the most important component of PKI and vouches for the validity of digital certificates. That is to say, the success of PKI depends on the security and availability of the CA since a principal must be able to correspond with the CA to get a certificate, check the status of another principal's certificate, acquire another principal's certificate, and so on.

Since the PKI has been deployed for wired networks [3] and some infrastructure-based wireless networks, therefore good connectivity can be assumed in these networks so the trusty research in such environment has focused on the security and scalability of the CA to handle a large number of requests. If we still rely on traditional cryptographic primitives, the secure mobile ad hoc networks will become very challenging.

It follows from what has been said that wire-based key management system is not fit for ad hoc networks, the specific key management schemes have to develop to adapt to the characteristics of mobile ad hoc networks. In this paper, we propose a new Cluster-Organized key management scheme. Our network model is based on clustering models in mobile ad hoc networks, and over which we propose a new mechanism to perform authentication. Our work aims at providing a secure, scalable, and distributed

authentication services.

II. KEY MANAGEMENT SCHEME FOR AD HOC NETWORKS

Cluster is a kind of resilient architecture, and ad hoc networks put it to good use. Actually, when the cluster extends its diameter for very large scale, it is similar to general ad hoc networks infrastructure and only exists one CA. Based on the one CA infrastructure in ad hoc networks exists the single failure point issue and limited power operational analysis constrain that are many hazards and unreliable factors. On the contrary, the diameter is zero that every node plays the CA role. In another word, the self-organized mobile ad hoc networks [3] is this special case. Furthermore, the self-organized has the follow drawbacks. First, this infrastructure is not scalable and finding the certificates paths between two nodes successful rate is lower especially in very large ad hoc networks. Secondly, it's prone to be attacked and hard to detect the malicious nodes. Thirdly, the flooding overhead is too heavy. However, based on the cluster infrastructure, it can solve aforementioned issues and supply the more strong and secure environment. The next few certificate authority schemes presented in this paper attempt to intelligently distribute the certificate authority functionality among mobile nodes. These very similar solutions take advantage of Shamir's secret sharing cryptographic technique introduced earlier. Under this technique, the encryption key is divided into *n* parts, and distributed among *n* cluster heads. Only acquiring *k* such pieces can the key be reconstructed. This scheme is termed 'Threshold Cryptography'. We will now evaluate a few schemes based on the technique and achieve the robust CA and strengthen our secure ability.

III. CBKM (CLUSTER-BASED KEY MANAGEMENT)

In our system, the certificates are stored and distributed by the cluster heads in a cluster-based manner. This architecture is devised to minimize the flooding of authority packets and promote the scalable capability. It is most suitable for large networks with numerous nodes. The entire network is divided into a number of overlapped, disjointed 2 or above hop-diameter clusters. Seriously the node with high secure level and stability is elected as a cluster head, and it has to maintain the cluster membership information. The identification of a cluster is by cluster head ID. Each node in the network knows its cluster head(s), therefore knows which cluster(s) it belongs to, and regards itself as in cluster ID *x* if it has bi-directional link to the head of cluster *x*. All the nodes broadcast hello messages periodically. The messages also contain information about the neighbor nodes, adjacent clusters, and certificate repositories, which are useful for maintaining synchronization of the cluster membership.

We propose the resilient hierarchical authentication scheme that is optimal for such architectures, and we introduce weight factors to the cluster head election algorithm that will be described in Section 6.

Our system scheme is two levels cluster-based architecture and consists of three types of entities:

A. Root cluster head (*Rch*): The root cluster head is the parent CA (Certificate Authority). It is responsible to combine the partial certificates into valid one and is the external cluster domains communication trust center.

B. Cluster head (*Ch*): The cluster head is the child CA. It is responsible to administrate authority, initialize itself cluster domain, store certificates in repository, and generate related key in response to normal node's requests.

C. Normal node: A client node in a cluster domain, which queries cluster head for certificates.

In brief, the *Rch* is the role of parent CA and the *Ch* is the child CA. *Rch* is the same as root CA in PKI. The generation of the *Ch* and *Rch* is described in Session 6. The scenario of the key management hierarchical model is as figure 1, and management scheme is as follows:

A. First step: *Rch* generates a key pair (*PK, SK*) in public key cryptograph. *SK* is the private key named system-secret-key and is used to generate the public key certificates of *Ch*s. *PK* is the system-public-key which verifies the authentication of cluster heads public key certificates. *PK* is distributed to every *Ch*s in the system and *SK* is protected by the Shamir's secret sharing scheme.

B. Second step: The cluster head *i* (*i=1,2,3,..,n*) generates a key pair (*Pchi, Schi*), and then sends the public key *Pchi* to *Rch*. When *Rch* receives *Pchi*, it generates the certificate Cert(*Pchi*) of *Pchi* using the system- secret-key *SK* and then sends Cert(*Pchi*) back to cluster head *i*. Once cluster head *i* receives this certificate Cert(*Pchi*), the Cert(*Pchi*) is verified by cluster head *i* using *PK* to make sure this certificate is

from correct *Rch*.

C. Third step: After *Rch* generates *n* certificates for *n* cluster heads, *Rch* divides *SK* into *n* shadows (sharing keys) $\{Sh_1, Sh_2, \ldots, Sh_n\}$ using Shamir's secret sharing, and encrypts Sh_i by the public key *Pchi* of cluster head *i*, and then sends Sh_i encrypted to cluster head *i*, for each $i = 1, 2, 3, \ldots n$. As soon as *Rch* knows that each cluster head receives his own correct public key certificate from it, *Rch* discards the system-secret-key *SK* in to order to provide system with more robust security.

These above steps are the initial phases of our system. Since cluster heads or normal nodes possibly join and depart from our system, a root cluster head may be cracked or replaced. The remainder in our key management scheme will describe how does the key management handle when a normal node joins or leaves a cluster domain, or a cluster head collapses for unanticipated situations.

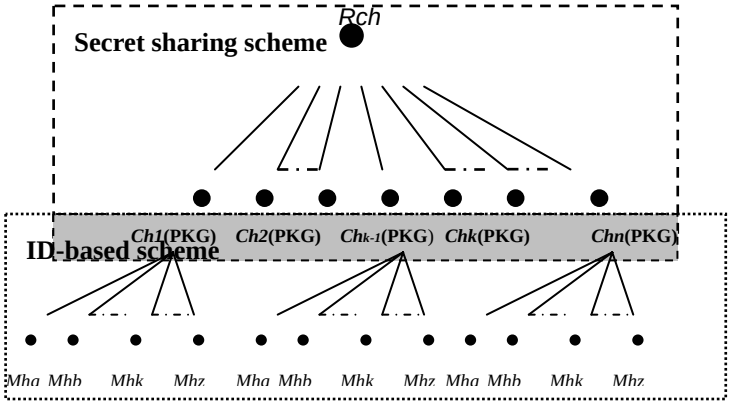


Figure 1. Cluster-Organized key management hierarchical model

IV. CBKM INTERNAL AUTHORITY FRAMEWORK

In every cluster domain, we introduce ID-based scheme. The cluster head acts as the authority for cluster members. When a node joins the networks, it is given belonged a cluster domain master-public-key. Besides, each node also needs a personal-private-key; the personal-private-key applies from its cluster domain head, and uses it to achieve the capability of encryption.

- **Initial phase:**

In initial phase, every cluster domain members elect the cluster head *Ch* as the domain CA in figure 2 by election algorithm. The CA will issue related keys according to client nodes applying for.

- **Joining/Leaving domain phase:**

When a node joins a cluster domain for the first time. The cluster head (by hello messages) detects new node will start on cluster head election algorithm for the remaining nodes not yet selected as a cluster head or assigned to a cluster domain. When a node leaves old cluster and joins another new cluster domain. The new cluster head treats it as a new node joining its cluster domain, and the old cluster head purges the entity of this node when it doesn't receive hello message for a certain predefined time interval. To prevent malicious nodes joining cluster domains, a mutual authentication procedure is performed between the moved node and a cluster head. If passing through mutual authentication, the cluster head then gives the joining node this cluster domain master-public- key.

- **Internal cluster members communication:**

When node *Mha* wants to communicate with node *Mhb* as figure 3, we employ cryptographic schemes to protect both data traffic and routing information. But uses of above schemes usually require a key management service, the key mechanism that we employ ID-based encryption scheme. For the advantage of ID-based system over public key certificates is no longer necessary a public key or certificates

store and without distributing it. It is implicitly certified from an identity (and therefore do not have to store public keys). This possibly results in a saving of space requirements. Similarly, ID-based secret key mechanisms avoid cluster head to maintain a large database containing the secret keys. This has the further advantage to offer a higher security level [10].

In ID-based encryption scheme consists of four algorithms as follows:

Setup:*Ch* takes as input a security parameter and outputs a cluster domain master-public/secret- key. The master-public-key will be publicly known while only the *Ch* (*PKG:* Private Key Generation service) knows the master-secret-key.
Extract:*Ch* takes as input the master-secret-key and *Mh* ID (an arbitrary string) and returns the personal-private-key S_{ID} corresponding to the *Mh* ID.
Encrypt: Sender takes as input the master-public-key, the *Mh* ID of the recipient, and a plain message and outputs a corresponding ciphertext.
Decrypt: Receiver takes as input the master-public-key, a personal-private-key S_{ID} and a ciphertext, and then outputs the corresponding plain message.

This algorithm setup is executed by *Ch* (PKG). The *Ch* also runs the algorithm to extract at the request of a node that wishes to obtain its personal-private-key.

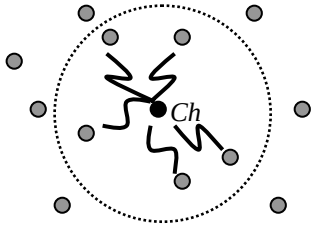


Figure 2. A cluster head in a domain

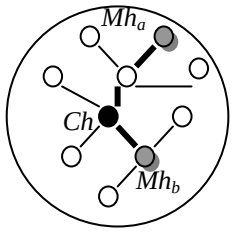


Figure 3. Internal members communication

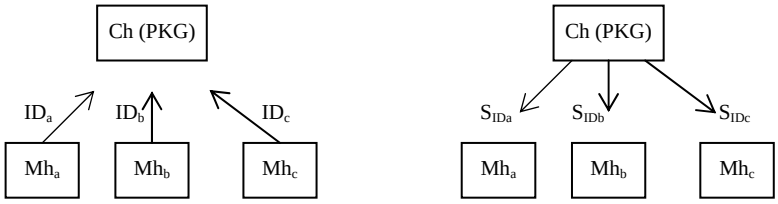


Figure 4. Personal-private-key requested in ID-based encryption

When Mh_i applies for a personal-private-key from *Ch*, the *Ch* computes the Mh_i's personal-private-key, and denotes S_{IDi} as follows:

$$S_{IDi}=H_k\,(ID_i, h_i^*),$$

Where $H_k(\ )$ is a keyed one-way function under the master-secret-key *k*. Examples of keyed one-way function includes DES, HMAC. The resulting key S_{IDi} is called personal-private-key of entity Mh_i, and argument h_i is starred to indicate that it is optional.

In setup phase, cluster head *Ch* players the PKG (private key generation) roles and generates the master-public/secret-keys for this cluster domain system. Hence Mh_a communicates with Mh_b, it gets personal-private-key from *Ch*. The *Ch* takes as input the master-secret-key and Mh_a's identity (such as MAC address which with non repudiation attribute and is useful for detection and isolation of compromised nodes) and returns to Mh_a's personal-private-key S_{IDa} as figure 4. After that, Mh_a takes as input the master-public-key, the identity of the recipient Mh_b, plain message and sends a ciphertext to Mh_b. Once Mh_b gets this ciphertext, it takes as input the master-public-key, the ciphertext, a personal-private-key and than returns the plain message. The Boneh-Franklin ID-based scheme seems quit suitable for message encryption. It is based on elliptic-curve cryptography, which gives savings in computation and communication [10].

To keep off the malicious nodes routing attacks, here we make use of node's ID and

personal-private-key as the key pair and adopt ARAN [2] cryptographic certificates to offer routing security. It consists of a preliminary certification process followed by a routing process that guarantees end-to-end routing authentication. Once setting up the internal secure routing path, source node applies the ID-based scheme to the message encryption.

V. CBKM EXTERNAL AUTHORITY FRAMEWORK

When Mh_a and Mh_b locate different cluster domains, we introduce external authority architecture.

- **Initial Phase:**

Here we assume that the whole ad hoc networks system depends on secure cluster election algorithm to elect the top stratum root *Ch* as *RCh*. In a secret sharing design, the system key pair is denoted as (*PK, SK*). Where *PK* is the system-public-key, *SK* is the system-secret-key, and *Rch* generates them. Once any *Ch*s (Cluster heads) pass through gateway nodes to join/leave system, the *Rch* will regenerate new system key pair (*PK, SK*) according to the Session 3 scenario. The newer *PK* is well know to all *Ch*s and *SK* is divided into n shadows (sharing keys) $\{Sh_1, Sh_2, \ldots, Sh_n\}$ by Shamir's secret sharing. Sh_i encrypted by the public key *Pchi* of cluster head *i*, then *Rch* sends Sh_i encrypted to cluster head *i*, and the receiver cluster head *i* decrypts by Sch_i. In Shamir's threshold secret sharing, *SK* can be shared by an arbitrary large community using a secret polynomial *f(x)*. If the degree of *f(x)* is *k-1*, then any *k* members of the community can reconstruct the secret key via Lagrange Interpolation, while any less than *k* members of the community reveal no information of the secret key. This is normally denoted as *k*-threshold secret sharing. In our scenario, *RCh* generates the secret key $SK = < S_d , n >$ and randomly selects a polynomial *f(x)* of degree *k-1*, $f(x) = S_d + f_1.x + \ldots\ldots + f_{k-1}.x^{k-1}$, the shared secret is $f(0) = S_d$. Each cluster head Ch_i ($i = 1,2,\ldots,n$) holds a shadow (sharing key) $Sh_i = (f(Ch_i) \bmod n)$. Any construction of *k* entities $\{Ch_1, Ch_2, Ch_3, \ldots\ldots, Ch_k\}$,the Lagrange Interpolation states that

$$S_d \equiv \sum_{i=1}^{k} (Sh_i . l_{Chi}(0) \bmod n) \equiv \sum_{i=1}^{k} SK_i \,(\bmod n) \ .$$

Where the $l_{Chi}(0)$ is the Lagrange Coefficients, each share holder Ch_i can compute an SK_j from its shadow (secret share) Sh_j by Lagrange Interpolation, and the *SK* will be recovered from the sum

$$S_d = (\sum_{i=1}^{k} SK_i \bmod n)$$

To defend the polynomial secret sharing and against the adversary could break into or compromise *k* or more cluster heads in enough time. We have to refresh periodical secret sharing updates with different polynomials by proactive secret sharing mechanism [12]. We construct another polynomial f_{k+1} from f_k, $f_{k+1} = f_k + g_k$, g_k is a random (*k-1*) polynomial, and the new secret share is $f_{k+1}(x_i) = f_k(x_i) + g_k(x_i)$ that can reconstruct the S_d. If the shadow (sharing key) is expiry date, considered suspect or compromised, cluster head will store it in its revocation list with a convicted accusation factor and forward the information to other cluster heads, who store it in revocation list with a suspect accusation and decrease the secure weight. When a cluster head *Ch* acquires *k* accusations, the suspect becomes convicted. Based on this model, we provide robust *RCh* fault tolerance. Once the *RCh* collapses or any *k* cluster heads convict it of malicious behaviors, by way of *RCh* election algorithm re-elects the new *RCh* and then cooperates with cluster heads to reconstruct the system secret-key *SK* from the *k* shadows (sharing keys) $\{Sh_1, Sh_2, \ldots, Sh_n\}$, and then regains normality.

- **Communication phase:**

When Mh_a and Mh_b locate different cluster domains, they feel like to communicate each other as figure 5. It will accompany with cryptographic certificates to achieve routing security. We propose CSBRP (cluster secure based routing protocol), which integrates CBRP (cluster-based routing protocol) with security, certification and signature process. The CSBRP is accomplished by CSRREQ message from a source Ch_s that is replied to CSRREP message by the destination Ch_x. Such routing messages are authenticated at each cluster head from source to destination, as well as on the reverse path from the destination to the source. CSBRP requires the use of *RCh* as a certificate server which with the repositories of all *Ch*s' key pair (*Pchi, Schi*) and whose public key is know to all *Ch*s. Our scenario introduces cluster-to-cluster authentication mechanism and the cluster head verifies the routing factuality. Details of authentication processes are explained below.

Our routing packets are protected by RSA digital signature algorithm approach for it can achieve encryption, signature and messages authentication. A source cluster head Ch_s in initial phase has got its certificate Cert(*Pchi*) from

Rch and we simplify Cert(*Pchi*) as $Cert_{chi}$. Source cluster Ch_s routes to destination Ch_x by broadcasting CSRREQ packets to its neighbor cluster heads. In order to prevent broadcast storm, the broadcasting messages only deliver inside a cluster ambit not flood over outside clusters.

Ch_s → Broadcast: $Cert_{chs}$, $X_s = Ek_{Schs}[T, PT_{csrrep}, IP_{chx}, Seq_{chs}, E]$

The CSRREQ packet includes a packet type identifier ("PT_{csrreq}"), destination cluster head IP address IP_{chx}, a time stamp *T* of when the packet was created, a time *E* at which the CSRREQ packet expires, and a packet sequence number, that all signed with Ch_s's private key S_{chs}. Table 1 summaries the notations. When a cluster head receives a CSRREQ packet, it sets up a reverse path back to the source by recording the neighbor cluster head from which it receives the CSRREQ. This is an anticipation of eventually receiving a reply message that will need to forward back to the source cluster head. The receiving node uses Ch_s's public key, which it extracts from Ch_s's certificate $Cert_{chs}$ to validate signature in $Cert_{chs}$ whether it is correct. Further using this public key verifies the CSRREQ packet to ensure that has not been tampered with. The receiving cluster head also check the (Seq_{chs}, IP_{chx}) tuple to verify the CSRREQ packet that has not already processed. The signature prevents spoofing attacks that may alter the route message. Let Ch_a be a neighbor cluster head that has received from Ch_s the CSRREQ broadcast, which it subsequently rebroadcasts.

Ch_a → Broadcast: $Cert_{cha}$, $X_a = Ek_{Schs}[Cert_{chs}, X_s]$

Upon receiving the CSRREQ, Ch_a's neighbor cluster head Ch_b validates the signature with the given certificate $Cert_{cha}$. Ch_b then discards Ch_a's certificate, decrypts X_a by Ch_a's public key, verifies this packet, and records Ch_a as its predecessor. After that, Ch_b uses self private key S_{chb} to encrypt the contents of the CSRREQ packet originally broadcast from Ch_s, appends its own certificate $Cert_{chb}$, and then forward rebroadcasts the CSRREQ packet to neighbor cluster heads.

Ch_b → Broadcast: $Cert_{chb}$, $X_b = Ek_{Schb}[Cert_{chs}, X_s]$

Each cluster head along the path repeats these steps of validating the previous node's public key certificate, discarding the previous node's certificate, decrypting the routing packet, recording the previous cluster head's IP, encrypting the original contents of the messages, appending its own certificate, and forward broadcasting the message. After receiving the CSRREQ packet, there are many loose source routing paths that pass through distinct cluster heads. The Ch_x figures out the result of strict source routing path by hop count or QoS factors and so on. The destination Ch_x unicasts a CSRREP packet back along the reverse strict source routing path to the source cluster head Ch_s. Let the first node that receives the CSRREP packet sent by Ch_x be cluster head Ch_c.

Ch_x → Ch_c: $Cert_{chx}$, $R_x = Ek_{Schx}[T, PT_{csrrep}, IP_{chs}, Seq_{chx}, E]$

The CSRREP packet includes a packet type identifier ("PT_{csrrep}"), the IP address of source cluster head Ch_s, current time *T*, the CSRREP packet expired time *E*, and packet sequence number, that all signed with Ch_x's private key S_{chx}. Each cluster head along the reverse strict source routing path unicasts back to the predecessor, encrypts the CSRREP packet, and appends its own certificate before forwarding the CSRREP packet to next cluster head. Let Ch_b be the next cluster head that receives Ch_c's CSRREP packet.

Ch_c → Ch_b: $Cert_{chc}$, $R_c = Ek_{Schc}[Cert_{chx}, R_x]$

Ch_b validates Ch_c's public key certificate on the received message, discards the certificate $Cert_{chc}$, then decrypts the contents of the packet R_c, appends its own certificate $Cert_{chb}$, and encrypts with self private key S_{chc} before unicasting the CSRREP packet to Ch_a.

Ch_b → Ch_a: $Cert_{chb}$, $R_b = Ek_{Schb}[Cert_{chx}, R_x]$

Each head along this strict routing path checks the CSRREP packet, validates public key certificate of the previous cluster head, then decrypts the encrypted CSRREP packet sent from previous head, and encrypts $Cert_{chx}$ of R_x by self private key, as the CSRREP packet returns back to the source. This avoids attacks that malicious nodes instantiate routes by impersonation and replay Ch_x's message. When the source cluster head Ch_s receives the

CSRREP packet, it verifies the public key certificate $Cert_{chx}$ returned by the destination.

Once cluster heads accomplish the secure routing mission to decide the strict source routing path. The source cluster head Ch_s starts to convey encrypted messages according to next head's public key. The receiver cluster head decrypts the cipher message and verifies the messages by its own private key to keep from the messages have been tampered with.

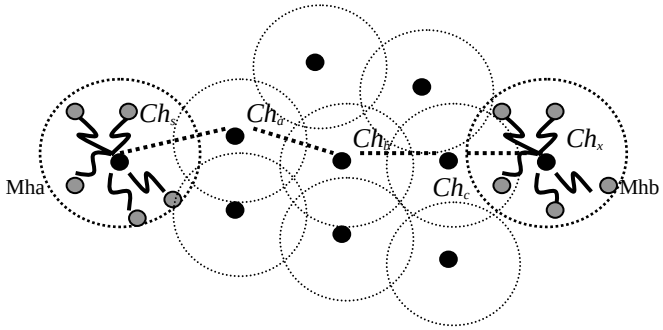


Figure 5. Different cluster domains communication

Table 1. Variables and notations

Ch_i	Cluster head i.
Pch_i	Public key of Ch_i.
Sch_i	Private key of Ch_i..
$Ek_{Schi}[d]$	Data d digitally signed by Ch_i's private key according to RSA algorithm.
$Cert_{chi}$	Certificate of Ch_i.
T	Timestamp of packet.
E	Packet expiration time (TTL).
IP_{chi}	IP address of Ch_i.
X_i, R_i	The digital signature result of Ch_i.
CSRREQ	Cluster-based secure route request packet identifier.
CSRREP	Cluster-based secure route reply packet identifier.
PT_{csreq}	Packet type of CSRREQ.
PT_{csrep}	Packet type of CSRREP.

VI. SECURE CLUSTER HEAD ELECTION

In our architecture, the cluster head is the most important role. To decide how well suited a node is for being a cluster head. Here we deduce Mainak Chatterjee's WCA (Weighted Clustering Algorithm for Mobile Ad Hoc Networks) [6]. Additional, we take secure weight factors into account and the others are degrees, transmission power, mobility, and battery power. We give descriptions as below:

Finding the neighbors of each node *v* within its transmission range, which defines its degree *dv* as

$$d_v = |N(v)| = \sum_{v' \in V, v' \neq v} \{dist(v, v') < tx_{range}\} .$$

The degree-difference, $\Delta_v = |d_v - \delta|$,for every node *v*, each cluster head can ideally support only δ(a defined threshold) nodes to ensure efficient medium access control (MAC) functioning. For every node *v*, computes the sum of distances $D_v$with all neighbors as

$$D_v = \sum_{v' \in N(v)} \{dist(v, v')\} .$$

The running average of the speed for every node till current time is *T*. This gives a measure of mobility and is denoted by M_v as

$$M_v = \frac{1}{T} \sum_{t=1}^{T} \sqrt{(X_t - X_{t-1})^2 + (Y_t - Y_{t-1})^2} .$$

Where (X_t,Y_t) and (X_{t-1},Y_{t-1}) are the coordinates of the node v at time t and time $(t-1)$, respectively. The cumulative time P_v, during which a node v acts as a cluster head. P_v implies how much battery power has been consumed which is assumed more for a cluster head than an ordinary node. To consider secure factors, we introduce a value of direct trust relationship as the trust class between two nodes in the same cluster domain. We apply the formula for calculation and combination of different trust values from the direct trust and the recommendation trust approach in [11]. It is a result of the computation of the direct trust values and is used for drawing a consistent conclusion when there are several derived trust relationships of the same trust class between two entities. We give a description as

$$V_{exam} = 1 - \prod_{i=1}^{m} \sqrt[n_i]{\prod_{j=1}^{n_i} (1 - V_{i,j})} \ .$$

Where $V_{ij} \neq 0$ (i=1…m,j=1…n), V_{exam} is the values of direct trust relationships. Additional, we define the $S_v = V_{sus}$ x V_{exam}, and we take account of aforementioned "suspect" issue as evaluated factor V_{sus}. Finally, calculating the combined weight W_v for each node v as follows:

$$W_v = w_1 \triangle_v + w_2 D_v + w_3 M_v + w_4 P_v + w_5 S_v.$$

Where $W_1, W_2, W_3, W_4,$ and W_5 are the weighted factors for corresponding system parameters. We choose the node with smallest W_v as the cluster head and all neighbors of the chosen cluster head are no longer allowed to participate in the election procedure. This mechanism is for the remaining nodes not yet selected as a cluster head or assigned to a cluster.

VII. ANALYSIS

On the basis of our model, we deduced the formula as follows:

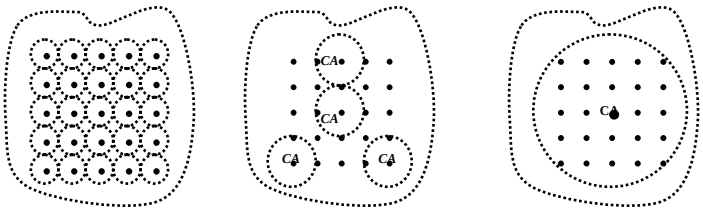
$$Num\_of\_system\_CA = \lim_{r \to n} \frac{m}{\alpha \bullet \gamma} .$$

Where r represents the hop count scope of a cluster head, m represents the total mobile nodes, and α is a ratio argument. These possible cases are as follows:

(1) n = 0, In our system model, every node is the cluster head who plays the role of certificate authentication (CA) as figure 6(a). It implies that the Srdjan Capkun's Self-Organized key management framework [3] is the special case.

(2) $n \in$ integer, our system model exists many cluster heads, and every head plays the role of CA that is responsible for its own service cluster domain range as figure 6(b).

(3) $n \rightarrow \infty$, the system model is reduced to one cluster head playing the role of CA. More precisely, it's a central certificate authentication architectural as figure 6(c).



(a) n = 0 (b) n = 1 (c) n → ∞

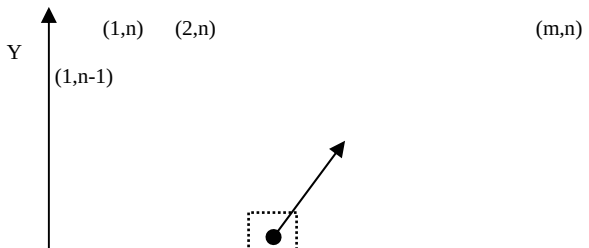
Figure 6. CA service range

VIII. PERFORMANCE EVALUATION AND SIMULATION

In performance evaluation conditions, we make some assumptions for this protocol designed, and describe as follows:

(1). Communication hop counts evaluation

In our model, we assume that ad hoc networks have m x n mobile nodes. These mobile nodes allocate on the intersections as figure 7. We feel like to compute the min-hop-count for any two nodes in this model. This subject is immensely complex, and we define some terminologies as follows to simplify.



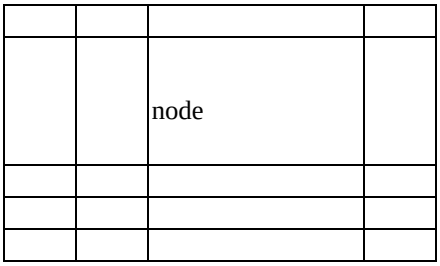


Figure 7. The mobile ad hoc networks model

N_{ab}: represents a mobile node allocated on coordinate (a, b).
$Min_{hop}(N_{ab}, N_{cd})$: represents the minimal hop count between node N_{ab} and N_{cd}.
$AVMin_{hop}$: represents the average minimal node-hop-count for any two nodes in this model.
$AVCBMin_{hop}$: represents the average minimal cluster-hop-count for any two clusters.
A: is a set denoting $\{1,2,\ldots,m\}$ or A=$\{1,2,\ldots m\}$.
B: is a set denoting $\{1,2\ldots,n\}$ or B=$\{1,2,\ldots n\}$.
$V = \sum_{a,c\in A;b\in B} Min_{hop}(N_{ab}, N_{cb})$.
$H = \sum_{b,c\in B;a\in A} Min_{hop}(Nab, Nac)$.
$R = \sum_{a,c\in A;b,d\in B} Min_{hop}(N_{ab}, N_{cd})$, $a \neq c$ and $b \neq d$.
B_{ab}: represents the number of a x b grid in the model, where a and $b \neq 1$.

In general, V represents the sum of Min_{hop} between two nodes parallel *Y-axis*. H represents the sum of Min_{hop} between two nodes parallel *X-axis*. B_{ab} represents the number of grids a x b in m x n model. R is the minimum hop count, Min_{hop}, sum of all diagonal line pair-nodes belonged in a x b grid. From aforementioned terminologies, we obtain the equations as follows:

$$V = n \bullet \sum_{i=1}^{m-1} (m-i)i = \frac{n(m-1)m(m+1)}{6} \quad \text{----------(1)}$$

$$H = m \bullet \sum_{i=1}^{n-1} (n-i)i = \frac{m(n-1)n(n+1)}{6} \quad \text{----------(2)}$$

$$B_{ab} = (m-a+1)(n-b+1) \quad \text{----------(3)}$$

$$R = 2 \bullet \sum_{b=2}^{n} \sum_{a=2}^{m} B_{ab} \bullet (a+b-2) = \frac{n(n-1)m(m-1)(n+m+2)}{6} \quad \text{-----(4)}$$

From equations (1)~(4), we can figure out the $AVMin_{hop}$ will be (m+n) / 3 as follows:

$$AVMin_{hop} = (V+H+R) / C_2^{nm}$$

$$= \left(\frac{nm(m^2-1)}{6} + \frac{nm(n^2-1)}{6} + \frac{n(n-1)m(m-1)(n+m+2)}{6} \right) \Big/ C_2^{nm}$$

$$= \frac{nm(n+m)(nm-1)}{6} \Big/ \frac{nm(nm-1)}{2} = \frac{m+n}{3} \quad \text{----------(5)}$$

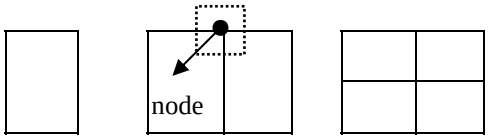
We give some instances to prove the equation (5) is exact. The nodes distributed instances are described in m x n model as figure 8.

A. m=n=2,there are 4 nodes, any two nodes average min. node-hop-count $AVMin_{hop}$ = $\frac{2 \bullet 2 \bullet 1 + 2 \bullet 2}{C_2^4} = \frac{8}{6} = \frac{4}{3} = \frac{2+2}{3}$

B. m=3,n=2,there are 6 nodes, any two nodes average min. node-hop-count $AVMin_{hop}$ = $\frac{2 \bullet (2 \bullet 1 + 1 \bullet 2) + 3 \bullet 1 + 2 \bullet 2 \bullet 2 + 2 \bullet 3}{C_2^6} = \frac{25}{15} = \frac{5}{3} = \frac{2+3}{3}$

C. m=n=3,there are 9 nodes, any two nodes average min. node-hop-count $AVMin_{hop}$ = $\frac{2\bullet(3\bullet 2\bullet 1+3\bullet 2)+4(2\bullet 2)+4(2\bullet 3)+2\bullet 4}{C_2^9}=\frac{72}{36}=\frac{6}{3}=\frac{3+3}{3}$

By the same way, in *4 x 4* model (m=n=4), the result is 8/3. It means the (m+n) / 3 is the generalized equation.



m=n=2 m=3,n=2 m=n=3

Figure 8. *m x n* model

Applying equation (5), we deduce the cluster distribution model and also compute the $AVCBMin_{hop}$ for any two clusters. We assume the cluster model is denoted as *a x b* model and satisfies three conditions:

1. The number of each cluster domain is the same.
2. Every cluster domain has $a \bullet b$ nodes; these nodes lie on *a x b* grid, every intersection allocates a node only.
3. The gateways between two clusters locate on the boundary lines.

In order to more manifest, we use an example to explain. Let ad hoc networks are *7 x 5* model, cluster domain is *4 x 2* model, each cluster allocates 8 nodes, and the double bold lines represent one cluster domain as the figure 9. According to above model, the system divides into 8 cluster domains. We realize that the cluster Ch_1 consists of node set = $\{N_{ij} | i=1,2,3,4.\ j=1,2\}$, and each cluster Ch_i (i=1,2,3…,8) has 8 nodes. The gateway nodes between cluster Ch_1 and Ch_3 are the node set=$\{N_{12},N_{22},N_{32},N_{42}\}$, that is the boundary between two rectangles. In our *m x n* networks model and *a x b* cluster model, where (*a-1*) divides (*m-1*) and (*b-1*) divides (*n-1*). Since we can imagine that each cluster can be regarded as a node, which represents a cluster head, and then the set of all clusters can be show as a $\left(\frac{m-1}{a-1}\right)\times\left(\frac{n-1}{b-1}\right)$ network model and to compute the $AVCBMin_{hop}$ is equivalent to compute $AVMin_{hop}$. We derive from equation (5) that the $AVCBMin_{hop}$ = $\frac{\left(\frac{m-1}{a-1}+\frac{n-1}{b-1}\right)}{3}$. If in *31 x 21* networks model and *4 x 3* cluster model are adopted. We derive from equation (5) that

$AVMin_{hop}=\frac{m+n}{3}=\frac{31+21}{3}=17.333$, and average minimal cluster-hop-count for any two clusters $AVCBMin_{hop}$ is equivalent to $\frac{\left(\frac{m-1}{a-1}+\frac{n-1}{b-1}\right)}{3}=\frac{\left(\frac{31-1}{4-1}+\frac{21-1}{3-1}\right)}{3}=\frac{10+10}{3}=\frac{20}{3}=6.666.$ The ratio of the communicative cost between any two cluster heads and any two nodes is $\frac{AVMin_{hop}}{AVCBMin_{hop}}=\frac{17.333}{6.666}$.

Generalizing this equation under *m x n* networks model and *a x b* cluster model is equivalent to

$$\frac{AVMin_{hop}}{AVCBMin_{hops}}=\frac{(m+n)/3}{\left(\frac{(m-1)}{(a-1)}+\frac{(n-1)}{(b-1)}\right)/3}=\frac{(m+n)}{\left(\frac{(m-1)}{(a-1)}+\frac{(n-1)}{(b-1)}\right)} \quad \text{-----(6)}$$

Where a and $b \neq 1$. In general case the equation (6) is greater than 1. This result implies that the cluster-based average minimum cluster-hop-count is below normal-based minimum node-hop-count. In another word, the cluster-based model is better then normal-based model.

If we take account of the additional communicative costs of normal-based $AVMin_{hop}$ and cluster-based $AVCBMin_{hop}$. We assume that communicative costs of normal-based neighbor two nodes is λ, and then the costs of cluster-based neighbor two clusters will be $2\bullet\lambda$, because of two clusters pass through a gateway node to communicate each other. Then we generalize equation (6) that will be equivalent to $\frac{(m+n)}{\left(\frac{(m-1)}{(a-1)}+\frac{(n-1)}{(b-1)}\right)\bullet 2}$. From our example model, we can realize the result is

$\frac{AVMin_{hop}}{AVCBMin_{hop}}=\frac{17.333}{2\times 6.666}=\frac{17.333}{13.332}$, cluster-based is still better than normal-based. Where we have to control the

λ argument to reach better communication performance.

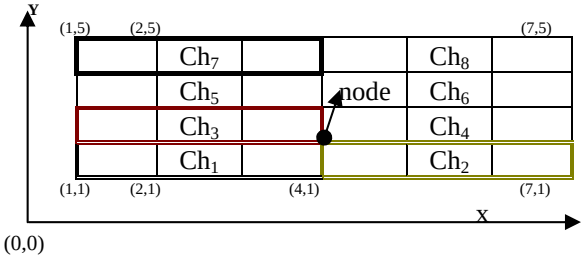


Figure 9. *7 x 5* ad hoc networks, *4 x 2* cluster domain

(2). Communication performance evaluation

Our goal is to show the CSBRP's packet delivery with reduced overhead, and evaluate how CSBRP scales to larger networks, and compare CSBRP with other ad hoc routing protocols (with/without local repair). We make use of NS2 (network simulator) with wireless extension, and the simulation results as follows:

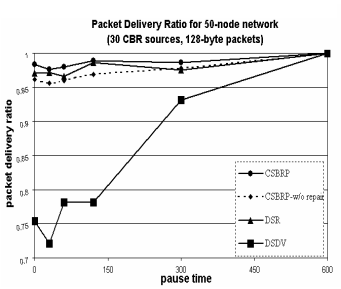


Figure 10. Packet delivery ratio with respect to network mobility

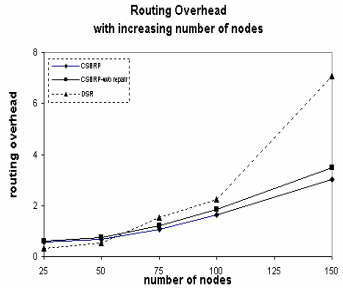


Figure 11. Routing overhead with respect to network size, the routing overhead = Num. of routing packets sent/Num. of data packets deli

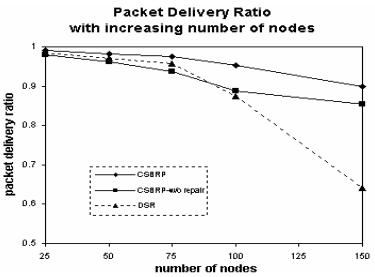


Figure 12. Packet delivery ratio with respect to network size

IX. Conclusion And Further Work

In this paper, we have presented the cluster-organized key management framework for mobile ad hoc networks. The approach provides a resilient way that distributes the key to threshold cluster head to keep off single failure point CA. Also, by defining a generic and flexible framework of describing self-organized and MOCA structure, the approach becomes very feasible for scalable ad hoc networks. Moreover, we introduce the cluster secure concept into CBRP that effectively provides authentication and non-repudiation securing routing in the managed-open environment. The future work for the study may include the exploration of mechanisms for the cluster head secure election algorithm, more precise on secure route performance comparison, and the improvement of simulation models.

References

[1]Hua-Yi Lin, Yueh-Min Huang," Information Service on Scalable Ad-Hoc Mobile Wireless Networks", in IEEE ICCNMC'03 Shanghai, China. p. 190.

[2]Kimaya Sanzgiri, Bridget Dahill, Brian N. Levine, and Elizabeth M. Belding-Royer," A secure routing protocol for ad hoc networks", In International Conference on Networks Protocol (ICNP), Paris, France, November.

[3]Srdjan Capkun, Levente Buttyan and Jean-Pierre Hubaux,"Self-Organized Public-Key Management for Mobile Ad Hoc Networks", IEEE Transactions On Mobile Computing, Vol 2, No.1, January-March 2003.

[4]Jean-Pierre Hubaux, L. Buttyan and S. Capkun, "The Quest for Security in Mobile Ad Hoc Networks", ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC), Long Beach, CA, USA, October 2001.

[5]Yvo Desmedt and Yair Frankel,"Threshold cryptosystems", Advances in Cryptology - Crypto '89, Proceedings, Lecture Notes in Computer Science 435 (G. Brassard, Ed.), Springer-Verlag, 1990, pp. 307-315.

[6]Mainak Chatterjee, Sajal K. Das, Damla Turgut, "WCA: A Weighted Clustering Algorithm for Mobile Ad Hoc

Networks", Cluster Computing 5, 193–204, 2002.
[7]T. Beth, B. Malte, and K. Birgit, "Valuation of Trust Computer Security", Sprintger – Verlag, New York, pp. 3-18, 1994.
[8]Elizabeth M. Royer, CK Toh, "A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks", IEEE Personal Communications, Apr.
[9]L. Zhou and Z. Haas, "Securing Ad Hoc Networks", IEEE Network Magazine, 13(6), November/December 1999.
[10]Marc Joye and Sung-Ming Yen, "ID-based Secret-Key Cryptography", ACM Operating Systems Review 32(4):33-39, 1998.
[11]Thomas Beth, Malte Borcherding, Birgit Klein,"Valuation of trust in open networks", In Proceedings of the Third European Symposium on Research in Computer Security - ESORICS 94, pages 3--18, Brighton, United Kingdom, November 1994.
[12]DR.Stinson, R.Wei,"Unconditionally Secure Proactive secret Sharing Scheme with Combinatorial", SAC'99, Springer Verlag LNCS 1758, 200-214.
[13]Seung Yi, Robin Kravets, "MOCA: Mobile Certificate Authority for Wireless Ad Hoc", 2nd Annual PKI Research Workshop Program (PKI 03), Gaithersburg, Maryland, April, 2003.