Fully Distributed Self Certified Key Management for Large-Scale MANETs

Zahra Moradlu
Computer Engineering Department
Shahed University
Tehran, Iran
Email:z.moradlou@shahed.ac.ir

Mohammad Ali Doostari
Computer Engineering Department
Shahed University
Tehran, Iran
Email:doostari@shahed.ac.ir

Mohammed Gharib
Computer Engineering Department
Sharif University of Technology
Tehran, Iran
Email: gharib@ce.sharif.edu

Ali Movaghar
Computer Engineering Department
Sharif University of Technology
Tehran, Iran
Email: movaghar@sharif.ir

***Abstract*—Mobile ad hoc networks (MANETs) have received considerable attention while their special characteristics make them vulnerable against different attacks. Providing security in such networks becomes a challenge and cryptography is an essential solution for it; to implement a cryptosystem, key management is the main challenge. In this paper a new deterministic scheme for key management in MANETs is proposed. Since public and private keys in this algorithm have an unforgeable relationship, there is no need for any certificate. Proposed scheme distributes the role of the key generation center (KGC) among all nodes, therefore the private key is issued by distributed KGCs (DKGCs) and the node itself. Furthermore, each pair of nodes can share a symmetric key in a non-interactive way while communicating with each other. The proposed scheme is certificate-less, does not require any trusted authority and also it can solve the key escrow problem. Moreover, the performance of the proposed algorithm is analyzed analytically and it is compared with previous works.**

***Index Terms*—Cryptography, Large-scale MANETs, Key Management, Self Certified Public Key.**

I. Introduction

MANETs are emerging as an important area in the field of ubiquitous networking. MANET consists of resource-limited mobile devices that are connect to each other through wireless links and multi-hop routing without any centralized infrastructure. Because of such characteristics, deploying security mechanisms is difficult in such environments.

Cryptography is a commonly used technique for providing security which needs a key management scheme while securely and efficiently distributing and providing keys for all nodes. Traditional networks use public key infrastructure (PKI) to manage public keys. In this approach certificate authority (CA) is responsible for issuing certificates for public keys. The need for infrastructure and CA is the main drawback of PKI to be used in MANETs. Either many works have been done about distributed CAs, not both the communication overhead and latency make them unsuitable for MANETs.

Another key management approach is called identity based key management. The concept of identity based systems first was introduced by Shamir in 1984 [1]. In this approach each nodes' public key is generated from its ID and because of the uniqueness of this ID, there is no need to have certificates. Instead each node has to received its private key from private key generation (PKG) center. The first functional identity based encryption scheme was introduced by Boneh and Franklin [2] using bilinear pairing. This approach has three important disadvantages. First, PKG is assumed trusted and knows the private keys of all nodes. Second, the unique ID for each node causes inability to update keys in the case of compromise. Third a secure channel is needed for transferring private keys.

Self certified public keys is another idea for key management introduced by Girault [3] and extended by Peterson and Horster [4]. In this scheme that is a combination of PKI and identity based key managements, each public key is generated based on node's identity in addition to a random number that acts as a witness for public key. Corresponding private key is also generated by cooperation of KGC and node itself. Every node should broadcast its own witness value for all others so that they can compute each node's public key. Each node can verify public keys implicitly and during the usage of keys. There is some disadvantages in this approach that make it inefficient for implementation. It is not scalable, it needs a reliable broadcasting protocol and large storage space for storing witness values, and it is opposed with freely moving and mobility of MANET nodes.

In this paper we propose a new key management algorithm that uses the self certified public keys concept and has solved its problems. In the proposed algorithm, an initiator defines system parameters and master public-private key pair. It also helps starting nodes to generate their public-private key pair. Starting nodes are the first coming nodes which are going to form the network. Then the initiator distributes the master private key between all nodes. After that, the initiator leaves the network and nodes start to communicate through the network. The nodes can establish symmetric keys between themselves in a non interactive manner, or they can use asymmetric cryptography that is introduced in this paper.

Every node joining the network gets its private key and also its share of master private key from specific number of neighbor nodes which are equal to threshold value (t) defined in secret sharing. Since each node has public-private key pair, the share can be transfered securely. Furthermore, it is resilient against compromising $t-1$ nodes due to applying threshold cryptography.

Publishing of witness values is also suggested to be done on demand instead of broadcasting, and can be transfered in the routing packets. On demand publishing decreases the network communication overhead and nodes receive the new values of witnesses every time they join the network. Also there is no need to store the witnesses values.

The proposed algorithm is analyzed in order to evaluate its performance. It is also compared with other deterministic key management algorithms while the results show that the proposed algorithm outperforms previous algorithms.

The rest of the paper is organized as following. Section 2 reviews the related works. Section 3 contains some preliminaries. The novel algorithm is proposed in section 4. Protocol analysis and results are presented in section 5. Finally, section 6 concludes the paper.

II. Related Works

This section reviews relevant prior works for key management schemes. We categorize public key management schemes into two categories: *certificate based systems* and *certificate-less systems*.

In certificate based systems, public keys are verified using certificates. Some schemes distribute the role of CA among nodes. When a certificate is being issued or revoked, threshold number of nodes participate to do the operation. Another schemes allow users to create, store, distribute and revoke their own public keys without any trusted authority. These schemes are called certificate chaining systems.

MOCA is a distributed certificate authority (DCA) based system [5] in which nodes that are more secure and powerful act as DCAs. Moreover, β-Unicast is introduced in which a node unicast its request to multiple nodes. Another scheme is DICTATE [6] in which there is a mother CA (mCA) that issues initial certificates for nodes and distributes its role between some server nodes (dCA). When the network is disconnected from the mCA, nodes can communicate with dCAs for query or update requests. Also Ge and Lam proposed a self initialized distributed certificate authority [7] in which specified number of nodes negotiate on the fundamental parameters including total number of DCA members, threshold value and list of DCA nodes at the beginning of the network.

Lack of efficiency, availability and security is the problem of partially DCAs. Because of the mobility of nodes and the limited number of DCAs, a requesting node may be some hop away from the DCAs. Furthermore, DCA nodes are known and they can be the bottleneck for the network. Also because every node for issuing, renewing and revoking its key should communicate frequently with t nodes, communication overhead and latency is high. Finally DCAs should store the issued certificates and exchange their information with each other in order to establish revocation list.

The first certificate chaining system was introduced by Capkun et al. [8]. Every node creates its own public-private key pair. If node A relies on the authenticity public key of node B, it issues a certificate that binds public key of node B to its identity. Public keys can be authenticated using a chain of valid certificates. Another scheme is introduced in [9] in which the chain is discovered under routing infrastructure instead of exchanging certificates. Despite fully self organized nature of these systems, they have some problems. First, forming trust relationship takes a long time and requires user interactions. Second, the transitivity of trust cause weak authentication. As the length of the certificate chain increases, the trustworthiness will decrease. Third, each node verifies a chain of certificates instead of one, which affect on efficiency.

In certificate-less systems, the certificate of public key is embedded in the public key itself and there is no need for CA and public key certificates. Identity based systems and self certified public key based systems are two groups of certificate-less systems.

Khalili et al. combined the idea of threshold cryptography and ID based cryptosystems to form the threshold PKG and used it in ad hoc networks [10]. Authors of [11] proposed a novel scheme in which public and private keys consist of two elements: node specific and network wide common, to provide the update ability for nodes. In [12] a partially distributed PKG was proposed to solve the key escrow problem. Chan proposed another identity based system [13] that uses the verifiable secret sharing for this distribution in order to establish initial trust between nodes.

These schemes have some problems that remain as a concern. They suggest two solutions to solve the problems of updating keys and key escrowing: distributing of PKG and adding another element to public key. However they need a secure channel for publishing secret shares and obtaining private keys. Also they are not scalable for ad hoc networks. Furthermore, compromising of threshold number of nodes causes the compromising of private keys. Moreover, for adding another element, schemes use an updating phase factor that just solves the updating problem. As a result and based on our knowledge it seems that there is not any suitable identity based approach introduced for ad hoc networks.

A lot of works have been done on self certified public keys; Lee and Kim suggested an explicit authentication for solving the denial of decryption attack [14]. In their scheme an identity publishes a signed witness value. [15] and [16] respectively introduced encryption and signature schemes based on self certified keys that are more lightweight in comparison with other schemes. Li et. al proposed a non interactive key agreement protocol [17]. In this scheme, keys are updated periodically according to time synchronization, or by agreement of communicating parts. Also [18] and [19] use self certified keys for key management.

Disadvantages of these key management systems can be summerized as follows: first, these schemes assume an off-line KGC that pre-distributes all the keys before the network formation. In some applications of MANET, specially in large scale networks, the number of nodes can not be specified before the network formation. So they are not scalable. Second, they need a reliable broadcasting protocol which guarantees all nodes will receive the witness value even in the case of link failures and mobility of the nodes. Also in MANETs, nodes can freely leave or join the network, so if the witness value is broadcasted when some nodes leave the network, they would not access the value after joining again. Finally if each node stores all the witness values, it would need more storage space.

III. Preliminaries

In this section some preliminary basics are described that are related to our proposed algorithm.

A. Secure key issuing protocol

This key issuing protocol is introduced in [4]. There is a centralized CA that generates system parameters and issues key pair for all nodes before network formation. Mentioned CA chooses two large prime numbers p and q with $q|p-1$, a generator g that is a member of multiplicative group of $\mathbb{Z}_p^*$ with order q and a collision free hash function H. Then CA chooses a random number $x_z \in \mathbb{Z}_q^*$ as its private key and computes its public key $y_z = g^{x_z}(modp)$ that is known to all other nodes.

Nodes' private keys are calculated interactively by CA and the nodes themselves. In the first step, CA chooses a random number $k_i \in \mathbb{Z}_q^*$ and computes $\tilde{r}_i = g^{k_i}(modp)$ for node i. Then node i chooses its own secret $a_i \in \mathbb{Z}_q^*$ and computes $r_i = \tilde{r}_i \cdot g^{a_i}(modp)$. Moreover, CA computes the signature parameter for node i based on r_i and its identity: $\tilde{x}_i = H(ID_i, r_i) \cdot x_z + k_i$; in fact this is the partial private key of the node i. Now, it can obtain its private key $x_i = \tilde{x}_i + a_i(modq)$. Node i sends its identity and random number r_i to other nodes so they can calculate its public key. The corresponding public key is $y_i = y_z^{H(ID_i,r_i)} \cdot r_i(modp)$.

B. Secret sharing

Secret sharing is a technique introduced by Shammir [20] for dividing a secret between some nodes so that a threshold number of nodes can reconstruct the secret. Given t points in the 2-dimensional plane $(x_1, y_1), ..., (x_t, y_t)$, with distinct x_i's, there is one and only one polynomial $f(x)$ of degree $t-1$ such that $f(x_i) = y_i$ for all i.

To distribute secret value X among n nodes, a trusted dealer chooses a large prime number q, and constructs a polynomial $f(x)$ of degree $t-1$ in a way that its coefficients has been chosen from $\mathbb{Z}_q^*$ and $f(0) = X$. Each node's share is equal to $s_i = f(ID_i)$ and the dealer transfers it securely to the node. Then, any group of t nodes can reconstruct the value of X using their shares as $\sum_{i=0}^{t} \lambda_i(0) s_i$ in which the $\lambda_i(0)$ is a Lagrange coefficient in the point 0 and the Lagrange coefficient is calculated as below

TABLE I
Table of Notations

p, q	Two large prime number
$\mathbb{Z}_p^*$	Multiplicative group of p
g	Generator of order q
(x_M, y_M)	Master private and public key
r'_i	random number of node i
ID_i	Identity of node i
h	Collision free hash function
(x_i, y_i)	Private and public key of node i
$f_M(x)$	Master polynomial with degree of $t-1$
s_i	Share of node i
$K_{i,j}$	Pairwise key between node i and j
$r'_{i,DKGC_j}$	Random number generated for node i by $DKGC_j$
$x'_{i,DKGC_j}$	Partial private key of node i generated by $DKGC_j$
$\lambda_i(x)$	Lagrange coefficient
δ	Random shuffling factor
m	Message
c	Cipher text of arbitrary message

$$\lambda_i(x) = \prod_{j=1, j\neq i}^{t} \frac{x - ID_j}{ID_i - ID_j} \quad (1)$$

C. Discrete logarithm problem

Let $\mathbb{Z}_q^*$ be a multiplicative group of q with generator g. Given $a \in \mathbb{Z}_q^*$ and q, finding an integer b such that $a = g^b(modq)$ is hard. In fact there is no polynomial time algorithm to solve this problem.

IV. Proposed Algorithm

In this section, a novel key management algorithm is proposed. The algorithm consists of four separate phase: network initialization, network formation, new node joining and, update and revocation. The network assumptions are described before proposing the algorithm's phases.

A mobile ad hoc network with n nodes is considered without any predefined trust relationship between its nodes. Nodes can move freely in the network, thus they can leave and join it again. There is an off-line initiator in the initialization phase that is responsible for initializing the network. Each node has a unique ID and also all nodes can discover the neighbor nodes that are in their own communication range. Also nodes themselves can detect whether their keys are compromised or not and when they need to update keys. Moreover, we consider that each time, even in the existence of adversary nodes, there are at least t honest nodes in the network that can provide KGC service. Table I specifies important notations used in the proposed algorithm.

A. Network initialization

The initiator chooses two large prime numbers p, q, with $q|p-1$. It also chooses generator g of $\mathbb{Z}_p^*$ with order q and a collision free hash function h. Then the initiator generates master private-public key pair: (x_M, y_M) in a way that $y_M = g^{x_M}(modp)$ and $x_M \in \mathbb{Z}_q^*$. Key pairs for starting nodes are generated as below:

1) The initiator chooses a random number $k_i \in \mathbb{Z}_q^*$ for node i. Then it calculates r'_i for node i such that $r'_i = g^{k_i}(modp)$.
2) The initiator calculates the partial private key of node i: $x'_i = x_M.h(ID_i) + k_i(modq)$.
3) Then the initiator sends r'_i and x'_i to node i.
4) Node i chooses a random number $a_i \in \mathbb{Z}_q^*$, and calculates its private key using the partial private key and its own secret: $x_i = x'_i + a_i(modq)$.
5) Node i calculates the witness value $r_i = r'_i.g^{a_i}(modp)$.
6) Other nodes can calculate node i' public key using its identity and witness value: $y_i = {y_M}^{h(ID_i)}.r_i(modp)$.
7) Node i can certify its public-private key pair by: ${y_M}^{h(ID_i)}.r_i(modp) = g^{x_i}(modp)$.

Also initiator generates a random polynomial $f_M(x) = \sum_{i=0}^{t-1} a_i x^i$ in which $f_M(0) = x_M$. The coefficients of this polynomial are chosen from $\mathbb{Z}_q^*$. According to secret sharing and using this polynomial, the master private key is distributed among starting nodes. Share of node i is equal to $s_i = f_M(ID_i)$.

B. Network formation

The initiator leaves the network and the network starts to work. Every two nodes that want to communicate, send the witness value to the other in the routing packets. For explicit authentication, nodes also send the signed witness. Each node calculates the public key and then verifies the signature. If the verification is successful, node i is authenticated. Any two nodes can establish a pairwise key and communicate securely. Pairwise key $K_{i,j}$ is established between nodes i and j as below:

$$K_{i,j} = h({y_j}^{x_i}(modp)) = h((g^{x_j})^{x_i}(modp)) = h((g^{x_i})^{x_j}(modp)) = h({y_i}^{x_j}(modp)) = K_{j,i} \quad (2)$$

C. New node joining

When a new node i joins the network, it should contact t nodes to do the joining operation. New node sends its request to t neighbor nodes. As new nodes can calculate and certify the neighbor nodes' public keys, it creates a session key and sends the encrypted session key to the neighbors. As a result the neighbors return the encrypted partial keys to the new node. The proposed public key encryption is described in the following sections. All the nodes can provide the KGC service for new nodes. So the nodes are called DKGCs. Each of the neighbor nodes chooses a random number from $\mathbb{Z}_q^*$ and creates $r'_{i,DKGC_j} = g^{k'_{i,DKGC_j}}(modp)$ for $j = 1, 2, ..., t$. Also based on the value of $k'_{i,DKGC_j}$, each of them creates a partial private key $x'_{i,DKGC_j} = s_{DKGC_j}.\lambda_{DKGC_j}(0).h(ID_i) + k'_{i,DKGC_j}(modq)$ in which $\lambda_{DKGC_j}(x)$ is the Lagrange multiplier of the node j in the point (x) and it is computed by equation 1.

The new node chooses a random number $a_i \in \mathbb{Z}_q^*$, then it extract its own private key:

$$x_i = a_i + \sum_{j=1}^{t} x'_{i,DKGC_j}(modq) \quad (3)$$

The witness value that node sends for others, is obtained from t values of $r'_{i,DKGC_j}$ and the random number chosen by node itself. It is equal to:

$$r_i = g^{a_i}.\prod_{j=1}^{t} r'_{i,DKGC_j}(modp) \quad (4)$$

The following equation proves that the private key obtained from partial private keys is equal to the private key obtained from the master private key.

$$\begin{aligned} x_i = a_i + \sum_{j=1}^{t} x'_{i,DKGC_j}(modq) = a_i + \sum_{j=1}^{t}(s_{DKGC_j}. \\ \lambda_{DKGC_j}(0).h(ID_i) + k'_{i,DKGC_j})(modq) = \\ \sum_{j=1}^{t} s_{DKGC_j}.\lambda_{DKGC_j}(0) + \sum_{j=1}^{t} k'_{i,DKGC_j}(modq) = \\ a_i + h(ID_i).x_M + \sum_{j=1}^{t} k'_{i,DKGC_j}(modq) \end{aligned} \quad (5)$$

Algorithm IV.1: INITIALIZATION & FORMATION (p, q, g, h)

```
$x_M \leftarrow$ MATERPRIVATEKEYGENERATION$(q)$;
$y_M \leftarrow$ MASTERPUBLICKEYGENERATION$(p, g, x_M)$;
$f_M \leftarrow$ MASTERPOLYNOMIALGENERATOR$(q)$;
for $i \leftarrow 0$ to $n$
  do
$k_i \leftarrow$ INITIATORSECRETGENERATOR$(q)$
$r'_i \leftarrow$ PARTIALRANDOMNUMBER$(p, g, k_i)$;
$a_i \leftarrow$ NODESECRETGENERATOR$(q)$;
$r_i \leftarrow$ NODERANDOMNUMBERGENERATOR$(p, g, a_i, r'_i)$;
$x_i \leftarrow$ NODEPRIVATEKEYGENERATOR$(x_M, ID_i, k_i, a_i)$;
$y_i \leftarrow$ NODEPUBLICKEYGENERATOR$(y_M, ID_i, r_i)$;
$s_i \leftarrow$ SHAREGENERATOR$(f_M(x), ID_i)$;

if two nodes want to communicate with each other
  {$y_i \leftarrow$ PUBLICKEYCALCULATION$(y_M, r_i)$;
  {$PairwiseKey \leftarrow$ PAIRWISEKEYESTABLISHMENT$(x_i, y_j)$;
  {return $(PairwiseKey)$
  else no operation is needed
```

It is also easy to prove that the collection of random numbers chosen by DKGCs and used in the calculation of partial private keys, is related to the published witness value.

$$r_i = g^{a_i} . \prod_{j=1}^{t} r'_{i,DKGC_j} (modp)$$
$$g^{a_i + \sum_{j=1}^{t} k'_{i,DKGC_j}} (modp) \quad (6)$$

Moreover, a new node needs a share of the master private key. Thus, all t nodes that help the new node to calculate its own private key, send a sub-share of master private key to the new node. Partial shares should be transferred securely, otherwise the adversary node can obtain the master private key by eavesdropping. t nodes encrypt the partial shares using symmetric key in order to securely transfer. Any sub-share is equal to the product of Lagrange multiplier at the ID of new node and the share of the node. Since any node could calculate Lagrange multiplier, so new node can calculate the share of the nodes from its sub-shares. Therefore, in the proposed algorithm, random shuffling is suggested to solve this issue.

$$s_i = s_{DKGC_j} . \lambda_{DKGC_j}(ID_i) \pm \delta \quad (7)$$

The parameter δ is the random shuffle factor which is agreed with both nodes. One of these two nodes adds it to the partial share and the other one subtracts it from partial share that is generated for the new node.

D. update and revoke

In the proposed scheme nodes themselves decide about updating their own keys to prevent compromising. Every node can update its keys without need to contact DKGCs. Clearly, update operation for each node would be done just by generating new random number r_i. Considering $x_{i,0}$ be the first private key which is obtained from master private key, each node generates new random witness value number and use equation 8 to update its public key and 9 to update its private key. Since nodes themselves update their keys, there is no need to revoke operation. This causes the communication and computation overhead decrease.

$$x_{i,t} = x_{i,0} . h(ID_i) + k_{i,t} (modq) \quad (8)$$

$$y_{i,t} = {y_{i,0}}^{h(ID_i)} . r_{i,t} (modp) \quad (9)$$

To provide more security, it is considered that nodes do not use their first keys for cryptographic applications. This makes nodes' first keys would never being compromised and the new keys would not being revealed. Every node keep the first key got from DKGCs as a secret and provides new keys based on this first key. Also it sends its first public key and the first signed witness value in addition to new signed witness value in order to allow other nodes authenticate its new keys.

Algorithm IV.2: NEW NODE JOINING (ID_{new})

```
for $j \leftarrow 0$ to $t$
  do
    $k'_{new,DKGC_j} \leftarrow$ DKGC-SECRETGENERATOR$(q)$;
    $x'_{new,DKGC_j} \leftarrow$ DKGC-PARTIALPRIVATEKEY(
    $p, g, ID_{new}, k'_{new,DKGC_j}$)
    $s'_i new, DKGC_j \leftarrow$ DKGC-PARTIALSHARE$(s_j, ID_{new})$;
if new node receive t partial private key and share
    $a_i \leftarrow$ NODESECRETGENERATOR$(q)$;
    $x_i \leftarrow$ NODEPRIVATEKEYGENERATOR$(x'_{new,DKGC_j}, a_i, q)$
    $s_i \leftarrow$ NODESHAREGENERATOR$(s'_i)$;
else new node should try to collect t response
```

E. Proposed Encryption scheme

To encrypt a message with public keys produced using the proposed algorithm, every node can do the following operation:

$$c = m^{y_i} (modp) \quad (10)$$

in which y_i is the public key of the destination node, and m is the plain message. Thus, encrypted message can be decrypted as bellow, in which the security of the operation is guaranteed based on discrete logarithm problem:

$$m = c^{g^{-X_i}} (modp) \quad (11)$$

V. PROTOCOL ANALYSIS

In this section, some security metrics of the proposed algorithm are discussed and then it is compared with IMKM [21] and B-BLS [22] to investigate the performance of the proposed algorithm.

A. Security Metrics

We argue some of most important requirements of a secure key management scheme that our algorithm can meet.

- **confidentiality.** In the proposed algorithm, partial private keys and shares are transformed securely.
- **Resistance.** Proposed algorithm is secure under the compromise of $t-1$ nodes. Even in the case of compromising t nodes, the adversary could not be able to discover the private keys of nodes.
- **Key escrow free.** In the proposed algorithm, each private key consists of two parts; one part is generated by DKGCs and another by nodes themselves.
- **Explicit key authentication.** Signing the value of witness and publishing it leads other nodes to be able to authenticate the public key before using it.
- **Completeness.** Any joining node can communicate with t neighbor nodes and receive partial private keys and shares in a polynomial time.

- **backward and forward secrecy.** Because the first key pair have never been compromised and new keys are independent, knowing one of the keys can not help the adversary to obtain other keys.
- **Efficiency.** There is no need to store witness values. Nodes only contact with others on demand. New nodes just need to contact with t neighbor nodes and symmetric keys are generated in non interactive manner.
- **scalability.** The role of KGC is distributed among all nodes. Thus, every node can act as a DKGC and any joining node can obtain its own keys and shares easily from its neighbors. If the number of neighbors is less than threshold, the node can move in area and find more neighbors.

B. *Performance analysis*

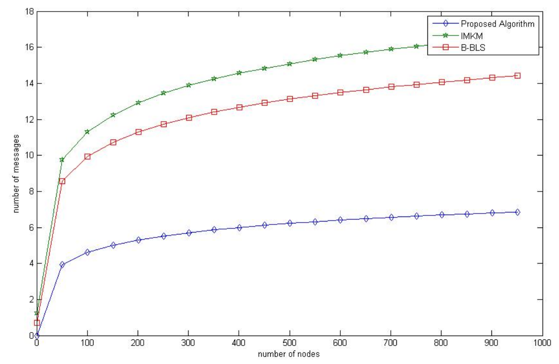
The comparison of algorithms consists of two aspects: communicational overhead and computational costs.

Three different key management phases including network formation, new node joining and key update are considered for communicational overhead comparison. Then the computational cost for these phases are compared. Two phases of network initialization and network formation of proposed algorithm are considered as one phase of network initialization to compare better with other two algorithms. Table II represents the time complexity of both compared algorithms in addition to our novel proposed algorithm. As it is clear from the table, the proposed algorithm outperforms both IMKM and B-BLS algorithms. It is noteworthy that network initialization in B-BLS can be either centralized or decentralized. In the case of centralized, a trusted dealer initiates the network without any communication overhead. Nevertheless in the decentralized manner, the communication overhead is $O(n^2)$.

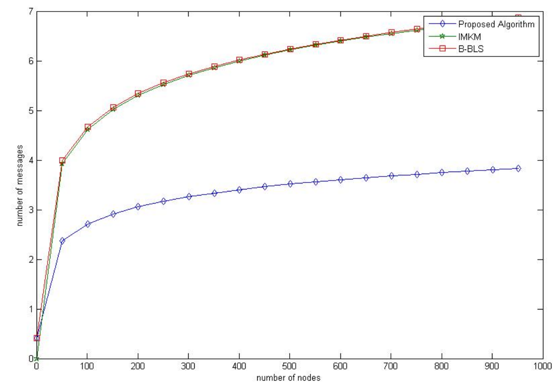
The exact number of packets flow in the network are calculated for different phases and then the communicational overhead in different network sizes are presented. Fig. 1 (a) is the communicational overhead in formation phase. In the proposed algorithm, nodes exchange witness values and number of messages is equal to n. Both the IMKM and B-BLS have broadcasting in this phase. The exact number of messages for these two algorithms are respectively equal to: $3n^2 + n^2t$ and $2n^2$. 1 (b) is the communicational overhead when a new node joins the network. In the proposed algorithm, new node only receives its keys and shares from one hop neighbors and only $3t$ messages is being sent and received. In IMKM, all of the nodes generate partial share for new node and number of messages is equal to n. In B-BLS, new node broadcasts its request and threshold number of nodes respond to this request. Number of messages in these cases is equal to $n + t$. Finallay Fig. 1 (c) is the communicational overhead for updating keys. In the proposed algorithm, nodes just send new witness value for other nodes they want to communicate. Number of messages is equal to n. In IMKM, nodes should broadcast new keys for others that need n^2 messages, and B-BLS does not have any update scheme. As a result, the proposed scheme has less traffic than others in larger networks.

TABLE II
COMMUNICATIONAL OVERHEAD

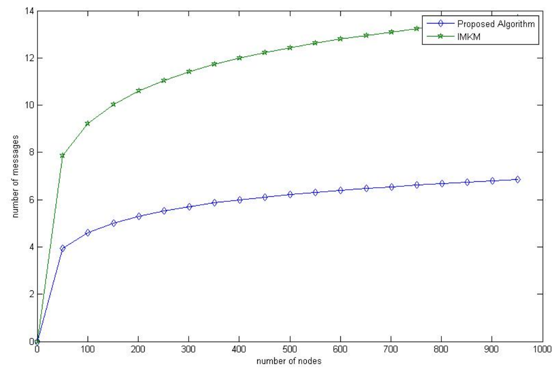
Phases	IMKM	B-BLS	Proposed Algorithm
network initialization	$O(n^2)$	$O(n^2)$	$O(n)$
new node joining	$O(n)$	$O(n)$	$O(t)$
key update	$O(n^2)$	$No - Update$	$O(n)$



(a)



(b)



(c)

Fig. 1. Number of messages flow in the network for different sizes and separate phases: (a) Network formation, (b) New node joining, and (c) Update.

TABLE III
COMPUTATIONAL OVERHEAD

Phases	IMKM	B-BLS	Proposed Algorithm
pairwise key	pairing	polynomial evaluation	modular exponentiation
share	Lagrange interpolation	matrix equation	Lagrange interpolation
authentication	modular exponentiation	pairing	modular exponentiation

For computational comparison, we choose some of the main computations and consider the most complex operations which are required in this computation. Based on this operations we compare the algorithms. These computations include pairwise key establishment, calculating the share of new joining node and key authentication.

For establishing pairwise keys, the proposed algorithm does modular exponentiation. IMKM uses pairing and B-BLS uses symmetric bivariate polynomials. Pairing has more complexity than modular exponentiation and evaluating one polynomial. In B-BLS, each node should solve a matrix equation with t equations and t unknown values for obtaining its share that needs a complex operations. Two other algorithms use Lagrange interpolation that outperforms B-BLS. In the proposed algorithm, authentication of public keys is based on modular exponentiation. In the B-BLS, pairing operation is needed to verify the token of each node. In IMKM keys authenticated via verifiable secret sharing. Modular exponentiation has less cost comparing to pairing and matrix equation. As a result, our scheme does not have many complex operations in comparison with two others and is suitable for ad hoc networks.

VI. Conclusion

Mobile ad hoc networks have become very interesting for researchers according to their advantages, but the main problem of these networks is the security issue. Cryptography can play an essential role to solve this problem, but its key management process among nodes will stay a big difficulty. Since there are some limitations for memory and process capabilities in MANETs, storing all the keys in whole nodes is not efficient, even if possible. This problem is more highlighted in large-scale MANETs. In this paper we proposed a novel key management algorithm for large-scale MANETs. The proposed algorithm is compared analytically with IMKM and B-BLS algorithms and it is shown that the novel proposed algorithm outperforms the others.

References

[1] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Proceedings of CRYPTO 84 on Advances in cryptology*. New York, NY, USA: Springer-Verlag New York, Inc., 1985, pp. 47–53.

[2] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Advances in Cryptology CRYPTO 2001*, ser. Lecture Notes in Computer Science, J. Kilian, Ed. Springer Berlin Heidelberg, 2001, vol. 2139, pp. 213–229.

[3] M. Girault, "Self-certified public keys," in *Advances in Cryptology EUROCRYPT 91*, ser. Lecture Notes in Computer Science, D. Davies, Ed. Springer Berlin Heidelberg, 1991, vol. 547, pp. 490–497.

[4] H. Petersen, P. Horster, and D. P. Horster, "Self-certified keys - concepts and applications," in *In Proc. Communications and Multimedia Security'97*. Chapman and Hall, 1997, pp. 102–116.

[5] S. Yi and R. Kravets, "Moca: Mobile certificate authority for wireless ad hoc networks," in *In Proceedings of the 2nd Annual PKI Research Workshop (PKI 03*, 2003.

[6] J. Luo, J.-P. Hubaux, and P. Eugster, "Dictate: Distributed certification authority with probabilistic freshness for ad hoc networks," *Dependable and Secure Computing, IEEE Transactions on*, vol. 2, no. 4, pp. 311–323, 2005.

[7] M. Ge and K.-Y. Lam, "Self-initialized distributed certificate authority for mobile ad hoc network," in *Advances in Information Security and Assurance*, ser. Lecture Notes in Computer Science, J. Park, H.-H. Chen, M. Atiquzzaman, C. Lee, T.-h. Kim, and S.-S. Yeo, Eds., vol. 5576. Springer Berlin Heidelberg, 2009, pp. 392–401.

[8] S. Capkun, L. Buttyan, and J.-P. Hubaux, "Self-organized public-key management for mobile ad hoc networks," *Mobile Computing, IEEE Transactions on*, vol. 2, no. 1, pp. 52–64, 2003.

[9] H. Dahshan and J. Irvine, "On demand self-organized public key management for mobile ad hoc networks," in *Vehicular Technology Conference, 2009. VTC Spring 2009. IEEE 69th*, 2009, pp. 1–5.

[10] A. Khalili, J. Katz, and W. Arbaugh, "Toward secure key distribution in truly ad-hoc networks," in *Applications and the Internet Workshops, 2003. Proceedings. 2003 Symposium on*, jan. 2003, pp. 342 – 346.

[11] Y. Zhang, W. Liu, W. Lou, and Y. Fang, "Securing mobile ad hoc networks with certificateless public keys," *IEEE Trans. Dependable Secur. Comput.*, vol. 3, no. 4, pp. 386–399, Oct. 2006. [Online]. Available: http://dx.doi.org/10.1109/TDSC.2006.58

[12] L. Li, Z. Wang, W. Liu, and Y. Wang, "A certificateless key management scheme in mobile ad hoc networks," in *Wireless Communications, Networking and Mobile Computing (WiCOM), 2011 7th International Conference on*, 2011, pp. 1–4.

[13] A.-F. Chan, "Distributed private key generation for identity based cryptosystems in ad hoc networks," *Wireless Communications Letters, IEEE*, vol. 1, no. 1, pp. 46 –48, february 2012.

[14] B. Lee and K. Kim, "Self-certificate: Pki using self-certified key," 2000.

[15] J. Lai and W. Kou, "Self-generated-certificate public key encryption without pairing," in *Public Key Cryptography PKC 2007*, ser. Lecture Notes in Computer Science, T. Okamoto and X. Wang, Eds. Springer Berlin Heidelberg, 2007, vol. 4450, pp. 476–489.

[16] D. Galindo and F. Garcia, "A schnorr-like lightweight identity-based signature scheme," in *Progress in Cryptology AFRICACRYPT 2009*, ser. Lecture Notes in Computer Science, B. Preneel, Ed. Springer Berlin Heidelberg, 2009, vol. 5580, pp. 135–148.

[17] Z. Li and J. Garcia-Luna-Aceves, "New non-interactive key agreement and progression (nikap) protocols and their applications to security in ad hoc networks," in *Mobile Adhoc and Sensor Systems Conference, 2005. IEEE International Conference on*, 2005, pp. 6 pp.–800.

[18] Z. Li and J. J. Garcia-Luna-Aceves, "Non-interactive key establishment in mobile ad hoc networks," *Ad Hoc Netw.*, vol. 5, no. 7, pp. 1194–1203, Sep. 2007. [Online]. Available: http://dx.doi.org/10.1016/j.adhoc.2006.07.002

[19] B. Vaidya, D. Makrakis, J. Park, and S.-S. Yeo, "Resilient security mechanism for wireless ad hoc network," *Wireless Personal Communications*, vol. 56, no. 3, pp. 385–401, 2011. [Online]. Available: http://dx.doi.org/10.1007/s11277-010-9978-7

[20] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.

[21] L.-C. Li and R.-S. Liu, "Securing cluster-based ad hoc networks with distributed authorities," *Wireless Communications, IEEE Transactions on*, vol. 9, no. 10, pp. 3072–3081, 2010.

[22] N. Saxena and J. H. Yi, "Noninteractive self-certification for long-lived mobile ad hoc networks," *Information Forensics and Security, IEEE Transactions on*, vol. 4, no. 4, pp. 946–955, 2009.