academicJournals

Vol. 8(17), pp. 793-802, 9 May, 2013
DOI: 10.5897/IJPS12.558
ISSN 1992-1950 ©2013 Academic Journals
http://www.academicjournals.org/IJPS

International Journal of Physical Sciences

Full Length Research Paper

Mobile agent based clustering and maintenance using secure routing protocol for mobile ad hoc network

R. Pushpa Lakshmi and A. Vincent Antony Kumar

PSNA College of Engineering and Technology, Dindigul, Tamilnadu, India.

Accepted 30 January, 2013

Routing in mobile ad hoc network is the challenging task due to the dynamic nature and resource limitations of network. Network clustering deals with partitioning network into clusters based on some rules. Clustering guarantees limited resource utilization and network scalability. In this paper, we mainly focused on using mobile agents for collecting information about cluster members and cluster maintenance. Ensuring security of mobile agent is a difficult task. In this work, the authorization of mobile agent is achieved using session key or secret key shared between mobile agent owner and node. We applied a distributed private key generation scheme, which generates secret key of cluster members based on 'n' key shares. The session key is generated based on past frequent traffic pattern that exist between the nodes. The behavior of mobile agent with respect to various security issues such as replay attack, non repudiation, denial of service and unauthorized access was discussed. The performance of the proposed scheme is evaluated under presence of varying number of malicious nodes. According to the observed results, the proposed protocol guarantees high packet delivery ratio and low delay, compared to cluster based routing protocol.

Key words: Cluster, mobile agent, distributed key generation, mobile ad hoc network, security.

INTRODUCTION

Mobile agents (MA) are commonly used in mobile ad hoc network (MANET) to collect network information and for network maintenance. MANET is wireless network with dynamic topology and without centralized control. MA is mobile software code that migrates from host to host. Due to its autonomy property, a MA can work without centralized control. MA also reacts automatically to the changes in the network environment. Due to dynamic nature, MA is suitable for MANET.

To collect information about network environment, more amounts of data need to be exchanged and processed across all nodes in the network. This increases the network traffic. MA reduces the network traffic, as it visits and collects data directly from mobile nodes. This reduces network latency (Wayne and Tom, 1999). Ensuring security of MA is difficult as the MA can be attacked by mobile node (MN) or other MA. A MA can also attack MN. One of the main challenges in using MA is to deal with security issues such as confidentiality, authentication, authorization, and non repudiation (Sarwarul Islam Rizvi et al., 2010; Marikkannu et al., 2011).

Previous work

In Pushpa Lakshmi and Vincent (2011), we proposed a secure dominating set based routing protocol which

*Corresponding author. E-mail: pushparaman@rediffmail.com.

applies a fuzzy logic controller to evaluate the trust value of nodes. The network is partitioned into clusters, where the cluster heads (CH) are elected based on their trust value and probability of future contact. The trust value of each node is computed based on their packet drop rate, packet forwarded successfully, and packet forwarded with alteration. Routing is carried out through trustable nodes in the network. To increase the level of security, a composite key management technique is applied. For key revocation process, mobile agent (MA) system is used to collect revoke point values for cluster members. Initially, revoke point value of all nodes is 0. When MA executes on mobile node (MN), the MN can update the revoke point value of suspected node. Based on revoke point value, the trust level of suspected node can be classified as not trustable, fully trustable, normal, low, and avg. The paper does not cover about security issues of MA code and data, election of common leader in cluster hierarchy, and selection of 'n' key serving nodes.

Ensuring security in MA system is important because MA code and data can be updated by other MA, affected by MN at runtime, and runtime MN may be affected by MA (Michael et al., 1998; Priyanka et al., 2010). In this current paper, we mainly focus on MA code integrity and authorization, MA owner authentication, MN authentication, and MA data protection. We also present improved MA system for key deactivation process, 'n' key serving nodes selection process, and common leader election process.

Related work

Secure image mechanism proposed in (Tarig, 2009) uses a secure image controller (SIC), which creates a copy of MA. The system classifies the node as trusted and untrusted. The original MA will directly be executed on trusted nodes. Whereas the encrypted copy of MA will be executed on untrusted nodes. When MA returns, SIC decrypt the MA code and compare hash digest of returned MA with hash digest of original MA code. The hash digest will be same, if the MA code is unaltered. Else SIC recorrect all the altered portion of MA code, using its backup copy. This method withstands eavesdropping and alteration attacks. But it does not address about attacks like masquerading, unauthorized access, replay attack, and non repudiation.

The security mechanism proposed in (Giovanni, 1997) maintains log files for agent's execution process. The log file records all activities performed during MA execution process. By checking log file, the agent owner can identify whether the MA is working properly as expected or not. This method requires maintenance of large log files. The security model proposed in (Sarwarul Islam Rizvi et al., 2010) is based on threshold cryptography. Changes to the agent code can be detected by computing message integrity code (MIC). The message is digitally signed by private key. The private key is generated by combining 'n' partial key shares. To decrypt the MA code and data, the MN must have proper private key. If MN is compromised, it will not generate correct key share. This method ensures security services like confidentiality, integrity and authenticity.

Environmental key generation method (Riordan and Schneier, 1998) generates key for encryption and decryption based on environmental parameters. Agent code and data is encrypted using the generated key. Agent can decrypt the code only when specified environmental condition exists. If the required environmental condition did not exist in the node, the decryption key will not be generated correctly. The agent cannot decrypt the code without knowing correct decryption key. Mobile agent model proposed in (Yikai, 2011) uses policy based cryptography. Authorization of MN is ensured by using policies defined by agent owner. MN will decrypt MA code only when it holds the policies defined by MA owner.

Mobile agent system proposed in (Shibli et al., 2009, 2010) uses security assertion markup language (SAML) ticket to authenticate MA code. SAML tickets are issued by policy decision point (PDP) server. Change to MA code is identified by verifying SAML tickets. Data accessibility is controlled by policy tokens, which specifies access control rules for MA and runtime node. Key distribution server generates group keys based on access policies of MA and MN. Data is encrypted by group key, which only allows the authorized MN to access the agent's platform data. It uses PKCS7 signed and enveloped data type to avoid unauthorized access of data (Shibli et al., 2010).

PROPOSED WORK

In this paper, we describe three mobile agents (MA) to perform the following tasks inside a cluster:

1. For key deactivation, MA collects each cluster member's trust opinion about other members in the cluster.
2. To elect common leader at higher level in hierarchy of clusters.
3. Collect trust value details of cluster members, used for selection of key serving nodes.

The process of MA is periodically initiated by CH. MA starts from CH and randomly moves to one of the neighboring cluster member. After reaching the neighbor, MA executes its process, collect the specified details and move to the next neighbor node. The movement of MA continues until it visits all nodes mentioned in the member list. Finally, when the MA return back to the CH,

the CH use the collected data for the process of key deactivation, common leader election and key serving nodes selection. Any malicious node in the network can modify MA code and its data. Malicious node can also create false MA. MA code is protected by applying one way hash function. To authenticate the node, a parallel key based scheme is proposed. The identity of CH and cluster members is validated using their secret key or session key. The CH will start a new version of MA, when old version fails to reach the CH within specified duration. The new version will be transmitted in different route.

Keys generation

Public key generation

The public key of new node is obtained by applying one way hash function on node's *ID*. If G1 is an additive group of prime numbers of order q and G2 is a multiplicative group of prime numbers of order q. The one way hash function $H:\{0,1\}^{*} \rightarrow G$ is defined. The public key is periodically updated based on node's trust value. The newly computed public key depends on old public key and node's current trust value and is shown in Equation (1).

$$Publickey, P_N = \begin{cases} H(ID_N), newnode \\ H(P_N \parallel trustvalue_N), existingnode \end{cases} \quad (1)$$

Private key generation

A distributed private key generation scheme is used, where the private key of node is generated by (n+1) serving nodes, based on Equation (2). 'n' serving nodes are elected by CH based on trust value of the node. The value of 'n' can be selected based on security level required by nodes, available resources and mobility of node.

$$\Pr ivatekey, S_N = S \cdot H(P_N) \quad (2)$$

Where $S = (S_1 + S_2 + S_3 + ... + S_N + S_{CH_{ID}})$,

$S_1, S_2, ...S_N$ – Secret key shares generated by 'n' serving nodes and $S_{CH_{ID}}$ – Secret key shares generated by *CH.*

The steps involved in private key generation of node N are as follows:

1. CH selects 'n' serving nodes based on trust value.

2. CH sends $KEYSHARE(P_{ID}, ID)$ to 'n' serving nodes.

3. Each serving node generate the secret key share based on Equation (3) and send $SHAREREPLY(P_{ID}, KS_{ID})$.

$$Keyshare, KS_i = h_{ID_i}{}^{r_i} \quad (3)$$

Where i=1 to n, CH, $h_{ID_i}{}^{r_i} = H(ID_i \parallel TV_i)^{r_i}$, ID_i – Identification number of node i, TV_i – Trust value of node I, r^i – Random number generated by node i

4. CH checks for correctness of key shares using Equation (4)

$$e(P_i, KS_i)^{r_i} \stackrel{?}{=} e(P_i{}^{r_i}, KS_i), \text{ i= 1 to 'n'} \quad (4)$$

The key share is correct, if Equation (2) is true. Else it shows the malicious behavior of the node. No other node in the cluster except CH knew about public key of serving nodes. The public key of serving nodes varies based on its trust value. So, other members cannot act like serving nodes.

5. Whole secret key is generated by CH based on Equation (5)

$$S_N = \sum_{i=1}^{n} (h_{ID_N}{}^{r_i})$$

Secret key of node N, (5)

Session key generation

The concept of mining based on backtracking is applied to generate the session key. The session key is generated based on previous traffic patterns that exist between the nodes. Each node maintains a table that holds details about last few traffic carried on in the network with other nodes. Session key is generated mainly based on frequent traffic pattern (FP) that exists between the nodes and is shown in Equation (6).

Session key shared by nodes *N1* and *N2*,

$$SK_{N1N2} = H(FP \parallel ID_{N1} \parallel ID_{N2} \parallel E_{PN1PN2}(r_{N1}P_{N1} \parallel r_{N2}P_{N2})) \quad (6)$$

Where r_{N1} and r_{N2} are random numbers selected by *N1* and *N2.* The random numbers generated by nodes are securely exchanged by applying encryption using node's public key. Nodes N1 and N2 exchange their random numbers as follows.

$$N1 \rightarrow N2 : E_{P_{N2}}(r_{N1})$$

Only the nodes N1 and N2 have similar traffic pattern. N1 and N2 will share the session key only when their FP is similar. If FP is different, it indicates the malicious behavior of the node.

$$N2 \rightarrow N1 : E_{P_{N1}}(r_{N2})$$

Key deactivation

MA maintains a hash table which includes CHID encrypted using secret or session key, node ID, trust value of the node, and revoke bit vector encrypted using secret key of the node or session key shared between the node and CH. Revoke bit vector is an 'n' bit vector, where 'n' represent number of nodes in the cluster. Hash index to access table entry is computed based on *N mod M*. *N* is the node ID and *M* is a constant selected based on cluster size. MA also maintains a route variable and an 'n' bit visited vector. Initially, route variable is set as null and visited vector bits are set as 0.

Mobile agent parameters

Mobile agent message package contains three parts: header, code and data. The header includes fields encrypted by MA's symmetric key. MA's symmetric key is assigned by the CH. The value of the header fields are updated by the MA code during its execution in each MN. The code represents the executable MA code. The data field contains a hash table with hash entries for each cluster members.

Header fields: Agent ID – a unique number assigned by MA owner to identify each mobile agent.
Exptime – It represent agent's life time. The agent get deleted automatically when it time expires. The agent must return back to the owner before its expiry period. When it fails, the owner initiates a new copy of MA with new agent ID. The maximum number of copies the owner initiates is limited and decided by the owner.
Time stamp – Time stamp of MA represent the time elapsed from beginning of its execution.
Visited vector – It is an 'n' bit vector initially assigned as 0. 'n' represents number of member nodes in the cluster. Whenever the MA visits a MN, the bit corresponding to that MN is set as 1. MA returns back to the owner when all the nodes are visited.
Member list – In the member list, the owner specifies the IDs of all nodes to be visited by MA. The bits in visited vector are assigned to MN based on the order of ID specified in the member list. For example, the first node in member list is N1, and then the first bit in visited vector is assigned to N1. The execution of MA code is limited only to the nodes listed in member list.
Route vector – This specifies the route information. Initially, the MA owner set it as null. Whenever a mobile node is visited, the ID of the node is appended in the route vector. This route vector information will be used by owner to forward future mobile agents.
Count – Number of nodes the agent visits during its traversal, calculated using number of 1's present in visited vector. MA executes its code only when counter <= *memcnt*. MA will be deleted if counter exceeds the limit. *memcnt* represents total number of member nodes present in the cluster.
Agent rights – The owner specifies the permission granted for MA. It includes details about resource utilization and read/write access.

Data field: Hash table – Hash table contains a hash entry for each node in member list. Each entry contains CHID, UNO, code hash, ID, revoke bit vector, and trust value.
CHID - Cluster head identification number.
UNO – Unique number assigned for MA.
Code hash – Hash value for the MA code computed by the CH.
ID – Identification number of cluster member.
Revoke bit vector - 'n' bit vector, where each bit corresponds to one cluster member.
Trust value – node's trust value.

CH initialization

The CH creates an entry for each member in the hash table. Each entry includes 4 fields: CHID, node ID, revoke bit vector, and trust value. Initially revoke bit vector and trust value are assigned as 0 for all members. CH also initializes the following:

Route vector = null; visited vector = 000000(n = 6); member list = {set of cluster member ID}
CH assigns a time limit for MA and starts the execution of MA process. The structure of the hash table is shown in Figure 1, in which *Skey* is current secret key of the node/session key shared between CH and the node.

The steps involved in CH initialization are as follows:

1. Initialize MA parameters: Agent ID, exptime, count = 0, visited vector = 000000, member list = {ID_1,ID_2,……,ID_n}, route vector = null.
2. Initialize hash table entries. For example, 0^{th} index entry is initialized as $E_{SkeyCHID_0}(CHID \mid ID_0 \mid 000000 \mid 0)$.

Each hash entry is encrypted by secret key of the node or session key shared between the node and CH.
3. Select the next node to traverse.
4. Move to the next selected node.

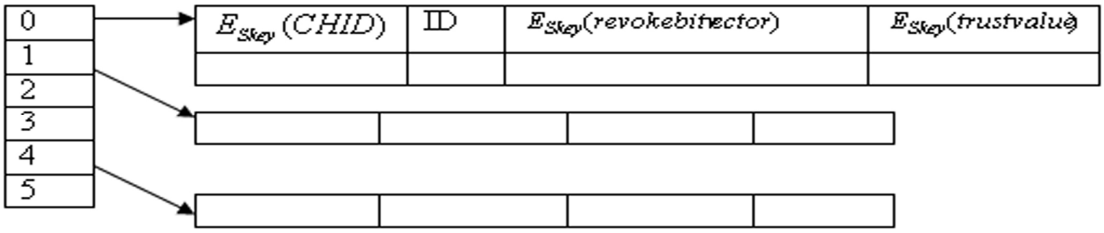

Figure 1. Data field of mobile agent. Data field contains a hash table. Each hash entry includes 4 fields. Field I: cluster head ID, agent unique number, hash value of agent code, agent rights. Field II: node ID, Field III: revoke bit vector, Field IV: trust value of node. The fields are encrypted using corresponding node's secret key or session key.

Mobile agent process

The steps involved in MN process are as follow:

Verification process at MN

1. Check the route vector. The MA code is already executed at current MN, if its ID is in route vector. In this case, MN stops the MA execution process.
2. Decrypt the code hash value using MN's secret key or session key.
3. Compute the hash value of the received MA code.
4. Compare the computed hash value with the decrypted hash value. If the hash values are equal, follow Step 4. If the hash value is invalid, the MN stops the MA execution process and does the following:

a. MN sends an error message (ERR), where PID represents previous node
ID.
b. On receiving ERR, CH sends UNTRUST (PID) to cluster members.
c. CH maintains a backup copy of MA code. It restarts the MA in different path excluding PID.
d. MN validates the owner of MA. Validation is done by decrypting the CHID specified in hash table entry.
e. Using hash index, MN identifies its corresponding hash table entry.
f. It decrypts the CHID either using the session key shared between them or using its current secret key.
g. If the agent is from malicious node, the decryption in the above step cannot take place correctly. In this case, MN terminates the MA execution process as follow:

i. MN verifies the access rights assigned for MA.
ii. It checks the life time of MA. If time expires, MN will stop the MA process.
iii. It checks the count value. If it exceeds the limit, MN stops the MA process.
iv. It checks the UNO of the MA. If UNO is invalid, MN will stop MA execution.

MA process – Header updation

1. Increment count by 1,
2. Update the visited vector. Set the bit corresponding to the current MN as 1,
3. Append the ID of current MN to route vector,
4. Encrypt all the header fields using MA's symmetric key.

MA process – Data updation

1. Set the bit value of revoke bit vector as 1, if MN suspects the trust worthiness of the node corresponding to that bit,
2. Encrypts the revoke bit vector using the session key shared between the current MN and CH or using MN's current secret key,
3. Compute MN's trust value using fuzzy logic controller. The trust value is within the range 0 and 10 ((0-1) not trusted, (2-3) low, (4-7) normal, (above 7) fully trustable).
4. Encrypts the trust value using the session key shared between the current MN and CH or using MN's current secret key,
5. Select the next node to be visited,
6. Move to the next node. Repeat from Step 1, until all nodes in member list are visited,
7. Return back to CH.

CH – Key deactivation

The steps involved in key deactivation decision making process by CH are as follows:

1. On receiving the MA before time expires, CH decrypts all revoke bit vectors. If time out occurs, CH starts a new version of MA and transmits in different route,
2. The CH identifies the node ID corresponding to the bit column having more number of ones,
3. The CH prepares a revocation list, which includes ID of the node whose key must be deactivated,
4. To ensure the security of the list, the node ID detail is encrypted using secret key or session key,

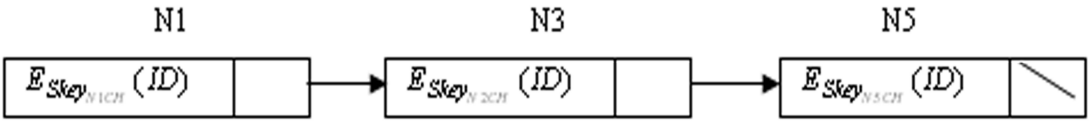

Figure 2. Key revocation list. The list is prepared by cluster head. Each node in the list contains ID of the suspected node. Each node value is encrypted using corresponding node's secret key or session key shared between node and cluster head. The list is transmitted across trustable nodes using mobile agent.

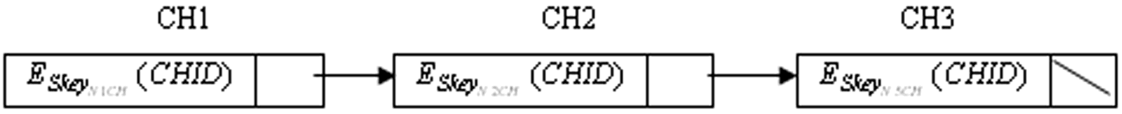

Figure 3. Common leader election. Cluster head with highest trust value is elected as leader. Elected leader ID is passed to all other cluster heads using a list. The ID is encrypted using secret key or session key of cluster head.

CH prepares a key revocation list which includes the ID of the suspected nodes, encrypted using session key or secret key. The revocation list is passed only to the trustable nodes in the cluster. To pass the revocation list, CH starts a MA in which member list includes only the ID of trustable nodes. MA follows the steps specified in MA code. On receiving the list, each node accesses its corresponding data, decrypt it and identify the ID of the suspected node. The cluster member will not consider any future messages received from the suspected node. The structure of the key revocation list is shown in Figure 2.

Selection of 'n' key serving nodes

MA – collection of trust values of members

Each node maintains information about their packet forwarding status and its neighbors. Trust value of the node is computed using fuzzy logic controller designed in our previous work (Pushpa Lakshmi and Vincent, 2011). The procedure follows the steps specified in MA code. In the computation process, MA activates the fuzzy controller to evaluate the trust value of the node. The trust value is secured by encrypting it with the secret key of the node or session key shared between the node and CH.

CH-key serving nodes selection

The steps involved in selection of 'n' key serving nodes are as follows:

1. On receiving the MA before time expires, CH decrypts the received trust value and sort the corresponding node IDs in descending order of their trust value. If MA is not returned within the time limit, the CH issues a new version of MA and transmits it in different route,
2. CH selects top 'n' node IDs, which will act as key serving nodes in secret key generation.

Common leader election

The common leader which acts as administrator in the hierarchical cluster is elected based on trust value. The CH with maximum trust value is elected as common leader in the higher level of the hierarchical structure. The CH1 initiates a MA for common leader election. The member list includes IDs of all CHs. The procedure follows the steps specified in MA code. MA travels to the next CH, using the virtual link established between the CHs:

1. CH1 decrypts the received trust values and find the CHID with maximum trust value,
2. CH1 prepares a list that includes the elected CHID encrypted using secret key or session key. The structure of the list is shown in Figure 3.

Security issues

The MA in the network perhaps affected by malicious node or by other MA. The MA may also utilize the resource of MN or access the information of MN, without the permission of MN. The behavior of our MA is discussed based on security aspects.

Masquerading

In this attack, the malicious node will act as other node and may change the MA code and data. In the proposed

model, the data in hash table is encrypted using the secret key of the node corresponding to the hash index or using the session key shared between the node and CH. The node that reads the data must first decrypt the message to validate the owner of MA. Decryption will be done only if the secret key or session key of the node is known. The secret key of the node is generated dynamically based on previous key and trust value of the node. The session key is generated based on past network traffic that exists between that node and CH. The malicious node can not identify the network traffic pattern of other node. So, the malicious node that acts as other node can not decrypt the MA data without knowing secret or session key. MA process continues only when the node is able to validate the owner. Updation to MA code can be detected using the hash value of MA code and data. Thus, the MA process ensures confidentiality, as the MA code and data are accessible only by authorized node.

Unauthorized access and alteration

Unauthorized access cannot be carried out by mobile node, as the data encrypted using secret key or session key of the node. Without permission, a node cannot access or update MA data. This ensures MA data integrity. Only the authorized MN can process MA. Even if the authorized MN later acts as malicious node, the MA data are secured as they are encrypted using corresponding node's secret key or session key. Any alteration in MA code can be detected using the code hash value. The malicious node is identified and informed to other members through UNTRUST message.

Replay attack

Time stamping is applied to avoid repeated transmission of MA. The owner of the MA attaches a timestamp which determines the lifetime of MA. The MA will be dropped when timestamp reaches expiry time. Route vector includes the ID of visited nodes. MA is transmitted only to the unvisited nodes. The current node stops the MA execution process, if it is already visited.

Non repudiation

The owner of MA, that is, CH include its CHID in the MA hash table. The CHID is used by MN for owner verification process. The CHID is encrypted by secret key or session key of the node. The secret key of each node is generated based on partial key distribution method by CH. The session key is based on past network traffic pattern that exists between the node and CH. Any node other than CH cannot generate secret or session key of a node. So, CH cannot later deny the transmission of MA. Thus, the proposed model ensures authentication of MA and MN.

Eavesdropping

The header section of MA is encrypted using MA's symmetric key. A unique symmetric key and UNO is assigned for each MA by the CH. So, a MA cannot access the encrypted header fields of another MA. The MA data fields are encrypted by node's session or secret key. The data fields are accessible only after the validation of MA owner, MA code and MA UNO. So, data fields of a MA are secured from other MA.

Denial of service

Agent rights specify the access rights assigned for MA. If MA violates the agent rights, the MN will stop the MA execution process.

RESULTS AND DISCUSSION

The simulation results of the protocol obtained using ns-2 simulator. The performance of the proposed protocol is evaluated by comparing the results with secure cluster based routing protocol. Table 1 summarizes the simulation parameters. We used the metrics packet delivery ratio, end to end delay, and packet drop rate for performance evaluation. Packet delivery ratio is the ratio of number of data packets delivered to the destination. End to end delay refers to the delay involved in transmitting packet from source to destination, which covers queuing delay, processing delay, propagation delay, and transmission delay. Packet drop rate is the number of packets dropped by the node. The network is simulated with 40 nodes, moving in an area of size 1500 $\times$ 1500 m. The source node transmit packet of size 1024 bytes. Packets are transmitted at rate of 4 per sec.

Figure 4 shows that the packets delivered by our proposed scheme is high even in the presence of malicious nodes. In our proposed scheme, the route is selected based on the trust worthiness of the nodes. The private key and session key are generated mainly based on the trust value of the nodes. The trust value of malicious nodes will be low, and they will be removed automatically while selecting trustable shortest routing path. When the number of malicious nodes is 5 our proposed scheme was able to transmit 90 packets, whereas the secure cluster based scheme transmits only 20 packets. When the malicious nodes count increased to 20 our proposed scheme was able to deliver 700 packets, whereas secure cluster based scheme delivers only 500 packets. Our proposed scheme produce high

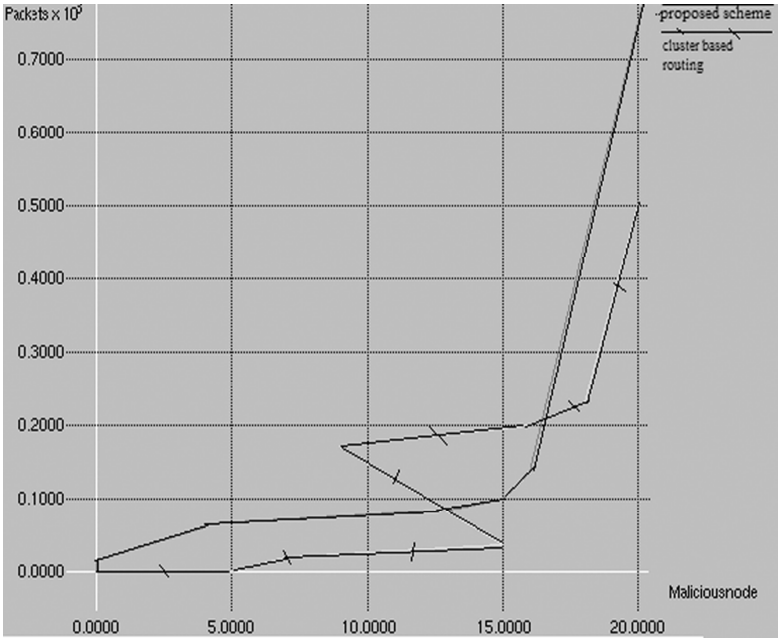

Figure 4. Packet delivered Vs Number of malicious nodes.

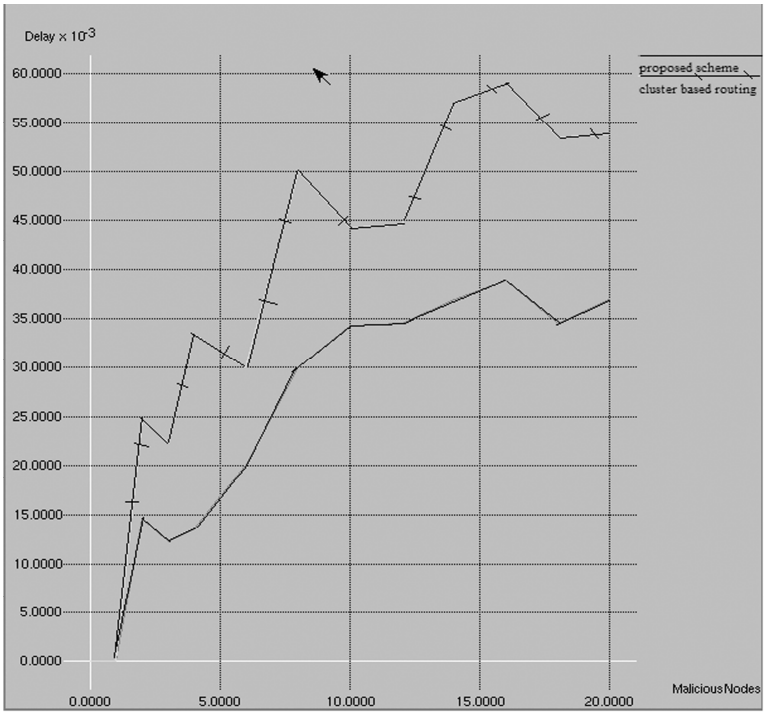

Figure 5. End to End delay Vs Number of malicious nodes.

packet delivery ratio even in presence of malicious nodes. The packet delivery ratio achieved in presence of varying number of malicious node is shown in Table 2. Figure 5 shows the end to end delay of proposed scheme and secure cluster based routing scheme. The end to end delay of our proposed scheme increases with increase in number of malicious nodes. However, the average end to end delay of our proposed scheme is less

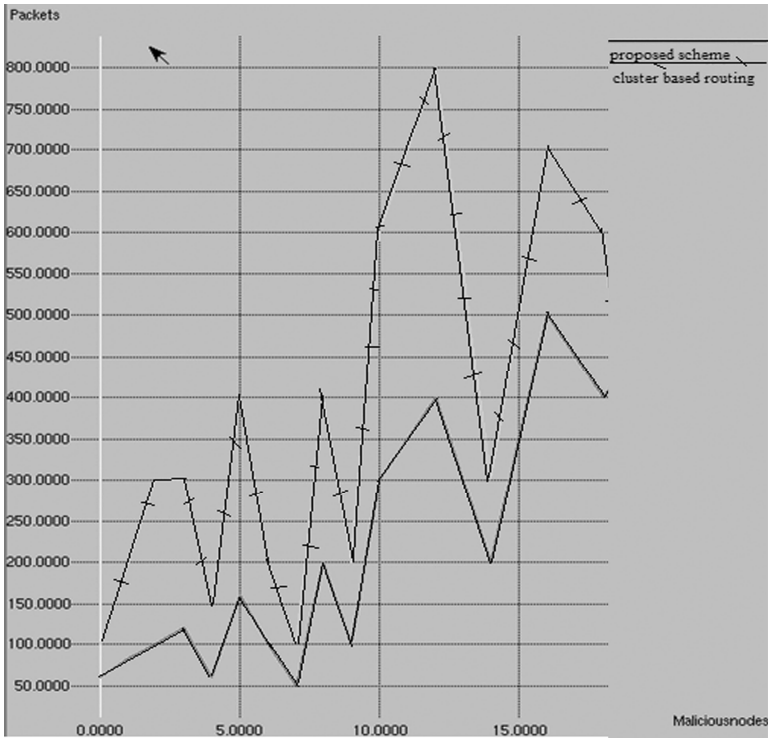

Figure 6. Packets dropped Vs Number of malicious nodes.

Table 1. Simulation parameters.

Parameter	Value
Transmitter range	250 m
Bandwidth	2 Mbps
Simulation time	20 s
Environment size	1500 m × 1500 m
Traffic type	Constant Bit Rate (CBR)
Packet rate	4 pkts/s
Packet size	1024 bytes

than delay incurred by secure cluster based routing scheme. When the number of malicious nodes in the network is 15, the end to end delay experienced by our proposed scheme is 38×10^{-3} which is less than 57×10^{-3} delay experienced by the other scheme. Our proposed scheme experienced nearly 15% less delay compared to secure cluster based routing scheme. The routing delay in presence of varying number of malicious node is shown in Table 2.

On average, the number of packets dropped by the other scheme is two times higher than the drop rate of our proposed scheme. The number of packets delivered by the proposed scheme is high even in presence of malicious nodes. Table 3 lists out routing delay and packet delivery ratio of network in presence of varying number of nodes with respect to processing time.

Figure 6 shows the number of packets dropped by the proposed scheme and secure cluster based routing scheme. The packet drop rate of the proposed scheme is low in number of malicious nodes.

Conclusion

In this paper, we presented mobile agent based cluster maintenance mechanism using secure routing protocol for mobile ad hoc network. Securing mobile agent from mobile node, mobile agent from other mobile agent and

Table 2. Delay and packet delivery ratio of network with varying number of malicious node.

No. of malicious nodes	Delay		Packets delivered		Saving in packet delivery
	CBRP	Proposed	CBRP	Proposed	
2	0	0	0	$0.02*10^{3}$	$0.02*10^{3}$
3	$5*10^{-3}$	$5*10^{-3}$	0	$0.03*10^{3}$	$0.03*10^{3}$
5	$32*10^{-3}$	$17*10^{-3}$	0	$0.08*10^{3}$	$0.08*10^{3}$
10	$44*10^{-3}$	$34*10^{-3}$	$0.03*10^{3}$	$0.08*10^{3}$	$0.05*10^{3}$
15	$57*10^{-3}$	$38*10^{-3}$	$0.04*10^{3}$	$0.1*10^{3}$	$0.06*10^{3}$
20	$54*10^{-3}$	$37*10^{-3}$	$0.5*10^{3}$	$0.8*10^{3}$	$0.03*10^{3}$

Table 3. Delay and packet delivery ratio of network with respect to time.

Time	Delay		No. of packets delivered	
	CBRP	Proposed	CBRP	Proposed
0	0	0	0	0
5	$4.2*10^{-3}$	$6*10^{-3}$	20	20
10	$15*10^{-3}$	$13.5*10^{-3}$	30	170
15	$26*10^{-3}$	$22*10^{-3}$	40	300
20	$37*10^{-3}$	$32*10^{-3}$	60	440

mobile node from mobile agent is achieved using cryptography based key generation mechanism.

Authorization of mobile agent and mobile node is ensured using the secret key of mobile node or session key shared between mobile agent owner and mobile node. Mobile agent is applied for various cluster maintenance tasks such as collection of cluster member details, selection of key serving nodes, key revocation, and selection of cluster leader. The simulation results show that the proposed scheme is effective. We compared the result with cluster based routing protocol (CBRP). The proposed scheme provides high packet delivery ratio and low delay compared to CBRP, in presence of malicious nodes.

REFERENCES

Giovanni V (1997). Protecting mobile agents through tracing. Proceedings of the Third ECOOP Workshop on Operating System support for Mobile Object Systems, Finland. pp. 137-153.

Marikkannu P, Adri-Jovin JJ, Purusothaman T (2011). An Enhanced Mobile Agent Security Protocol. Eur. J. Sci. Res. 5(3):321-331.

Michael SG, Jennifer CB, David GH (1998). Mobile agent and security, IEEE Communications Magazine, July. pp. 76-85.

Priyanka D, Kamlesh D, Govil MC (2010). Security Issues in Mobile Agents. Int. J. Comput. Appl. 11(4).

Pushpa Lakshmi R, Vincent AKA (2011). A Secure Dominating set based routing and key management scheme in Mobile Ad hoc Network. WSEAS Trans. Commun. 10(10):297-307.

Riordan J, Schneier B (1998). Environmental Key Generation towards Clueless Agents Mobile Agents and Security, G. Vigna, ed., Springer-Verlag. pp. 15-24.

Sarwarul Islam Rizvi SM, Zinat S, Bo S, Md. Washiqul I (2010). Security of Mobile Agent in Ad hoc Network using Threshold Cryptography, World Academy of Science. Eng. Technol. pp. 424-427.

Shibli MA, Giambruno A, Muftic S, Lioy A (2009). MagicNET: Security System for Development, Validation and Adoption of Mobile Agents, 3rd IEEE International Conference on Network and System Security. pp. 389-396.

Shibli MA, Yousaf I, Muftic S (2010). MagicNET - Security System for Protection of Mobile Agents. 24th IEEE International Conference on Advanced Information Networking and Applications (AINA) APR 20-23, Perth, Australia. pp. 1233-1240.

Tarig MA (2009). Using Secure-Image Mechanism to Protect Mobile Agent against malicious Hosts, World Academy of Science. Eng. Technol. pp. 439-444.

Wayne J, Tom K (1999). Mobile Agent Security, National Institute of Standards and Technology, Special Publication 800-19, August, US department of commerce, Technology administration, National Institute of Standards and Technology.

Yikai W (2011). Policy-based secure agents, Master of computer science thesis, University of Wollongong, http://ro.uow.edu.au/theses/3261/.