International Journal of Security and Its Applications
Vol.8, No.6 (2014), pp.273-284
http://dx.doi.org/10.14257/ijsia.2014.8.6.24

Improved Life of Watchdog Nodes in Ad hoc Networks

Anitha M[1] and Dr. Rhymend Uthariaraj V[2]

[1]*Assistant Professor,Department of Computer Science and Engineering, Velammal Engineering College, Chennai 600066, India*
anithajulian@gmail.com
[2]*Director and Professor,Ramanujan Computing Centre, Anna University, Chennai 600 025, Indiarhymend@annauniv.edu*

Abstract

Clustering or Cluster formation of nodes in wireless networks facilitates resource reuse and increases the performance capacity of the system. This paper presents a cluster-based solution to improve life of watchdog nodes that are designed to monitor intrusions in ad hoc network. The proposed approach ensures load sharing, increased robustness, added security and high reliability such that the entire network becomes fault tolerant. Unlike the existing systems, the algorithm intends to create multiple cluster heads for each cluster, thereby balancing the monitoring nodes' work load and reducing reelection overhead. Simulation results show the competence of the proposed work over the existing approaches in terms of performance metrics such as percentage of alive nodes, cluster head election time, re-election time, election rate and communication overhead.

Keywords: *Intrusion detection, Monitoring nodes, Multi-cluster head, Trust-based, Clustering*

1. Introduction

Wireless Ad hoc networks differ from traditional networks in terms of possessing the capability of connecting several nodes without a centralized access point. Security checks are normally provided by using network monitoring modules such as Intrusion Detection Systems (IDSs) [9, 16]. The extreme case will be to run the network monitoring module on every node in the network. This however creates the need to balance on the resource constraints and decrease the percentage of potential threats, due to the fact that the mobile nodes are energy-constrained. To overcome these issues, the network is divided into a set of one hop clusters where each node belongs to at least one cluster. The nodes belonging to a cluster elect a cluster head (leader node) to serve as the IDS for the entire cluster [18].

Existing research works in [8, 13, 10, 2] suggest different clustering schemes that elect a single cluster head that will act as the monitoring node, running its IDS. The schemes suggest that the cluster head could be elected either randomly or by applying trust based approaches. All existing works recommend the election of a single cluster head or a group of trust worthy cluster heads that are given the overall responsibility to act as the monitoring node(s) thereby increasing the overhead of IDS and also making the system still vulnerable to attack on the IDS itself. The proposed work aims at developing a hierarchical framework of multi-head clusters formation to implement IDS which can act as a multi-layered self-defensive monitor. The proposed solution considerably reduces the overhead of the IDS implemented on selected nodes. The approach gives promising results in terms of reducing reelection overhead and extending cluster lifetime.

ISSN: 1738-9976 IJSIA
Copyright ⓒ 2014 SERSC

The implementation of each ad hoc node as a monitoring node, results in the scenario that each node runs any of the existing intrusion detection models. Each of these nodes have limited battery power, and running the IDS on all the nodes tends to disproportion the resource consumption among all the nodes and reduce the lifetime of the network.

Authors in [8], address the run-time resource constraint problem using a cluster-based detection scheme where periodically a node is elected randomly as the ID agent for a cluster. Compared with the scheme where each node runs its own ID agent, this scheme is much more efficient while maintaining the same level of effectiveness. The suggested method suffers from a major drawback that there has to be a guarantee that the IDS cannot be compromised, or at the least, attacks against the IDS can be detected. If a compromised node happens to be elected as the cluster head, it can launch attacks without being detected because it is the only node that should run IDS and its IDS may have been disabled already.

A solution for an uncompromised cluster head is to use a tamper-resistant device to protect the IDS on each node, which has been implemented in [13]. The authors have put forth an approach for electing trust worthy leaders to reduce the resource consumes. The problems due to selfish nodes are overcome by providing the nodes incentives in the form of reputations to encourage nodes in honestly participating in the election process. The algorithm decreases the percentage of leaders, single node clusters, maximum cluster size and increases average cluster size. The approach addresses the issue of electing a selfish node as the leader. The proposed work focuses on the election of optimal number of cluster heads to reduce the overhead on the cluster head that runs the IDS. This makes IDS fault-tolerant by way of having supporting cluster heads.

The rest of this paper is organized as follows: Section 2 puts forth the problem statement. Section 3 analyzes the existing cluster formation techniques and leader election algorithms and identifies the best approach for the proposed work. Section 4 describes the mechanism for implementing ‘k’ optimal cluster heads to ensure load balancing and fault-tolerance of the network. Section 5 presents the empirical results of the proposed work and Section 6 concludes the work and highlights future extensions to the work.

2. Problem Statement

The discussion first considers an ad hoc scenario where threat level is low. It is extremely advantageous to have a single monitoring node in the cluster namely the cluster head that runs its IDS. Cluster head election is normally done based on varied criteria and under each criterion it chooses the most efficient node. As an example, if energy is the chosen criterion, the node with the highest battery power and residual energy is chosen as the cluster head, thereby overcoming power shortage problems. The proposed work considers adverse scenarios where the threat level is high; a single monitoring node may not be able to cater to the effective monitoring of the entire cluster. Electing more than one monitoring node is likely to balance the overhead incurred by the single monitoring node. Secondly every node in the network is required to act as a cluster head sometime making it mandatory to have the trained IDS models per-installed on all nodes [8]. The proposed implementation of multiple cluster heads is no additional overhead in running the IDS from multiple cluster heads at a pre-defined time interval.

Finally, in networks that favor single cluster heads, on assumption of fair election, meaning to say that every node has to be given a fair chance to serve as a cluster head, it requires to conduct re-election at regular intervals even though the elected cluster head might still be capable of continuing its role. The time for re-election is far more reduced if multiple cluster heads are elected in a single election and at a later stage it only switches over among

 Copyright ⓒ 2014 SERSC

the elected cluster heads. There is always the trade-off factor between efficiency and trust. The proposed work considers trust factor of the nodes as prime importance and implements 'k' number of optimal cluster heads acting as monitoring nodes, to reduce overhead of a monitoring node and also make the IDS fault-tolerant.

3. Existing Approaches of IDS

Clustering is an important approach to manage ad hoc networks. The clustering mechanism can be described as the classification of nodes in a network hierarchically into equivalence classes with respect to certain attributes such as geographical regions or small neighborhood of one or two hops from special nodes called the cluster heads [10]. Considerable works in the area of cluster head election have been implemented in [2, 17, 19]. The need for a cluster head is common to any clustering application such as routing [2] and key distribution [3, 4] in the implementation of IDS in ad hoc networks. All these methods ensure that one node can stay alive and actively operational while others can be in the energy-saving mode.

Authors of [10] suggest a cluster based approach for dynamic routing in networks. Assuming the dynamic network as an undirected graph, G = (V, E), a node n_1 can communicate with another node n_2 only within a limited geographical region around it, which can be termed as node coverage area of radius 'r'. An edge (n_1, n_2) connects node n_1 and node n_2 if the corresponding nodes are in the vicinity and thus a communication link is established between them. A host may sometime be isolated where there are no other mobile hosts in its vicinity. Such a host will be represented in the graph by a disconnected node.

In comparison with existing and conventional routing protocols, the suggested cluster-based approach incurs lesser overhead during updates in topology and also has faster merger of nodes. The effectiveness of this approach also lies in the fact that existing routing protocols can be directly applied to the network, replacing the nodes by clusters. The advantage of this algorithm is that it does not directly elect cluster heads. Hence, a selection function of the user's choice can be effectively computed on a random set of inputs obtained from each member nodes, during the cluster head election.

3.1 Secure Leader Election

Traditionally, most of the clustering election algorithms select only one cluster head in a cluster. The research work in [11] shows the selection of two cluster heads. A multi-head clustering algorithm for VANETs given in [15] uses the cluster structure to facilitate the finding, uploading, and downloading of multimedia files.

An implementation on the computational complexity analysis of multicast models in distributed systems is elaborated in [5]. The LeaSel model [6] is very difficult to attack as the model consists of two entities that manage and control the groups and subgroups, namely deputy controller and leader. The former decides the rank of all the members in the subgroup and maintains a rank list. The member which ranks first is designated as the leader. The leader is authorized to perform key generation and distribution. In continuation to this work, authors in [12, 14] suggest a model based on this LeaSel multicast model, wherein concept of multiple leaders can be used for load sharing, increased robustness and added security. Survey on existing works shows that that there is no existing work on implementing multiple cluster heads for running IDS.

Some of the evaluation methods for cluster election in ad hoc networks are:

- ID-based method in which a node with the largest or smallest identification number (ID) in its neighborhood can be elected.

Copyright ⓒ 2014 SERSC

- In the degree-based method, a node with the largest degree value (indicating the number of its direct neighbors) in its neighborhood is elected.
- Local nodes with a competition-based method can compete to be elected as proposed in Max-Min D-hop clustering Algorithm [1] and Random Competition based Clustering.
- A node with stable relative mobility to its neighbors is elected in the mobility-based method.
- A hybrid multiple-metrics method can be obtained by combining one or more evaluation factors together by a weighted average function to elect the cluster head.

4. Proposed Mechanism

The proposed approach initially partitions the ad hoc network under study, into uniform clusters using the algorithm suggested in [10]. At the lowest layer of defense, we propose and evaluate a multi cluster head clustering scheme for grouping IDS data as shown in Figure1. During the cluster head election phase, 'k' number of cluster heads is elected for each cluster. At any given time interval, only one cluster head acts as the leader and the cluster head is alternated for every transaction. In this way, the 'k' cluster heads share the overhead of running the IDS among themselves.

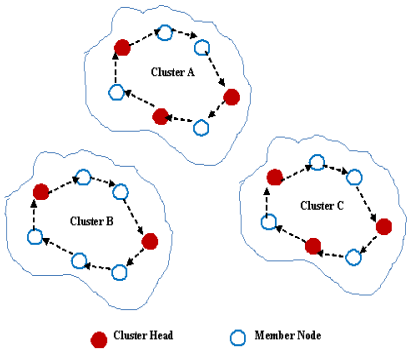

Figure 1. Optimal Cluster Heads in each Cluster

Moreover, self-defense of the IDS is also achieved as any attack on the cluster head involves attacking all the 'k' cluster heads instead of one. Electing and selecting an optimal number for depends upon the application and the environment in which the cluster head is being elected.

4.1. Trust Computation

Trust computations consist of three components experience (also termed as direct trust), recommendation and knowledge [20]. A hybrid method to compute trust on a node is based on direct and also recommendations from other nodes. The trust evaluation method [7] has been considered here omitting the consideration of choosing the most relevant factor. The trust of node *a* about node *b*, $(T_a(b))$ is given in (1). The proposed work represents this computed value asδ.

$$T_a(b) = ((1 - \alpha))\, Q_a(b) + \alpha R_a(b),\ 0 \le \alpha \le 1,\ 0 \le Q_a(b) \le 1,\ 0 \le R_a(b) \le 1 \quad (1)$$

where $Q_a(b)$ represents the trust node *a* has on node *b* based on its own observations and $R_a(b)$ is the aggregate value of the recommendations from all other neighbors of *b*. Further,

 Copyright ⓒ 2014 SERSC

$$Q_a(b) = \beta E_a(b) + (1 - \beta) T_a(b),\ 0 \leq \beta \leq 1 \quad (2)$$

where $E_a(b)$ represents the trust value obtained by the judgment of the actions of *b* and $T_a(b)$ gives the last trust level value stored about node *b* on node *a*. In a clustered network once the nodes compute their mutual trust values, the cluster head assists the member nodes in further computation.

4.2. Cluster Head Election

The practical use considered in this work is that of assigning the cluster head is proposed with the intrusion monitoring task. It is extremely logical to consider the node that is in the center of the cluster as the cluster head. This enables similar data transmission paths for all member nodes to the head node. In an ad hoc scenario as the mobile nodes undergo frequent changes in their position and the cluster head should possess stable relative mobility to its members in order to reduce frequent reelections. From the survey of existing work in section 3.2, the concepts of center position and relative mobility of each mobile node [15], has been chosen as the criterion for cluster head election in the propose work.

At regular intervals each mobile node in the network broadcasts a hello message to its neighbors that are in one-hop distance. The sent message carries information about the location, motion vector, and relative mobility of the mobile node. A node maintains a neighbor table to record the mobility data of its current one-hop neighbors by listening to these messages. It can be understood that clusters with nodes having different moving directions are unstable. Hence all nodes in a cluster are restricted to have the same moving direction. If a node has *m* entries in its neighbor table, it can be assumed that the first entry records its own mobility-related data. Three data fields (x_i, y_i) are recorded in the i^{th} entry, which indicate the current location (x_i, y_i) and the current moving speed (v_i) of the recorded node. The Selection Function (SF) for cluster head election is defined in terms of RPM of a node is computed using the following steps:

Compute the center position from these *m* entries in the neighbor table as (3).

$$P_c = (x_c, y_c) = \frac{1}{m} \left(\sum_{i=1}^{m} x_i, \sum_{i=1}^{m} y_i\right) \quad (3)$$

Compute the relative distance to this center position as (4).

$$RelDist_{1c} = |P_1 - P_c| = \sqrt{(x_1 - x_c)^2 + (y_1 - y_c)^2} \quad (4)$$

Sort this*m* moving speeds (scalar values) and find the median as (5).

$$v_c = Median\ \{Sort\{v_1, v_2, \ldots, v_m\}\} \quad (5)$$

Compute the relative speed to the median speed as (6).

$$RelSpeed_{1c} = |v_1 - v_c| \quad (6)$$

Compute the SF as (7).

$$SF = \delta . \frac{RelDist_{1c}}{Max\{RelDist_{jc}, j \epsilon\ 1 \sim m\}} + (1 - \delta) \frac{RRelSpeed_{1c}}{Max\{RelSpeed_{jc}, j \epsilon\ 1 \sim m\}} \quad (7)$$

where δ is the trust value computed as explained in section 4.1. The proposed algorithm ranks the nodes using the SF value for each node.

4.3. Proposed Algorithm

During the first phase the mobile node are ranked according to the SF value in its neighbor table. SF value as explained earlier has been computed in terms of the trust of node ‘a’ over the trust of another node ’b’. The trust value comprises of

- Node n_a’s self-evaluated trust on n_b. n_a computes this by directly monitoring n_b.
- Weighted sum of other nodes’ trust on n_b evaluated by n_a.

Copyright ⓒ 2014 SERSC

The node with the maximum SF value declares itself as cluster head (CH), means of Advertisement Message. During the next phase CH chooses the next optimal number of nodes with the highest SF value as assistant cluster heads (ACHs). This information is however known only to the ACHs and not to all other member nodes (MNs) in the network. The pre-installed IDS in ACHs are activated by CH and the ACHs are in constant communication with CH for IDS data aggregation. If at any point of time if the CH fails, the topmost ACH takes over as CH. Reelection is initiated when the number of ACHs reaches a threshold value θ which is assumed as

$$\theta = (\text{optimal number of ACHs selected by CH}) / 2 \quad (8)$$

Algorithm 1 is the case where only one cluster head is elected. Thus the node with the maximum value of SF gets elected as the CH. The elected CH runs its own IDS. When the CH is compromised, reelection is initiated and there is considerable idle time on part of the IDS performance (time for reelection) and also lacks fault-tolerance.

Algorithm 1

1. *Compute SF in terms of δ (trust value of nodes)*
2. *Rank nodes in network by SF value*
3. *Select node with max(SF) as CH*
4. *CH activates IDS on itself*
5. *If CH is compromised*
6. *call for re-election*

Algorithm 2 shows the proposed algorithm for improving the life of the monitoring nodes in ad hoc networks. The fault-tolerance has been implemented with the election of multiple cluster heads rather than the traditional approach of electing a single cluster head. As illustrated by the simulation results, such an election of multiple cluster heads reduces the reelection time and increases the percentage of alive nodes at a given point of time.

As the case in Algorithm 1, here too the value for SF is computed as in equation (7).

Algorithm 2

1. *Compute SF in terms of δ (trust value of nodes)*
2. *Rank nodes in network by SF value*
3. *Select node with max(SF) as CH*
4. *CH selects next optimal number of nodes with max (SF) as ACHs*
5. *CH activates IDS on elected ACHs*
6. *If CH compromised*
7. *While n(ACH) ≥n(ACHs)/2*
8. *ACH with max(SF) selected as CH*
9. *ACH takes over CH functionality*
10. *n(ACHs)--*
11. *Call for Re-election*

 Copyright ⓒ 2014 SERSC

5. Performance Evaluation

The simulations to evaluate the performance of our proposed algorithm have been carried out using NS2 (Network Simulator). The major objective of the proposed simulation is to study the fault-tolerance of the IDS under situations of attacks. In order to show the effect of the frequent election of cluster head that proposes to run the monitoring action (act as IDS), the following performance metrics have been chosen. *Fraction time without Cluster head, Percentage of alive nodes in the network show* the effectiveness of the IDS functionality as intrusions are detected and prevented on time. *Re-election Time* is the mean time elapsed between the time in which a nodes starts participating in the cluster head election and the time at which the node knows the identity of its cluster head. *Election Rate* is the average number of elections that a node takes part in per unit time. *Message overhead* is the average number of messages sent by a node during a single election.

Table 1. Simulation Parameters

Parameter	**Value**
Network Size	2000m x 2000m
Node deployment model	Random Waypoint
Average Speed of node	15m/s
Packet rate	4 packets/sec
Traffic Patterns	CBR / UDP Traffic

The simulation environment is as shown in Table 1. Initially all nodes are acting as member nodes and a server node starts the election process. A node declares itself a cluster head when its SF value is the smallest one in its neighbor table. The new cluster head broadcasts an invitation to join to its neighbors. Any normal node joins to the cluster when it receives an invite packet from a cluster head in its neighborhood. After joining the node sends a reply packet indicating the join to the cluster head.

The first set of results is obtained using *Algorithm 1*, where the simulation is for electing a single cluster head per cluster. Table 2 shows the influence of the number of neighbors each node considers during trust evaluation [21]. The trust level error for trust computation decreases as the number of neighbors increases. Thus it can be concluded that the trust value calculated is at its best when the trust calculation combines the information from the experience calculation ($Ea(b)$) and the recommendation calculation ($Ra(b)$) to derive a trust level.

Table 2. Trust Error Level for Varying Number of Neighbors

Number of Neighboring Nodes under consideration	3	5	7	15	30
After 5 seconds	0.050	0.040	0.030	0.019	0.010
After 10 seconds	0.037	0.037	0.027	0.018	0.008
After 15 seconds	0.039	0.039	0.029	0.018	0.008
After 20 seconds	0.040	0.038	0.028	0.018	0.008
After 25 seconds	0.035	0.036	0.026	0.018	0.008

Copyright ⓒ 2014 SERSC

During cluster head election it is advantageous to broadcast Election messages as they reduce the Election messages and also the number of Acknowledgement messages [19]. The performance metrics in the conducted study are:

Fraction of time without Cluster head: When a single cluster head is elected, the network happens to be without cluster head once the elected cluster head is compromised. But if more than one cluster head is elected, even if the cluster head is compromised, the IDS operation is shifted to the next available cluster head and thus 99.99% of the time the cluster has a cluster head as the re-election time does not happen without a cluster head. Figure 2 shows the time fraction for the cluster without its cluster head.

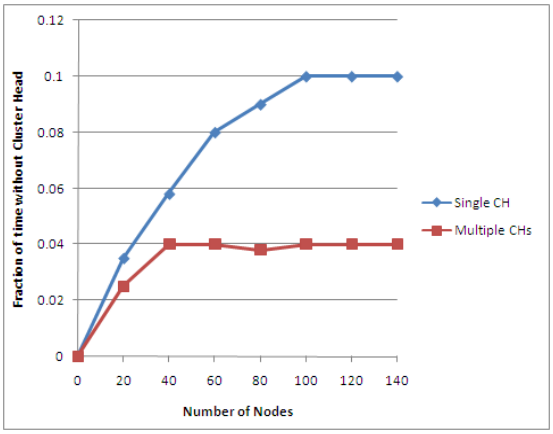

Figure 2. Fraction of time without CH

Percentage of Alive Nodes: The option as in *Algorithm 1* leaves the network without cluster heads for larger fraction of the time. This has disabled the IDS and thus attack on the nodes is more vulnerable and thereby we can calculate the alive nodes after the cluster has been re-elected.As shown in Figure 3, the number of alive nodes increases in using *Algorithm 2* as the IDS is enabled most of the time due to the switching of cluster heads and the IDS is kept active for a greater time period.

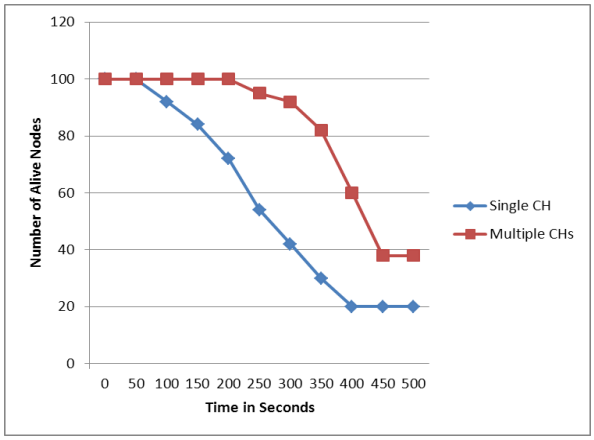

Figure 3. Percentage of Alive Nodes

Copyright ⓒ 2014 SERSC

As for *Algorithm 1*, as already mentioned, the *Fraction time without cluster head* initially increases with increase in number of nodes but eventually drops after a particular number of nodes is reached. This behavior is similar to that of the trends observed in *Election Rate*. The nodes are mobile and when the number of nodes is small most of the nodes are isolated. When the number of nodes increases, it results in frequent departure from their cluster heads and thus an increase in *Election Rate*. But beyond a threshold value (say number of nodes is greater than 100), the node density (nodes per unit area) become quite high and most of the nodes belong to a large connected component, that remains connected for longer durations and, so *Election Rate* decreases as shown in Figure 4.

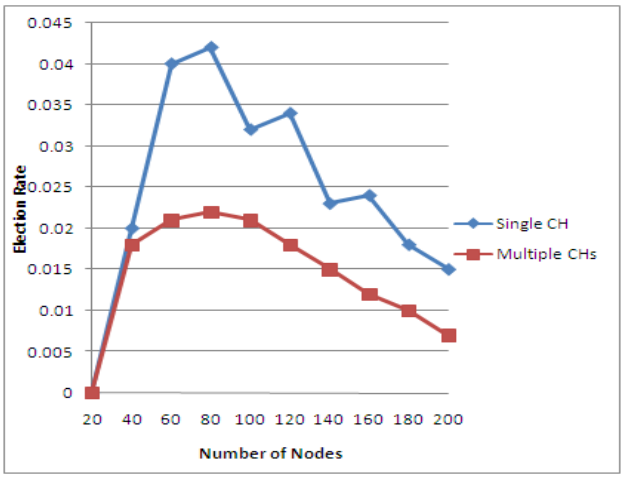

Figure 4. Election Rate

It has been observed that even during high mobility scenarios, the disconnection from the cluster head is very brief and thus *Election Rate* decreases. The values for *Election Rate* are all the more reduced for the multiple cluster head scenario that uses *Algorithm 2*, where the number of elections per unit time is very minimal. Since we are interested to elect more than one cluster head during an election and use them subsequently when the CH gets compromised, the *Re-Election Time* is prolonged in the case of multiple CHs.

Considering the metric of *Message Overhead,* any node (except for the source) sends at least one unicast *Child* message and one *Acknowledge* message to its parent during an election. Each node, in addition also sends at least 2 broadcast messages, namely, one *Election* message upon joining the election and one *Cluster Head* message upon termination. As the number of nodes increase, both the broadcast message overhead and unicast message overhead also increase. But in our case since we are prolonging the Re-election time interval our algorithms certainly cut down on message overhead too.

6. Conclusion

The propose work has introduced the concept of electing multiple cluster heads in a clustered network thereby balancing the workload of a cluster head. The chosen performance metrics such as *Fraction time without Cluster head*, *Percentage of alive nodes in the network, Re-election Time, Election Rate and Message overhead* clearly indicate the advantages of using multiple cluster heads rather than a vulnerable single cluster head. In all the simulation results obtained remarkable increase in efficiency due to the proposed solution has been seen. Further to this, a practical application for this concept by taking the case of using the cluster

Copyright ⓒ 2014 SERSC

head as the network monitoring node or in other words the node that runs the IDS software has also been illustrated. Future extensions to this work can include application of game theoretical techniques for increasing the trustworthiness of the nodes.

References

[1] A. D. Amis, R. Prakash, T. H. P. Vuong and D. T. Huynh, "Max-Min d-Cluster Formation in Wireless Ad hoc Networks", In Proceedings of IEEE Infocom Conference, Tel Aviv, Israel, **(2000)**, pp. 32-41.
[2] S. Basagni, "Distributed Clustering for Ad hoc Networks", In proceedings of the IEEE International Symposium on Parallel Architectures, Algorithms, and Networks (ISPAN), Perth/Fremantle, WA, **(1999)**, June 23-25, pp. 310-315.
[3] M. Bechler, H. J. Hof, D. Kraft, F. Pahlke and L. Wolf, "A Cluster-based Security Architecture for Ad hoc Networks", In IEEE INFOCOM, Hong Kong, China, **(2004)** March, pp. 2404–2413.
[4] B. DeCleene, L. Dondeti, S. Griffin, T. Hardjono, D. Kiwior, J. Kurose, D. Towsley, S. Vasudevan and C. Zhang, "Secure Group Communications for Wireless Networks", In Proceedings of the IEEE Military Communications Conference (MILCOM), **(2001)** October, pp. 28-31, McLean, VA, USA.
[5] E. R. Blessing and V. R. Uthariaraj, "LeaSel: An Efficient Key Management Model for Scalable Multicast System", In Proceedings of ICORD India, **(2002)** December 27-31, Anna University, India.
[6] E. R. Blessing and V. R. Uthariaraj, "Evaluation and Analysis of Computational Complexity for Secure Multicast Models", Springer Verlag, Lecture Notes in Computer Science, vol. 2668, **(2003)**, pp. 684-694.
[7] K. Govindan and P. Mohapatra, "Trust Computations and Trust Dynamics in Mobile Ad hoc Networks A Survey", IEEE Communications Surveys and Tutorials, **(2012)**, pp. 1-30.
[8] L. W. Huang, "A Cooperative Intrusion Detection System for Ad hoc Networks", In Proceedings of the ACM Workshop on Security of Ad Hoc and Sensor Networks, **(2003)**, pp. 135-147.
[9] K. Scarfone and P. Mell, "Guide to Intrusion Detection and Prevention Systems (IDPS)", NIST Special Publication 800-94 Revision 1 (Draft), **(2012).**
[10] N. H. Krishna, M. V. Chatterjee and D. K. Pradhan, "A Cluster Based Approach for Routing in Dynamic Networks", In Proceedings of the ACM SIGCOMM Computer Communication Review, **(1997)**, pp. 49-64.
[11] T. D. C. Little and A. Agarwal, "An information propagation scheme for VANETs", In proceedings of IEEE Intelligent Transportation Systems Conference, **(2005)**, pp. 155-160.
[12] M. S. Vennila, S. Srinivasan, T. C. Rangarajan, V. Sankaranarayanan and R. V. Uthariaraj, PLEASE, 'P'-LEAder Selection for Multicast Group Communication. IJCSNS International Journal of Computer Science and Network Security, vol. 6, no. 11, **(2006),** pp. 277-283.
[13] N. Mohammed, O. H. L. Wang, M. Debbabi and Bhattacharya, "Mechanism Design-Based Secure Leader Election Model for Intrusion Detection in MANET", IEEE Transactions on Dependable and Secure Computing, **(2011)**, pp. 1-15.
[14] R. V. Uthariaraj, T. C. Rangarajan, S. Srinivasan, S. M. Vennila and V. Sankaranarayanan, "A Security-Centric Comparative Study of PLEASE with Existing GKM Protocols", Communication Networks and Services Research, CNSR '07. Fifth Annual Conference on, **(2007)**, pp. 192-202.
[15] S.-C. Lo, Y.-J. Lin and J.-S. Gao, "A Multi-Head Clustering Algorithm in Vehicular Ad Hoc Networks", International Journal of Computer Theory and Engineering, vol. 5, no. 2, **(2013),** pp. 242-247.
[16] S. Northcut, "Network Intrusion Detection", Sams Publishing, **(2002)**.
[17] K. Sun, P. Peng, P. Ning and C. Wang, "Secure Distributed Cluster Formation in Wireless Sensor Networks", In Proceedings of the 22nd Annual Computer Security Applications Conference on Annual Computer Security Applications Conference, Washington, DC, USA, IEEE Computer Society, **(2006)**, pp. 131–140.
[18] T. Anantvalee and J. Wu, "A Survey on Intrusion Detection in Mobile Ad Hoc Networks", Wireless/Mobile Network Security, DOI: 10.1007/978-0-387-33112-6_7, **(2008)**, pp. 170–196.
[19] S. Vasudevan, B. DeCleene, N. Immerman, J. Kurose and D. Towsley, "Design and Analysis of a Leader Election Algorithms for Mobile Ad hoc Networks", In Proceedings of the 12th IEEE International Conference on Network Protocols, Berlin, Germany, **(2008)**.
[20] P. B. Velloso, R. P. Laufer, D. O. Cunha, O. C. M. B. Duarte and G. Pujollel, "Trust Management in Mobile Ad hoc Networks Using a Scalable Maturity-based Model", IEEE Transactions on Network Service Management, vol. 7, no. 3, **(2010)**, pp. 172–185.
[21] K. Xu and M. Gerla, "A Heterogeneous Routing Protocol Based on a New Stable Clustering Scheme", In Proceedings of Milcom Conference, vol. 2, **(2002)**, pp. 838-843.

 Copyright ⓒ 2014 SERSC

Authors

Anitha M, she received her bachelor's degree in Computer Science and Engineering from Pondicherry University (1990) and the master's degree in Computer Science and Engineering from Anna University Chennai (2007). She is currently working as Assistant Professor in the Department of Computer Science and Engineering, Velammal Engineering College, India and is involved with the research activities of TIFAC-CORE in Pervasive Computing Technologies. She is a Life member of Computer Society of India and is currently pursuing her Ph.D. in Ad hoc Networks under the guidance of Dr. V. Rhymend Uthariaraj.

Rhymend Uthariaraj V., he received the master's degree in Computer Science and Engineering and Ph.D. (1999), from Anna University, Chennai, India. He is working as a Professor of Information Technology. Currently he is the Director of Ramanujan Computing Centre of Anna University Chennai and also the Secretary of Tamil Nadu Engineering Admissions. Also he is the syndicate member of Anna University Chennai. His research interests include Network Security, Pervasive computing and Optimization.

Copyright ⓒ 2014 SERSC

Copyright ⓒ 2014 SERSC