International Journal of Wireless Networks and Communications.
ISSN 0975-6507 Volume 4, Number 1 (2012), pp. 71-81

http://www.irphouse.com

Using VCG based Designing and Electing Secure Leader Model for Intrusion Detection System in Manet

Laiphangbam Melinda and S. Madhan Kumar

[1]M.E (CSE), [2]M.E, Assistant Professor
Veltech Multitech Dr. Rangarajan Dr. Sakunthala Engineering College, Chennai, India
E-mail: mel.laiphangbam20@gmail.com, madhan866@gmail.com

Abstract

This paper is to elect leader (cluster head) from clusters in the presence of selfish node to serve as the IDS for the entire cluster and to balance the resource consumption among all nodes and prolong the lifetime of an MANET. A Mechanism design theory called VCG (Vickrey, Clarke, and Groves) and globally optimal election result with low cost are proposed to address the issue of selfish node and optimal election issue, respectively. Two possible application settings namely, Cluster-Dependent Leader Election (CDLE) and Cluster-Independent Leader Election (CILE) are used to address the above issues.

Index Terms: leader election, intrusion detection system, mechanism design and MANET security.

Introduction

A mobile ad hoc network is a collection of wireless devices moving in seemingly random directions and communicating with one another without the aid of an established infrastructure. So a mobile device is required to play the role of a router for forwarding the packets of its neighbors in dynamic environment where node movements cause frequent topology changes. MANETs have no fixed chokepoints /bottlenecks where Intrusion Detection Systems (IDs) can be deployed. Hence, a node may need to run its own IDS and cooperate with others to ensure security. This is very inefficient in terms of resource consumption since mobile nodes are energy-limited. Solution to above is to divide the MANET into a set of 1-hop clusters where each node belongs to at least one cluster. The nodes in each cluster elect a leader node

(cluster head) to serve as the IDS for the entire cluster. The leader-IDS election process can be either random or based on the connectivity. Both approaches aim to reduce the overall resource consumption of IDSs in the network. With the random model, each node is equally likely to be elected regardless of its remaining resources. The connectivity index-based approach elects a node with a high degree of connectivity even though the node may have little resources left. With both election schemes, some nodes will die faster than others, leading to a loss in connectivity and potentially the partition of network. Although it is clearly desirable to balance the resource consumption of IDSs among nodes, this objective is difficult to achieve since the resource level is the private information of a node. Unless sufficient incentives are provided, nodes might misbehave by acting selfishly and lying about their resources level to not consume their resources for serving others while receiving others services.

Problem Statement

Every node has an IDS and a unique identity. Two challenges arise in electing the most cost-efficient as leader. First, the resource level that reflects the cost of analysis is considered as private information. As a result, the nodes can reveal fake information about their resources if that could increase their own benefits. Second, the node might behave normally during the election but then deviate from normal behavior by not offering the IDS service to their voted nodes.

In this model, MANET is considered as an undirected graph G = (N, L), where N is the set of nodes and L is the set of bidirectional links. We denote the cost of analysis vector as C= (c_1, $c_{2,}$…, c_n), where n is the number of nodes in N. We denote the election process as a function vt_k (C,i), where vt_k (C, i)= 1 if a node i votes for a node k; vt_k (C, i)= 0, otherwise. Each elected leader is assumed to allocate the same budget B (in the number of packets) for each node that has voted for it. Knowing that the total budget will be distributed among all the voting nodes according to their reputation. Thus, the model will be repeatable. The objective of minimizing the global cost of analysis while serving all the nodes can be expressed by the following Social Choice Function (SCF):

$$SCF = S(C) = \min \sum_{k \epsilon N} c_k .(\sum_{i \in N} vt_k \ (C, i).B)$$

To minimize this SCF, incentives must be designed for encouraging each node in revealing its true cost of analysis value c and also election algorithm must be designed that can provably minimize the above SCF while not incurring too much of the performance overhead.

Leader Election Mechanism

The leader election mechanism is presented for truthfully electing the leader node.

Mechanism Design Background

The balance of IDS resource consumption problem can be modeled using mechanism design theory with an objective function that depends on the private information of

the players. In this case, the private information of the player is the cost of analysis which depends on the player's energy level. The main goal of using mechanism design is to address this problem by: 1) designing incentives for players (nodes) to provide truthful information about their preferences over different outcomes and 2) computing the optimal system-wide solution.

A mechanism design model consists of n agents where each agent $i \in \{1, \dots, n\}$ has a private information $\theta_i \in \theta_i$ known as the agent's type. And also, it defines a set of strategies A_i for each agent i. The agent can choose any strategy $a_i \in A_i$ to input in the mechanism. According to the inputs $(a_i,\dots,a_n)$ of all the agents, the mechanism calculates an output $o= o(a_1,\dots,a_n)$ payment vector $p = (p_1, \dots, p_n)$, where $p_i = p_i(a_1, \dots ,a_n)$. The preference of each agent from the output is calculated by a valuation function $v_i (\theta_i , o)$. The utility of a node is calculated as $u_i = p_i - v_i(\theta_i , o)$.The utility is the combination of output measured by valuation function and the payment it receives from the mechanism.

Mechanism Model

The following components are formulated:

1. Cost of analysis function: It is needed by the nodes to compute the valuation function.
2. Reputation system: It is needed to show how:
    a. Incentives are used once they are granted.
    b. Misbehaving nodes are catched and punished.

3. Payment design: It is needed to design the amount of incentives that will be given to the nodes based on VCG.

Cost of Analysis Function

In the design of the cost of analysis function, the two problems arise: First, the energy level is considered as private and sensitive information and should not be disclosed publicly. Second, if the cost of analysis function is designed only in terms of nodes' energy level, then the nodes with the low energy level will not be able to contribute and increase their reputation values. To solve the above problems, the cost of analysis function is designed with the following two properties: *Fairness* and *Privacy*. The former is to allow nodes with initially less resources to contribute and serve as leaders in order to increase their reputation. On the other hand, the latter is needed to avoid the malicious use of the resources level, which is considered as the most sensitive information. To avoid such attacks and provide fairness, the cost of analysis is designed based on the reputation value, the expected number of time slots that a node wants to stay alive in a cluster, and energy level.

Reputation System Model

Reputation system is used to: a) motivate nodes to behave normally and b) punish the misbehaving nodes. And also is can be used to determine whom to trust. To motivate the nodes in behaving normally in every election round, relate the cluster's services to nodes' reputation. This will create a competition environment that motivates the

nodes to behave normally by saying the truth. To enforce the mechanism, a punishment system is needed to prevent nodes from behaving selfishly after the election. Misbehaving nodes are punished by decreasing their reputation and consequently are excluded from the cluster services if the reputation is less than a predefined threshold. The abstract model of the reputation system is shown below:

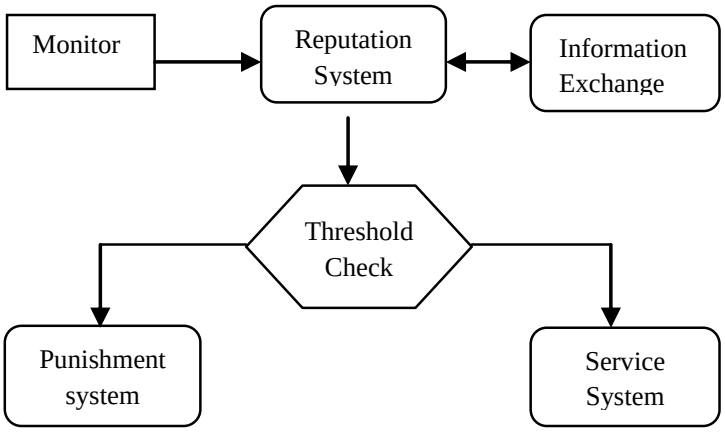

Figure : Reputation System Model

The system has the following components:

- Monitor or watchdog: It is used to monitor the behavior of the elected leader. To reduce the overall resource consumption, a set of node is elected randomly known as checker, to perform the monitoring process. The selected checkers mirror a small portion of the computation done by the leader, so the checker can tell whether the leader is actually carrying out its duty
- Information Exchange: It include two types of information sharing:
    - The exchange of reputation with other node in other clusters.
    - To reduce the false positive rate, the checkers will exchange information about the behavior of the leader to make decision about the leader behavior

- Reputation system: It is defined in the form of a table that contains the ID of other nodes and their respective reputation R. The node that has the highest reputation can be considered as the most trusted node and is given priority in the cluster's services.
- Threshold check: It has two main purposes:
    - To verify whether nodes' reputation is greater than a predefined threshold. If the result is true than node' services are offered according to nodes reputation.
    - To verify whether a leader's behavior exceed a predefined misbehaving threshold. According to the result the punishment system is called.

- Service System: In order to motivate the nodes to participate in every election round, the amount of detection service provided to each node is based on the nodes' reputation. Only limited services can be offered by the system. Packets of highly reputed node are always forwarded. If the source node has an unacceptably low reputation then its packets will have less priority. So, in every round node will try to increase their reputation by becoming the leader in order to increase their services.
- Punishment System: To improve the performance and reduce the false positive rate of checkers in catching and punishing a misbehaving leader, a cooperative game-theoretical model has been formulated to efficiently catch and punish misbehaving leader with low false positive rate.

CILE Design

In CILE, each node must be monitored by a leader node that will analyze the packets for other ordinary nodes. Based on the cost of analysis, nodes will cooperate to elect a set of leader nodes that will be able to analyze the traffic across the whole network and handle the monitoring process. This increases the efficiency and balances the resource consumption of an ID in the network. The mechanism provides payments to the elected leaders for serving others (i.e., offering the detection service). The payment is based on a per-packet price that depends on the number of votes the elected nodes get. The nodes that do not get any vote from others will not receive any payment. The payment is in the form of reputations, which are then used to allocate the leader's sampling budget for each node.

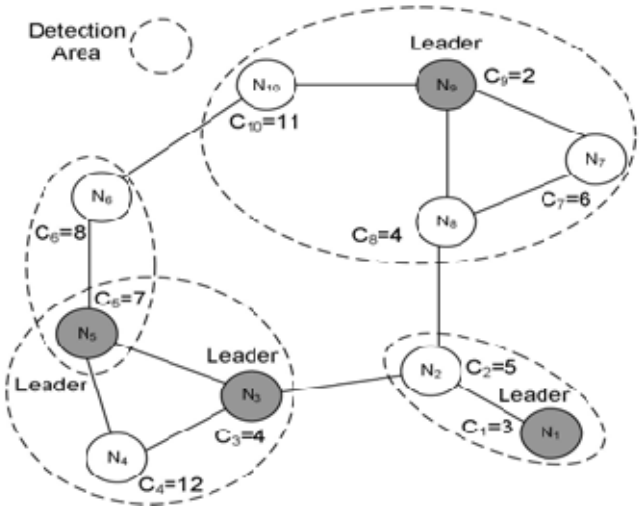

Figure : An example of leader election

CDLE Design

In CDLE, the whole network is divided into a set of clusters where a set of 1-hop neighbor nodes forms a cluster. Here, use the scheme of to cluster the nodes into 1-

hop clusters. Each cluster then independently elects a leader among all the nodes to handle the monitoring process based on nodes' analysis cost. Main objective is to find the most cost-efficient set of leaders that handle the detection process for the whole network. Like CILE, CDLE provides payment to the elected node and the payment is based on a per-packet price that depends on the number of votes the elected node gets. Finally, selfish nodes might misbehave after election, which motivates us to select random checkers to ensure a catch-and-punish scheme in order to motivate an elected node to be faithful during the detection process.

Security Analysis of the Mechanism

The main objective of the mechanism is to motivate selfish nodes and enforce them to behave normally during and after the election process.

Presence of Selfish Nodes

The misbehaving leader can be catch and punish by the checker. A caught misbehaving leader will be punished by receiving a negative payment. Thus, it discourages any elected node from not carrying out its responsibility. Now, it can conclude that the mechanism is truthful and it guarantees a fair election of the most cost-efficient leader.

Presence of Malicious Nodes

A malicious node can disrupt the election algorithm by claiming a fake low cost in order to be elected as a leader. Once elected, the node does not provide IDS services, which eases the job of intruders. To catch and punish a misbehaving leader who does not serve others after being elected, a decentralized catch and-punish mechanism are proposed using random checker nodes to monitor the behavior of the leader.

Although not repeated here, this scheme can certainly be applied here to the malicious nodes by catching and excluding them from the network. Due to the presence of checkers, a malicious node has no incentive to become a leader since it will be caught and punished by the checkers. After a leader is caught misbehaving, it will be punished by receiving a negative reputation and is consequently excluded from future services of the cluster. Thus, the mechanism is still valid even in the presence of a malicious node.

Leader Election Algorithm

For running the election mechanism a leader election algorithm that helps to elect the most cost-efficient leaders is proposed with less performance overhead compared to the network flooding model. The needed messages are devised to establish the election mechanism taking into consideration cheating and presence of malicious nodes. Moreover, the addition and removal of nodes is considered to/from the network due to mobility reasons. Finally, the performance overhead is considered during the design of the given algorithm where computation, communication, and storage overhead are derived.

Objectives and Assumptions

To design the leader election algorithm, the following requirements are needed: 1) To protect all the nodes in a network, every node should be monitored by a leader and 2) to balance the resource consumption of IDS service, the overall cost of analysis for protecting the whole network is minimized.

Leader Election

For starting a new election, the election algorithm uses four types of messages. *Hello*, used by every node to initiate the election process; *Begin-Election*, used to announce the cost of a node; *Vote*, sent by every node to elect a leader; and *Acknowledge*, sent by the leader to broadcast its payment, and also as a confirmation of its leadership. For describing the algorithm, we use the following notation:

- service-table (k): The list of all ordinary nodes, those voted for the leader node k.
- reputation-table (k): The reputation table of node k. Each node keeps the record of reputation of all other nodes.
- neighbors (k): The set of node k's neighbors.
- leadernode (k): The ID of node k's leader. If node k is running its own IDS, then the variable contains k.
- leader (k): A boolean variable that sets to TRUE if node k is a leader and FALSE otherwise.

Initially, each node k starts the election procedure by broadcasting a Hello message to all the nodes that are 1 hop from node k and starts a timer T1.

Algorithm 1 (Executed by every node)
/* On receiving Hello, all nodes reply with their cost */
1. if (received Hello from all neighbors) then
2. Send Begin-Election (IDk, costk);
3. else if (neighbors (k)=ø) then
4. Launch IDS.
5. end if.

On expiration of T1, each node k checks whether it has received all the hash values from its neighbors. Nodes from whom the Hello messages have not received are excluded from the election.

Algorithm 2 (Executed by every node)
/* Each node votes for one node among the neighbors */
1. if ($\forall n \in neighbor, \exists\ i \in n(k) c_i \leq c_n$) then
2. Send Vote (IDk, IDi, $cost_{j \neq i}$);
3. leadernode (k): = i;
4. end if.

On expiration of T2, the node k compares the hash value of Hello to the value received by the Begin- Election to verify the cost of analysis for all the nodes. Then, node k calculates the least-cost value among its neighbors and sends Vote for node i as in Algorithm 2. The Vote message contains the ID_k of the source node, the ID_i of the proposed leader, and second least cost among the neighbors of the source node$cost_{j\neq 1}$.

Algorithm 3 (Executed by Elected leader node)
/* Send Acknowledge message to the neighbors nodes*/
1. *Leader* (i):= TRUE
2. Compute Payment, Pi;
3. updateservice_table (i);
4. updatereputation_table(i);
5. Acknowledge = P_i + all the votes;
6. Send Acknowledge (i);
7. Launch IDS.

On expiration of T3, the elected node i calculates its payment using and sends an Acknowledge message to all the serving nodes as in Algorithm 3. The Acknowledge message contains the payment and all the votes the leader received.

Adding a New Node

Including a new node to the IDS service, four messages are needed: *Hello, Join, status* and *Acknowledge. Hello* is used to announce its presence in the network. After receiving the Hello message, all the neighbors reply a *Status* message. On receiving the Status messages from the neighbors, the new node send *Join* to the leader node. After getting the *Join* message, the leader node adds the new node to its service list and divides its budget according to nodes reputation.

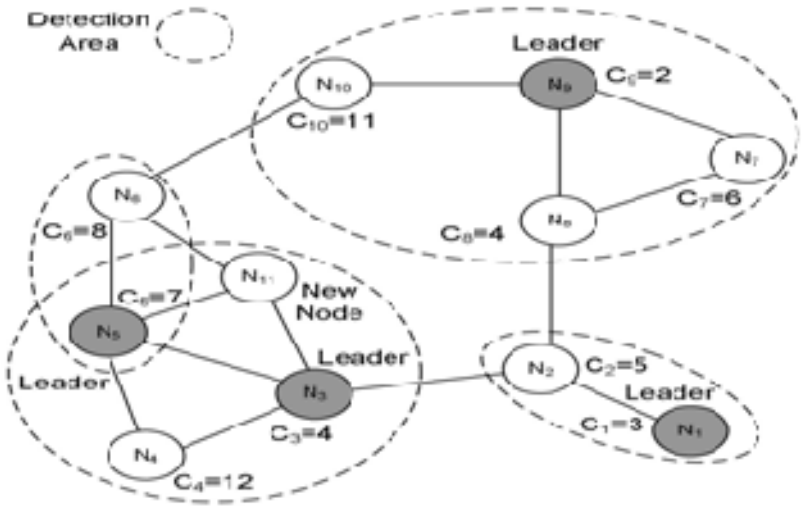

Figure : MANET after adding a new node

Removing a Node

When a node is disconnected from the network the neighbor nodes have to reconfigure the network. Whenever a node dies, its neighbors are aware of it. At first, a Dead (n) message is circulated to all neighbors to confirm the removal of node n. On receiving the Dead (n) message, the neighbors node k checks whether node n is its leader node or not. If node n is the leader node of node k, then node k announce a new election and updates its reputation table.

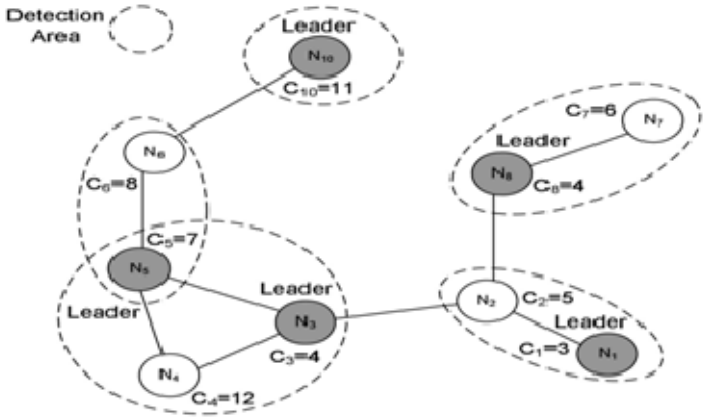

Figure : MANET after removing a node.

Modules

There are 5 modules in the project:

1. Network model.
2. Detection of selfish node.
3. Leader Election Mechanism.
4. Security Analysis.
5. Performance evaluation.

The following are the screenshots:

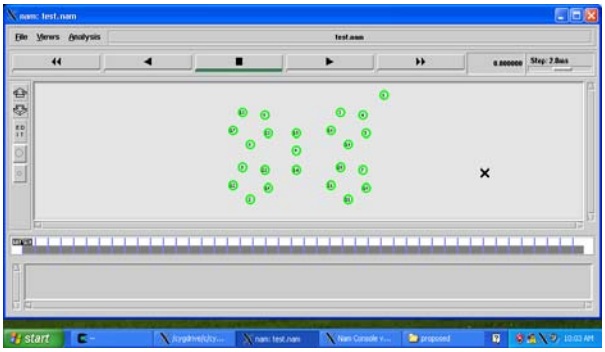

Figure : Network Model

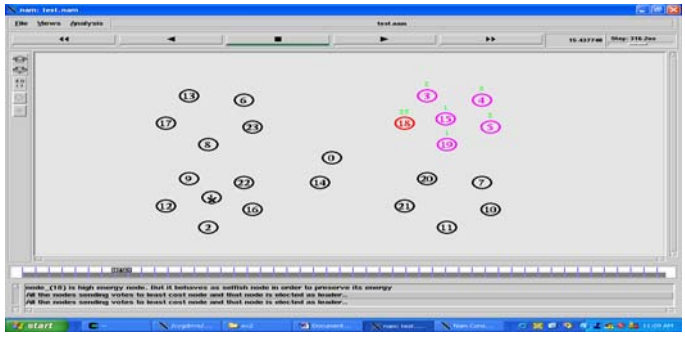

Figure : Detection of Selfish node in CLUSTER 1

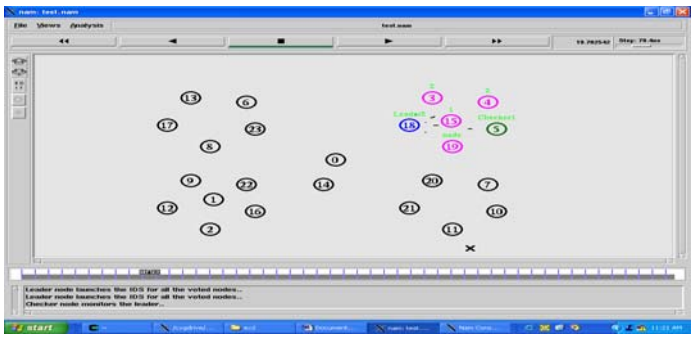

Figure : Selfish node becomes leader

An MANET composed of 24 nodes labeled from N0 to N23. These nodes are located in four regions where 6 nodes belong to one cluster and have limited resources level. Assume that each node has different energy level, which is considered as private information. Here, node 18 is detected as selfish node and by providing incentives from the neighboring node this node is elected as leader and served the IDS for the neighboring node. Likewise, same process is repeated for each region.

Conclusion

The model is able to prolong the lifetime and balance the overall resource consumption among all the nodes in the network. The motivated nodes truthfully elect the most cost-efficient nodes that handle the detection duty on behalf of others. And reputation is computed using the well-known VCG mechanism by which truth-telling is the dominant strategy. The sum of the elected leader is globally optimal and also analyzed the performance of the mechanism in the presence of selfish and malicious nodes. Two algorithms are used to implement the mechanism i.e., CDLE and CILE. CDLE requires nodes to by cluster before running the election mechanism whereas CILE does not require any preclustering. And also it is able to minimize the percentage of leaders, single node clusters, maximum cluster size and increase average cluster size.

References

[1] "A Survey on Intrusion Detection in Mobile Ad Hoc Networks, "Wireless/Mobile Network Security, Springer, 2006. T. Anantvalee and J. Wu.

[2] "Performance analysis of clustering protocol in MANET". Sharmila John Francis and Elijah Blessing Rajsingh.2008.

[3] " SCAM :Scenario based clustering algorithm for MANET" V.S. Anitha and M.P. Sebastian, 2009.

[4] H. Otrok, N. Mohammed, L. Wang, M. Debbabi, and P. Bhattacharya, "A Game-Theoretic Intrusion Detection Model for Mobile Ad-Hoc Networks," J. Computer Comm., vol. 31, no. 4, pp. 708-721, 2008.

[5] "Trust Based Clustering and Secure Routing Scheme or MANET".Pushpita Chatterjee. 2009.

[6] "TACA:A Topology Adaptive Clustering Algorithm for MANET". Suchismita Chinara and Santanu Kumar Rath.

[7] "Leader Election Modes of the Service Distributed Protocol for Ad Hoc Networks". Mohamed Handy and Birjitta Konig-Ries.