Intrusion Detection In Manets Using Secure Leader Election

Yasmeen Shaikh, V. K. Parvati
Lecturer, Department of Computer Science and Engineering, VDRIT, Haliyal
HOD, Dept. of Information Science and Engineering, SDMCET, Dharwad

ABSTRACT

In this paper, we study leader election in the presence of selfish nodes for intrusion detection in mobile ad hoc networks (MANETs). To balance the resource consumption among all nodes and prolong the lifetime of a MANET, nodes with the most remaining resources should be elected as the leaders. However, there are two main obstacles in achieving this goal. First, without incentives for serving others, a node might behave selfishly by lying about its remaining resources and avoiding being elected. Second, electing an optimal collection of leaders to minimize the overall resource consumption may incur a prohibitive performance overhead, if such an election requires flooding the network. To address the issue of selfish nodes, we present a solution based on mechanism design theory. More specifically, the solution provides nodes with incentives in the form of reputations to encourage nodes in honestly participating in the election process. The amount of incentives is based on the Vickrey, Clarke, and Groves (VCG) model to ensure truth-telling to be the dominant strategy for any node. To address the optimal election issue, we propose a election algorithm that can lead to globally optimal election results with a low cost. We address these issues by assuming cluster of nodes. Finally, we justify the effectiveness of the proposed scheme through extensive experiments.

KEYWORDS *: Leader election, intrusion detection systems, mechanism design and MANET security.*

I. INTRODUCTION

The Mobile Ad hoc Networks (MANET) have no fixed chokepoints/bottlenecks where Intrusion Detection Systems (IDSs) can be deployed [3], [7]. Hence, a node may need to run its own IDS [14], [1] and cooperate with others to ensure security [15], [26]. This is very inefficient in terms of resource consumption since mobile nodes are energy-limited. To overcome this problem, a common approach is to divide the MANET into a set of one-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 [16] or based on the connectivity [19]. Both approaches aim to reduce the overall resource consumption of IDSs in the network. However, we notice that nodes usually have different remaining resources at any given time, which should be taken into account by an election scheme. Unfortunately, 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. Moreover, even when all nodes can truthfully reveal their resource levels, it remains a challenging issue to elect an optimal collection of leaders to balance the overall resource consumption without flooding the network.

A Motivating Example

Figure 1.illustrates a MANET composed of ten nodes labeled from N1 to N10. These nodes are located in 5 one-hop clusters where nodes N5 and N9 belong to more than one cluster and have limited resources level. We assume that each node has different energy level, which is considered as private information. At this point, electing nodes N5 and N9 as leaders is clearly not desirable since losing them will cause a partition in the network and nodes will not be able to communicate with each other. However, with the random election model [16], nodes N5 and N9 will have equal probability, compared to others, in being elected as leaders. The nodes N5 and N9 will definitely be elected under the connectivity index-based approach due to their connectivity

indices [19]. Moreover, a naive approach for electing nodes with the most remaining resources will also fail since nodes' energy level is considered as private information and nodes might reveal fake information if that increases their own benefits. Finally, if the nodes N2, N5 and N9 are selfish and elected as leaders using the above models, they will refuse to run their IDS for serving others. The consequences of such a refusal will lead normal nodes to launch their IDS and thus die faster.

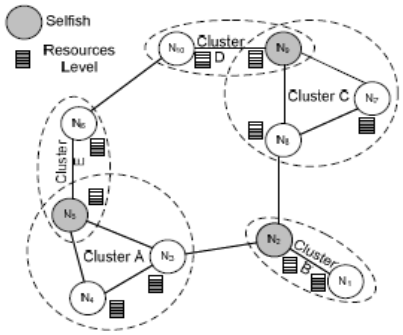

Figure 1: Example MANET

B Proposed system

In this paper, we propose a solution for balancing the resource consumption of IDSs among all nodes while preventing nodes from behaving selfishly. To address the selfish behavior, we design incentives in the form of reputation to encourage nodes to honestly participate in the election scheme by revealing their cost of analysis. The cost of analysis is designed to protect nodes' sensitive information (resources level) and ensure the contribution of every node on the election process (fairness). To motivate nodes in behaving normally in every election round, we relate the amount of detection service that each node is entitled to the nodes' reputation value. Besides, this reputation value can also be used to give routing priority and to build a trust environment.

The design of incentives is based on a classical mechanism design model, namely, Vickrey, Clarke, and Groves (VCG). The model guarantees that truth-telling is always the dominant strategy for every node during each election phase. On the other hand, to find the globally optimal cost-efficient leaders, a leader election algorithm is devised to handle the election process, taking into consideration the possibility of cheating and security flaws, such as replay attack. The algorithm decreases the percentage of leaders, single node clusters, maximum cluster size and increases average cluster size. The leaders are elected according to the received votes from the neighbor nodes, after the network is formulated into multiple clusters. Hence the leaders are elected in an optimal way in the sense that the resource consumption for serving as IDSs will be balanced among all nodes overtime. Finally, we justify the correctness of proposed methods through analysis and simulation. Empirical results indicate that our scheme can effectively improve the overall lifetime of a MANET.

The main contribution of this paper is a unified model that is able to:

1. Balance the IDS resource consumptions among all nodes by electing the most cost-efficient leaders.
2. Motivate selfish nodes to reveal their truthful resources level.

II. PROBLEM STATEMENT

We consider a MANET where each node has an IDS and a unique identity. To achieve the goal of electing the most cost efficient nodes as leaders in the presence of selfish and malicious nodes, the following challenges arise: 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 nodes might behave normally during the election but then deviate from normal behavior by not offering the IDS service to their voted nodes.

In our model, we consider MANET 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 = {c1, c2, . . . , cn} where n is the number of nodes in N. We denote the election process as a function vtk(C, i) where vtk(C, i) = 1 if a node i votes for a node k; vtk(C, i) = 0, otherwise. We assume that each elected leader allocates 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. This will motivate the nodes to cooperate in every election round that will be held on every time TELECT.

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):

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

Clearly, in order to minimize this SCF, the following must be achieved. First, we need to design incentives for encouraging each node in revealing its true cost of analysis value c, which will be addressed in Section III. Second, we need to design an election algorithm that can provably minimize the above SCF while not incurring too much of performance overhead. This will be addressed in Section V.

III. LEADER ELECTION MECHANISM

A The Mechanism Model

We treat the IDS resource consumption problem as a game where the N mobile nodes are the agents/players. Each node plays by revealing its own private information (cost of analysis) which is based on the node's type θi. The type θi is drawn from each player's available type set Θ_i={*Normal, Selfish*}. Each player selects his own strategy/type according to how much the node values the outcome. If the player's strategy is normal then the node reveals the true cost of analysis. We assume that each player i has a utility function [21]:

$u_i(\theta_i) = {}_{pi} - vi(\theta_i, \mathbf{o}(\theta i, \theta_{-i}))$ ..... (1)

where,

• θ_{-i} is the type of all the other nodes except i.

• vi is the valuation of player i of the output **oєO,** knowing that O is the set of possible outcomes. In our case, if the node is elected then v_i is the cost of analysis c_i. Otherwise v_i is 0 since the node will not be the leader and hence there will be no cost to run the IDS.

• p_i є R is the payment given by the mechanism to the elected node. Payment is given in the form of reputation. Nodes that are not elected receive no payment.

B Cost of Analysis Function

During the design of the cost of analysis function, the following two problems arise: First, the energy level is considered as private and sensitive information and should not be disclosed publicly. Such a disclosure of information can be used maliciously for attacking the node with the least resources level. 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, we design the cost of analysis function 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 to 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. Note that the expected number of slots and energy level are considered as the nodes' private information.

To achieve our goal, we assume that the nodes are divided into l energy classes with different energy levels. The lifetime of a node can be divided into time-slots. Each node i is associated with an energy level, denoted by E_i, and the number of expected alive slots is denoted by ${}_{nTi}$. Based on these requirements, each node i has a power factor $PF_i = E_i/nT_i$. We introduce the set of l − 1 thresholds P = {$\rho_1, \ldots, \rho_{l-1}$} to categorize the classes as in Equation 2.

$$CL = \begin{cases} cl_1 & if\ PF < \rho_1 \\ cl_i & if\ \rho_{i-1} \le PF < \rho_i;\ i \in [2, l-1] \\ cl_l & if\ PF \ge \rho_{l-1} \end{cases}$$

.....(2)

The reputation of node i is denoted by R_i. Every node has a sampling budget based on its reputation. The c_i notation represents the cost of analysis for a single packet and E_{ids} is used to express the energy needed to run the IDS for one time slot. The cost of analysis of each node can be calculated based on energy level. However, we considered energy level, expected lifetime and the present PS of node to calculate the cost of analysis. We can extend the cost of analysis function to more realistic settings by considering the computational level and cost of collecting and analyzing traffic. Our cost of-analysis function is formulated as follows:

$$c_i = \begin{cases} \infty & if\ (E_i < E_{ids}) \\ \frac{PS_i}{PF_i} = \frac{\frac{R_i}{\sum_{i=1}^{N} R_i} \times nT_i}{E_i} & otherwise \end{cases}$$

Where PS is percentage of sampling defined as follows,

$$PS_i = \frac{R_i}{\sum_{i=1}^{N} R_i}.$$

According to the above formulation, the nodes have an infinite cost of analysis if its remaining energy is less than the energy required to run the IDS for one time slot. This means that its remaining energy is too low to run the IDS for an entire time-slot. Otherwise, the cost of analysis is calculated through dividing the percentage of sampling by the power factor. The cost of analysis c is proportional to the percentage of sampling and is inversely proportional to the power factor. The rationale behind the definition of the function is the following. If the nodes have enough PS, they are not willing to loose their energy for running the IDS. On the other hand, if PF is larger, then the cost-of-analysis becomes smaller since the nodes have higher energy levels. In the rest of the paper, we will use cost and cost-of-analysis interchangeably. As the time goes by, the nodes belonging to lower energy class gains more budget while the budget of higher classes decreases. This justifies that our cost function is able to balance the energy of the nodes and gives a fair budget to all nodes.

C Reputation System Model

Before we design the payment, we need to show how the payment in the form of reputation can be used to: (1) Motivate nodes to behave normally and (2) punish the misbehaving nodes. Moreover, it can be used to determine whom to trust. To motivate the nodes in behaving normally in every election round, we 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 our 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. As an extension to our model, we can extend our reputation system to include different sources of information such as routing and key distribution with different assigned weights.

IV. IMPLEMENTATION

A Leader Election

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. (2) To balance the resource consumption of IDS service, the overall cost of analysis for protecting the whole network is minimized. In other words, every node has to be affiliated with the most cost efficient leader among its neighbors. Our algorithm is executed in each node taking into consideration the following assumptions about the nodes and the network architecture:

- Every node knows its (2-hop) neighbors, which is reasonable since nodes usually maintain a table about their neighbors for routing purposes.
- Loosely synchronized clocks are available between nodes.
- Each node has a key (public, private) pair for establishing a secure communication between nodes.

To start 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; *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 one hop from node k and starts a timer T1. This message contains the hash value of the node's cost of analysis and its unique identifier (ID)

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* message has not received are excluded from the election. On receiving the *Hello* from all neighbors, each node sends *Begin-Election* as in Algorithm 1, which contains the cost of analysis of the node and then starts timer T2. If node k is the only node in the network or it does not have any neighbors then it launches its own IDS.

Algorithm 2 (Executed by every node)

/* Each node votes for one node among the neighbors */

1. **if** ($\forall$ n _ neighbor(k), $\exists$ i _ n : $c_i \leq c_n$) **then**

2. send Vote(IDk, ID_i , $cost_j$_=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 IDk of the source node, the IDi of the proposed leader and second least cost among the neighbors of the source node costj_=i. Then node k sets node i as its leader in order to update later on its reputation. Note that the second least cost of analysis is needed by the leader node to calculate the payment. If nodek has the least cost among all its neighbors then it votes for itself and starts timer T3.

Algorithm 3 (Executed by Elected leader node)

/* Send Acknowledge message to the neighbor nodes */

1. Leader(i) := TRUE;

2. Compute Payment, P_i;

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 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. The leader then launches its IDS.

V. RESULTS AND DISCUSSIONS

A Simulation Environment

We simulate the scheme using Network Simulator 2(NS2). The main objective of our simulation result is to study the effect of node selection for IDS on the life of all nodes. To implement the model, we randomly assign 60 to 100 joules of energy to each node. We assume that the energy required for running the IDS for one time slot as 10 joules. We ignore the energy required to live and transmit packets to capture the silent aspect of the problem. We set the transmission radius of each node to 200 meters. Two nodes are considered as neighbors if their Euclidean distance is less than or equal to 200 meters. Besides, we deploy 20 nodes in an area of 500 × 500 square meters. It helps us to measure the performance of the nodes.

B Experimental Results

Nodes can behave selfishly before and after the election. A node shows selfishness before election by refusing to be a leader. On the other hand, selfishness after election is considered when nodes misbehave by not carrying out the detection service after being a leader. Both kinds of selfishness have serious impact on life of the normal nodes. As selfish nodes do not exhaust energy to run the IDS service, it will live longer than the normal nodes. In order to avoid this, election is based upon the reputation. Selfish nodes are punished with negative reputations. On other hand, node with highest reputation is elected as leader. Hence normal nodes will not launch IDS and will live longer. The nodes repetitively elect a set of leaders with maximum resources. Thus, we consider some nodes to be selfish and study their impact on our model. The experiment indicates that our model results in a higher percentage of alive nodes as shown in fig 2.

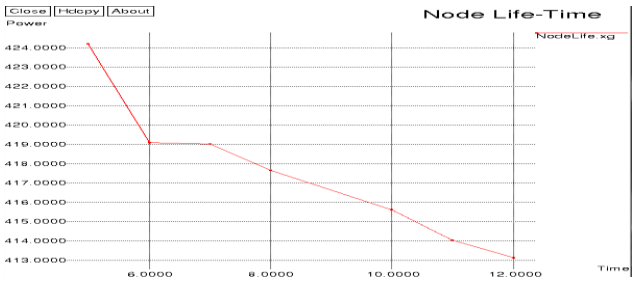

Figure 2. Node Life-Time

Our paper ensures that data can be transmitted in an easier, faster and secure way without intrusion. Leader election helps in identifying the intruders and hence data transmission rate is increased, thereby improving the throughput as shown in fig 3.

In our model, the node that has least cost of analysis becomes the leader. All nodes can keep a balance of their energy level with time. A leader has more energy hence capable of delivering large number of packets, thereby decreasing packet drop ratio and increasing packet delivery ratio as shown in fig 4 and 5 respectively.

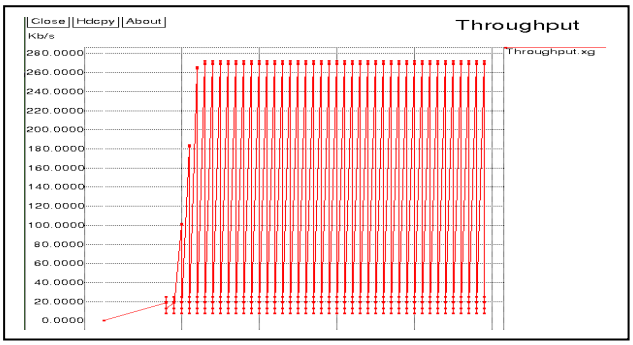

Fig 3 Throughput

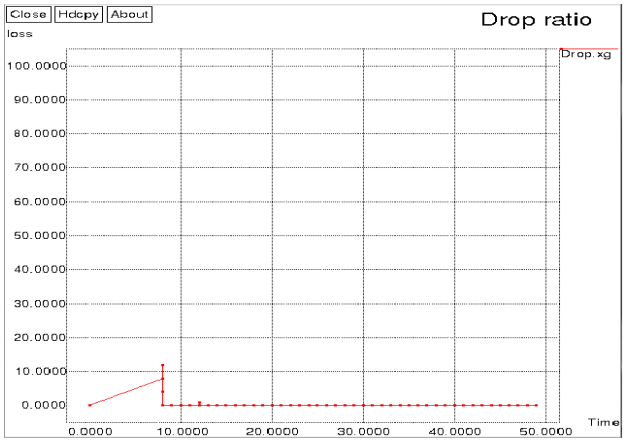

Figure 4 Packet drop ratio

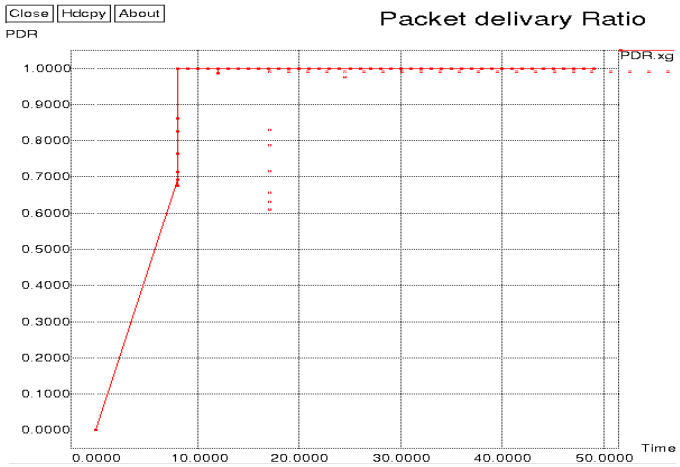

Figure 5 Packet Delivery ratio

VI. PERFORMANCE AND SECURITY ANALYSIS

A Performance

In this section, we analyze the performance overhead of our proposed leader algorithm. In summary, our algorithm has four steps. In the first 3 steps, all the nodes broadcast *Hello*, *Begin-Election* and *Vote* messages consecutively. In the last step, only the leader node sends an *Acknowledge* message to others.*1) Computation Overhead:* Each node i signs its messages sent in the first 3 steps. Also, each node verifies the messages it received in these steps. In the 4th step, the leader node signs the Acknowledge message and others verify. Hence each normal node signs 3 messages and verifies 3|Ngi| + 1 messages where Ngi is the number of neighboring nodes. On the other hand, the leader node signs 4 messages and verifies 3|Ngi| messages. Note that each node must find the least cost node which requires O(log(Ngi)). Therefore, each node approximately performs O(Ngi) verifications, O(1) signatures and O(log(Ngi)) to calculate the least cost node. Thus the computation overhead for each node is O(Ngi) + O(1) + O(log(Ngi)) ≈ O(Ngi). Since our algorithm involves more verification than signing, nodes can use the public key cryptosystem of [13] to verify a signature in 0.43s. Since leader election will take place after a certain interval, this computational overhead is tolerable.

2) Communication Overhead: Each node i broadcasts one message in the first 3 steps and only the leader node broadcasts a final *Acknowledge* message in the 4th step. Hence, the total communication overhead of our election algorithm is 3|Ngi|+ 1 ≈ O(Ngi), where |Ngi| is the number of neighboring nodes.

3) Storage Overhead: According to the algorithm, each node maintains a *reputation-table*, *neighbors* list and two variables: *Leadernode* and *leader*. The leader node keeps an extra *service-table*. Hence, each normal node needs |Ni| + |Ngi| + 2 storage and the leader node needs |Ni| + |Ngi| + |Vi| + 2. Knowing that |Ni| is the number of nodes in the network, |Vi| is the number of votes the leader node received where |Ni| > |Ngi| > |Vi|. Therefore, the total storage for each node is in the order of O(Ni).

B Security

The main objective of our mechanism is to motivate selfish nodes and enforce them to behave normally during and after the election process. Here, we analyze the election mechanism in the presence of selfish and malicious nodes.

1. Presence of Selfish Nodes

A selfish node i will deviate from our mechanism if doing so increases its utility, u_i. Here we consider two type of untruthful revelation, namely, node i might either *under-declare* or *over-declare* the true value c_i of its cost of analysis. Node i may under-declare its valuation function with a fake value $\hat{c}_i(\hat{c}_i < c_i)$. By under-declaring, node i pretends that it has a cheaper valuation function than reality. Since payments are designed based on VCG, playing by under-declaration will not help the node for two reasons. First, suppose the node i indeed has the lowest cost of analysis c_i, so it will win the election even by declaring its true value. In this case, reporting a lower value $\hat{c}_i$ will not benefit the node because the payment is calculated based on the second best price and does not depend on the value declared by node i. Therefore, the utility of node i remains the same because it will be the difference between the payment and the real value c_i. Second, suppose that the node it does not have the cheapest valuation function but tries to win the election by revealing a lower value $\hat{c}_i$. This will help the node i to win the election but it will also lead to a negative utility function u_i for node i, because the payment it receives will be less than the real cost of analysis. That is, the node i will have to work more than what it has paid for. On the other hand, the node i might over-declare its valuation by revealing a fake $\hat{c}_i$ ($\hat{c}_i > c_i$). Following such a strategy would never make a player happier in two cases. First, if the node i indeed has the cheapest valuation function, then following this strategy may prevent the node from being elected, and therefore it will lose the payment. On the other hand, if node i still wins, then its utility remains the same since the payment does not depend on the value it reports. Second, suppose the real valuation function ci of node i is not the lowest, then reporting a higher value will never help the node to win. Last but not least, the checkers are able to catch and punish the misbehaving leaders by mirroring a portion of its computation from time to time. A caught misbehaving leader will be punished by receiving a negative payment. Thus it discourages any elected node from not carrying out its responsibility. We can thus conclude that our mechanism is truthful and it guarantees a fair election of the most cost efficient leader.

2. Presence of Malicious Nodes

A malicious node can disrupt our 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. 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, our mechanism is still valid even in the presence of a malicious node.

VII. ADVANTAGES AND DISADVANTAGES

A. Advantages

- Ensures easier, secure and faster data transmission.
- Balanced resource consumption in the presence of selfish and malicious nodes.
- Decreases the percentage of leaders, single node clusters, maximum cluster size and increase average cluster size.
- Life time of nodes is increased as nodes with maximum resources are elected as leaders.
- Leaders are elected randomly based upon the reputation.

B. DISADVANTAGES

- If any new node is added or removed from the network, it is not possible to dynamically configure the network, which may lead to partitioning in the network.
- It is not possible to make a specified node as leader who would launch IDS, as leaders are elected randomly.

VIII. FUTURE SCOPE

In future work, we plan expanding our model to dynamically configure the network when any new node is created or when any existing nodes is removed from the network. We'll also plan to make a specified node as leader by manipulating the leader election mechanism.

IX. CONCLUSION

The unbalanced resource consumption of IDSs in MANET and the presence of selfish nodes have motivated us to propose an integrated solution for prolonging the lifetime of mobile nodes and for preventing the emergence of selfish nodes. The solution motivated nodes to truthfully elect the most cost efficient nodes that handle the detection duty on behalf of others. Moreover, the sum of the elected leaders is globally optimal. To achieve this goal, incentives are given in the form of reputations to motivate nodes in revealing truthfully their costs of analysis. Reputations are computed using the well known VCG mechanism by which truth-telling is the dominant strategy. We also analyzed the performance of the mechanisms in the presence of selfish and malicious nodes. To implement our mechanism, we devised an election algorithm with reasonable performance overheads. Simulation results showed that our model is able to prolong the lifetime and balance the overall resource consumptions among all the nodes in the network. Moreover, we are able to decrease the percentage of leaders, single node clusters, maximum cluster size and increase average cluster size. These properties allow us to improve the detection service through distributing the sampling budget over less number of nodes and reduce single nodes to launch their IDS.

REFERENCES

[1] T. Anantvalee and J. Wu. A survey on intrusion detection in mobile adhoc networks. Wireless/Mobile Network Security, 2006.

[2] L. Anderegg and S. Eidenbenz. Ad hoc-VCG: A truthful and cost efficient routing protocol for mobile ad hoc networks with selfish agents.
In proc. of the ACM International Conference on Mobile Computing and Networking (MobiCom), 2003.

[3] F. Anjum and P. Mouchtaris. Security for Wireless Ad Hoc Networks. John Wiley & Sons. Inc., USA, 2007.

[4] S. Basagni. Distributed and mobility-adaptive clustering for multimedia support in multi-hop wireless networks. In proc. of the IEEE International Vehicular Technology Conference (VTC), 1999.

[5] S. Basagni. Distributed clustering for ad hoc networks. In proc. of the IEEE International Symposium on Parallel Architectures, Algorithms, and Networks (ISPAN), 1999.

[6] M. Bechler, H. Hof, D. Kraft, F. Pahlke, and L. Wolf. A cluster based security architecture for ad hoc networks. In proc. of the IEEE INFOCOM, 2004.

[7] P. Brutch and C. Ko. Challenges in intrusion detection for wireless adhoc networks. In proc. of the IEEE Symposium on Applications and the Internet (SAINT) Workshop, 2003.

[8] S. Buchegger and J. L. Boudec. Performance analysis of the CONFIDANT protocol (cooperation of nodes - fairness in dynamic adhoc networks). In proc. of the ACM MOBIHOC, 2002.

[9] K. Chen and K. Nahrstedt. iPass: An incentive compatible auction scheme to enable packet forwarding service in MANET. In proc. of the International Conference on Distributed Computing Systems, 2004.

[10] 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 proc. of the IEEE Military Communications Conference (MILCOM), 2001.

[11] J. Feigenbaum, C. Papadimitriou, R. Sami, and S. Shenker. A BGP based mechanism for lowest-cost routing. In proc. of the ACM symposium on Principles of distributed computing (PODC), 2002.

[12] J. Feigenbaum and S. Shenker. Distributed algorithmic mechanism design: Recent results and future directions. In proc. of the AMM International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIALM), 2002.

[13] N. Gura, A. Patel, A. Wander, H. Eberle, and S. C. Shantz. Comparing elliptic curve cryptography and RSA on 8-bit CPUs. In proc. of the Cryptographic Hardware and Embedded Systems (CHES), 2004.

[14] S. Gwalani, K. Srinivasan, G. Vigna, E. M. Beding-Royer, and R. Kemmerer. An intrusion detection tool for AODV-based ad hoc wireless networks. In proc. of the IEEE Computer Security Applications Conference (CSAC), 2004.

[15] Y. Hu, A. Perrig, and D. B. Johnson. Ariadne: A secure on-demand routing protocol for ad hoc networks. In proc. of the ACM International Conference on Mobile Computing and Networking (MOBICOM), 2002.

[16] Y. Huang and W. Lee. A cooperative intrusion detection system for adhoc networks. In proc. of the ACM Workshop on Security of Ad Hoc and Sensor Networks, 2003.

[17] L. Hurwicz and S. Reiter. Designing Economic Mechanisms. Cambridge University Press, 1st edition, 2008.

[18] J.Green and J.Laffont. Incentives in Public Decision-Making. Springer Netherlands, USA, 1996.

[19] O. Kachirski and R. Guha. Efficient intrusion detection using multiple sensors in wireless ad hoc networks. In proc. of the IEEE Hawaii International Conference on System Sciences (HICSS), 2003.

[20] P. Krishna, N. H. Vaidya, M. Chatterjee, and D. K. Pradhan. A cluster based approach for routing in dynamic networks. In proc. of the ACM SIGCOMM Computer Communication Review, 1997.

[21] A. Mas-Colell, M. Whinston, and J. Green. Microeconomic Theory. Oxford University Press, New York, 1995.

[22] P. Michiardi and R. Molva. Analysis of coalition formation and cooperation strategies in mobile adhoc networks. Journal of Ad hoc Networks, 3(2):193 – 219, 2005.

[23] A. Mishra, K. Nadkarni, and A. Patcha. Intrusion detection in wireless ad hoc networks. IEEE Wireless Communications, 11(1):48 – 60, 2004.

[24] N. Mohammed, H. Otrok, L. Wang, M. Debbabi, and P. Bhattacharya. A mechanism design-based multi-leader election scheme for intrusion detection in manet. In proc. of the IEEE Wireless Communications & Networking Conference (WCNC), 2008.

[25] P. Morris. Introduction to Game Theory. Springer, 1st edition, 1994.

[26] P. Ning and K. Sun. How to misuse AODV: A case study of insider attacks against mobile ad-hoc routing protocols. In proc. of the IEEE Information Assurance Workshop, 2003.

[27] N. Nisan and A. Ronen. Algorithmic mechanism design. In Games and Economic Behavior, pages 129–140, 1999.

[28] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani. Algorithmic Game Theory. Cambridge University Press, 1st edition, 2007.

[29] H. Otrok, N. Mohammed, L. Wang, M. Debbabi, and P. Bhattacharya. A game-theoretic intrusion detection model for mobile ad-hoc networks. Journal of Computer Communications, 31(4):708 – 721, 2008.

[30] A. Perrig, R. Canetti, D. Tygar, and D. Song. The TESLA broadcast authentication protocol. RSA Cryptobytes, 5(2):2 – 13, 2002.

[31] J. Shneidman and D. Parkes. Specification faithfulness in networks with rational nodes. In proc. of the ACM Symposium on Principles of Distributed Computing, 2004.

[32] K. Sun, P. Peng, P. Ning, and C. Wang. Secure distributed cluster formation in wireless sensor networks. In proc. of the IEEE Computer Security Applications Conference (ACSAC), 2006.

[33] S. Vasudevan, B. DeCleene, N. Immerman, J. Kurose, and D. Towsley. Leader election algorithms for wireless ad hoc networks. In proc. of the IEEE DARPA Information Survivability Conference and Exposition (DISCEX III), 2003.

[34] S. Vasudevan, J. Kurose, and D. Towsley. Design and analysis of a leader election algorithm for mobile ad hoc networks. In proc. of the IEEE International Conference on Network Protocols (ICNP), 2004.

[35] Y. Zhang and W. Lee. Intrusion detection in wireless ad-hoc networks. In proc. of the ACM International Conference on Mobile Computing and Networking (MobiCom), 2000.