Available online at www.sciencedirect.com
ScienceDirect
Computer Communications 31 (2008) 708–721
computer communications
www.elsevier.com/locate/comcom

A game-theoretic intrusion detection model for mobile ad hoc networks

Hadi Otrok *, Noman Mohammed, Lingyu Wang, Mourad Debbabi, Prabir Bhattacharya

Computer Security Laboratory, Concordia Institute for Information Systems Engineering, Concordia University, Montreal (QC), Canada

Available online 22 October 2007

Abstract

In this paper, we address the problem of increasing the effectiveness of an intrusion detection system (IDS) for a cluster of nodes in ad hoc networks. To reduce the performance overhead of the IDS, a leader node is usually elected to handle the intrusion detection service on behalf of the whole cluster. However, most current solutions elect a leader randomly without considering the resource level of nodes. Such a solution will cause nodes with less remaining resources to die faster, reducing the overall lifetime of the cluster. It is also vulnerable to *selfish nodes* who do not provide services to others while at the same time benefiting from such services. Our experiments show that the presence of selfish nodes can significantly reduce the effectiveness of an IDS because less packets are inspected over time. To increase the effectiveness of an IDS in MANET, we propose a unified framework that is able to: (1) Balance the resource consumption among all the nodes and thus increase the overall lifetime of a cluster by electing truthfully and efficiently the most cost-efficient node known as leader-IDS. A mechanism is designed using Vickrey, Clarke, and Groves (VCG) to achieve the desired goal. (2) Catch and punish a misbehaving leader through checkers that monitor the behavior of the leader. A cooperative game-theoretic model is proposed to analyze the interaction among checkers to reduce the false-positive rate. A multi-stage catch mechanism is also introduced to reduce the performance overhead of checkers. (3) Maximize the probability of detection for an elected leader to effectively execute the detection service. This is achieved by formulating a zero-sum non-cooperative game between the leader and intruder. We solve the game by finding the Bayesian Nash Equilibrium where the leader's optimal detection strategy is determined. Finally, empirical results are provided to support our solutions.

Keywords: MANET security; Intrusion detection system; Cooperative and non-cooperative game theory; Mechanism design

1. Introduction

The open nature of Mobile Ad hoc Networks (MANET) leads to new security problems that have recently attracted significant attentions [1,2]. The cooperation among nodes is critical to MANET in many perspectives, including intrusion detection, as evidenced by recent studies [1,3]. Cooperative intrusion detection systems were proposed as a second line of defense [1,2,4,5] to detect and thwart security threats. Current cooperative IDS solutions are costly with respect to resource consumptions, which directly affect the effectiveness of an IDS over a long period of time. Electing randomly a leader node to handle the detection process on behalf of the whole cluster (that is, a group of one-hop neighbor nodes who can overhear each other) is recently proposed as a solution for reducing the performance overhead of an IDS [4]. However, the solution does not consider the potential selfish behavior of nodes. Nodes may misbehave since they are not willing to consume their resources for serving others and at the same time they want to benefit from others' services.

The second limitation of the random model [4] is that it causes normal nodes with less remaining resources to die faster because they will be the only nodes performing the detection service. This fact reduces the overall lifetime of a cluster. Non-cooperative behavior (that is, *selfishness*) is first studied under the cooperation enforcement discipline where the seriousness of selfish behaviors has been revealed

* Corresponding author.
E-mail addresses: h_otrok@ciise.concordia.ca (H. Otrok), no_moham@ciise.concordia.ca (N. Mohammed), wang@ciise.concordia.ca (L. Wang), debbabi@ciise.concordia.ca (M. Debbabi), prabir@ciise.concordia.ca (P. Bhattacharya).

0140-3664/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2007.10.024

and solutions for solving the selfishness problem in routing already exist [6–8]. On the other hand, the selfishness problem naturally exists in MANET, although the problem has not been widely studied to our best knowledge. In particular, selfish nodes may deviate from telling the truth about their private information, such as remaining resources, during an election of the leader node if that would increase their benefits. These limitations of existing solutions motivate us to propose a unified model that takes into consideration the selfishness issue during the election and after. Our model is based on well studied economic solutions in *Game theory* and its subfield *mechanism design*.

In this paper, we balance the resource consumption of an IDS among all the nodes in a cluster by electing the most cost-efficient node as the leader for detecting intrusions. We propose a unified solution that can deal with potential selfish nodes during and after the election of a leader and can also maximize the leader's detection probability. First, during the election, incentives are given in the form of reputation to motivate nodes to cooperate. Incentives are calculated based on the truth-telling mechanism Vickrey, Clarke, and Groves (VCG). Moreover, to motivate nodes to participate in every election, a leader will analyze packets for each node in the cluster according to its reputation value. Hence, each node will be assigned a sampling budget out of the leader's total-budget. Second, to prevent an elected leader from misbehaving, we design a catch-and-punish mechanism to monitor the behavior of a leader with *checker* nodes. To reduce the false-positive rate of checkers, a cooperative decision game is formulated where checkers are the players. We also introduce a multi-stage catch mechanism for reducing the performance overhead of checkers. Third, to maximize the probability of detection, we model and solve a zero-sum non-cooperative game where the leader and intruder are players. For the leader to optimally distribute the node's budget among all the node's incoming-links, we formulate a zero-sum non-cooperative game with incomplete information about the intruder. We solve the game by finding the Bayesian Nash equilibrium where the optimal plan for distributing the nodes' sampling budget is derived.

In summary, the main contribution of the paper is an integrated framework that can:

- Increase the overall lifetime of an IDS in MANET by truthfully electing the most cost-efficient node to handle the detection process on behalf of the whole cluster. This is achieved by balancing the resource consumption for the detection service among all the nodes in a cluster.
- Encourage selfish nodes to truthfully reveal their cost of analysis during a leader election. This is achieved by a reputation system based on the truth-telling mechanism Vickrey, Clarke, and Groves (VCG) and by binding the reputation of a node to the amount of services the node is entitled to.
- Encourage an elected leader to carry out its responsibility of intrusion detection. This is achieved with a decentralized catch-and-punish mechanism using random *checker* nodes.
- Reduce the false-positive rate of checkers in catching the misbehaving leader. This is achieved by formulating a cooperative decision game among the checkers and by a multi-stage catch mechanism.
- Maximize the probability of detection by optimally distributing the node's sampling budget among all its incoming-links. This is achieved by modeling a zero-sum non-cooperative game between the leader and intruder with incomplete information about the intruder.

The rest of the paper is organized as follows. Section 2 presents the related work. Section 3 discusses a motivating example to build intuitions about a series of issues addressed in this paper. Section 4 describes the leader election mechanism. The election process is illustrated through a concrete example and justified through analysis. Section 5 presents the catch-and-punish mechanism. We illustrate our cooperative game theoretic model with an example. Section 6 presents a zero-sum non-cooperative game between the intruder and defender. The formulation of the game is given followed by its solution and an example. Empirical results are presented in Section 7. Finally, Section 8 draws a conclusion with a future work.

2. Related work

In this section, we review related work on intrusion detection systems for MANET, the application of mechanism design in infrastructure and ad hoc networks, and the application of game theory to intrusion detection.

2.1. Intrusion detection systems in MANET

The difference between wired infrastructure networks and mobile ad hoc networks [1] has motivated researchers to study IDSs that can handle new security challenges such as securing routing protocols. A cooperative intrusion detection model is proposed in [5] where every node participates in running its IDS in order to collect and identify possible intrusions. If an anomaly is detected with a weak evidence then a global detection process is initiated for further investigation about the intrusion through a secure channel. An extension of this model is proposed in [4] where a set of intrusions can be identified with their corresponding sources. Moreover, the authors addresses the problem of run-time resource constraints of IDSs through modeling a repeatable and random leader election framework. An elected leader is responsible for detecting intrusions for a predefined period of time. The proposed models, however, do not consider the selfish or malicious behavior of nodes which could reduce the probability of detection. Watchdog and Pathrater are proposed in [9] to

improve the throughput in MANET in the presence of misbehaving nodes. Watchdog's goal is to identify compromised nodes in the network, whereas Pathrater's goal is to prevent routing protocols from using misbehaving nodes.

Most existing models do not include any punishment procedure that can enforce nodes to behave normally. Hence, misbehaving nodes can continue operating in the network and benefit from other normal nodes' services. Due to the negative impact of misbehaving nodes, researchers have developed different solutions to cope with such a problem. Proposed cooperation enforcement models are based on threshold cryptography [7], micro-payments [6] and reputation [8]. CORE [8] is a cooperative enforcement mechanism based on monitoring and reputation systems. The goal of this model is to detect selfish nodes and to enforce them to cooperate. Each node keeps track of others' cooperation using reputation as the cooperation metric. CORE ensures that misbehaving nodes are punished by gradually stopping communication services and by providing incentives, in the form of reputation, for nodes to cooperate. Reputation is usually given based on data monitored by local nodes and information provided by other nodes involved in each operation. In our model, the amount of reputation is computed based on mechanism design to motivate nodes to truthfully reveal their private information.

2.2. Mechanism design application

Mechanism design is a sub-field of microeconomics and game theory [10]. It uses game theory tools to achieve a desired goal. The main difference between game theory and mechanism design is that the former is used to study what could happen when independent players act selfishly, whereas mechanism design allows us to define the game in such a way that the outcome of the game, known as the social choice function (SCF), will be played by independent players according to the rules set by the mechanism designer. Mechanism design has been extensively used in microeconomics for modeling solutions for various economical problems such as auctions. It has been used in computer science by Nisan and Ronen [11] for solving least cost path and task scheduling problems using algorithmic mechanism design. Distributed mechanism design based on VCG is first introduced in [12] to compute the lowest cost routes for all source-destination pairs and payments for transit nodes on all the routes. It is a direct extension of Border Gateway Protocol (BGP), which causes modest increases in routing table size and convergence time.

In [12], the author compute payments, instead of the social choice function, in a distributed manner (notice that the computation of a social choice function actually needs more computational efforts than computing payments). An extension of this work is given in [13] where the authors considered consequences that may appear after the computation phase is finished. To achieve their objectives, the authors introduce the concept of checker nodes that mirror what the elected node is computing through recomputing by itself. Currently in MANET, mechanism design is mainly used for routing purposes. In [14], the authors present a truthful ad hoc-VCG mechanism to find the most cost-efficient route in the presence of selfish nodes. In [6], the authors provide an incentive compatible auction scheme to enable packet forwarding service in MANET using VCG. A continuous auction process runs to determine who should obtain how much of the bandwidth and at what price. Incentives are in the form of monetary rewards.

Leader-election in IDS is significantly different from the above problems. Our unique contributions, which are not present in the work of routing, include but are not limited to: The design of payment scheme, binding payments to services, having checkers to watch dishonest leaders. We address selfishness in IDS leader election, which is a real problem that has not been addressed by previous approaches, and it's our belief that the main contribution of this paper does not lies in how we apply VCG, but where we apply it.

2.3. Game theory

Game theory [15] has been successfully applied to many disciplines including economics, political science, and computer science. Game theory usually considers a multi-player decision problem where multiple players with different objectives can compete and interact with each other. Game theory classifies games into two categorizes: Non-cooperative and cooperative. Non-cooperative games are games with two or more players that are competing with each other. On the other hand, cooperative games are games with multi-players cooperating with each other in order to achieve the greatest possible total benefits. To predict the optimal strategy used by intruders to attack a network, the authors of [16] model a non-cooperative game-theoretic model to analyze the interaction between intruders and the IDS in a wired infrastructure network. They solve the problem using a zero-sum non-cooperative game with complete information about the intruder. In complete information game, the type, strategy spaces, and payoff functions of both players are known. In [17], the authors aim at demonstrating the suitability of game theory for development of various decision, analysis, and control algorithms in intrusion detection. They address some of the fundamental network security tradeoffs, and give illustrative examples in different platforms. They propose two different schemes based on game theoretic techniques and consider a generic model of distributed IDSs equipped with a network of sensors. Bayesian Nash is used in [18] to analyze the interaction between the intruder and defender in static and dynamic scenarios. The authors provide a hybrid detection approach with lightweight and heavyweight monitoring systems.

These existing studies clearly show that game theory is a strong candidate for providing the much-needed mathematical framework for analyzing the interaction between IDSs and intruders. To the best of our knowledge, our work is among the first efforts on improving the performance of intrusion detection in MANET. Our solution is unique in many perspectives. First, we elect the most cost-efficient nodes using mechanism design for providing intrusion detection service where the payments are given in the form of reputation and calculated based on VCG. Our solution can thus motivate selfish nodes to behave normally during the election process. Second, we ensure that a misbehaving leader will be caught and punished through a catch-and-punish scheme. To reduce the false positive rate of such a scheme, a cooperative game-theoretic model is used the checker's decision. Third, a zero-sum non-cooperative game based on Bayesian Nash equilibrium is used to model the interaction between the leader and intruder, taking into consideration that the precise location of intruders is typically unknown. The solution of such a game helps the leader to optimally distribute node's sampling budget over its incoming-links so the probability of detection will be maximized.

3. Motivating example

In this section, we present a concrete example of intrusion detection in MANET in order to build intuitions and motivate further discussions. We illustrate a series of issues through the example, which will be addressed one by one in the following sections. Fig. 1 illustrates two clusters in a MANET. All nodes in a cluster are one-hop away so they can overhear each other. Between the two clusters, we are more concerned with cluster A, which has ten nodes numbered from N_1 to N_{10}. The node N_5 is a member of both clusters. We assume the nodes N_2 and N_8 are selfish in that they do not want to consume their energy for serving others. An adversary node I is outside the two clusters and it is sending malicious packets to cluster A aiming to disrupt its normal operations. Assuming each node in cluster A has an IDS that can detect malicious packets sent by the adversary node. However, running IDSs at all the nodes in cluster A would be too expensive in terms of energy consumption. Therefore, the nodes in cluster A decide to repeatedly elect a leader to carry out the intrusion detection service on behalf of the whole cluster.

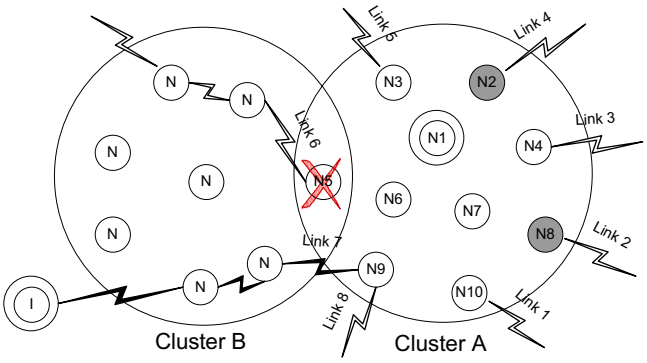

Fig. 1. An example of two clusters in a MANET.

First consider the random election model in [4], which picks a random node as the leader without taking into account their remaining resources. Therefore, all the nodes in cluster A have the same probability of being elected. Suppose the remaining energy of each node in cluster A is shown in Table 1. Electing the node N_5 as the leader is apparently not desirable in this case. First, N_5 has the least remaining energy, which means it may die quickly while carrying out the intrusion detection service as a leader. Moreover, N_5 happens to be the only node between the two clusters, which means nodes in different clusters will not be able to communicate with each other after N_5 dies (this may as well be the objective of the adversary). Another issue with the random election model is that it does not consider the presence of selfish nodes. In this case, if the selfish nodes N_2 and N_8 are elected as the leader, then the whole cluster will be subject to attacks since they would not carry out any intrusion detection service.

In Section 4, we shall present a solution to the following two issues, that is the unbalanced resource consumption and the presence of selfish nodes. First, instead of electing a leader randomly, we select the most cost-efficient node (a node with the most remaining energy in the simple case of Fig. 1) as the leader. Therefore, node N_7 will first be elected as the leader, and then N_3 will likely be the next leader, and so on. Under this election strategy, nodes with more resources will carry out more responsibility, leading to a balanced resource consumption over time. However, two issues arise due to the presence of selfish nodes. First, a selfish node, such as N_2 or N_8, will try to avoid being elected by declaring a fake (lower) value of remaining energy. Second, such a selfish node will not carry out the responsibility of detection, even if it is elected as the leader. In Section 4, we devise a reputation system based on mechanism design to motivate selfish nodes to reveal their true cost of analysis (remaining energy in this case). To address the second issue, we incorporate a catch-and-punish scheme. Randomly selected checker nodes will monitor the behavior of the leader to ensure it is carrying out its responsibility. Selfish nodes are encouraged to work normally, because the consequence of being caught misbehaving will be devastating.

Two more issues arise with the introduction of checker nodes. First, a checker node may make mistakes in identifying a misbehaving checker. This is partly due to the fact that a checker only mirrors a small portion of work done by the leader, and such observation will be affected by

Table 1
Remaining energy of nodes in cluster A

Nodes	N_1	N_2	N_3	N_4	N_5	N_6	N_7	N_8	N_9	N_{10}
Energy	50	45	80	77	10	25	94	34	23	69

channel interference, collision, and other uncertainty inherent to a MANET. Such false positives may cause an honest leader, such as node N_5 in Fig. 1, to be excluded from the cluster, if a leader is determined as misbehaving as soon as a checker says so. In Section 5, we propose a solution to reduce the false positive rate by requiring checkers to cooperate in order to determine a misbehaving leader. Second, although the cooperation of a large number of checkers will apparently improve the accuracy in catching a misbehaving leader, this also implies significant performance overhead of checkers. In Section 5, we address this issue through a multi-stage catching scheme, which increases the number of checkers dynamically upon discovering weak evidences of a misbehaving leader.

Finally, an issue arises when considering how an elected leader should divide the corresponding node's sampling budget among node's incoming-links to increase the probability of detection. For example, in Fig. 1 suppose node N_1 is elected as the leader after a certain time. The leader can only analyze a finite number of packets for each node in the cluster due to its limited resources. That is, node N_1 has a sampling budget for each node. Unlike wired network, a MANET does not have any gateway node where sampling can be done. Therefore, sampling will be handled at node's incoming-links. In Fig. 1, there are totally 8 links to cluster A among which 2 links are for node N_9. An adversary node can inject a malicious packet using link 7 to attack node N_9. It is clearly desirable for the leader node N_1 to allocate the node's sampling budget to link 7 so the probability of detection will be increased. However, the fact that link 7 will carry malicious packets is not known, and can only be estimated based on historical information. On the other hand, the adversary node will also try its best to avoid sending malicious packets through a link with the most sampling budget in order to avoid detection. To address this issue, Section 6 models a zero-sum non-cooperative game between the leader and adversary to derive the optimal sampling strategy for the leader in order to improve its probability of detection.

4. A truthful and efficient leader election mechanism

In this section, we first give our assumptions, definitions, and notations. We present the proposed mechanism based on standard mechanism design notations followed by an illustrative example and the analysis of the mechanism.

4.1. Assumptions, definitions, and notations

4.1.1. Assumptions

In this paper, we model the MANET as an undirected graph $G = (N,L)$ where N is the set of mobile nodes and L is the set of bidirectional links. The network is divided into different clusters where every cluster has a set of nodes $n \in N$ and a set of links $l \in L$. One-hop neighbor nodes form a cluster and nodes might belong to more than one cluster. It is assumed that each node has an IDS and a unique identity. Moreover, the neighbor nodes can always overhear each other using an omnidirectional antenna. We consider the presence of selfish nodes that are not willing to participate in detecting network intrusions in order not to consume resources such as battery, memory and CPU time. We assume that every node will increase the reputation rate of a cooperative node since reputation is used to track whom to trust. We consider the presence of outsider intruders who do not belong to the cluster but want to disrupt the cluster's normal operation. We do not consider the collusion among nodes in misbehaving since nodes in an ad hoc network usually belong to different organizations. We assume nodes are rational in the sense that they want to maximize the amount of services they are entitled to receive from the leader. Therefore, nodes are motivated to participate in the election process in every round because a node's reputation is bound to the amount of services it can receive.

4.1.2. Definitions

We define *selfish node* as an economically rational node whose objective is to maximize its benefits (payoffs). Therefore, *incentives* must be given to nodes to motivate them in cooperating. Incentives are modeled in terms of the reputation of node. *Reputation* is used to decide whom to trust and motivate nodes to reveal truthfully their private information about their cost of analysis. The *cost* function aggregates the cost of energy used to analyze traffic, cost of collecting traffic, current battery and computational (CPU and memory) level. In our model, the default value of the reputation at the cluster formation time is a fixed value R_0. A misbehaving node is punished by decreasing its reputation and consequently withholds cluster's services when the reputation is less than the predefined threshold TH.

4.1.3. Notations

$R^t_{n_i}(n_j)$ represents the reputation calculated by node n_i at time t for node n_j behavior. On the other hand, the reputation of node i is denoted by R_i. Every node has a sampling budget based on its reputation. This is indicated by percentage of sampling, $PS_i = \frac{R_i}{\sum_{i=1}^{N} R_i}$. The C notation is used to express the cost of analysis for 100 packets/sec and E_{ids} is used to express the energy needed to run the IDS for that period of time. Moreover, $LeaderIDS_{cluster^A}$ is the leader of $cluster^A$ that will be responsible for protecting the nodes with good reputation rate from intrusions. Finally, $cluster^A_{-n_i}$ represents the nodes in $cluster^A$ except n_i.

4.2. Cost of analysis function

In our cost of analysis function, we consider *fairness* where a node with less resources can have the opportunity to be elected, after a period of time, as a leader. Hence, its reputation is increased. We consider N nodes in a cluster, which are divided into k energy classes with different energy

levels. Let n_i be the number of nodes in class i ($i = 1, \ldots, k$). We assume the lifetime of nodes can be divided into time-slots. Each node in class i is associated with the same energy level, denoted by E_i, and an alive slot, denoted by nT_i. Based on these notations, we say each node in class i has a power factor $PF_i = E_i/nT_i$. We introduce a set of $k-1$ thresholds $P = \{\rho_1, \ldots, \rho_{k-1}\}$ to categorize the classes as follows:

$$CL = \begin{cases} cl_1 & \text{if } PF < \rho_1 \\ cl_i & \text{if } \rho_{i-1} \leqslant PF < \rho_i; \quad i \in [2, k-1] \\ cl_k & \text{if } PF \geqslant \rho_{k-1} \end{cases}$$

The cost of analysis of each node can simply be calculated based on its energy level, but we choose to also consider the expected lifetime and the present PS of a node in calculating the cost of analysis. We believe that our idea of calculating cost of analysis can further be extended to more realistic settings involving computational level, cost of collecting and analyzing traffic, and other relevant factors. The following shows our cost of analysis function.

If ($E_i < E_{ids}$)

$C_i = \infty$

Else

$$C_i = \frac{PS_i}{PF_i} = \frac{\frac{R_i}{\sum_{i=1}^{N} R_i} \times nT_i}{E_i}$$

Selfish nodes want to maximize its PS using less energy. To achieve that, nodes need to calculate their cost efficiently. According to the given function, nodes have an infinite cost of analysis if the remaining energy is less than the energy required to run the IDS for one time-slot, because the remaining energy is too low to run the IDS for the entire time-slot. Otherwise, nodes will calculate their C according to the given function. The C value is calculated through dividing the percentage of sampling by the power factor. The cost C is proportional to the percentage of sampling since it is inversely proportional to the power factor. The rationale behind the function is the following. If the nodes have enough PS, they are not willing to loss their energy for running the IDS. On the other hand, if PF is larger, the cost becomes smaller since nodes have higher energy level.

Next we show the effect of our cost function over PS through an example. We consider 20 nodes divided equally to 4 energy classes where nodes in class 4 have more resources. In Table 2, we give the respective PS for nodes of each class against time.

Table 2
PS calculated using the cost function

PS (percentage of sampling) (s)	$Class_4$ (%)	$Class_3$ (%)	$Class_2$ (%)	$Class_1$ (%)
After 200	55	20	15	10
After 600	45	24	18	13
After 1000	40	26	20	14

In Table 2, initially nodes belonging to lower energy level have a small budget. As time goes by, the nodes belonging to lower energy class gains more budget while the budget of higher classes gradually decreases. This shows that our cost function is able to balance the energy of nodes and gives a fair budget to all the nodes even if they have initially a very low energy level.

4.3. Our mechanism model

Here, we define our model using standard mechanism design notation [19]. We treat our problem as a game where mobile nodes are the players. Each node holds a private information θ_i about its preferences (θ_i is known as the type of player i). The type θ_i is drawn from each player's available type set $\Theta_i = \{Normal, Selfish\}$, it describes how each player values all possible outcomes. Moreover, we define S_i as the available set of strategies for player i. In our model, we are using direct revelation mechanism [19] in which $\Theta_i = S_i$. We assume that player i has a quasilinear utility function [10]:

$$u_i(\theta_i, o(\theta_i, \theta_{-i})) = p_i - v_i(\theta_i, o(\theta_i, \theta_{-i})) \tag{1}$$

where,

- θ_{-i} is the type of all nodes in a cluster except i.
- v_i is the valuation of player i to the output $o \in O$, knowing that O is the set of possible outcomes. In our case, v_i is the cost of analysis C_i that will be revealed by i after selecting its type θ_i.
- $p_i \in \Re$ is the payment given by the mechanism to a selected node. Payment is given in the form of reputation.

Note that, u_i is what the player usually seeks to maximize. It reflects the amount of benefits gained by player i if he follows a specific type θ_i. Players might deviate from revealing the truthful valuation of the cost of analysis if that could lead to better payoff. Therefore, our mechanism must be strategy-proof where truth-telling is a dominant strategy. To play the game, every node selects a type θ_i and a valuation function $v_i(\theta_i, o)$ where the set of valuations is the input of our mechanism. For each input vector, the mechanism calculates its corresponding output $o = o(\theta_1, \ldots, \theta_n)$ and a payment vector $p = (p_1, \ldots, p_n)$. Payments are used to motivate players to behave in accordance with the mechanism goals. Our desired goal (i.e., the most efficient IDS) is expressed by the Social Choice Function (SCF) that is the objective for our mechanism to implement:

$$\text{SCF} = \min \sum_{i \in n} v_i(\theta_i, o(\theta_i, \theta_{-i})) \tag{2}$$

To achieve the desired goal, payments are computed using VCG mechanism where truth-telling is proved to be dominant.

$$p_i = R_i = \sum_{j \in -n_i} v_j(\theta_j, o(\theta_i, \theta_{-i})) \quad (3)$$

where R_i is the reputation and $\sum_{j \in -n_i} v_j(\theta_j, o(\theta_i, \theta_{-i}))$ denotes, according to the standard notation in mechanism design, the best price excluding n_i (i.e., second best price).

4.4. Leader election mechanism

Our MANET is modeled as a set of clusters where a set of one-hop neighbor nodes forms a cluster. To defend against intrusions, each node has an IDS that is able to detect intrusions targeting communication protocols. Based on the cost of analysis C, nodes will cooperate to elect a leader node that will handle the monitoring process. This increases the efficiency of an IDS in a cluster. To balance the resource consumption of an IDS in a cluster, we design an efficient and truthful cooperative intrusion detection model in the presence of selfish nodes. Our model objective is to find the most cost-efficient node that handles the detection process. Without loss of generality, we will find the leader node at $cluster^A$ where nodes are asked to reveal truthfully their cost of analysis by motivating them through incentives. Incentives are given in the form of reputations and computed based on VCG mechanism, where truth telling is the dominant strategy. Reputations are needed to decide whom to trust among the nodes in a cluster. In our model, nodes are asked to directly reveal their utility function to compute the SCF, which is the least cost of analysis value. Payments are computed using VCG. The SCF is computed in a distributed manner where all the nodes decide about the leader node. This guarantees that the same leader is elected by all.

4.4.1. IDS leader election

To form the MANET into clusters, we use the cluster formation algorithm [20]. Every node is aware of its neighbor nodes. Here, we describe our distributed election protocol where every node executes the following steps. Note that step one is executed by any node to start the election process.

(1) $n_i \rightarrow cluster^A_{-n_i}$: *Begin-Election* $(ID_{n_i}, H(ID_{n_i}, C_i, TS_i), T_1)$
(2) $n_i \rightarrow cluster^A_{-n_i}$: *Election* (ID_{n_i}, C_i, TS_i)
(3) If $LeaderIDS \neq n_i$;
    $n_i \rightarrow LeaderIDS$: *Done-Election*.
    $LeaderIDS \rightarrow n_i$: *Confirm Leadership*.
    $n_i \rightarrow LeaderIDS$: *Deliver payment* $= R^t_{n_i}(n_j)$.
(4) Else after T_2;
    $n_i \rightarrow cluster^A_{-n_i}$: *Confirm Leadership*.
    $cluster^A_{-n_i} \rightarrow n_i$: *Deliver payment* $= R^t_{n_j}(n_i)$.

In the first step, node $n_i \in [1,n]$ sends a *Begin-Election* message to all nodes in $cluster^A$. This message includes the identity ID_{n_i} of n_i and the hash value $H()$ of the identity ID_{n_i}, the cost of analyzing the traffic C_i and the time-stamp TS_i. This time stamp helps to avoid replay attacks. The hash function is used to avoid nodes from cheating and delivering a fake C_i as shown after. The time T_1 is used to identify the election start time. After all the nodes exchange the *Begin-Election* message within time T_1.

In the second step, node n_i sends the *Election* message which includes its identity ID_i, the cost of analyzing the traffic C_i and the time stamp TS_i. Nodes that did not contribute in sending the *Begin-Election* message will be excluded from cluster's services. On receiving the *Election* messages from $cluster^A_{-n_i}$, node n_i verifies each received message with its corresponding hash value that has been sent in *Begin-Election* message. After the verification is accomplished, each node computes the SCF, which is the minimum valuation[1] of cost of analysis as in Eq. (2).

In the third step, if the leader is different from n_i then it sends a *Done-Election* message to inform the leader that he has been elected. In this case, elected leader forward a *Confirm Leadership* message that indicates its acceptance of leadership. Then, n_i calculates the payment $(R^t_{n_i}(n_j))$ using the VCG mechanism as in Eq. (3) and sends a copy of the payment back (i.e., n_i increases its reputation table for n_j by p_j where n_j is the elected node). Note that a node needs the copy of payment to calculate its reputation and compare it to the threshold TH to avoid punishment.

In the fourth step, if the leader is n_i, it sets a timer T_2 then starts verifying the origin of all the *Done-Election* messages. If T_2 expires without receiving all the *Done-Election* messages, then nodes who did not participate are excluded from the cluster's services. Last but not least, once the leader has been chosen by all the nodes, all contributing nodes will be added to the protected list. To ensure fairness, we enforce the reelection procedure every time T_{ELECT}. If the cluster did not change after T_{ELECT} expires, we omit the formation step and start leader election. Moreover, we enforce the reelection procedure either when the leader-IDS misbehave or quit from the cluster before T_{ELECT} expires.

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. Note that a random election ensures fairness. In our model, the selected checker is assumed to be cooperative since the benefit of the intrusion detection service dominates resources consumption. This is because we assume that the elected checkers mirror a portion of the computation done at the elected node which have a marginal effect on resource consumption. Catched misbehaving leader, for example n_j, is punished by receiving a negative payment $-R^t_{n_i}(n_j)$

[1] If more than one node has the same most cost-efficient of analysis, then we assume that the node with the highest reputation is elected.

4.5. Illustrative example

We consider a cluster of 10 nodes where 20% of the nodes are selfish (but rational) and the presence of an outsider intruder that does not belong to the cluster but targets it. Nodes have to cooperate to truthfully elect the most cost-efficient node that handles the detection responsibility. This prolongs the life of the IDS in a cluster and therefore the percentage of packet analysis is improved. Since our model is repeatable, we present the election example at the 10th round. We assume that the reputation at the 9th round is given in the first row of Table 3. where N_5 is the leader-IDS at this round. The detection service is given to the nodes proportional to their reputations. Hence, the IDS budget (100 packets/s) is distributed as in the second row of Table 3, where the detection service is offered according to nodes' sampling percentage.

To elect a new leader in the 10th round, every node computes its corresponding cost of analysis as in the third row of Table 3 using the equation in Section 4.2. According to its type (selfish or normal), nodes reveal the cost of analysis following the election protocol steps. Using Eq. (2) in Section 4.3, node N_9 has the most cost-efficient value. Nodes calculate the payment for elected node as in Eq. (3) in Section 4.3, which is equal to 12 U of reputation. All the nodes increase the reputation of the elected node N_9 by +12. Nodes send a copy of their payment to elected node in order to confirm its leadership. Using Eq. (1) in Section 4.3, the utility of N_9 is $12 - 10 = 2$, which represents the benefits gained by the node. In other words, the leader is doing the job with cost of 10 U of reputation while receiving the payment of 12 U, which is used for increasing the detection service later on. Note that if a leader misbehaves after election then its reputation will go down by −12.

4.6. Mechanism analysis

In this section, we show that our model satisfies the truthfulness and cost-efficiency properties in the presence of selfish nodes. This is done by showing that our mechanism is strategy-proof where truth-telling is the dominant strategy. Hence, the cost-efficiency property is satisfied since truthfulness is guarantied. A strategy is dominated by another strategy if the second strategy is at least as good as the other one regardless of the other players' strategy. It is expressed as follows:

$$p_i - v_i(\theta_i^*, o(\theta_i^*, \theta_{-i}^*)) = u_i^* \geqslant u_i = p_i - v_i(\theta_i, o(\theta_i, \theta_{-i}^*))$$

where θ_i^* is non-selfishness and θ_i is selfishness. For node i's utility to be maximized, it has to receive payment p_i. Therefore, payment must be designed in a way to eliminate cheating and selfishness. Here, we show that our payment is designed according to VCG mechanism where deviating from truth-telling will not make node i happier (here, happiness means maximizing its utility function u_i). We consider two cases for untruthful revelation where node i might *under-declare* or *over-declare* its v_i function (i.e., cost of analysis).

Node i under-declares its valuation function by revealing $\hat{v}_i$ where $\hat{v}_i < v_i$. By under-declaring this value, node i pretends that it has a cheaper valuation function than reality, which could help it to win the election and become a leader. Since payments are designed based on Eq. (3), playing by under-declaration will not help the node for two reasons. Firstly, node i might win the election even if it declares fake valuation $\hat{v}_i$ since in reality, using truth valuation v_i, it has the cheapest valuation (i.e., the most cost-efficient cost of analysis). Using this strategy would never benefit the node since payment depends on the second best price and never changes. Therefore, its utility remains the same since it is computed with respect to the real valuation v_i. Secondly, if in reality node i does not have the cheapest valuation but tries to win the election by revealing a fake valuation $\hat{v}_i$. This would lead to a negative utility function u_i since the payment received is less than the real valuation (i.e., cost of analysis). Thus, node will receive the payment in the form of reputation but the work it does is more than the received payment.

On the other hand, node i might over-declare its valuation by revealing a fake $\hat{v}_i$, where $\hat{v}_i > v_i$. Following such strategy would never make a player happier for two reasons. Firstly, if in reality node i has the cheapest valuation then following this strategy leads the node not to be elected and therefore it will lose the payment. If the node with this fake $\hat{v}_i$ has been elected then its utility remains the same since payment would not change. Secondly, if the real valuation v_i of node i is not the cheapest then following this strategy would never lead to better solutions. Last but not least, checkers are able to catch and punish misbehaving elected leader by mirroring a portion of its computation every period of time. A catched leader is punished by computing the payment in negative. Finally, we can conclude that our mechanism is truthful and guarantee the election of the most cost-efficient node (IDS).

Table 3
Leader-IDS election example

Nodes	N_1	N_2	N_3	N_4	N_5	N_6	N_7	N_8	N_9	N_{10}
Reputation	12	14	10	8	13	6	9	16	1	11
Sampling	12%	14%	10%	8%	13%	6%	9%	16%	1%	11%
Cost of analysis	12	17	20	14	16	18	16	19	10	15

5. Cooperative catch and punish model

Due to un-control problems such as channel collision, the leader-IDS could not be able to monitor and analyze the traffic of some protected nodes for a specific period of time. Hence, a checker that is monitoring the behavior of the leader-IDS could report a misbehaving event and therefore the leader-IDS is punished and a new leader is

elected. This problem motivated us to propose a cooperative game theoretical model that is able to efficiently catch and punish misbehaving leader-IDS with less false positive rate. We propose the concept of detection-levels to be $DL = \{dl_1, \ldots, dl_k\}$ which enables us to respond better to misbehaving leader-IDS depending on which detection-level it belongs to. Hence, the percentage of checkers will vary with respect to the detection-level. Our catch and punish model is made up of k detection-levels, each level represents the severe behavior of the leader-IDS. We introduce the set of $k-1$ thresholds T to categorize the detection-levels where $T = \{t_1, \ldots, t_{k-1}\}$. Now, we introduce the aggregate function to be:

$$F(n) = \sum_{i \in n} R_i \times f(i) \tag{4}$$

where R_i is the reputation of checker i, n is the set of checkers and $f(i)$ is the catch function that takes a value between 0 and 5 according to the severe behavior of the leader-IDS. To decentralize the catch decision, $F(n)$ will be calculated by all the checkers after the secure exchange of $f(i)$. $F(n)$ sums up the catch-function of each checker i, in a cluster n, while considering the reputation of each checker. Now, we categorize the detection-levels as follows:

$$DL = \begin{cases} dl_1 & \text{if } F(n) < t_1 \\ dl_i & \text{if } t_{i-1} \leqslant F(n) < t_i; \quad i \in [2, k-1] \\ dl_k & \text{if } F(n) \geqslant t_{k-1} \end{cases} \tag{5}$$

Categorizing the misbehavior of leader-IDS into different levels help in catching the selfish leader-IDS with less false positive rate and reduce the performance overhead of the checkers. We can use statistically trained data to assign values to thresholds T and detection-levels DL in order to have better results with respect to catch and punish. Here, cooperative game theory is used to formally illustrate the problem.

5.1. Cooperative game theory

The design and analysis of our proposed model is done using cooperative game theory [21]. The l checkers will be modeled as a set N of l players in an *N-person* game with $N = \{N_1, \ldots, N_l\}$ [15]. We introduce a coalition in cooperative game theory to be:

$\Delta \subseteq N$ and $\forall x \in \Delta$.

In other words, we define a coalition to be a set of checkers, where each checker reports the behavior of elected leader-IDS. Therefore, each checker in Δ reports a risk in the cluster. Let δ be the number of checkers in a coalition in a cluster. We use the aggregate function over Δ,

$$\sum_{x \in \Delta} R_x \times f(x)$$

to assign the behavior of leader-IDS to its equivalent detection-level dl_j.

Assigning an anticipated marginal contribution to each player (checker) in the game with respect to a uniform distribution over the set of all permutations on the set of players will be presented by Shapley value [21]. To find the contribution of checker N_i in coalition Δ, we consider all the different permutations for the checkers, Π_Δ, in the coalition. Then we calculate the difference between the function including all checkers in the permutation before checker N_i, including N_i, and the function of all checkers prior to N_i, excluding N_i. We define $P_\pi^{N_i}$ to be the set of checkers before the node N_i in the permutation $\pi \in \Pi_\Delta$. Then taking the average of all these differences, we get the marginal contribution of checker N_i in coalition Δ. In other words, the contribution would be the following:

$$\phi_{N_i}(\Delta) = \frac{1}{\delta!} \sum_{\pi \in \Pi_\Delta} F(P_\pi^{N_i} \cup \{N_i\}) - F(P_\pi^{N_i}) \tag{6}$$

Replacing $F(P_\pi^{N_i} \cup \{N_i\})$ by $\sum_{x \in P_\pi^{N_i} \cup \{N_i\}} r_x \times f(x)$ and $F(P_\pi^{N_i})$ by $\sum_{x \in P_\pi^{N_i}} r_x \times f(x)$.

The Equation is reduced to the following:

$$\phi_{N_i}(\Delta) = \frac{1}{\delta!} \sum_{\pi \in \Pi_\Delta} F(\{N_i\}) \tag{7}$$

which simply reduces to the following:

$$\phi_{N_i}(\Delta) = F(\{N_i\}) \tag{8}$$

Now, to calculate the marginal contribution (Shapley value) of checker N_i, in the cluster, we should take the average of this value over all possible coalitions, which is:

$$\phi_{N_i} = \frac{1}{\gamma} \sum_{N_i \in \Delta, \Delta \in N} F(\{N_i\}) \tag{9}$$

where γ is the number of possible coalitions in the cluster. Coalitions with enough power to impose a decision collectively are called winning coalitions. Here, a coalition is a winning coalition if the aggregation of their values can change the detection-level. Therefore, the value of a coalition corresponding to a level is either zero or one. It is one in the case of a winning coalition and zero otherwise. Thus, the effect of checker N_i on detection-level dl_i can be written as $\frac{1}{\gamma}|\Delta'|$, where Δ' is the set of the winning coalitions: i.e. $\sum_{N_i \in \Delta'} F(\Delta') \geqslant t_i$.

The Shapley value of checker N_i indicates the relative contribution for a given threshold t_i. Therefore, we can tune the values of the thresholds using statistical data in order to reduce the false positives. In each detection-level, we have a different response varies from adding more checkers to dropping the leader-IDS and electing new one. This illustrates the importance of analyzing the relative contribution of a checker to decide the detection-level. The following example illustrates our model.

5.2. Illustrative example

Consider five checkers in a cluster of 20 nodes. A checker noticed that the elected leader-IDS is not doing

the job of detection. Note that the leader could misbehave either due to a channel collision or due to selfish behavior. To efficiently catch and punish the misbehavior leader-IDS and reduce false positive, a cooperative decision is made among the checkers in the cluster.

As an example, let us consider that we have four detection-levels $DL = \{dl_1,dl_2,dl_3,dl_4\}$. Knowing that dl_1 indicates normal behavior, dl_2 and dl_3 mean more checkers have to be added to monitor the behavior of node while dl_4 means a new leader-IDS has to be elected. The threshold set $T = \{2,4,6\}$, the detection-levels DL are classified as follows: $dl_1 < 2$, $2 \leqslant cl_2 < 4$, $4 \leqslant dl_3 < 6$, $dl_4 \geqslant 6$, the reputation of nodes $R_1 = 0.5$, $R_2 = 0.8$, $R_3 = 0.2$, $R_4 = 0.5$, $R_5 = 0.6$, and $f(1) = 3, f(2) = 4, f(3) = 1, f(4) = 2, f(5) = 5$. Using Eq. (9), we calculate in the following table the participation of each checker in catching the misbehaving leader-IDS.

Checker nodes	N_1	N_2	N_3	N_4	N_5
Contribution value	19.2	40.96	2.56	12.8	38.4

Moreover, using $\frac{1}{\gamma}|\Delta'|$, we evaluate the contribution of each checker on each detection-level as shown in Fig. 2. In this Figure for example, the contribution of N_3, with other checkers in the coalition, changes the detection-level from dl_3 to dl_4, which means that the leader-IDS is not behaving properly and a new leader has to be elected. Now, we use $\sum_{N_i \in \Delta'} F(\Delta') \geqslant t_i$ to find the winning coalitions. For example, the winning coalitions that decide the change from detection-level dl_3 to dl_4, where N_3 belongs to these coalitions is the following set: $\{\{N_2,N_3,N_5\}, \{N_1,N_2,N_3,N_5\}, \{N_2,N_3,N_4,N_5\}, \{N_1,N_2,N_3,N_4,N_5\}\}$.

6. Intruder-defender game theoretical model

In this section, we play a game between the leader-IDS and the intruder. The objective of the intruder is to inject a malicious packet to attack node $i \in N$ by selecting a routing path to i. In order to detect the intrusion, the leader-IDS samples packets on the target's incoming-links according to the target's sampling-budget B_i which is equivalent to reputation R_i. We define the traffic flowing on link e as f_e and the sampling rate as s_e. The expected probability of detecting a malicious packet on link e is $p_e = \frac{s_e}{\mu_{t_{k+1}}(\theta|a_e)f_e}$ bounded by the sampling budget B_i and having the sampling constraint $\sum_{e \in E} s_e \leqslant B_i$. Knowing that $\mu_{t_{k+1}}(\theta|a_e) > 0$ is the belief evaluation function. It informs the leader-IDS about the belief that link e has been used maliciously or not at time t_k by observing the occurring action a_e. Here, we assume that the belief of the IDS regarding the sender is equivalent to the belief of the link that the sender is using to communicate with the target node i. Therefore $\mu_{t_{k+1}}(\theta|a_e)$ is calculated by observing the occurring action a_e at time t_k using the following Bayes rule:

$$\mu_{t_{k+1}}(\theta|a_e) = \frac{\mu_{t_k}(\theta)P(a_{e_{t_k}}|\theta)}{\sum_{\theta \in \Theta} \mu_{t_k}(\theta)P(a_{e_{t_k}}|\theta)} \tag{10}$$

Note that, $P(a_{e_{t_k}})$ is the probability of having action a_e at time t_k given the type θ.

Now, we define the strategies of the two players:

The intruder pure strategy is to select one of the victim's incoming-links $e \in E$ where $E = \{e_1, \ldots, e_l\}$ is the set of all incoming-links directed to victim i. The intruder mixed strategy is to assign a probability for selecting each link where the probability vector is $Q = (q_{e_1}, \ldots, q_{e_l})$; such that $\sum_{e \in E} q_e = 1$. The intruder then selects a link $e \in E$ with probability q_e.

The leader-IDS pure strategy is to choose the sampling rate s_e on link e such that $\sum_{e \in E} s_e \leqslant B_i$. We introduce the set of detection probability vector $V = (p_{e_1}, \ldots, p_{e_l})$; such that $\sum_{e \in E} \mu_{t_{k+1}}(\theta|a_e) f_e p_e \leqslant B_i$. The strategy for the IDS is to pick a set of detection probabilities at the links which belongs to V.

6.1. The game definition

We model our game as zero-sum non-cooperative game with incomplete information about the players where each player has a private information about his/her preferences. In our case, the leader-IDS type is known to all the players while the sender type is selected from the type set $\Theta = \{Malicious, Nomral\}$. Knowing that the sender type is a private information. *Bayesian Equilibrium* [22] dictates that sender's action depend on his/her type θ.

The following is the leader-IDS utility function:

$$U_i = \sum_{e \in E} p_e q_e \tag{11}$$

where,

- p_e is the expected probability of detecting a malicious packet at link e.
- q_e is the probability of selecting link e.

The leader-IDS objective is to maximize this utility function as follows:

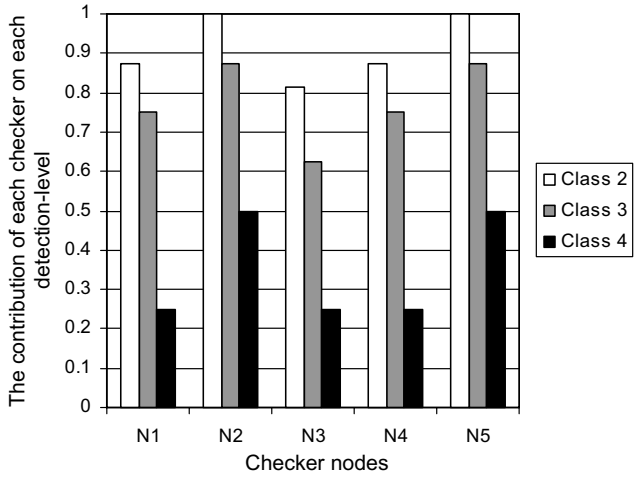

Fig. 2. The effect of checkers on each detection-level.

$$\max_{p\in V}\sum_{e\in E}p_e q_e \tag{12}$$

On the other hand, the sender objective is to choose his/her type θ from the type set Θ. If his/her $\theta = Malicious$, then the intruder objective is to minimize the maximization as follows:

$$\min_{q\in Q}\max_{p\in V}\sum_{e\in E}p_e q_e \tag{13}$$

Using minmax theorem [22], we can find the Nash equilibrium where the following holds:

$$\beta = \min_{q\in Q}\max_{p\in V}\sum_{e\in E}p_e q_e = \max_{p\in V}\min_{q\in Q}\sum_{e\in E}p_e q_e \tag{14}$$

6.2. The game solution

The game can be solved by considering the problem from either the intruder's or leader-IDS's standpoint. Here, we consider the intruder's problem which is:

$$\min_{q\in Q}\max_{p\in V}\sum_{e\in E}p_e q_e \tag{15}$$

For a fixed q_e, the problem is reduced to the following:

$$\max_{p\in V}\sum_{e\in E}p_e q_e \text{ where : } \sum_{e\in E}\mu_{t_{k+1}}(\theta|a_e) f_e p_e \leqslant B_i \tag{16}$$

Using the dual variable algorithm [23], we take the dual variable Y with respect to p_e. Hence, Eq. (16) is reduced to the following optimization problem:

$$\min B_i Y$$

subject to:

$$\mu_{t_{k+1}}(\theta|a_e) f_e Y \geqslant q_e, \quad \sum_{e\in E} q_e = 1 \text{ and } Y \geqslant 0 \tag{17}$$

To find Y, we have to find the value that makes Y satisfies the constraints. To achieve this, we interpret q_e as the flow on link e. Hence, the following constraint:

$$q_e \leqslant \mu_{t_{k+1}}(\theta|a_e) f_e Y \tag{18}$$

restricts the flow on link e to be $\mu_{t_{k+1}}(\theta|e) f_e Y$. Thus, the constraint $\sum_{e\in E} q_e = 1$ is interpreted as the summation of all the flows in E is equal to 1. In other words, the constraint $\sum_{e\in E} q_e = 1$ can be written as follows:

$$\sum_{e\in E}\mu_{t_{k+1}}(\theta|a_e) f_e Y = 1 \tag{19}$$

Then, the objective is to find the smallest value of Y that satisfies the constraint in Eqs. (18) and (19). Thus, Y is equal to $\sum e \in E \mu_{t_{k+1}}(\theta|a_e) f_e$.

Finally, the leader-IDS strategy is to compute the belief of each one-hop node from the protected node i using Eq. (10). Knowing that the belief of the sender is equivalent to the link's belief $e \in E$ that has been used by the sender. Thus, the leader-IDS can calculate the following: $\sum e \in E \mu_{t_{k+1}}(\theta|a_e) f_e$. Finally, the leader-IDS optimal sampling strategy is to sample the incoming link e of node i with:

$$s_i = \frac{B_i \mu_e f_e}{\sum_{e\in E}\mu_e f_e} \tag{20}$$

The main differences between this game and the work done in [16] are:

- In this game, the type of the sender is either normal or malicious and therefore the defender has incomplete information about the type of the sender. On the other hand, in [16] the sender type is known to the defender which is always malicious.
- In our game, the leader-IDS calculates the belief function for each sender which help the leader-IDS to evaluate the link usage (i.e., either used maliciously or normally).

6.3. Illustrative example

Let us consider a cluster of 5 nodes with their reputations $\{N_1 = 25, N_2 = 17, N_3 = 15, N_4 = 23, N_5 = 20\}$ where N_5 is the leader-IDS with detection budget $B = 100$ packets/s. N_5 is responsible to sample the incoming packets for each protected node according to the nodes' assigned sampling budget which is equivalent to nodes' reputation. Hence, each node sampling budget will be: $\{B_1 = 25,\ B_2 = 17, B_3 = 15, B_4 = 23, B_5 = 20\}$. Now, we demonstrate how the game is played between the leader-IDS N_5 and an intruder where the intruder identity is unknown. As an example, we select node N_1 as the target node where an intruder is targeting to attack. Fig. 3, describes an attack scenario where an intrusion could be directed to node N_1 either through node A, B or C. At the beginning of the game, nodes A, B, and C choose their type from the type set $\Theta = \{Malicious, Normal\}$. Even if the types of A, B, and C are normal, an intrusion could still be directed to node N_1 through these intermediate nodes. Hence, the leader-IDS will use the belief function of Eq. (10) to calculate the belief for each incoming-link $e \in E$. In our example, the incoming set of links to N_1 is $E = \{AN_1, BN_1, CN_1\}$ and leader-IDS belief for each link for time t_{k+1} is $\{\mu(\theta|a_{AN_1}) = 0.3, \mu(\theta|a_{BN_1}) = 0.5, \mu(\theta|a_{CN_1}) = 0.2\}$. Knowing that the flow on link $e \in E$ is determined by how much the node is willing to forward to the target node N_1 which depends on whether the node is a source or an intermediate. We can compute the link's flow using the work in [24]. As an example, we assume that the flow on each link $e \in E$ is $\{f_{AN_1} = 15, f_{BN_1} = 18, f_{CN_1} = 17\}$. To defend node N_1 efficiently, the leader-IDS's optimal strategy is to sample the links, using Eq. (20), as follows: $\{AN_1 = \frac{0.3\times 15}{16.9}, BN_1 = \frac{0.5\times 18}{16.9}, CN_1 = \frac{0.2\times 17}{16.9}\}$. Note that, our solution will be iterated until the budget constraint is satisfied.

7. Simulation results

We simulate our unified model in a cluster of 20 mobile nodes in the presence of selfish nodes. Our model is divided

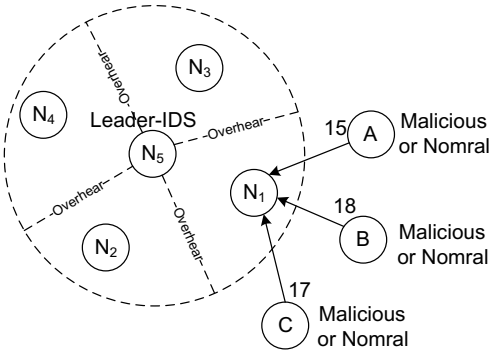

Fig. 3. Leader-IDS and intruder game example.

into three phases: The first one is the election process with reputation and punishment system based on *VCG* mechanism to elect the most cost-efficient IDS. The second phase uses checker nodes to catch and punish the misbehaving leader node cooperatively. Here, we show that both phases are needed to motivate selfish nodes in behaving normally. Finally, the leader-IDS distribute his/her budget according to nodes' reputation. The detection is done using our Bayesian game model.

In Fig. 4a, we present the effect of checkers on packet analysis in the random election model [4] where the percentage of checkers varies from 0 to 80% step by 20. We assume that each checker is sampling 5% of packets analyzed by leader IDS in order to ensure leader performance. As the percentage of checkers increases the percentage of packet analysis increases. This is because with more checkers we can analyze more packets and identify the selfish leader-IDS more quickly and accurately. Once the selfish node is identified, a new leader is elected. So far there is no punishment system that is able to punish misbehaving nodes. Hence, selfish nodes can still benefit from others' services and affect others' life time as shown in Fig. 4b. Therefore, a reputation model is needed to punish misbehaving nodes. In our model, we assume that all nodes at the formation phase have a reputation of one.

Fig. 5a shows our reputation model for different number of elections where 10% of nodes are selfish. Our model is able to punish selfish nodes 5 and 15 by decreasing their reputation. We consider the value of the punishment threshold *TH* to be zero. Nodes with negative reputation are excluded from cluster's services. Hence our mechanism ensures that misbehaving nodes are punished. In Fig. 5b, we compare our model with random one to show the percentage of alive nodes and packet analysis with respect to time. Due to the absence of a catching mechanism in the random model, selfish nodes are undetected. Therefore, normal nodes that handle the detection process will die faster since random model does not support any motivation strategy. On the other hand, our model gives better results since selfish nodes are given incentives to cooperate. Thus, the node with minimum cost of analysis becomes the leader. This balances the energy level among all the nodes in the cluster over time.

Fig. 6a describes the percentage of packet analysis for both models. This percentage decreases for random model since percentage of selfish nodes in the cluster increases over time as only normal nodes are losing their energy. The figure shows improved results due to *VCG* and checkers of our model that motivates nodes to behave normally. In our model, nodes do not remain selfish as they care for their reputations. Fig. 6b shows that our model balances the energy of all nodes while in the random one several normal nodes die.

In Fig. 7a, we compare our proposed detection-level framework with fixed cooperative model of 20 and 60% of checkers. Our model is better than 20% checkers with respect to false positive rate since the number of checkers varies according to the severe behavior of leader-IDS. On the other hand, 60% checkers perform better than ours but consumes much more energy. Hence, our detection-level framework has a tradeoff between energy consumption and false positive rate. In Fig. 7b, we compare our model with uniform one regarding the probability of detection. Knowing that the uniform model distributes

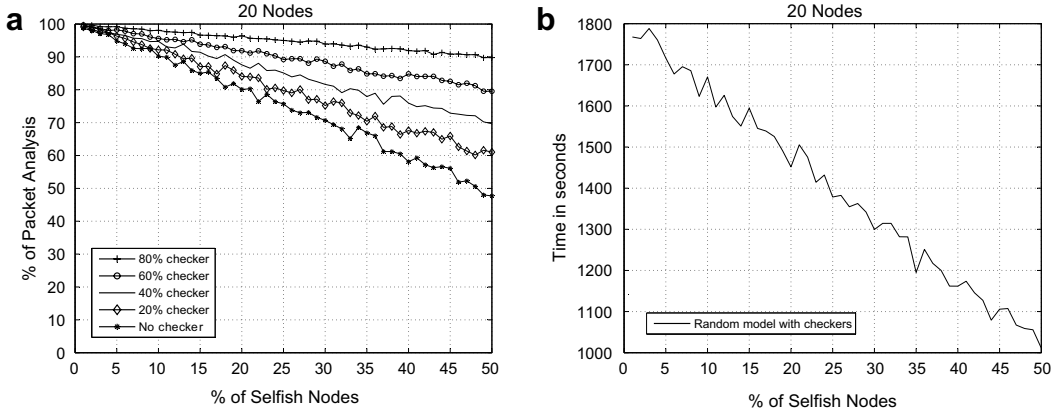

Fig. 4. (a) Total packet analysis. (b) Time taken for normal node to die.

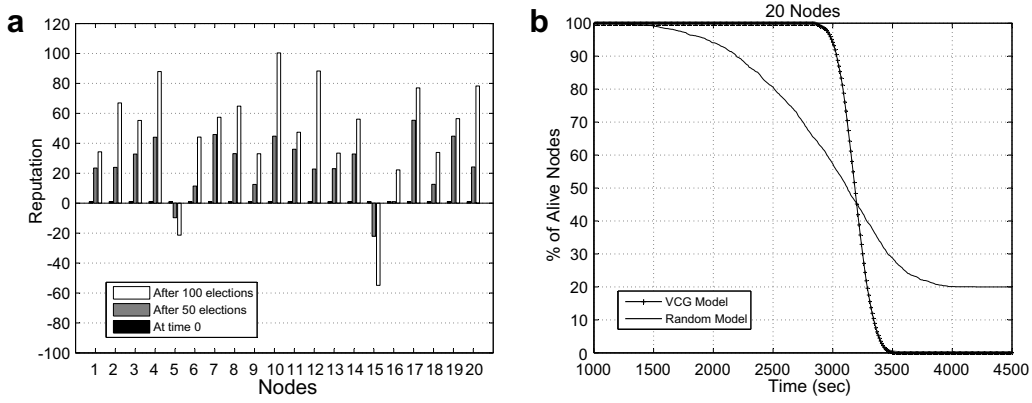

Fig. 5. (a) Reputation of nodes. (b) Percentage of alive nodes.

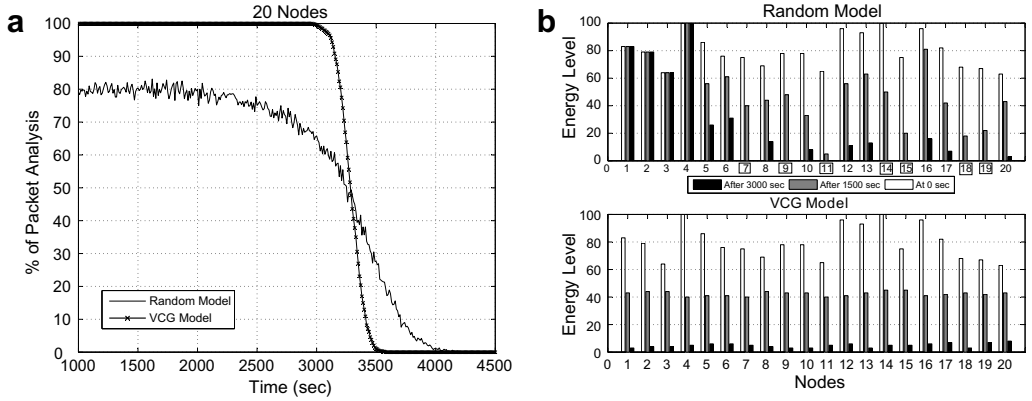

Fig. 6. (a) Percentage of packet analysis. (b) Energy level of nodes.

the node's budget equally over node's incoming-links. Our model performs better than uniform irrespective of the budget since our model distribute the node's budget using our Bayesian game result. As the number of links increases the performance of uniform model decreases due to the equal distribution of the sampling effort. While, our model is not affected by the increase of links since sampling is handled taking into consideration the link's belief.

8. Conclusion and future work

We proposed a unified framework that is able to prolong the lifetime of IDS in a cluster by balancing the resource consumptions among all the nodes. This was achieved by truthfully electing the most cost-efficient node (IDS) that handles the detection process. Incentives were given in the form of reputations to motivate nodes in revealing truthfully their costs of analysis. Reputations

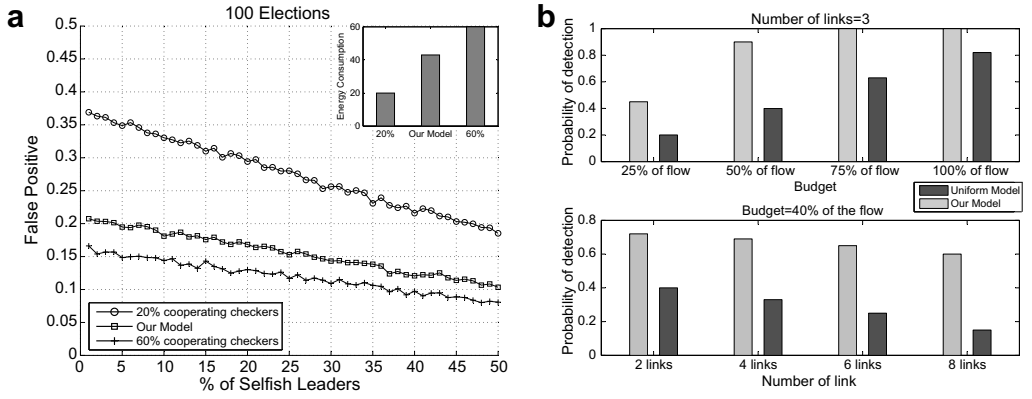

Fig. 7. (a) False positive rate and energy. (b) Probability of detection.

are computed using the well known VCG mechanism where truth-telling is the dominant strategy. Our distributed mechanism was able to elect the most cost-efficient node and to punish misbehaving nodes by withholding cluster's services. Moreover, our framework was able to catch and punish misbehaving leader that would deviate from detecting intrusions after election. A cooperative decision game theoretical model was proposed to efficiently catch the misbehaving leader-IDS with less false-positive rate. Additionally, a zero-sum non-cooperative game was given to help the leader-IDS to maximize the probability of detection. This game was played between the leader-IDS and intruder with incomplete information about the intruder's identity. The solution of the game advised the leader-IDS to his/her optimal sampling strategy. Our simulation results showed that our framework was able to elect the most cost-efficient node, reduce the catch false-positive rate by the checkers and maximize the probability of detection by the leader-IDS. Finally, our current catch and punish mechanism solution adopts a binary approach, which either regards a leader as honest, or dishonest. While this approach works in principle, it may certainly be refined as a quantitative approach that rates the leader with numerical values. We will investigate this possibility in future work.

Acknowledgments

This material is based upon work supported by Natural Sciences and Engineering Research Council of Canada under Discovery Grant. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsoring organizations. The authors are grateful to the anonymous reviewers for their valuable comments.

References

[1] A. Mishra, K. Nadkarni, A. Patcha, Intrusion detection in wireless ad hoc networks, IEEE Wireless Communications (2004) 48–60.
[2] Y. Zhang, W. Lee, Y. Huang, Intrusion detection techniques for mobile wireless networks, in: ACM Wireless Networks, ACM, 2003, pp. 545–556.
[3] Y. Hu, A. Perrig, A survey of secure wireless ad hoc routing, IEEE Security and Privacy (2004) 28–39.
[4] Y. Huang, W. Lee, A cooperative intrusion detection system for ad hoc networks, in: Proceedings of the 1st ACM Workshop Security of Ad Hoc and Sensor Networks, ACM, Virginia, 2003, pp. 135–147.
[5] Y. Zhang, W. Lee, Intrusion detection in wireless ad hoc networks, in: Proceedings of the MOBICOM 2000, ACM, 2000, pp. 275–283.
[6] K. Chen, K. Nahrstedt, ipass: an incentive compatible auction scheme to enable packet forwarding service in manet, in: Proceedings of the 24th ICDCS'04, IEEE, 2004.
[7] H. Yang, X. Meng, S. Lu, Self-organized network-layer security in mobile ad hoc networks, in: Proceedings of the ACM Workshop on Wireless Security, ACM, 2002, pp. 11–20.
[8] P. Michiardi, R. Molva, Analysis of coalition formation and cooperation strategies in mobile ad hoc networks, Elsevier Journal: Ad hoc Networks 3 (2005) 193–219.
[9] S. Marti, T. Giuli, K. Lai, M. Baker, Mitigating routing misbehavior in mobile ad hoc networks, in: Proceedings of the 6th Annual International Conference on Mobile Computing and Networking, ACM, 2000, pp. 255–265.
[10] A. Mas-Colell, M. Whinston, J. Green, Microeconomic Theory, Oxford University Press, New York, 1995.
[11] N. Nisan, A. Ronen, Algorithmic mechanism design (extended abstract), in: Proceedings of the 31st Annual ACM Symposium on Theory of Computing, ACM, 1999, pp. 129–140.
[12] J. Feigenbaum, C. Papadimitriou, R. Sami, S. Shenker, A BGP based mechanism for lowest-cost routing, in: Proceedings of the 21st annual symposium on Principles of distributed computing, ACM, 2002, pp. 173–182.
[13] J. Shneidman, D. Parkes, Specification faithfulness in networks with rational nodes, in: Proceedings of the PODC'04, ACM, 2004.
[14] L. Anderegg, S. Eidenbenz, Ad hoc-vcg: a truthful and cost-efficient routing protocol for mobile ad hoc networks with selfish agents, in: Proceedings of the MobiCom'03, ACM, San Diego, California, 2003, pp. 245–259.
[15] P. Morris, Introduction to Game Theory, First ed., Springer, 1994.
[16] M. Mehrandish, H. Otrok, M. Debbabi, C. Assi, P. Bhattacharya, A game theoretic approach to detect network intrusions: The cooperative intruders scenario, in: Proceedings of the 49th annual of IEEE GLOBECOM, IEEE, 2006.
[17] T. Alpcan, T. Basar, A game theoretic analysis of intrusion detection in access control systems, Proceedings of the 43rd IEEE Conference on Decision and Control (CDC), Vol. 2, IEEE, 2004, pp. 1568–1573.
[18] Y. Liu, C. Comaniciu, H. Man, A bayesian game approach for intrusion detection in wireless ad hoc networks, in: Proceedings of the GameNets06, ACM, 2006.
[19] R. Dash, N. Jennings, D. Parkes, Computational mechanism design: a call to arms, IEEE Intelligent Systems (2003) 40–47.
[20] P. Krishna, N.H. Vaidya, M. Chatterjee, D.K. Pradhan, A cluster-based approach for routing in dynamic networks, in: Proceedings of the ACM SIGCOMM Computer Communication Review, ACM, 1997, pp. 49–64.
[21] A.E. Roth, The Shapley Value: Essays in Honor of Lloyd S.Shapley, Cambridge University Press, 1988.
[22] M. Willem, Minimax Theorem, Birkhauser, USA, 1996.
[23] C.H. Papadimitriou, K. Steiglitz, Combinatorial Optimization Algorithms and Complexity, Dover Publications, 1999.
[24] V. Srinivasan, P. Nuggehalli, C.F. Chiasserini, R.R. Rao, Cooperation in wireless ad hoc networks, in: Proceedings of the INFOCOM03, IEEE, 2003, pp. 808–817.