Towards Optimal Monitoring in Cooperative IDS for Resource Constrained Wireless Networks

Amin Hassanzadeh, Radu Stoleru
Department of Computer Science and Engineering, Texas A&M University
{hassanzadeh, stoleru}@cse.tamu.edu

Abstract—**The problem of cooperative intrusion detection in resource constrained wireless networks (e.g., adhoc, sensor) is challenging, primarily because of the limited resources available to participating nodes. Although the problem has received some attention from the research community, little is known about the tradeoffs among different objectives, e.g. network performance, power consumption, delay in information being collected and security effectiveness. This paper proposes, to the best of our knowledge for the first time, to distribute cooperative intrusion detection functions that take into account, simultaneously, multiple objectives. We formulate the problem of identifying the type of intrusion detection each node runs as a multi-objective optimization problem and motivate/develop a genetic algorithm to solve it. Through extensive simulations we demonstrate that our solution is characterized by: a small variance in the normalized fitness values of individual/single objectives; and a smaller attack detection and reporting delay than state of art solutions. In a real implementation/evaluation of our cooperative intrusion detection system, we demonstrate that it achieves a higher detection rate (93%) than state of art solutions (60%-73%).**

I. Introduction

Intrusion detection systems (IDS) are an essential part of security for resource constrained wireless networks (e.g., adhoc and sensor networks deployed for emergency response [1]), as they are for any high survivability network. The IDS deployed in these networks differ from those used in wired networks. Because of a lack of concentration points where traffic can be analyzed, and reliance on noisy, intermittent wireless communication channels that provide limited bandwidth, researchers have proposed decentralized architectures [2]–[6]. Most decentralized approaches suggest placing an intrusion detection agent in each node. These agents use local and reliable audit data, in conjunction with a local IDS to detect malicious activities, and *collaborate* among themselves.

To conserve resources and manage the intrusion detection mechanism, some IDS functionality can be distributed to a few nodes (i.e., monitoring nodes) and still provide a sufficient degree of detection. In these *cooperative IDS*, the network topology used for communicating intrusion detection reports has an important effect on network performance and resource consumption. Finding the optimum network topology for nodes that execute cooperative IDS has been shown to be an NP-hard problem [7], [8].

Previous research investigated distributed solutions in which nodes, using information about their neighbors, elect a local candidate for executing cooperative IDS functions [9], [10]. These solutions, however, are suboptimal and incur high communication overhead. In this paper, we propose a solution in which a base station (in our application scenario, the Command&Control Center of emergency response operations [1]), which has knowledge about network (e.g., node resources, locations, etc.) and security requirements (e.g., maximum permissible delay in reporting an event, minimum network coverage), computes the optimal distribution of roles specific to cooperative IDS. Our proposed solution allows execution of sophisticated algorithms that optimize multiple objectives related to network performance and security effectiveness. More precisely, this paper makes the following contributions:

- Models are developed for individual objectives, such as node energy, permissible delay in reporting data, information contained in reports, and network coverage.
- A genetic algorithm (GA), employing a penalized function, is developed for solving the multi-objective optimization problem.
- Extensive simulations that demonstrate that our GA converges to a solution characterized by: a small variance in the normalized fitness value of single objectives; and by a small attack detection and reporting delay.
- A proof-of-concept implementation and an evaluation of the proposed architecture in a resource constrained wireless network deployed in a realistic setting.

II. Related Work

Early research [6] has focused on node cooperation for intrusion detection, but has not considered how cooperating nodes are selected. More recently, several methods have been proposed for selecting nodes that run intrusion detection functions. While these nodes are primarily selected based on connectivity in wired networks, in resource constrained wireless networks the selection criteria is much more complicated. The proposed methods fall largely in two categories: distributed algorithms and centralized algorithms.

In the distributed solutions the concept of local elections and voting is used to select monitoring nodes (and the cluster of nodes each monitor is responsible for), based on a criteria, their neighbors' capability, e.g., remaining power [10] or degree of connectivity [9] or simply random [11]. The problem with this type of algorithm is that the best possible clustering is only optimal at the local level due to lack of global information. Centralized solutions guarantee that the decision will be made based on more complete information about the nodes and the network. For this, a powerful central

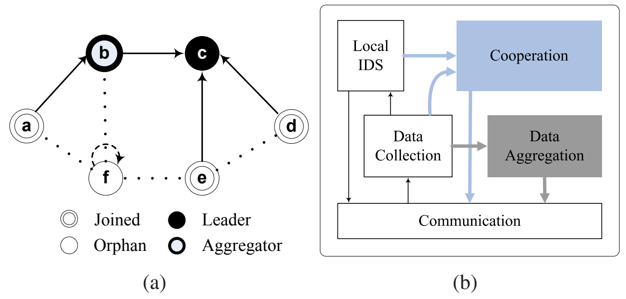


Fig. 1. (a) Example of a network with a cooperative IDS (nodes responsibilities vary). (b) A generic architecture for nodes in a cooperative IDS.

node is required to run more complicated algorithms. There are several centralized algorithms (e.g., [7], [8]) that formulate the monitoring problem in binary integer linear programming considering only node coverage as an optimization objective. Taking into account other issues, as our proposed solution does, e.g., amount of information collected by the nodes, total power consumption, and delay in detection process, may significantly affect the role assignment.

III. Proposed Cooperative IDS Architecture and Problem Formulation

Our system [1] consists of a resource constrained wireless network with a single base station (i.e., C&C operations center). The network is loosely time synchronized and all nodes know their locations. The base station periodically collects network information and uses this information to decide which IDS functions to run, and where to run them. This decision is done periodically, or when network conditions change.

The decision of how to distribute IDS functions results in a network organized as a set of cluster trees, each running a distributed cooperative IDS. Figure 1(a) depicts an example of a network cooperative IDS which consists of a cluster tree with 5 nodes and one additional node, not associated with a cluster tree. Figure 1(b) shows a generic node architecture for a node. The nodes forming cluster trees have different levels of intrusion detection responsibilities.

Inspired by previous intrusion detection research [6], [12] we identify four distinct roles for the nodes in our distributed IDS: *Joined* nodes are leaves in a cluster tree. They monitor local activity (e.g., communication, processes running, data produced) and run a local IDS as depicted in Figure 1(b). Results are reported to the parent, which can be an Aggregator or a Leader; *Aggregator* nodes also monitor local activity, as a joined node, receive reports from children, either joined or other aggregator nodes, and aggregate this data with their information, using the "Data Aggregation" module in Figure 1(b). The aggregated data is reported to a parent, either another aggregator or leader; *Leader* is the root of a cluster tree. A leader receives reports from its children, either joined or aggregator nodes, and executes sophisticated IDS functions as part of a cooperation module (Figure 1(b)). The results are reported to the base station. Leaders of all cluster trees form a connected graph, which contains the base station. As shown in Figure 1(a) there is a single-cluster tree, with node c as its leader; *Orphan* nodes are not part of a cluster tree. They run local IDS and do not forward their observations to their neighbors.

The nodes with the aforementioned tasks can largely be described as: *Tasked* nodes if they are either Joined, Aggregator, or Leader; *Untasked* nodes - the Orphan nodes. Similarly we will refer to all children of node i as *Follower*s.

Individual nodes, organized in cluster trees, communicate and aggregate data for the purpose of detecting, in a collaborative manner, intruders. The proposed cluster tree organization impacts energy consumption, hence network lifetime, event reporting delay, network coverage, and quality of data collected. Each of these properties represent individual optimization objectives in a multi-objective optimization problem, formally described in Section III-B.

A. Detection System and Attacker Model

The proposed system architecture, based on cooperative IDS, is the main focus of our paper. For the intrusion detection system we employ existing IDS engines, e.g. snort [13], which is based on rulesets. For our approach, it is critical to observe that more complex actions performed by the detection engine (e.g., number of rules evaluated, processing stages involved) will pose a higher demand on available resources (e.g., computation, communication). Consequently, the configuration of the IDS engine presents opportunities to trade off intrusion detection accuracy for resource availability. Thus, we consider three types of intrusion detection engines: lightweight (LW-DS), employed by joined and orphan nodes; medium (RE-DS), employed by aggregators; and heavy (HW-DS), employed by leaders.

In this paper, we consider both insider (i.e. compromised nodes) and outsider attackers. The latter are nodes that are not members of the network, and hence, have not been assigned any roles (e.g., leader, aggregator, joined or orphan). We assume that the attacker runs attack only against nodes located in its radio range (i.e., single-hop attack). If the attack is multi-hop, the intrusion detection problem is simplified, due to the opportunity of other nodes, possibly executing more sophisticated engines (RE-DS or HW-DS), to detect the attack. Additionally, a single hop attack enables us to investigate the impact that the detection engine type (LW-DS, RE-DS, or HW-DS) has on detection accuracy. The types of attacks we consider are the following: 1) *Flooding:* the purpose of this attack is to exhaust both network and host resources by sending a rapid succession of many packets. As examples for this attack, we consider the SYN and ICMP flood attacks. For detecting these attacks a LW-DS is sufficient; 2) *Port scanning:* this is a generic attack that probes a target node for open ports. As an example, we consider TCP port scanning employing TCP SYN and FIN packets. For detecting a port scanning attack, an RE-DS or HW-DS is required; 3) *Web exploits:* for this, the attacker hosts an HTTP server and executes HTML exploits (e.g., information disclosure), against clients. Due to the complexity of this attack, only a HW-DS is capable of detecting it.

B. Problem Formulation

Given a set $N = \{n_1, n_2, ..., n_k, b\}$ of $k+1$ nodes, which includes base station b, and the roles of nodes identified at the beginning of the section, let L be the set of leader nodes, A the set of aggregator nodes, J the set of joined nodes and O the set of orphan nodes in the network. Each node n_i in the network is assigned a single responsibility. More formally, $N \setminus \{b\} = L \cup A \cup J \cup O$ and $r_i \cap r_j = \emptyset$ where $r_i, r_j \in \{L, A, J, O\}$ and $i \neq j$. Using the proposed cooperative IDS architecture, nodes are organized in cluster trees. Let $G = \{T_1, ..., T_q\}$ be the set of q cluster trees formed in the network. Each cluster tree T_i has one leader, and one or more aggregators A_{T_i} and joint nodes J_{T_i}: $T_i = L_{T_i} \cup A_{T_i} \cup J_{T_i}$. As mentioned, the set of leader nodes forms a connected graph. All nodes communicate with radio range R. Leaders can communicate over greater distances using radio range $R' = \beta R$. More formally, we define $G_L(L, L \times L)$ as the graph formed by leaders $L_i \in L$ such that $\text{dist}(L_i, L_j) \leq R', \forall i \neq j$.

Different cluster tree topologies exhibit different characteristics for energy consumption, event reporting delay, network coverage, and data accuracy. Some of these properties need to be minimized (e.g., energy consumption and delay), while others need to be maximized (e.g., network coverage and data accuracy). Individually, each of these properties can be treated naively as a single objective optimization problem. However, a more complex, but more commonly sought goal in a cooperative IDS for a set of cluster trees, is to find the lowest aggregate energy consumption and delay with the highest aggregate data accuracy and network coverage. It is obvious that single objective optimizations may negatively impact the performance of other objectives in a multi-objective environment. Therefore, we define the network performance as a multi-variate objective function:

$$\begin{aligned} f(G) &= w_1(1 - \frac{P_G}{C_p}) + w_2\frac{H_G}{C_h} + w_3(1 - \frac{D_G}{C_d}) + w_4\frac{C_G}{C_c} \\ &= w_1\left(1 - \frac{\sum_{i=1}^{q}\sum_{j=1}^{|T_i|} p_{ij}}{C_p}\right) + w_2\frac{\sum_{i=1}^{q}\sum_{j=1}^{|T_i|} h_{ij}}{C_h} \\ &+ w_3\left(1 - \frac{\sum_{i=1}^{q}\sum_{j=1}^{|T_i|} d_{ij}}{C_d}\right) + w_4\frac{\sum_{i=1}^{q}\sum_{j=1}^{|T_i|} c_{ij}}{C_c} \end{aligned}$$

, where w_i ($i = 1, ..., 4$) are weighting constants, q is the number of cluster trees, $|T_i|$ is the size of cluster tree i and P_G, H_G, D_G and C_G are functions for Power, Information, Delay and Coverage, respectively, in network G. Similarly, C_p, C_d, C_h and C_c are normalizing constants for the individual objective functions. This multi-variable objective function trades off Power and Delay, two minimization objectives, in order to improve Information and Coverage, two maximization objectives. Consequently, we define our multi-objective optimization (MOO) problem, as follows:

$$\begin{aligned} \text{maximize} \quad & f(G) && (1) \\ \text{subject to:} \quad & |L_{T_i}| = 1, \forall i = 1, ..., q && (2) \\ & \text{if } n_j \in A_{T_i}, J_{T_i} \text{ then parent } (n_j) \in A_{T_i} && (3) \\ & \text{and parent } (n_j) \in, L_{T_i}, \forall i = 1, ..., q && \\ & |J_{T_i}| \geq 1, \forall i = 1, ..., q && (4) \\ & \forall L_i \in G_L, \text{base } b \in G_L, \text{ and } p_{L_i} \geq p_L^{th} && (5) \\ & |T_i| \leq T_{th} && (6) \end{aligned}$$

, where constraint (2) indicates that a single leader node is in each cluster tree; constraint (3) enforces the construction of the cluster tree (to conform with our cooperative IDS architecture); constraint (4) indicates that at least one joined node must be in each cluster tree; and constraint (5) says that the remaining power of the leader has to be higher than a defined threshold. Finally, constraint (6) prohibits the creation of large cluster trees. This constraint forces the solution to produce a set of trees instead of one large tree - a single point of failure.

For investigating solutions for MOO, we will make use of the corresponding single objective optimization (SOO) problems. As an example, for maximizing Information in the network, the corresponding SOO is:

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^{q}\sum_{j=1}^{|T_i|} h_{ij} && (7) \\ \text{subject to:} \quad & \text{same constraints as MOO} && (8) \end{aligned}$$

, where h_{ij} represent the information available to node j in cluster tree i. Similarly we can formulate single optimization problems that minimize Delay and Power, and that maximize Coverage (omitted here due to space constraints).

Alternatively, the optimization problem defined in Equation 1 may be solved by choosing a single objective as a fitness function and by using the other objectives as constraints. For example, maximization of Information may be used as a fitness function while Power, Delay and Coverage objectives are constrained to threshold values P^{th}, D^{th}, $|N| = k+1$, as follows: $\sum_{i=1}^{q}\sum_{j=1}^{|T_i|} p_{ij} \leq P^{th}$, $\sum_{i=1}^{q}\sum_{j=1}^{|T_i|} d_{ij} \leq D^{th}$, $\sum_{i=1}^{q}\sum_{j=1}^{|T_i|} c_{ij} = |N|$.

IV. Proposed Solution for Optimal Monitoring in Cooperative IDS

In this section we propose mathematical models for P_G, H_G, D_G, C_G, the Power, Information, Delay, and Coverage functions, respectively, and our solution for the optimal monitoring in cooperative IDS.

A. Power Model

Nodes in the network, depending on their roles, consume different amounts of energy for communication and computation. Leader and aggregator nodes perform functions that consume more power; cooperation module in leader and aggregation module in aggregator. Similarly, cluster trees may vary in the number of tasked and untasked nodes and in the number and distribution of followers for each leader and aggregator. This means that the total power consumption of any set of cluster trees will vary from other possible sets. Thus, one single objective optimization problem is to minimize total power consumed by all the nodes in a set of cluster trees G, formed in the network, as given by:

$$P_G = \sum_{i=1}^{k} p_i = \sum_{i=1}^{|L|} p_{L_i} + \sum_{i=1}^{|A|} p_{A_i} + \sum_{i=1}^{|J|} p_{J_i} + \sum_{i=1}^{|O|} p_{O_i} \quad (9)$$

where p_i is the power consumption of node i for both communication and computation activities, which will depend on the role assigned to it (i.e., leader, aggregator, joint or orphan). In the remaining part of the section, we present the power consumption models for leader nodes p_{L_i}, aggregator nodes p_{A_i}, joined nodes p_{J_i}, and orphan nodes p_{O_i}.

For our power consumption model, we extend the one proposed in [14]. We consider the power consumed for radio communication or processing/computation. We denote by P_{TX} and P_{RX} the power consumed for transmitting and for receiving a report, respectively. We denote by P_{Rep} and P_{Recv} the power consumed for transmitting and for receiving an intrusion detection alarm, respectively. The power consumed for processing/computation, denoted by P_{Proc}, depends on the role assigned to a node. The following notations are with reference to the proposed system architecture, shown in Figure 1(b) - the "Cooperation" and "Data Aggregation" modules. For an aggregator node, which has the "Data Aggregation" module active, $P_{Proc} = P_{ModA} + P_{CodA}$ where P_{ModA} is a fixed power consumed for maintaining the module active and P_{CodA} is a variable power, consumed for coding/aggregating data from follower nodes. For a leader node, which has the "Cooperation" module active, $P_{Proc} = P_{ModC} + P_{CodC}$ where P_{ModC} is a fixed power consumed for maintaining the module active and P_{CodC} is a variable power, consumed for coding/cooperating data from follower nodes.

We denote by E_i an event vector, which is a set of observed security parameters reported by node i. We denote by l_{E_i} its length ($l_{E_i} = |E_i|$). We remark here that l_{E_i} is the same for all joined nodes (since they do not have followers) and that it increases with the number of descendants. Considering our system architecture, each tasked node i, except for leaders, sends E_i in each time slot, at a rate λ packets per second. We assume that λ is equal for all nodes. We use F_i as the set of followers of node i. Considering the tasks mentioned in Section III the power consumed becomes:

$$\begin{aligned} p_{L_i} &= P_{RX_i} + P_{Proc_i} + P_{Rep_i} + P_{Recv_i} \\ &= p_{rx} \sum_{j=1}^{|F_i|} \lambda l_{E_{ij}} + P_{CodC} \left[\lambda l_{E_i} + \sum_{j=1}^{|F_i|} \lambda l_{E_{ij}} \right] \\ &+ P_{ModC} + \eta l_A (p_{tx} + p'_{tx} + p'_{rx}) \end{aligned}$$

, where p_{tx} and p_{rx} are the power consumptions required to transmit and receive one bit, respectively, and p'_{tx} and p'_{rx} are the power consumptions required for transmitting 1 bit over a long distance between leaders. Similarly, $p_{A_i} = p_{L_i} + P_{TX_i} = p_{L_i} + p_{tx} \lambda l_{E_i}$.

Reporting consumes $P_{Rep} = \eta l_A p'_{tx}$ depending on the attack frequency η in the network ($0 \leq \eta \leq 1$) where l_A is the size of alarm. Leaders need to communicate with each other, using long distance communication, to exchange alerts in case of attack detection as well as sending alert to the followers, using regular radio range communication. Since aggregators forward alerts to their children, they consume P_{Rep} and P_{Recv} for transmitting/ receiving alarms, respectively.

In contrast to leaders and aggregators, the only task done by joined nodes is transmitting their event vector to their parents and receiving alarms from them. Therefore, the power consumption of joined nodes is $p_{J_i} = P_{TX_i} + P_{Rep_i} = p_{tx} \lambda l_{E_i} + \eta l_A p_{rx}$. Since orphan nodes does not do any of these processes, their power consumption is 0 ($p_{O_i} = 0$).

B. Information Model

For IDS cooperation, in which each node has a local IDS and sends reports to its parent, the way nodes are organized in cluster trees and cooperate affects the amount of data collected. An optimal solution for our MOO problem collects as much data as possible, but, at the same time reduces the amount of useless data collected. Thus, for defining one additional objective in our MOO, we develop a model for data collected in a cooperative IDS.

We already mentioned that each node i sends an event vector E_i to its parent. Here, for simplicity, we assume that each vector contains only one security parameter which is a Random variable with a Gaussian distribution ($x_i \sim N(\mu, {\sigma_i}^2)$), where x_i is the value of a security parameter observed by node i. From information theory, the amount of information contained in x is its entropy, $H(x)$. In our system, $H(x_i)$ is the data entropy of a reporting parameter (variable) by node i and $H(x_i, x_j)$ is the joint entropy of two variables x_i and x_j where node i is the parent of node j. So, the data entropy at any aggregator is the entropy of all received data from its followers combined with its own observation. For higher levels of aggregation, aggregators receive the aggregated data of all descendants.

For any set of cluster trees, the Information function is defined as summation of all data entropy at the nodes as $H_G = \sum_{i=1}^{k} h_i$, where $h_i = H(x_i)$ for any joined node and $h_i = H(x_i, ..., x_n)$, as the entropy of all parameters received by aggregator i combined with its own security parameter.

The joint entropy of n Gaussian random variables is $H(x_1, ..., x_n) = \frac{1}{2} \log[(2\pi e)^n det(\Sigma)]$, where Σ is the covariance matrix of the joint variables [15]. Matrix Σ contains all σ_i's as variances of Gaussian random variables and ρ_{ij} as the correlation coefficient of any pair of variables i and j as follows:

$$\Sigma_{x_1,x_2,...,x_n} = \begin{pmatrix} \sigma_1^2 & \rho_{12}\sigma_1\sigma_2 & \cdots & \rho_{1n}\sigma_1\sigma_n \\ \rho_{21}\sigma_2\sigma_1 & \sigma_2^2 & \cdots & \rho_{2n}\sigma_2\sigma_n \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{n1}\sigma_n\sigma_1 & \rho_{n2}\sigma_n\sigma_n & \cdots & \sigma_n^2 \end{pmatrix}$$

In Σ, the correlation coefficient ρ_{ij} is a function of the location of nodes i and j as $\rho_{ij} = f(X_i, Y_i, X_j, Y_j)$. This means that ρ depends on the location, length and orientation of the line segment between two nodes. Assuming that the location of the attacker has a random distribution in the network, then the variable x is a stationary random variable independent of node location, resulting in $\rho = f(d_{ij})$. Assuming the same circular radio range for all nodes $A = \pi R^2$, then $\rho_{ij} = (A_i \bigcap A_j)/A$, if $d_{ij} \leq 2R$, and $0,$ otherwise, where $A_i = A_j = A$.

C. Delay Model

In our cooperation model, we assume that aggregator nodes process all received reports at time slot τ and send the aggregated report to their parents at time slot $\tau + 1$. The total delay between a report is transmitted by a node, until it is received by a leader located at a distance of μ_i hops is thus $D_G = \sum_{i=1}^{k} \mu_i$.

D. Network Coverage Model

The final objective function to be maximized is coverage of the nodes in the network. A node which is a member of cluster tree T_i, regardless of its role, is considered covered (if $\exists T_i$ where $n_j \in T_i \Rightarrow c_{ij} = 1$). Accordingly, the only nodes in the network that are not covered are orphan nodes. Thus, the coverage function becomes $C_G = 1 - \frac{|O|}{k}$.

E. Genetic Algorithm for Multi Objective Optimization

The optimization problem, described by Equations 1-6, has properties that limit the methods that can be used for solving it. First, the number of possible cooperation topologies grows exponentially with the number of nodes. This makes the search space of the problem very large and, in terms of time complexity, an NP-hard problem. Next, the problem has some non-linear constraints which are not simple to define(e.g., constraint (3)). Given these constraints, a Genetic Algorithm (GA) is a suitable technique for solving the optimization problem. A GA requires that the problem be presented in an appropriate format. All objectives must be encapsulated in a single fitness function. In the multi-objective case, this is made possible through the use of secondary constraints to the objective function, called penalty methods. The GA, with pseudocode shown in Algorithm 1, begins with an initial random population and iteratively solves the problem using ideas borrowed from biology, such as: selection, crossover, and mutation, over many generations. Problem encoding, fitness function refinement, and all other GA-related steps are discussed below.

Encoding the problem: GA operates on (i.e., its solutions are) chromosomes, or bitstrings of specific length. A solution to our problem is a set of cluster trees $T \in G$ obtained from a given graph G. The set of cluster trees T can be conveniently represented as an adjacency matrix. Consequently, an immediate way to encode our problem for a GA (i.e., represent a solution to our problem as a chromosome) is to assemble the chromosome by concatenating the rows of T's adjacency matrix, in a single bitstring. Hence, the chromosome is composed of multiple segments, where each segment is a row from T's adjacency matrix. As an example, Figure 2 depicts, in the chromosome titled "Instant Chromosome", the encoding for the cluster tree presented in Figure 1(a). Based on our proposed architecture, i.e., a set of tree clusters, a chromosome must have a single bit set to 1 in each segment. Figure 2 also depicts two infeasible solutions (non-chromosomes) in the bitstrings titled "Infeasible Solution". In one infeasible solution, two bits in the same segment are set to 1, while in the second a node has an invalid parent. Through our proposed encoding, it is easy to observe that the length of a chromosome is $|V|^2$.

Algorithm 1 GA Optimization

```
 1: $NumSol \leftarrow S$
 2: Collect Network Info()
 3: Create Adjacency Matrix()
 4: Generate First Population($NumSol$)
 5: Compute Fitness Values(0, $NumSol$, $\alpha$)
 6: for $NumGen = 1$ to $G$ do
 7:     $Elite$ = Find Optimal Solution($NumGen$, $NumSol$)
 8:     $PreGen$ = Selection($NumGen$, $NumSol$)
 9:     $PreGen$ = Crossover($PreGen$)
10:     $PreGen$ = Mutation($PreGen$)
11:     Replace Elite($Elite$)
12:     Compute Fitness Values($NumGen$, $NumSol$, $\alpha$)
13: end for
```

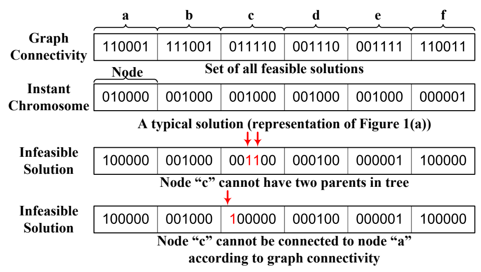


Fig. 2. The connectivity graph is encoded as a string suitable for genetic algorithm. All feasible solutions can be derived from graph connectivity string.

Initial population: The initial population, consisting of S individual chromosomes -each representing a single solution- is randomly generated. A population of size S allows the GA to start with large set of feasible solutions. A chromosome is obtained from a randomly generated bitstring, if it abides by constraints (2-6) of our optimization problem. If the bitstring conforms, then the GA considers it a solution (Algorithm 1, Line 4).

Computation of fitness: The fitness value of a chromosome (i.e., solution) represents its ability to keep genetic properties for next generations. For our particular problem, we compute the fitness value of a solution as a combination of individual functions to be optimized (as presented in Section III). We compute (in Algorithm 1 Line 5) the fitness value of a solution T, which consists of q trees, through a Penalized Function technique. The penalized function we use

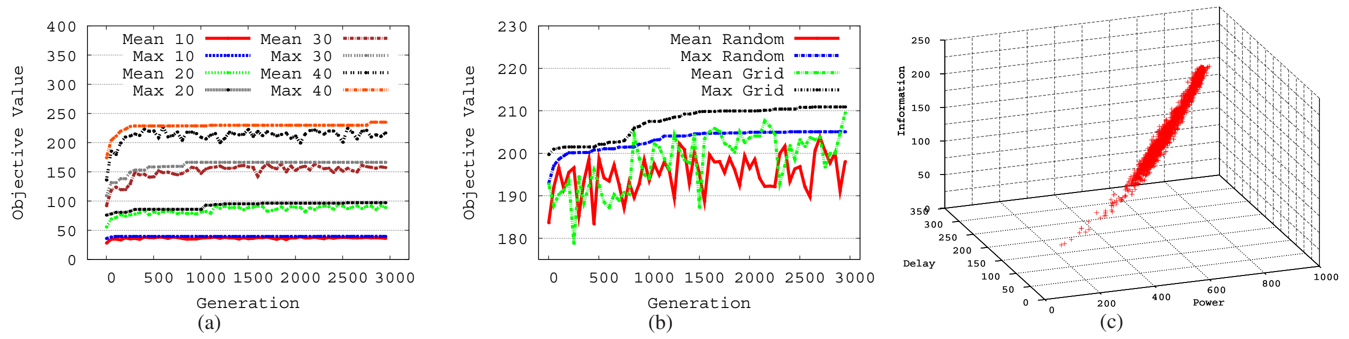


Fig. 3. (a) Convergence of single objective optimization for Information in networks of different sizes. (b) Convergence of a penalized function in a multi-objective optimization problem comparing the best and average fitness values across all generations. (c) A Pareto diagram emphasizes the relationship between objective functions. The high degree of correlation is evidenced by the narrow surface displayed.

is $f(T) = \left[2 - \frac{P_G}{C_P} - \frac{D_G}{C_D}\right] \times \left(\frac{1+\lfloor \frac{C_G}{k} \rfloor}{2}\right) H_G$, where a set of constraints penalizes the fitness value for each solution: the penalty method maximizes data entropy (H_G), while normalized values of power (P_G) and delay (D_G) operate as inverse coefficients of the fitness value. We use coverage as a penalty parameter as well (C_G). To find the constants C_P and C_D, the maximum delay and power consumption, two single objective problems are used. The maximum outcomes of several iterations of these SOO problems are assigned as constants C_P and C_D.

Our GA uses Elitism (Line 7), Selection (Line 8), Crossover (Line 9), and Mutation (Line 10) steps for each generation. We use "tournament" and "roulette wheel" mechanisms, weighted with the chromosome's fitness value, for selection. In addition, our GA algorithm uses three different methods for crossover and two mutation mechanisms (i.e. bitwise or blockwise). Since these methods have been applied to GA before, and due to space constraints, we omit their in-depth description.

V. Performance Evaluation

We implemented the proposed GA in Matlab for both single and multi-objective optimizations. In this section, first, we investigate the convergence, to the best fitness values, of the GA for the single objective optimization problem and demonstrate the robustness of the algorithm for different runs and different types of selection, crossover and mutation. Next, using results from single objective optimization, we run GA with a penalized fitness function to solve the multi-objective case. Results show how considering other highly correlated functions as constraints in a penalized function affects the produced cluster trees.

A. Single Objective Optimization

We investigate the single objective optimization problem for two reasons: to evaluate the robustness of the GA for different evolution methods (e.g., selection, mutation, etc.) and to obtain the values of scaling factors C_P and C_D of the penalized multi-objective optimization function (Section IV-E). The former was done by several runs of single objective optimizations. In this section we present results of single objective optimization for Information. Data entropy, the amount of information collected by each cluster members in different networks is the single objective of our optimization problem. Similar results were obtained for Power and Delay functions but are omitted due to space constraints.

For this set of experiments we employ networks with random node placement. We create four different deployments of size 10, 20, 30, and 40 nodes in a 200×200m^2 area and ran the simulation for solving the SOO problem in each network. We ran 10 simulations for each of the two configurations. In the first configuration, we ran GA on the same initial population with different combinations of selection, crossover and mutation methods to see the correctness of implemented methods. In the second configuration, we selected the best combination of these methods for the maximization case and ran it for different initial populations.

Figure 3(a) shows the convergence of the Information value for four different network sizes and the effect of the network size on it. The variation of average fitness value and best fitness value of each generation is depicted in this figure as well, indicating that the GA successfully approaches an optimal solution. Due to space constraints, the results for all different aforementioned methods and different initial populations are omitted. The results verify the correctness of all the methods and how they approach to surprisingly same optimal value starting from different random populations.

In our optimization problem (Equations 1-6), constraint (6) prohibits the creation of large cluster trees. We chose different values for T_{th} for different network sizes (e.g. 5 for 10-node network and 7 for 20-node network). By changing T_{th} we expected that different numbers of cluster trees will appear in the GA solution. Hence, we used root mean square error (RMSE) to show the bias and variance of number of cluster trees obtained by the GA. We normalized the expected number of cluster trees, for different network sizes to 1 and then calculated RMSE for a given network size. The RMSE values of our estimator's expected value (number of clusters) for networks of sizes 10, 20, 30, and 40 were 0, 0.011, 0.018, and 0.025, respectively. These very small RMSE values indicate that solutions produced by our GA, in terms of number of cluster trees, approach an intuitively optimal value, obtained when cluster trees are equally sized.

B. *Multi-objective Optimization*

As mentioned in Section IV-E, the multi-objective case of our optimization problem (Equations 1-6) is solved using a penalized fitness function. The optimization problem is to maximize the Information while keeping Delay and Power as small as possible, and Coverage as large as possible. We used the solution for the SOO problem to obtain the maximum values for C_P and C_D.

We performed simulations using two networks of 49 nodes in grid and random deployments. Figure 4(a) depicts the obtained cluster tree topologies by the GA, when the objective to maximize is Information objective, but do not consider optimizing Power and Delay. The obtained topologies are largely linear and, while producing solutions that maximize Information, would incur high Power and Delay costs. To assess the effect of the penalized fitness function on the output solutions, we ran GA on the same networks using Power and Delay as penalties to the fitness function. Figure 4(b) show that the results are smaller cluster trees that achieve a balance between all objectives. We omit the results for grid topology, due to space constraints.

The convergence of the algorithm when solving the MOO, using the penalized function, is depicted in Figure 3(b). In addition to algorithm convergence, the figure shows that given a network, the solution for the MOO problem provides less information than when SOO is used. Not represented in this graph, the objective values for Power and Delay remained very low. It also depicts the variation of average fitness and best fitness during convergence time. As shown, the multi-objective fitness value varies more than the single objective case and the average value in the multi-objective function fluctuates more than the single objective fitness value. This fluctuation happens since maximization and minimization objectives are not independent. Whenever a new solution, having higher Information, is obtained, then Power and Delay increase the penalty ratio. This results in removing the good solutions characterized by higher Information values.

A Pareto curve shows the non-dominated solution space of each objective, i.e., Power, Information and Delay, in terms of the other two objectives. It presents the optimal front and the diversity across objectives along this front. The Pareto curve of the penalized function proposed in Section IV-E, is shown in Figure 3(c). As shown, we can identify the maximum possible value for Information that can be collected in a given network, while the other objectives are constrained. The Pareto curve shows that Power, Information and Delay are highly correlated, giving one other explanation for the observed fluctuations in the convergence graph of the multi-objective fitness value.

VI. Security Analysis and Evaluation

A. *Intrusion Detection Delay*

In this section, we show how different cluster tree topologies (i.e., solutions), obtained by solving the single objective and multi-objective optimization problems, handle outsider and insider security attacks. We investigate the intrusion detection

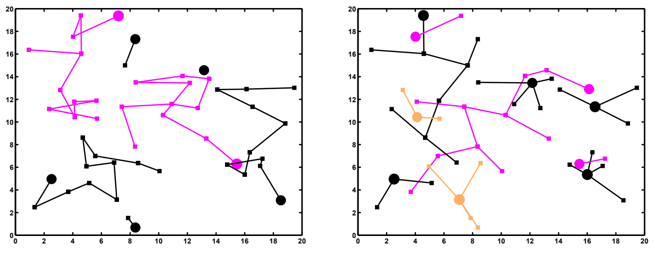

(a) (b)

Fig. 4. Effects of constraints on the output of the optimization problem: (a) Single objective optimization of Information in a random topology; (b) Penalized multi-objective optimization of Information in a random topology.

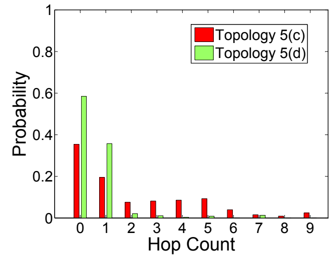
delay for insider and outsider (randomly distributed in the network) attackers. The cooperative IDS solutions we focus on are those obtained in Section V-B where the proposed GA was used for solving SOO and MOO problems, results depicted in Figure 4(a) and Figure 4(b), respectively. For this investigation, we considered the most sophisticated attack reported in this paper, namely *Web exploits*. In our scenario, a leader node within the radio range of an outsider attacker, can detect the attack with a delay of 0 time units (i.e., timeslots employed by our TDMA protocol), since a HW-DS intrusion detection engine is employed by a leader.

In our simulations, we considered 10^5 random locations for the outsider attacker. For the insider attacker case, we randomly chose non-leader nodes as attackers. We estimated the detection delay as the shortest hop count between a node within attacker's radio range, and node's leader. Our simulation results for an outsider attacker scenario are presented in Figures 5(a), where two cooperative IDS solutions are analyzed (i.e., Figures 4(a) and 4(b)). Our results indicate that most of the attacks will be detected in 0 or 1 hops, if the solution is obtained by solving the MOO. If the cooperative IDS solution used is the one obtained from solving the SOO (Figure 4(a)), then the detection delay is much larger. More precisely, the mean detection delay for the SOO solution is 2.28, and for the MOO solution it is 0.58.

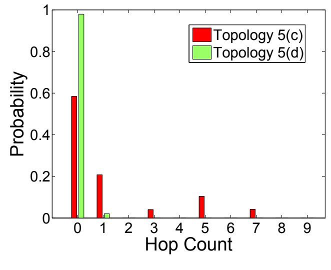
The simulation results for the scenario of an insider attacker are presented in Figure 5(b). Similar with the outsider attacker scenario, a GA solution for the MOO problem is able to detect an insider attacker much quicker than the case when the solution is only for the SOO problem formulation. For the insider attacker scenario, the mean detection delay for the SOO solution is 1.19, and for the MOO solution it is 0.027.

B. *System Implementation*

As a proof of concept of our proposed architecture for cooperative IDS, we built a resource constrained wireless network, consisting of six laptops, and deployed it in an indoor area. The laptop configurations were: 1.8GHz Intel processor with 2MB cache, 1GB RAM running Linux (Ubuntu 6.10); and Intel Core 2 Duo 2GHz, 2MB cache, 4GB RAM running Windows 7. For the intrusion detection engine we used Snort in NIDS mode, the most complex and configurable mode, that analyzes captured network traffic to find any matches between



(a)



(b)

Fig. 5. Probability mass function for delay in reporting attacks to the nearest leader for two architectures depicted in Figures 4(a) and 4(b). (a) Outsider attackers. (b) Insider attackers.

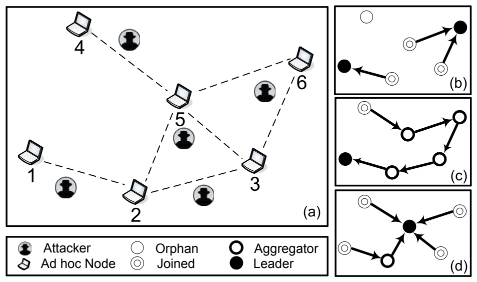


Fig. 6. (a) Our wireless network and the random attacker locations. (b) Random topology including two cluster trees and one Orphan node. (c) SOO topology with one linear tree. (d) MOO topology with one cluster tree.

them and activated rule sets [13]. Although we did not consider event reporting among nodes, the different types of detection engines we used (i.e, LW-DS, RE-DS and HW-DS) emulated such costs, if we had had reporting capabilities available.

To have different configurations of Snort with diverse power, memory and CPU consumption and detection power, we disabled proportions of all available rulesets. We disabled 2/3 of detection rules excluding *bad-traffic* and *icmp* in the LW-DS configuration (run by either orphan or joined nodes). Thus, nodes that had LW-DS version only detected flooding attacks. For aggregator nodes that used the RE-DS configuration, only 1/3 of rules, excluding *bad-traffic icmp* and *scan*, were disabled, so that flooding and port scanning attacks could be detected. Leader nodes ran the the HW-DS version of Snort which used all rulesets for detecting any possible attack, among the three we consider in this paper.

The locations of the 6 adhoc nodes were randomly chosen, and are shown in Figure 6(a). We selected 5 random positions for the outsider attacker. Attackers were enabled to run the 3 types of attacks considered in this paper. Adhoc nodes had different configurations of Snort depending on their assigned role by our proposed cooperative IDS architecture. To evaluate the effectiveness of our collaborative IDS, we evaluated 5 different cluster tree topologies. Two were obtained by using the proposed GA solution for the SOO and MOO problems. These solutions are depicted in Figure 6(c) and (d), respectively. A third solution we considered was a "Random" solution, shown in Figure 6(b), which assigns detection responsibilities in a random manner. The final two solutions we considered, not depicted, were "All LW" and "All HW" where all 5 nodes execute LW-DS and HW-DS detection engine, respectively.

The results show that the "All HW" scenario achieves a detection rate (DR) of 100%. In this solution, however, all nodes execute the most sophisticated IDS, and thus exhaust resources the fastest. At the other extreme, when all nodes execute the lightest IDS, the detection rate suffers - it was only 33%. The solution obtained by the proposed GA that solves both SOO and MOO achieve high detection rates of 73% and 93%, respectively, superior to a random assignment of roles to nodes, which achieves only a 60% detection rate.

VII. Conclusions

Given the limited resources available to some wireless networks [1], intrusion detection can be difficult. The task is complicated because individual node parameters, like power, must be considered alongside network parameters like degree of coverage and permissable delay. Optimizing for just one of these parameters can have dramatic negative effects on others. In this paper, we have investigated the relationship between energy efficiency and parameters that affect security effectiveness. Through extensive simulations and system implementation we show the superior network performance and intrusion detection rates obtained by our collaborative IDS.

Acknowledgement. This work was funded in part by NSF grant CNS 0923203.

References

[1] S. George, W. Zhou, H. Chenji, M. Won, Y. O. Lee, A. Pazarloglou, R. Stoleru, and P. Barooah, "Distressnet: a wireless ad hoc and sensor network architecture for situation management in disaster response," *Communications Magazine, IEEE*, vol. 48, no. 3, pp. 128 –136, 2010.
[2] S. Sen and J. A. Clark, "Intrusion detection in mobile ad hoc networks," *Guide to Wireless Ad Hoc Networks*, pp. 1–28, 2009.
[3] F. Hugelshofer, P. Smith, D. Hutchison, and N. J. Race, "OpenLIDS: a lightweight intrusion detection system for wireless mesh networks," in *Conf. on Mobile Computing and Networking*, 2009.
[4] G. Li, J. He, and Y. Fu, "A distributed intrusion detection scheme for wireless sensor networks," in *ICDCS*, 2008.
[5] A. P. R. da Silva, M. H. T. Martins, B. P. S. Rocha, A. A. F. Loureiro, L. B. Ruiz, and H. C. Wong, "Decentralized intrusion detection in wireless sensor networks," in *Q2SWinet*. ACM, 2005, pp. 16–23.
[6] B. Sun, K. Wu, and U. W. Pooch, "Alert aggregation in mobile ad hoc networks," in *Workshop on Wireless Security*, 2003.
[7] D.-H. Shin and S. Bagchi, "Optimal monitoring in multi-channel multi-radio wireless mesh networks," in *MobiHoc*. ACM, 2009, pp. 229–238.
[8] D. Subhadrabandhu, S. Sarkar, and F. Anjum, "A framework for misuse detection in ad hoc networks-part i," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 274 – 289, feb. 2006.
[9] O. Kachirski and R. Guha, "Effective intrusion detection using multiple sensors in wireless ad hoc networks," in *System Sciences*, 2003.
[10] T. Srinivasan, V. Mahadevan, A. Meyyappan, A. Manikandan, M. Nivedita, and N. Pavithra, "Hybrid agents for Power-Aware intrusion detection in highly mobile ad hoc networks," in *Systems and Networks Communication*, 2006.
[11] Y. an Huang and W. Lee, "A cooperative intrusion detection system for ad hoc networks," in *Workshop on Security of AdHoc and Sensor Networks*, 2003.
[12] I. Krontiris, Z. Benenson, T. Giannetsos, F. C. Freiling, and T. Dimitriou, "Cooperative intrusion detection in wireless sensor networks," in *Wireless Sensor Networks*. Springer-Verlag, 2009, pp. 263–278.
[13] *SNORT Users Manual v2.9.0*, The Snort Project, September 2010.
[14] L. Galluccio, S. Palazzo, and A. Campbell, "Efficient data aggregation in WSN: An entropy-driven analysis," in *PIMRC*, 2008.
[15] T. M. Cover and J. Thomas, *Elements of Information Theory*. Wiley, 1991.