Optimising IDS Sensor Placement

Hao Chen, John A. Clark
Department of Computer Science
The University of York
York, UK
Email: chenhao,jac@york.ac.uk

Siraj A. Shaikh, Howard Chivers, Philip Nobles
Department of Informatics and Sensors
Cranfield University
Shrivenham, UK
Email: s.shaikh,h.chivers,p.nobles@cranfield.ac.uk

***Abstract*—In large network environments multiple intrusion detection sensors are needed to adequately monitor network traffic. However, deploying and managing additional sensors on a large network can be a demanding task, and organisations have to balance their desire for detecting intrusions throughout their network with financial and staffing limitations. This paper investigates how intrusion detection system (IDS) sensors should best be placed on a network when there are several competing evaluation criteria. This is a computationally difficult problem and we show how Multi-Objective Genetic Algorithms provide an excellent means of searching for optimal placements.**

I. Introduction

In large network environments, particularly those with many network segments and those with multiple Internet access points, network administrators have generally placed multiple IDS sensors along the network perimeters, typically around firewalls, or near the node to be protected, to monitor network traffic.

There are many reasons for using multiple IDS sensors. For example, by deploying sensors on various network segments, we can tune each of them to the traffic that we typically see on that segment, which means we could identify and locate suspicious activities more quickly. However, the detection of intrusions in large volumes of data, in the absence of semantic hints provided by prior knowledge of the intrusion type, is fundamentally limited by the low ratio of malicious events [1]. For example, it is not obvious that deploying IDS sensors in larger numbers would improve detection quality – diminishing returns are likely to be evident early. Neither is it feasible to deploy more and more sensors given the costs and the manual engagement required to monitor for potential intrusions.

Furthermore, determining where to place a set of sensors to create cost effective intrusion detection is a difficult task. There may be several evaluation criteria for placements, seeking to maximise various desirable properties (e.g. various attack detection rates), whilst seeking to reduce undesirable properties (such as false alarm rates as well as purchase, management, and communications costs). Subtle tradeoffs may need to be made between the properties; different placements may have complementary strengths and weaknesses, with neither placement being uniformly better than the other.

However, engineering regularly deals with such difficult *multi-criteria optimisation* problems and has developed a powerful suite of technical tools to facilitate the search for high performing solutions. In this paper we show how a multi-objective genetic algorithm (MOGA) can be harnessed to address the sensor placement problem.

The optimal placement of sensors depends on what we wish to achieve. A placement may be optimal for the detection of one type of attack, but not for a second type of attack. We may seek a placement that gives good chances of detecting each of several types of attack; this may yield a different optimal placement. To determine the "optimal" placement we need a means to evaluate a particular placement. In some cases, this may be carried out with respect to statically assigned information (e.g. location of firewalls and servers). In others, we may need to simulate attacks and measure the effectiveness of the placement. Thus the specific evaluation mechanism may differ but the overall technique remains the same: find a placement P that optimises some evaluation function $f(P)$, or a set of evaluation functions $f_1(P), \ldots, f_n(P)$. Such a situation is a suitable target for the application of heuristic optimisation.

The Genetic Algorithm (GA) [2] is a heuristic optimisation technique based loosely on natural selection and has been applied successfully in the past to a diverse set of problems. Its general idea is that populations evolve according to rules that will in general support the emergence of ever fitter individuals (that is, ones with higher evaluation value). As with other search methods, GA can be used in conjunction with Multi-Objective Optimisation (MOO) techniques [3]. MOO aims to find solutions that satisfy more than one objective, so that a solution's ability to solve a problem is assessed by a set of objective functions $f_1, \ldots, f_n$. MOO methods return a set of solutions in a single run, and each solution achieves a different balance between multiple objectives. In this paper, we experiment with GA and MOO to evolve optimal sensor placements. These experiments serve as proof of concept and to demonstrate the validity and potential of the proposed approach. Researchers have used Genetic Programming (GP) and Grammatical Evolutioin to

978-0-7695-3965-2/10 $26.00 © 2010 IEEE
DOI 10.1109/ARES.2010.92

determine IDS detection rules [4], but our experiments here report the first use of heuristic optimisation techniques to evolve optimal IDS sensor placements.

II. Related Work

Noel and Jajodia [5] propose to use attack graph analysis to find out optimal placement of IDS sensors. Attack graphs represent a series of possible paths taken by potential intruders to attack a given asset. Such graphs are constructed in a topological fashion taking into account both vulnerable services that allow nodes to be exploited and used as launch pads, and protective measures deployed to restrict connectivity. The purpose is to enumerate all paths leading to given assets and where optimal placement is devised to monitor all paths using minimal number of sensors. This is seen as a set cover problem: each node allows for monitoring of certain graph edges and the challenge is to find a minimum set of routers that cover all edges in the graph; a greedy algorithm is then used to compute optimal placement. The use of attack graphs provides an efficient mapping of network vulnerabilities in the network. A vulnerability-driven approach to deploying sensors overlooks factors such as traffic load however. As a result the placement is optimised such that the more paths that go through a node the more likely it is chosen for placement.

Rolando [6] introduces a formal logic-based approach to describe networks, and automatically analyse them to generate signatures for attack traffic and determine placement of sensors to detect such signatures. Their notation to model networks is simple yet expressive to specify network nodes and interconnecting links in relevant detail. While there are advantages to using a formal model, such an approach may not be scalable. The formal notation allows for a more coarse-grained specification but it is not clear whether the resulting sensor configurations are even likely to be feasible for real environments. Moreover, the notation does not allow for modelling any system-level characteristics.

III. Experimental Setup and Evaluation

A. Network Simulation

We use Network Simulator NS2 [7] to simulate our experimental network as shown in Figure 1. The whole network consists of 180 nodes, where node 0 represents the outside world, nodes 1 to 19 are the routers interconnecting various parts of the network, nodes 20 to 39 are servers offering valuable services to users and therefore critical assets that need to be protected, and nodes 40 to 180 are ordinary clients some of which may be compromised by intruders to attack critical assets. The network is organised as such that the servers are distributed over six subnets and the clients are distributed over seven separate subnets.

We simulate real intrusive behaviour to analyse how such behaviours could be efficiently detected by the proposed approach. The intrusive behaviour we simulated is to do with probing and information gathering, the purpose of which is to assess a potential target's weaknesses and vulnerabilities [8]. For example, an intruder may strive to detect active hosts and networks that are reachable and the services they are running that could be successfully exploited. Detecting and preventing such probes therefore is important both to inhibit exposure of information and prevent attacks that follow.

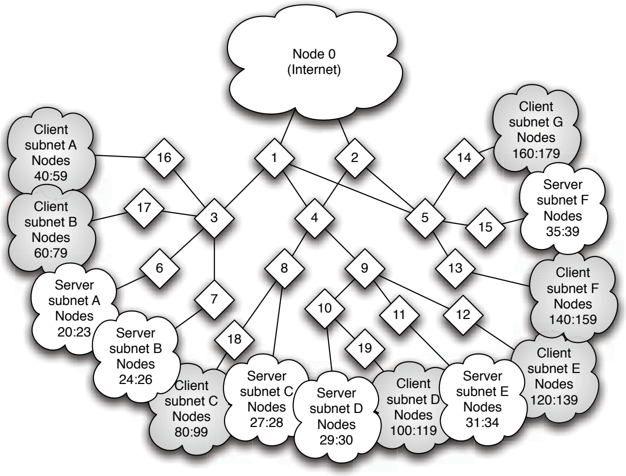
Node 0
(Internet)
Client subnet A Nodes 40:59
Client subnet B Nodes 60:79
Server subnet A Nodes 20:23
Server subnet B Nodes 24:26
Client subnet C Nodes 80:99
Server subnet C Nodes 27:28
Server subnet D Nodes 29:30
Client subnet D Nodes 100:119
Server subnet E Nodes 31:34
Client subnet E Nodes 120:139
Client subnet F Nodes 140:159
Server subnet F Nodes 35:39
Client subnet G Nodes 160:179
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19

Figure 1. Simulated Network

We simulate a probe attack scenario where various servers are probed from the outside (through node 0) and inside from clients, hence the simulation consists of both external and internal attacks. An intruder may subvert a node in any of the client subnets to probe any of the servers. Intruders (picked randomly) from each of the client subnets, that are client nodes 45, 78, 95, 111, 133, 157 and 178, probe server nodes 20 to 38. In addition, node 45 also attempts a probe on neighbours 46 and 47. A total number of of 154 instances of probe attack are injected.

Note the simulation of attacks so far in our experiments is simple for the purposes of demonstration. Simulation of more subtle intrusive behaviours and research of how such behaviours could be effectively and efficiently detected by our approach are currently under investigation.

In order to investigate how the false alarms may influence sensor placement strategy, we simulate not only a number of attacks but also background network traffic. The background network traffic is generated randomly by NS2 traffic source generator *cbrgen*. In the experiment, we assume that traditional IDS metrics such as false positive rate and false negative rate are already known. This hypothesis stands as all IDS evaluation work so far is trace-driven [9], suggesting when evaluating IDSs, we use a data set where we know the ground truth, i.e., what data are attacks and what data are normal. Thus we can easily find out the metrics such as false positive rate and false negative rate. If the testing data set is a very representative sample of the operation environment, we can use the metrics in the testing data to approximate the real world situation. In our experimental framework we assume all sensors are identical and configured to exhibit a detection rate of 95% and a false positive rate of 0.1%. These figures are in accordance with the features claimed by most IDS products.

We characterise expected monitoring costs for the network. Such costs are dependant on the load of the traffic at a specific location in the network: the busier the location, the higher the levels of activity monitored (including false alarms), and therefore bigger the effort. We restrict the costs to a range of values 1 to 10 to express relative monitoring costs for different locations on a network. In the experiments, we characterise expected monitoring costs to reflect an operational network in the real world: routers 1 – 19 serving at the heart of the network are assigned a cost relatively much higher.

Router nodes 1 and 2 are assigned a cost of 8, as they serve to link the entire network with the outside world, and also interconnect internal traffic between routers 3, 4 and 5. Router nodes 3, 4, 5 and 9, further down the hierarchy, are all assigned a cost of 7. The rest of the cost assignments broadly follow from this. Nodes 8 and 10 are assigned a cost of 6. Nodes 6, 11 and 15 are assigned a cost of 5 to indicate they link more servers and hence are busier. We assign a flat cost of 4 for all the other subnet router nodes. All server nodes are assigned a cost of 3 and client nodes a cost of 1.

B. Fitness Measurement

The fitness of a sensor placement is determined by its ability to satisfy four objectives: the number of sensors, detection rate, false alarm rate and monitoring cost.

Equation (1) is used to minimise the number of sensors, and the $nSensors$ represents the number of sensors.

$$f_1(P) = nSensors \tag{1}$$

Equation (2) is used to maximise the detection rate of a sensor placement. The $nDetectedAttacks$ represents the number of distinct attacks that have been detected; $nAttacks$ represents the number of all simulated attacks we have injected in the data set (i.e. 154 probe attacks). Note that we implement the sensors to detect attacks in a cooperative manner, which means duplication of alarms is avoided, and also cooperating sensors are able to detect attacks they may not detect independently.

$$f_2(P) = \frac{nDetectedAttacks}{nAttacks} \tag{2}$$

Equation (3) is used to minimise the false alarm rate of a sensor placement. The $nFalseAlarms$ represents the number of false alarms that are raised by the sensors. The $nAllAlarms$ represents the number of all alerts that are reported by the sensors. It is a sum of the number of detected attacks (a.k.a. true alarms) and the number of false alarms. So $f_3(P)$ follows precisely the definition of false alarm rate.

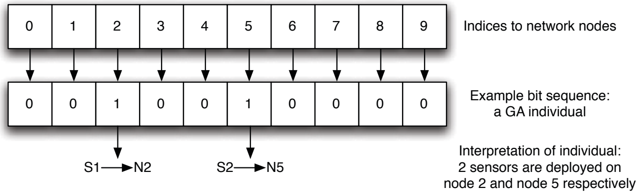
0
1
2
3
4
5
6
7
8
9
Indices to network nodes
0
0
1
0
0
1
0
0
0
0
Example bit sequence: a GA individual
S1
N2
S2
N5
Interpretation of individual: 2 sensors are deployed on node 2 and node 5 respectively

Figure 2. Sensor Placement Representation

$$f_3(P) = \frac{nFalseAlarms}{nAllAlarms} \tag{3}$$

Equation (4) is used to minimise the total monitoring cost. We use $TotalCost$ to express the total monitoring cost of a set of sensors which are deployed on a network.

$$f_4(P) = TotalCost \tag{4}$$

C. Sensor Placement Representation

In our implementation, a feasible sensor placement is represented by n (i.e. the number of network nodes) bits. Figure 2 is an example of how to interpret a bit sequence into a feasible sensor placement. In this example, we are going to deploy IDS sensors onto a small network of 10 nodes. There are 1023 (i.e. $2^{10} - 1$) distinct individuals, hence 1023 feasible sensor placements in total. Note that the figure 2 serves as a simple example, our actually experiment network has 180 nodes.

D. Parameters for the Search

Our implementation makes use of the versatile toolkit ECJ [10]. The major parameters for the GA search are as follows: the population size is 1500; the number of generations is 250; the crossover probability is 0.95 whereas the mutation probability is 0.05; the selection method is tournament of size 2.

To carry out multi-objective optimisation, an implementation of the Strength Pareto Evolutionary Algorithm 2 (SPEA2) algorithm was written as an extension to ECJ, which followed precisely the original algorithm specified by Zitzler et al [11]. The algorithm retains an archive of non-dominated individuals, which are individuals that cannot be improved upon in terms of all objectives by any other single individual within the archive. The algorithm attempts to use the archive to approximate the pareto front, a surface of non-dominated individual with objective space. We set the archive size for the multi-objective SPEA2 to 128.

The settings for parameters not listed here are given by the parameter files *simple.params* and *ec.params* supplied with the ECJ Toolkit. One of our experiment purposes

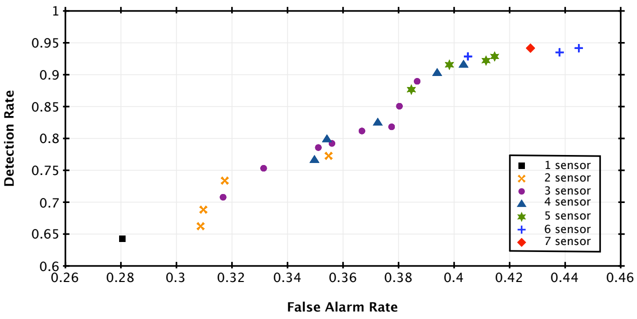

Figure 3. Relation between the number of sensors and the pair of detection rate and false alarm rate

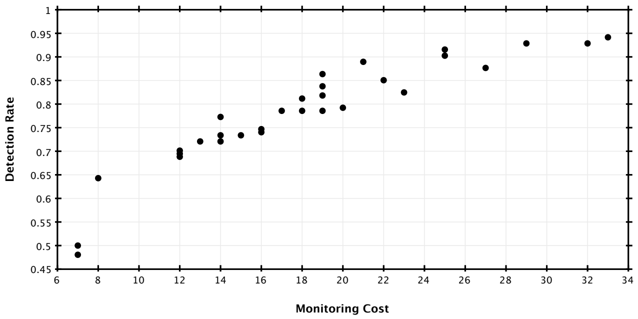

Figure 4. Detection rate increases as the budget monitoring cost increases

is to demonstrate the validity and potential of the multi-objective approach to functional trade-offs in general, and so no parameter tuning was attempted.

E. Experiment Results

In the first experiment, we investigate the relations between the number of sensors and detection quality (in terms of the pair of detection rate and false alarm rate), and search for placement given constraints on the number of sensors available to deploy. We plot our experiment results in Figure 3, where each point corresponds to a placement's properties in the objective space. Note that this is not a ROC curve. The FA rate counts the fraction of FAs in the set of alerts generated, which is not equal to the false positive rate (fraction of non-attack events which raise an alarm). The results validate our multi-objective optimisation approach and demonstrate that functional trade-offs are indeed possible for sensor placement problem.

Figure 3 shows the trend that the more sensors we use, the more attacks we will be able to detect (higher detection rates), whilst the more false alarms (higher false alarm rate) we will have to dismiss. Intuitively, by deploying multiple sensors on various network segments, we can tune each of them to the traffic that we typically see on that segment; due to the increased network visibility, more attacks are detected as more sensors are deployed. False alarm rate depends in practice on many factors (e.g. signature quality, volume of background traffic etc.). In this experiment, because we use sensors with the same settings, the false alarm rates were dominated by the volume of background traffic at different nodes. The more sensors are deployed, the higher volume of background traffic they will see, hence the higher false alarm rate.

Note that deploying more sensors may help to reduce false alarm rate in some situations. For example, both the placement with 7 sensors deployed on nodes 1, 12, 15, 16, 17, 18, 19 (the red diamond point on top right; also see Table I) and the placement with 6 sensors deployed on nodes 3, 8, 9, 15, 16, 19 (the blue cross on top right) have a detection rate of 94.15%. However, the placement with 7 sensors has a lower false alarm rate of 42.75%. The false alarm rate of the placement with 6 sensors is 44.49%. This result means we may get better detection quality with one more sensor.

The second experiment is designed to determine the minimum monitoring cost needed to detect certain amount of attacks, and the criteria of amount of sensors is omited. We find that attack detection rate increases as the budget monitoring cost increases.

Observe that how half of attacks are detected using a budget of 7, and over two-thirds of attacks are detected using only a budget of 12, whereas to detect 90% of the attacks, a total budget of 25 is needed. Beyond this, the entire budget needs to be 33 to achieve only a marginal gain. It is safe to conclude that the return in terms of attack detection is diminished, as shown in Figure 4, as more budget is sanctioned. Nevertheless, given a reasonable budget, it is possible to effectively detect a majority of the attacks if the sensors are optimally placed. Note that we use monitoring cost to replace the number of sensors as a search criteria, hence the placements found in the second experiment are not necessary identical with the placements found in the first experiment. For example, the placement with one sensor deployed on node 3 (monitoring cost of 7, detection rate of 50%) is not found in the first experiment.

In practice, we may often have to deal with budget constrained questions, for example, if we have a budget of 20, how should we choose IDS and how to deploy and configure IDS sensors? In the third experiment, we try to answer this question using the multi-objective optimisation techniques. We ask our program to search for placements which have monitoring costs in the range of 16 to 22 (i.e. from -20% percent to +10% of the original budget of 20). We plot our experiment results in Figure 5.

Each point on Figure 5 represents a sensor placement, and the figure on the right of each point is the monitoring cost of the placement. Observe that we actually have a number of choices in this range. The red circle, which represents a placement which has monitoring cost of exact 20, has four sensors deployed on nodes 1, 12, 17 and 18. It is, however,

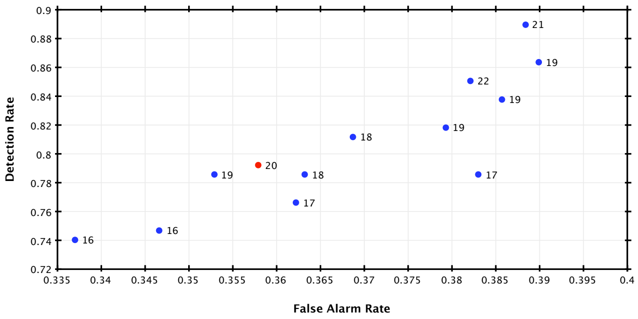

Figure 5. DRFARandSensors

not necessary the best option we could have in this budget range. For example, we could have higher detection rate with a lower monitoring cost of 18 or 19. Although we will have a little higher false alarm rate with these options, we do save budget and achieve a higher detection rate. On the other side, if we would like to accept a little bit more budget, for example a cost of 21, we could get even higher detection rate of 89%. The third experiment we report here successfully demonstrate the multi-optimisation technique can be a very powerful tool to help to find cost-effective sensor placements.

In the interests of brevity, we list some selected placements options that were determined in the experiments in Table I. For example, the first sensor placed on node 1 is able to detect 64.29% of all attacks. This is due to the location of node 1 (see Figure 1). It provides a strategic advantage, as it serves to link over half of the network (through nodes 4 and 5) with the other half (through node 3).

IV. Conclusions and Further Work

Means to reason and compare IDS sensor placements are important to judge the potential ability of such sensors to make a difference individually or in combination. The nature of sensor placement problem is such that there are too many criteria to consider when making a cost-effective decision, hence a multi-objective optimisation problem. Our experiments demonstrate the validity and potential of the multi-objective approach to sensor placement trade-offs and provide incremental placement options.

The work presented in this paper is a deliberate attempt to use GA and MOO techniques to assist network administrators to choose IDS sensor placement that effectively satisfies multiple criteria. The placement strategies generated, although simple, are typical places that network administrators would likely deploy IDS sensors. The ease with which the approach generated placements satisfying realistic security requirements merits further investigation of the technique. Experimentation and our general knowledge of intrusion detection systems have allowed us to identify numerous possible improvements to the approach and tool support. These are outlined below.

A straightforward extension of this work would be to incorporate an increased number of security requirements. Sensor placement is critical to providing effective defence. Optimal placement for this purpose would seek to minimise damage caused by intrusions. Placements that seek to maximise the number of victims detected could be useful in identifying locations best for detecting attacks likely to have more adverse impact. Such placements could be particularly important to detect and mitigate worm propagation and network probes (such as ping sweeps).

So far in the experiments we have dealt with network nodes in equal importance. In practice, some nodes are more significant to merit monitoring depending on the level of risk associated with individual nodes. Such level of risk needs to take into account both the value of assets and services offered and the likelihood of intrusions targeting them. One future work we are planning is to assign quantitative information (e.g. level of risk) to individual nodes and provide a model (e.g. the sensor deployment model by Shaikh [12]) to assess the information and incorporate it into the multi-objective optimisation framework.

References

[1] P. Helman and G. Liepins, "Statistical foundations of audit trail analysis for the detection of computer misuse," *IEEE Transactions on Software Engineering*, vol. 19, no. 9, pp. 886–901, 1993.

[2] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1989. [Online]. Available: http://portal.acm.org/citation.cfm?id=534133

[3] C. A. C. Coello and L. Nacional, "An updated survey of ga-based multiobjective optimization techniques," *ACM Computing Surveys*, vol. 32, pp. 109–143, 1998.

[4] W. Lu and I. Traore, "Detecting new forms of network intrusion using genetic programming," in *Proceedings of the 2003 Congress on Evolutionary Computation*, 2003.

[5] S. Noel and S. Jajodia, "Attack graphs for sensor placement, alert prioritization, and attack response," in *Cyberspace Research Workshop*, 2007.

[6] M. Rolando, M. Rossi, N. Sanarico, and D. Mandrioli, "A formal approach to sensor placement and configuration in a network intrusion detection system," in *SESS '06: Proceedings of the 2006 international workshop on Software engineering for secure systems*. ACM, 2006, pp. 65–71.

[7] T. Issariyakul and E. Hossain, *An Introduction to Network Simulator Ns2*. Springer, 2008. [Online]. Available: http://www.springer.com/engineering/signals/book/978-0-387-71759-3

[8] S. A. Shaikh, H. Chivers, P. Nobles, J. A. Clark, and H. Chen, "Network reconnaissance," *Network Security*, vol. 11, pp. 12–16, 2008, elsevier.

No. of Sensors	Detection Rate	False Alarm Rate	Placement Options	Monitoring Cost
1	64.29%	28.06%	Node 1	8
2	73.38%	31.74%	Nodes 1, 3	15
2	77.27%	35.48%	Nodes 3, 4	14
3	78.57%	35.11%	Nodes 1, 3, 12	19
3	88.96%	38.67%	Nodes 3, 5, 9	21
4	82.47%	37.25%	Nodes 1, 3, 12, 19	23
4	91.56%	40.34%	Nodes 3, 8, 9, 15	25
5	87.66%	38.46%	Nodes 1, 3, 12, 18, 19	27
5	91.56%	39.83%	Nodes 3, 4, 12, 15, 19	27
6	92.86%	40.50%	Nodes 1, 3, 12, 15, 18, 19	32
6	94.16%	44.49%	Nodes 3, 8, 9, 15, 16, 19	33
7	94.16%	42.75%	Nodes 1, 12, 15, 16, 17, 18, 19	33

Table I
EXAMPLE PLACEMENT OPTIONS OF VARYING QUALITY AND COST

[9] G. Gu, P. Fogla, D. Dagon, W. Lee, and B. Skoric, "Measuring intrusion detection capability: an information-theoretic approach," in *ASIACCS '06: Proceedings of the 2006 ACM Symposium on Information, computer and communications security*. ACM, March 2006, pp. 90–101.

[10] S. Luke, "A java-based evolutionary computation research system," 2008, available as http://cs.gmu.edu/ eclab/projects/ecj/.

[11] E. Zitzler, M. Laumanns, and L. Thiele, "Spea2: Improving the strength pareto evolutionary algorithm," Swiss Federal Institute of Technology, Tech. Rep. 103, 2001.

[12] S. A. Shaikh, H. Chivers, P. Nobles, J. A. Clark, and H. Chen, "A deployment value model for intrusion detection sensors," in *3rd International Conference on Information Security and Assurance*, ser. Lecture Notes in Computer Science, vol. 5576, 2009, pp. 250–259.