3-2009

Network Security Using Self Organized Multi-Agent Swarms

Eric M. Holloway
Air Force Institute of Technology

Gary B. Lamont
Air Force Institute of Technology

Gilbert L. Peterson
Air Force Institute of Technology

Recommended Citation

E. M. Holloway, G. B. Lamont and G. L. Peterson, "Network security using self organized multi agent swarms," 2009 IEEE Symposium on Computational Intelligence in Cyber Security, Nashville, TN, USA, 2009, pp. 144-151, doi: 10.1109/CICYBS.2009.4925102.

Network Security Using Self Organized Multi Agent Swarms

Eric M. Holloway and Gary B. Lamont and Gilbert L. Peterson
Department of Electrical and Computer Engineering
Graduate School of Engineering and Management
Air Force Institute of Technology
Wright-Patterson AFB, Dayton, OH 45433
(eric.holloway, gary.lamont, gilbert.peterson@afit.edu)

***Abstract*—Computer network Cyber-security is a very serious concern in many commercial, industrial, and military environments. This paper proposes a new computer network security approach defined by self organized agent swarms (SOMAS) which provides a novel computer network security management framework based upon desired overall system behaviors. The SOMAS structure evolves based upon the partially observable Markov decision process (POMDP) formal model and the more complex Interactive-POMDP and Decentralized-POMDP models. Example swarm specific and network based behaviors are formalized and simulated. This paper illustrates through various statistical testing techniques, the significance of this proposed SOMAS architecture.**

***Index Terms*—computer network security, cyberspace, agent swarms, self-organization**

I. Introduction

With the increasing number of computer network threats, intrusions and internal anomalies, computer security has become a serious concern of commercial, industrial, and military organizations from financial activities and power system operations to internet information communication and aircraft reconnaissance and attack activities. Thus, outside intrusion detection systems, insider covert network detection, and system anomaly detection techniques are important security tools in cyberspace. Many, many such systems have been proposed using standard hierarchical management structures with identification of features employing classical pattern recognition algorithms. Also evolved are bio-inspired approaches from artificial immune system constructs [16] to particle swarm approaches [9] with support vector machines [19], as well as incorporating multi-objective aspects [8]. Various commercial packages embed many of the associated algorithms. Nevertheless, such network security systems are generally quite slow and limited when operated automatically. Humans in the loop provide better performance due to insight and intelligence.

The goal is to provide better effective and efficient network security real-time performance using a swarm of autonomous self organized agents that evolve a non-hierarchical entangled cyberspace security management structure. In particular, this paper defines a self-organized multi-agent swarm (SOMAS) system with a limited number of desired behaviors. However, these particular scenarios or behaviors only demonstrate SOMAS feasibility and are not comprehensive of the system's capability. SOMAS can be applied to a wide variety of other network security problems, such as intrusion detection, network defense, hiding high value targets, etc. as well as issues not only restricted to network defense.

The paper discusses current computer network threats in Section II. Section III presents the Self Organized Multi Agent Swarms (SOMAS) for decentralized security with its design developed in Section IV. The formalism of the agent swarm behavior is discussed in Section V. The experimental testing of the SOMAS concept is presented in Section VII with testing and analysis to follow in Section VIII. Finally, a summary of the investigation along with avenues of further research are discussed in Section IX.

II. Current Network Threats Defense

Network security threats run the gambit from internal to external attacks. Network defense of these attacks can be structured from a centralized or decentralized framework. All these elements of network security are addressed.

A. Nature of Threat

Web Based Insider Attacks: According to the multitude of internet analysis, we are fighting a losing battle against those who create malware. Although, for example, the external worm threat is currently under control, the computer virus signature types have grown to astronomical numbers. While any computer that is hooked up to the internet is guaranteed to be attacked within 20 minutes, most attacks are ineffective. Consequently, the malware contemporary effort has moved to the client side, embedding exploits in web pages and emails.

The implication is that the threat has moved inside the network. Thus, border control is no longer adequate. Intrusion detection must look both at network traffic and host activity. Any kind of defensive system must be able to handle an internal threat by identifying it, quarantining it, and eliminating the malicious entities involved.

Denial of Service Attacks: Even though the threat of external infiltration by worms and viruses is low, denial of service attacks are still a very real problem, as recent events in Estonia show. The threat is heightened when the attacking computers can be within as well as outside the targeted network, brought about by internal intrusion.

Information Exploitation and Corruption: The degradation of network performance results in the corruption and destruction of information. Malicious agents also exploit and remove confidential information from the networks.

Counter Defense: Finally, the defenders of all the information are the indirect, yet primary, target of malicious agents; since defense keeps attackers from desired information.

B. Defensive Network Environment Designs

Secure Middleware: To defend against a network attack, a comprehensive solution is needed, allowing security to quickly be pushed out to all nodes and rapidly report back incidents. This security solution is a form of middleware, a distributed computing system that all users of the network interact. The Air Force is developing secure middleware called Cybercraft [15]. While the exact implementation design is in development, it is based on a container model. Each node receives and deploys software "payloads" governed by a policy.

Efficiency and Flexibility of Distributed Systems: Stytz, et al [26], argue military networks require a distributed intelligent agent framework for security to avoid the weaknesses of centralized control structures, i.e. lack of scalability, single points of failure, fragility. Servat and Drogoul [24] predict future networks are characterized by a ubiquity of mobile end devices, even nanotechnology. Such large, heterogeneous environments make centralized control extremely difficult and costly, further implying that a MAS is necessary for system control.

In [12] and [23], the authors argue a MAS is given a greater range of ability by mobilizing software agents. Still, there is a danger of devolving into chaos; and mobility adds new degrees of freedom, threatening to produce unstable systems.

III. Empirical and Theoretical Basis of Self Organization, and Application to Decentralized Control

The usefulness of SO is demonstrated and applied to the problem of decentralizing control in a MAS.

The approach to creating mobile, decentralized agent swarms lies in the much explored, but not fully understood, subject of self organization (SO). SO consists of global properties, such as goals and patterns, emerging from autonomous, local interactions, observations, and knowledge of agents in a MAS. Many natural ecosystems exhibit SO, such as termite mounds and bacteria cultures, and SO is prevalent throughout the natural world [5].

These levels are called emergent, since they are both completely composed and constructed by a simpler sub-system, yet are not reducible to the subsystem [4]. For example, chemical laws operate based on physical laws, but physical laws cannot generate chemical laws, which is why chemistry and physics are distinct fields. The formal irreducibility of emergent levels is suggested by Gödel's incompleteness theorem [4]. This evidence demonstrates the terms "emergence" and "self-organization" refer to objective phenomena and are useful topics of study and application, instead of subjective phenomena caused by limitations of the human mind.

SO is useful for security because a self organized MAS can respond to dynamic environments without centralized control and develop continuously changing tangled hierarchies. Tangled hierarchies are hierarchies where different levels of control mutually influence each other in a multi-directional feedback loop. Tangled hierarchies circumvent the problems of normal hierarchies while providing systemic control.

In order to create a MAS that exhibits SO, it is necessary to rigorously identify when SO is occurring. The approach taken by Nagpal [20] is to define a set of rules that produce SO behavior. Another approach is to create a general SO metric, as Shalizi, et al [25] have done. This metric algorithmically highlights the sections that humans visually identify as organized in a spatiotemporal network simulation. It is considered consistent with information theory [22]. The metric is usable as an objective function in solution space searches. The wide variety of rules that can be used to construct the swarm is a benefit of this top-down approach, as opposed to the limited rule set required by the bottom up design of SO. On the other hand, the search space to optimize the SO objective is very large, and the bottom up approach can generate a SO swarm more quickly. The two approaches are not mutually exclusive and both are more effective when combined.

In this paper, self organization is only identified visually. Later work uses the self organization metric to quantitatively analyze the effectiveness of self organization.

IV. Self Organized Multi Agent Swarm Design

In the following design of Self Organized Multi Agent Swarms (SOMAS) [13], desired system behaviors, major assumptions and risks are discussed. The general approach to generating the SOMAS is described and a formal model is selected to represent the individual agents and their interaction with each other as well as the generic computer network environment. The general objectives and the model development complexity is discussed as associated with finding each individual agent policy and an overall policy or process to achieve the desired system behaviors.

Given the overall goal from the Introduction, *the generic SOMAS behavioral objectives addressed in this paper are to minimize the activity on a network node, that is, minimizing agent movement, creation, and deletion on a node. And, to identify the vital network nodes, that is those nodes that can cause the largest network degradation if removed.* Such objectives and constraints require mathematical optimization formulations in order to develop a computational model. Constraints considered are the use of a container model, the specified agent sensors, the given set of agent rules, and a static network topology.

The implementation model used for the agents simplifies much of the complexity of a real world implementation. The agents are assumed to be implemented within a software or hardware container. This assumption is valid if they are to be deployed in the Cybercraft environment, described in section II. Container models significantly simplifies decentralized swarm implementation, abstracting agents from hardware and operating system details. It is assumed containers use a

common interface to provide agents with information about the hosts they are on. Agents need only know the interface to access the information they need and interact with their host through their container. The use of the container model implies the agents do not need specialized mechanisms for interacting with their host. Additionally, communication is simplified to direct delivery between nodes, i.e. an agent is merely moved from one container's data structure and placed in another's data structure.

A major risk is the nature of solutions generated by approximate SOMAS generation. Approximation means the solution is imperfection and the SOMAS does not react appropriately to certain events and environments. The second risk is the degree of self organization the swarms exhibit. Lack of self organization results in trivial or chaotic swarms, both of which are ineffective. Additionally, the chaotic swarm is dangerous. Finally, the complexity and seriousness of the problem may mean that the SOMAS approach does not provide the necessary gain for the risk involved.

Two approaches are used for the generation and adaptation of SOMAS: off-line and on-line. The off-Line approach makes use of simulations and a DEC-POMDP model to generate agents. This model is chosen because the global network state can be observed for evaluating the global fitness of the swarm during off-line generation. On the other hand, the on-line approach uses data from the actual network and an I-POMDP model to generate agents. The I-POMDP model is chosen because it presupposes individual agents are only capable of local observations of other agents in the swarm. Both approaches make use of evolutionary operators. The off-line production uses an explicit evolutionary algorithm and the on-line production's evolutionary algorithm is implicit in the combination of the SOMAS and the environment.

Various Markov decision process (MDP) models are appropriate for modeling the off-line and on-line agent swarms, because the agent schema outlined by Russel and Norvig in AIMA [21] can be represented with an MDP. The AIMA agent schema consists of sensors, actuators, and state. These can be encompassed in the elements of MDP models, which are at least state and an action-state transition function, but can also include elements such as observation and reward.

Since most agent swarms do not have full observation of their environment, a partially observable Markov decision process (POMDP) is appropriate. The elements of the AIMA agent schema correspond to the POMDP $< S, A, T, O, R >$ tuple in this way: S is the agent's state, A are the agent's actuators, O are the agent's observations, and R are the rewards the agent can receive for its utility function. T is not found explicitly in the AIMA agent schema. It represents how agents' actions change their state.

Along with the basic schema in [21], AIMA also describes a taxonomy of agents, ranging from purely reactive agents to decision theoretic agents. The mapping parameters T and Ω in the interactive POMDP (I-POMDP) [10] tuple $< IS_i, A, T_i, \Omega_i, O_i, R_i >$ are the symbolic equivalent of cognitive capabilities in agents that make decisions based on their knowledge of other agents. R in the decentralized (DEC-POMDP) [2] is a group reward, allowing the utility for an entire swarm to be modeled. Table I compares the agent scale characteristics of common MDP models and their respective elements. Regional can comprehend both local and global. Both are extreme points of regionality.

Type	Scale	S	O	A	T	Ω	R
MDP	L	L	N/a	L	L	L	L
DEC-MDP	G	L	N/a	L	L	L	G
POMDP	L	L	L	L	L	L	L
DEC-POMDP	G	G	L	L	R	L	G
I-POMDP	R	R	R	L	R	R	R
R-MTDP	G	R	L	R	R	L	G

TABLE I
TAXONOMY OF COMMONLY USED MDP MODELS. L = LOCAL, R = REGIONAL, G = GLOBAL

The DEC-POMDP and I-POMDP models are chosen over the other MDP models because they encompass most of the others. The exception to this is that they do not group actions into roles like the R-MTDP model. As well as encompassing the other models, the DEC-POMDP is chosen because it allows group rewards, which are useful for the off-line development of the swarm from a global perspective. The I-POMDP models the agents interacting with each other and directly changing each others' parameters, and thus describes the agent observable on-line behavior of the swarm.

V. AGENT SWARM BEHAVIOR FORMALISM

In order to properly develop the SOMAS, a formal set of behavior definitions are required with associated agent rules using first order predicate logic well formed formulæ. Formal overall behavioral goals and specific individual agent goals need to be addressed for DEC-POMDP processing.

A. Agent Rules

Each agent has a set of decision rules that it uses to determine whether to execute a given actuator based on its observation of its state and environment.

A basic weighted discriminant function is used in this application, equations (1) and (2).

$$\text{convert}(\vec{data}) > 0 \rightarrow \textbf{actuator}.\text{execute} \quad (1)$$

$$\text{convert}_{weighted}(\vec{data}) = \text{mean}(\vec{weights}_{actuator}.\times\vec{data}) \quad (2)$$

$\vec{data}$ is statically sized in this rule, and $\vec{weights}_{actuator}$ is a particular set of vector weights for the given actuator in the range (-1, 1). $.\times$ is element-wise matrix multiplication.

$\vec{data}$ is a vector composed of state and environment observations, a random variable, and a set of fixed parameters.

Each agent has the same set of 4 sensors.

- Fitness value of agent
- Fitness values of all agents on node
- Fitness values of all agents' chromosomes on the node
- Fitness values of neighboring nodes one edge away

Each agent has the same set of 6 actuators/rules. Many actuators have dynamic parameter values provided by each agent's chromosome through a genetic algorithm process.

- Change location
- Change fitness value of agent
- Mutate one of the agent's chromosomes
- Crossover one of the agent's chromosomes with a local agent
- Create local agent (also marks the node)
- Delete local agent

The fitness value of a node is the sum of all the fitness values of the agents on the node; not global, not local, but regional.

B. Formal SOMAS Objectives

Since there is often a tradeoff between objectives, it is usually impossible to find a single solution that minimizes each objective. In operations research, there are two main techniques for dealing with multi-objective optimization. The objectives are either reduced to a single objective, or solutions are compared based on their Pareto dominance. [6] Since the latter technique is more general, it is used to rank the solutions into Pareto equivalent sets, as detailed in equation (3).

$$PF_{TRUE} = \forall f(\text{argmin}^{P}_{s \in \mathcal{S}}(f(s)) : f \in \mathcal{F}) \tag{3}$$

The best set is called the PF_{TRUE}, since it is the Pareto dominant set out of all the solutions, also known as the Pareto front. In the case of a search for an approximate solution the solution set is PF_{KNOWN} since it is the known best set, but not necessarily the optimal set. argmin^{P} is an argmin function that uses a Pareto dominance ranking metric to find the Pareto dominant set of solutions.

Non Intrusive:

This objective is to minimize agent activity on a node, which entails minimizing agent movement, creation, and deletion. Two objective functions measure this activity. Both functions are of the form of equation (4). $\mathcal{A}_t$ in the first function is the set of agents created during each simulation step, per formula (5). $\mathcal{A}_t$ in the second function is the set of agent movements between nodes during each simulation step, per formula (6).

$$\sum_{t \in \mathcal{T}_{t>1}} |\mathcal{A}_t| \tag{4}$$

$$\forall a\{a \in \mathcal{A}_t : a \in s_t \wedge a \notin s_{t-1}\} \tag{5}$$

$$\forall a\{a \in \mathcal{A}_t : a \in n_t \wedge a \notin n_{t-1} \wedge a \in s_{t-1}\} \tag{6}$$

$\mathcal{T}_{t>1}$ is the set of all simulation time steps, not including the initial step. n_t is a given node during time step t. s_t is the simulation state during time step t.

Vital Node Identification:

Identifying the vital nodes in a network is the scenario specific goal of SOMAS. As mentioned in the introduction, this scenario is meant to show the feasibility of using a SOMAS for network security. It should not be considered the only SOMAS security behavior, since multi agent systems are used to solve many security problems. A broad range of behaviors are covered in future work.

The vital node set is the minimal set of nodes in a network that causes the greatest network degradation when removed. Finding the vital nodes and edges in a network is an NP-Hard problem [1]. The difficulty of the problem is increased by the fact that the problem information is only partially observed, each agent only has a partial view of the whole network.

It is important to design networks so they cannot be easily damaged by resources being deactivated. Identifying the vital nodes is relevant for any security professional concerned with robust network topologies and resource allocation.

To simplify the objective evaluation, the network is constructed such that a specified node has a significantly greater degree than all other nodes, making it the vital node. Measuring swarm effectiveness consists of counting the number of agents that correctly identify the vital node, as detailed by equation (7).

$$\frac{|\mathcal{M}_i| + 1}{|\mathcal{M}_c| + |\mathcal{M}_i| + 1} \quad \forall \mathcal{M}_c \forall \mathcal{M}_i \{\mathcal{M}_c, \mathcal{M}_i \in \mathcal{S}\} \tag{7}$$

$\mathcal{S}$ is the set of all simulation states. $\mathcal{M}_c$ is the set of marks on the correct node and $\mathcal{M}_i$ is the set of incorrect marks. The top and bottom are offset by 1 so the result is not ambiguous ($\frac{0}{0}$) if both sets are empty.

VI. Algorithm domain

The search algorithm, various operator, and associated parameter values must be defined for multi objective computational execution [6]. Also, a testing environment must be selected. These issues are addressed in this section.

Since finding a DEC-POMDP policy is NP-Complete [2] and finding an I-POMDP policy is harder or impossible due to its recursive nature [11], it is not tractable/possible to find an exact solution in most I-POMDP and DEC-POMDP problem domains. The SOMAS problem domain is not simpler. It is a planning problem of determining the right agent actions to perform at the right place at the right time. Solving the planning problem is NEXP-Complete [21]. In such difficult problems, it is better to look for a good local optima than the optimal solution.

The field of metaheuristics provides numerous general approximation search techniques, such as simulated annealing, tabu search, and evolutionary algorithms [17]. Genetic algorithms (GA) are most suited for the SOMAS problem domain because they search the global solution space and an effective swarm needs to incorporate multiple kinds of behaviors. The algorithm used for this paper employs the standard GA operators of uniform crossover and mutation, and selects the operands with the IBEA multi-objective selection algorithm [28]. Figure 1 shows how the genotype is translated to the phenotype in the GA.

A. Search Heuristics

According to the No Free Lunch Theorem (NFLT) [27], one search technique is just as preferable to another without specialized domain knowledge. However, the NFLT only applies across all problem spaces or to particular domains exhibiting certain characteristics. The vast majority of countable problem

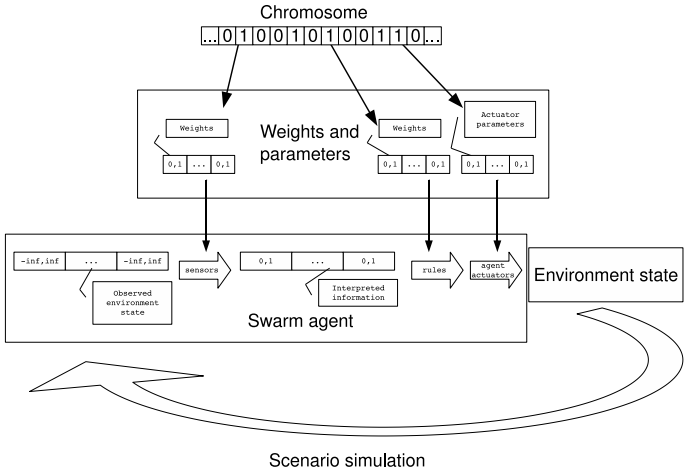

Fig. 1. Translating the genotype to the phenotype

domains (classes of functions) do not have the conditions necessary for the NFLT to hold [14], but many other problem domains have these conditions, such as uncountable problem domains. Engineers need to know the problem domain characteristics, how to encode its specialized knowledge, and how to define metrics and measures to evaluated system performance.

One problem domain metric is intentional development. Intentional development creates order whereas lack of intention results in disorder. This is seen when comparing human artifacts to natural artifacts [7]. A simple demonstration is to view the amount of order in New York City compared with the amount of order in the Amazon rain forest. Order and disorder can be characterized by the entropy of the system, and consequently the amount of information in the system. A system with high information exhibits a very concise coding whereas a system with low information exhibits a very large coding, as implied by Shannon's theory of information. The amount of order or disorder in the system, in turn, helps guide the amount of local and global information to use in the search. Since any existing system is produced either through a mechanical and stochastic process, or through intention (human or animal), a very general, yet potentially very useful, heuristic is hypothetically possible to guide the information or specialized knowledge design of the SOMAS for a particular problem domain. If this hypothesis is true, then the NFLT does not hold for algorithms operating in many existing systems.

B. Operators and parameter selection for MOEA

Multiple objectives in MOPs do not generally map in a regular pattern to the decision space. Additionally, the curse of dimensionality implies the objective vector mean lies in a radius around the vector composed of the mean of each objective, and the radius increases with the number of objectives. Therefore, it is useful for a MOEA to be more exploratory as the number of objectives increase.

The implemented mutation operator looks at each bit in the chromosome and flips it according to a probability. The probability of mutation is set to 1.0 and the probability that a particular allele is mutated is set to 0.1. Consequently, the likelihood that at least one allele is mutated is $1-0.9^{\alpha}$, where α is the number of alleles in the chromosome. Since mutation in a GA is traditionally set to a very low value, such as around 0.01, this parameter setting is comparatively very high for a chromosome of significant length.

On the other hand, the crossover probability is very low in regard to normal GA settings, due to the characteristic of MOPs where the good building blocks tend to have high epistasis and occur rarely [6]. The crossover operator is a uniform crossover. The probability the crossover operator is used is set to 1.0 and the probability of crossover is 0.1 for each allele. The likelihood at least one allele is crossed over is the same as the likelihood of at least one allele mutation. Traditional GAs tend to use multi point crossover, instead of uniform crossover. So, while the likelihood of using the crossover operator is very high, the likelihood the same number of alleles is crossed over as in a traditional GA is quite low. Thus, the likelihood a building block of high epistasis is disrupted is reduced.

VII. Design of Experiments

In the design of experiments, the test objectives must be specified, followed by a methodology and plan for accomplishing the objectives.

A. Test objectives

Best algorithm: There are a wide variety of MOEA algorithms in existence, and they each have their own characteristics which make them suited for particular problem domains. As a result, it is important to determine what the best algorithms are for finding solutions on or near the different regions of the PF_{TRUE} for the SOMAS problem domain. This is possible despite the No Free Lunch theorem (NFLT) since the NFLT only applies to all problem domains as a whole and only a small portion of individual problem domains, refer to section V.

Effective Self Organized Behavior: The general goal behind the SOMAS approach is to generate swarms that can accomplish global objectives with only local information and interaction, section III. The behavior in this case is to identify the vital nodes in the network, section V. Additionally, the behavior should be self organized.

B. Test Methodology

Algorithm Evaluation: In evaluating MOEA effectiveness, various metrics must be appropriately selected with the resulting data statistically analyzed. The PISA testing framework [3] is used because of its extended utility for various MOEAs.

The two algorithms which are tested against each other are NSGA2 and SPEA2. They are run on the 3 different problem sizes: pedagogical, local, and campus. 30 simulation runs are conducted for each chromosome.

The MOEAs are run 4 times each for the two smaller size networks, and 2 times for the largest network, resulting in 18 runs altogether. Table III shows the settings for μ and λ.

According to the Central Limit Theorem and empirical evidence, 25 experiments suffice to produce significant results

[18]. Thus, each chromosome is evaluated on 30 randomly initialized networks to generate its objective values. The extra evaluations are used to improve the significance.

In order to determine the comparative effectiveness of MOEAs, their respective PF_{KNOWN} surfaces must be compared. The metric for this comparison should be Pareto compatible, although there may exist a good reason for Pareto non-compatibility [6]. If the metric is not Pareto compatible, then it may rank a dominated front higher than the dominating front. Suggested MOEA metrics include error ration, hyperarea or hypervolume ratio, epsilon indicator along with Pareto attainment functions [6]. The Pareto compatible metric selected is hypervolume ratio, since it is easily computable.

To test comparisons between algorithms, it is important to first determine whether the results follow the normal distribution. If they do, then a p-test can be used. If the results do not, then a non parametric test must be used. In this paper, since the distribution is not known, non parametric tests are used. The tests are the Fisher independent test and the Mann-Whit test, which are both non-parametric. The statistics are calculated by comparing the hypervolumes generated by the PF_{KNOWN} of each algorithm.

Behavior Evaluation: In order for the behavior to be effective, it must perform statistically better than random behavior. However, even if the behavior is effective, that does not necessarily mean it is self organized. This paper does not use the self organization metric discussed in section III. The metric is used in later work. Instead, a visual analysis of self organization is used. If the swarm exhibits a organized, global behavior that requires the local interactions of the swarm's agents, then the behavior is self organized according to the definition in section III.

To determine the statistical significance of the behavior's effectiveness, the upper bound on the random behavior is 30 trials to mark the correct node, where each trial only consists of one mark attempt. Since each chromosome is evaluated on 30 randomly initialized simulations, the number of trials is set to 30. The number of mark attempts per trial is set to 1 because if the number of mark attempts per trial is increased, the likelihood of getting the same percentage of correct marks averaged over all the trials decreases. Thus, 1 mark attempt per trial is the lowest possible number of attempts that provides the greatest likelihood of success. The probability of the swarm selecting a particular node is $\frac{1}{|\mathcal{N}|}$ where $\mathcal{N}$ is the set of nodes. The percentage of correctly identified nodes averaged across the simulation runs is compared to likelihood of the same percentage being produced using a binomial distribution with the random behavior's parameter settings. If the likelihood is below 0.01, then the swarm's effectiveness is considered statistically significant.

VIII. Results and Analysis

To evaluate SOMAS performance, analysis of approximate or known Pareto fronts is addressed.

A. Algorithm Effectiveness

Even though the p-values in the statistics do not fall below the significance a-value af 0.05, it is important to see a difference does exist. The lack of statistically significant results is possibly due to the fact only 4 samples are compared against each other, instead of the recommended 20. For the small network, NSGA2 creates the dominant PF_{KNOWN}, while SPEA2 outperforms NSGA2 for the larger networks. This is in-line with the algorithms' known behavior. SPEA2 is observed to converge more effectively than NSGA2.

The knowledge of the algorithms' comparative effectiveness depending on problem parameters fulfills the first test objective, section VII-A, to an extent.

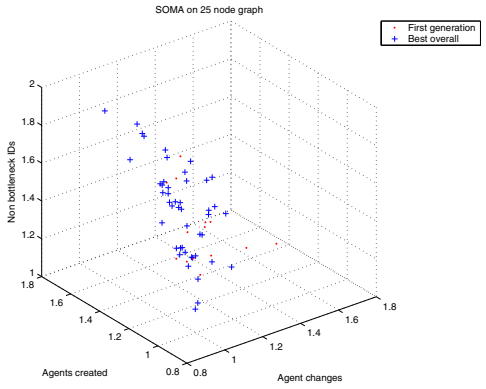

Fig. 2. PF_{KNOWN} of first generation and overall for SOMA on 25 node graph

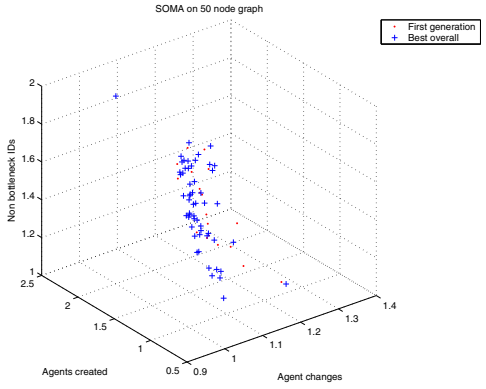

Fig. 3. PF_{KNOWN} of first generation and overall for SOMA on 50 node graph

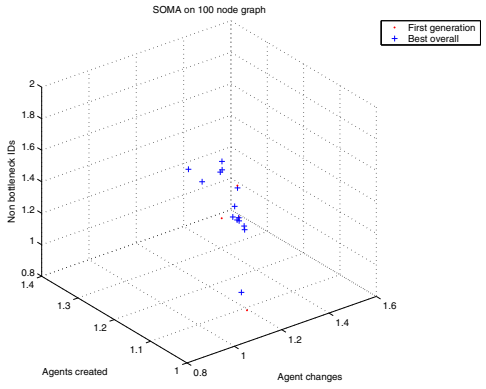

Fig. 4. PF_{KNOWN} of first generation and overall for SOMA on 100 node graph

Graph Size	Test Type	Result
25 node	Fisher	SPEA2 $>$ NSGA2 w/ p-value of 0.500
	Independent	NSGA2 $>$ SPEA2 w/ p-value $<$ 0.500
25 node	Mann-Whit	SPEA2 $>$ NSGA2 w/ p-value of 0.841
		NSGA2 $>$ SPEA2 w/ p-value of 0.159
50 node	Fisher	SPEA2 $>$ NSGA2 w/ p-value $<$ 0.500
	Independent	NSGA2 $>$ SPEA2 w/ p-value of 0.501
50 node	Mann-Whit	SPEA2 $>$ NSGA2 w/ p-value of 0.159
		NSGA2 $>$ SPEA2 w/ p-value of 0.841
100 node	Fisher	SPEA2 $>$ NSGA2 w/ p-value $<$ 0.500
	Independent	NSGA2 $>$ SPEA2 w/ p-value of 0.501
100 node	Mann-Whit	SPEA2 $>$ NSGA2 w/ p-value of 0.159
		NSGA2 $>$ SPEA2 w/ p-value of 0.841

TABLE II
HYPOTHESIS TESTS OF ALGORITHM DOMINANCE FOR GRAPHS.

Type	Nodes	Pop	Gens	Correct	Prob
NSGA2	25	100	8	42.0%	4.01e-11
	25	10	8	33.8%	1.39e-07
	50	100	5	25.5%	9.60e-08
	50	10	8	21.3%	2.34e-05
	100	100	8	19.3%	4.66e-07
SPEA2	25	100	8	40.5%	6.95e-10
	25	10	8	33.3%	1.39e-07
	50	100	5	29.0%	4.79e-09
	50	10	8	25.1%	9.60e-08
	100	100	8	23.7%	1.61e-08

TABLE III
BEST RESULTS FOUND WITH GIVEN PARAMETERS

B. Behavior Effectiveness

As shown in table III, all variations of the experiment produce a behavior that is statistically more effective than the random behavior. Therefore, SOMAS production produces significant solutions in the problem domain, fulfilling the effectiveness criterion of the second test objective, section VII-A.

C. Behavior Self Organization

1) Human Engineered Chromosome: Comparing a human engineered chromosome to sampled evolved chromosomes demonstrates a number of interesting differences. The human engineered solution is very active and many agents move, are created, and deleted. The general trend, when the solution works, is for the agents to stabilize so there is a group of very mobile agents concentrated at the bottleneck, while the rest of the nodes primarily contain stationary agents.

2) Observation Self Organized Behavior in Evolved Chromosomes: On the other hand, the two sampled evolved solutions are created from agents that tend to have extremely high parameters. The first solution is made from a single very mobile agent that has a medium likelihood of creating an agent, and a stationary agent that is very likely to delete other agents. The second solution is similar to the first, except the mobile agent is also very likely to create new agents.

Both evolved solutions have a comparatively very small amount of network activity, and in some cases the agents are completely removed from the network. Generally, there is a brief period of high activity, where agents rapidly visit the whole network, followed by a very rapid decline in activity until there are almost no agents left. The second chromosome has an interesting additional feature where the network in some cases stabilize with a single set of agents constantly creating and deleting each other on a single node, which in turn is often the vital node. This is a very ideal result, although it does not happen consistent. However, it is interesting to see that such a solution can be evolved with a very small number of rules, simple parameters, and simple observations.

Since this behavior is achieved by the agents without the agents knowing the global utility of their actions or the global swarm organization, and the behavior is fairly stable, the behavior is self organized. This result fulfills the self organization criterion of the second test objective, section VII-A. The behavior entirely accomplished by the dual level feedback mechanism of agent creation and deletion, and evolutionary operators; as well as the local utility function of going to the node with the highest pheromone concentration.

D. Pareto Front Analysis

As can be seen in the plots of the approximate Pareto front, the selection algorithms are not able to find a good spread of solutions on the two larger networks. The solutions tend to be grouped in the center. This lack of exploration is directly related to the problem domain and represents the difficulty in minimizing each objective independently from the others. The number of agents created and the number of agent changes on a node are very closely tied, but not identical. The number of agent changes is limited by the number of agents created, but the converse does not apply. That is why the agent creation axis is better explored than the agent change axis. On the other hand, the incorrect ID axis is largely independent of the other two since it is scale independent, although a small amount of activity is necessary to produce a value.

It is noteworthy that the final PF_{KNOWN} does not extend significantly far beyond the original PF_{KNOWN}, although it comprehends more of the Pareto front. This observation suggests the solutions are very close, if not on, PF_{TRUE}. Assuming the PF_{KNOWN} is close to optimal, then some optimal solutions are quite easy to find, since the PF_{KNOWN} for the algorithms' first generations is produced from fairly small populations.

IX. CONCLUSION

In this paper, a self organized multi agent swarm approach to network security is proposed. Its derivation comes from considerations of self organization and mathematical Markov decision models. A metric for self organized behavior is recommended, as well as a heuristic. The desired behaviors of the swarm are formalized and SOMAS is statistically tested on a preliminary benchmark. The tests demonstrate the effectiveness of particular algorithms based on problem parameters. The SOMAS approach [13], at least in this fairly simple problem domain, has shown itself to be effective at accomplishing a given objective, as shown by table III. Additionally, evolved behaviors have demonstrated self organized behavior, when individual chromosomes are visualized, although their reliability needs to be improved.

Yet, despite the limited success, many SOMAS developmental aspects can be improved in the approach, methodology, and experimental design and evaluation. More extensive testing is required of different agent rules, scenarios, and scenario sizes. Additionally, the self organization metric should be used to determine the influence of self organization on the swarm's effectiveness and efficiency. The intentional development heuristic also needs to be formalized and tested.

The feasibility of the SOMAS approach opens many promising areas of research. As mentioned in the introduction, our electronic world is faced with ever mounting complexity of events and structures, complexity for which our traditional hierarchical approaches are not well suited. However, natural organisms handle much greater complexity on an everyday basis, without a centralized hierarchy. Armed with a means of rigorously quantifying self organization, the SOMAS architecture shows promise as being able to solve these problems, with intriguing implications.

ACKNOWLEDGEMENTS

This investigation is a research effort of the AFIT Cyberspace Technical Center of Excellence, Director: Dr. Rick Raines, www.afit.edu/ccr, as supported by the Air Force Research Laboratory, National Security Agency, and the Department of Homeland Security.

Eric Holloway would like to thank Adam Holloway, Abraham Lewis, David Kristof, and Elliot Temple for the helpful discussions and critiques.

REFERENCES

[1] A. Bar-Noy, S. Khuller, and B. Schieber. The complexity of finding the most vital arcs and nodes. Technical report, IBM Research Division and University of Maryland, 1995.

[2] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein. The complexity of decentralized control of markov decision processes.

[3] S. Bleuler, M. Laumanns, L. Thiele, and E. Zitzler. PISA — a platform and programming language independent interface for search algorithms. In C. M. Fonseca, P. J. Fleming, E. Zitzler, K. Deb, and L. Thiele, editors, *Evolutionary Multi-Criterion Optimization (EMO 2003)*, Lecture Notes in Computer Science, pages 494 – 508, Berlin, 2003. Springer.

[4] F. Boschetti and R. Gray. Emergence and computability. In *Emergence: Complexity and Organization*, volume 9. The Complexity Society, the Institute for the Study of Coherence and Emergence, and Cognitive Edge, 2007.

[5] S. Camazine, J. Deneubourg, N. Franks, J. Sneyd, G. Theraulaz, and E. Bonabeau. *Self-Organization in Biological Systems*. Princeton University Press, 2003.

[6] C. A. C. Coello, G. B. Lamont, and D. A. V. Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems*, chapter MOEA Parallelization. Springer, 2007.

[7] L. N. de Castro. *Fundamentals of Natural Computing (Chapman & Hall/Crc Computer and Information Sciences)*. Chapman & Hall/CRC, 2006.

[8] R. Dewri, nayot Poolsappasit, I. Ray, and D. Whitley. Optimal security hardening using multi-objective optimization on attack tree models of networks. *CCS*, 2007.

[9] G. Doziert, D. Brownf, J. Hurley, and K. Cainf. Vulnerability analysis of ais-based intrusion detection systems via genetic and particle swarm red teams. Technical report, Auburn University and Clark-Atlanta University and The Boeing Company, 2004.

[10] P. J. Gmytrasiewicz and P. Doshi. Interactive pomdps: Properties and preliminary results. *Association for the Advancement of Artificial Intelligence*, July 2004.

[11] P. J. Gmytrasiewicz and P. Doshi. Exact solutions of interactive pomdps using behavioral equivalence. *Association for the Advancement of Artificial Intelligence*, May 2006.

[12] C. G. Harrison, D. M. Chess, and A. Kershenbaum. Mobile agents: Are they a good idea? Technical report, IBM, T. J. Watson Research Center, Yorktown Heights, New York, Mar. 1995.

[13] E. Holloway. Self organized multi agent swarms (somas) for accomplishing network goals. Master's thesis, Air Force Institute of Technology, March 2009.

[14] C. Igel and M. Toussaint. On classes of functions for which no free lunch results hold. *Elsevier Science*, 2003.

[15] D. R. Karrels and G. Peterson. Cybercraft: Protecting air force electronic systems with lightweight agents. November 20007.

[16] J. Kim and P. J. Bentley. Towards an artificial immune system for network intrusion detection: An investigation of clonal selection with a negative selection operator. Technical report, University College London, 2001.

[17] Z. Michalewicz and D. B. Fogel, editors. *How to Solve it: Modern Heuristics*. Springer, 1998.

[18] J. S. Milton and J. C. Arnold. *Introduction to Probability and Statistics*, chapter Chapter 7 Estimation. McGraw Hillo, 2003.

[19] S. Mukkamala, G. Janoski, and A. Sung. Intrusion detection using neural networks and support vector machines. *IEEE*, 2002.

[20] R. Nagpal. Programmable self-assembly using biologically-inspired multiagent control. In *AAMAS*, pages 418–425. ACM, 2002.

[21] P. Norwig and S. Russell. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2003.

[22] M. Prokopenko, F. Boschetti, and A. Ryan. An information-theoretic primer on complexity, self-organisation and emergence. *Advances in Complex System*, 2006.

[23] T. Schlegel, P. Braun, and R. Kowalczyk. Towards autonomous mobile agents with emergent migration behaviour. In *AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 585–592, New York, NY, USA, 2006. ACM.

[24] D. Servat and A. Drogoul. Combining amorphous computing and reactive agent-based systems: a paradigm for pervasive intelligence? In *AAMAS*, pages 441–448. ACM, 2002.

[25] C. R. Shalizi, R. Haslinger, J.-B. Rouquier, K. L. Klinkner, and C. Moore. Automatic filters for the detection of coherent structure in spatiotemporal systems, July 29 2005.

[26] M. R. Stytz, D. E. Lichtblau, and S. B. Banks. Toward using intelligent agents to detect, assess, and counter cyberattacks in a network-centric environment. Technical report, Institute for Defense Analysis, 2005.

[27] D. H. Wolpert and W. G. Macready. No free lunch theorems for search. Technical report, Santa Fe Institute, February 1995.

[28] E. Zitzler and S. Künzli. Indicator-based selection in multiobjective search. In X. Yao et al., editors, *Parallel Problem Solving from Nature (PPSN VIII)*, pages 832–842, Berlin, Germany, 2004. Springer-Verlag.

Eric M. Holloway received a B.Sc. degree in computer science from Biola University, California, 2005, and graduated Summa Cum Laude. He received a Masters of Computer Science degree in March, 2009, at the Air Force Institute of Technology, WPAFB, Dayton, OH, 45433. He is also a communications officer in the United States Air Force.

Gary B. Lamont is a Professor in the Department of Electrical and Computer Engineering, Graduate School of Engineering and Management, Air Force Institute of Technology, WPAFB, Dayton, OH, 45433, USA; B. of Physics, 1961; MSEE, 1967, PhD, 1970; University of Minnesota. He teaches courses in computer science and computer engineering. His research interests include: evolutionary computation, artificial immune systems, information security, parallel and distributed computation, combinatorial optimization problems (single objective and multi-objective), software engineering, digital signal processing, and intelligent and distributed control. He has advised many MS and PhD students in these disciplines. Dr. Lamont has authored several textbooks (Multi-Objective EAs, Computer Control), various book chapters as well as numerous papers in the above areas.

Gilbert L. Peterson is an Associate Professor of Computer Engineering, Department of Electrical and Computer Engineering, Air Force Institute of Technology, WPAFB, Dayton, OH, 45433; BS Architecture, University of Texas at Arlington, 1995; MS, Computer Science, University of Texas at Arlington, 1998; PhD, University of Texas at Arlington, 2001. His research interests include uncertainty in artificial intelligence, robotics, machine learning, datamining, and digital forensics.