Mixed-Initiative Cyber Security: Putting humans in the *right* loop

Jereme N. Haack[1], Glenn A. Fink[1], Wendy M. Maiden[1], David McKinnon[1], and Errin W. Fulp[2],

[1] Pacific Northwest National Laboratory, Richland WA, USA
[2] Wake Forest University, Computer Science Department, Winston-Salem, NC, USA
{jereme.haack, glenn.fink, wendy.maiden, david.mckinnon}@pnl.gov, fulp@wfu.edu

Abstract. Organizations and their computer infrastructures have grown intertwined in complex relationships through mergers, acquisitions, reorganizations, and cooperative service delivery. Consequently, defensive actions and policy changes by one organization may have far-reaching negative consequences on the partner organizations. Human-centric and machine-centric approaches are insufficient for defending the security of today's increasingly complex computer infrastructures. The former are slow but highly adaptive, while the latter are fast but highly specialized. We believe the solution lies in mixed-initiative defenses combining the complementary qualities of both human- and machine-based approaches. We describe the Cooperative Infrastructure Defense (CID), a new cyber-defense paradigm designed to unify complex-adaptive swarm intelligence, logical rational agents, and human insight. CID will enable cooperative defense of infrastructure through situational awareness using visualization, security policy dialogue between humans and agents, shared initiative in solving cyber problems, and a foundation for building trust between humans and agents within and between organizations.

Keywords: agents; security; mixed-initiative

1 Background

Computer infrastructures consist of multiple interdependent organizations that share computational resources in a formal or *ad hoc* arrangement. In critical infrastructures such as electrical power grids, organizations may be so interdependent that failure of some part of a single company could produce cascading failures with consequences on the local, national or international scale. Regardless of the degree of their interdependence, infrastructure members remain economically independent organizations. They have separately managed systems, few shared policies, differing business drivers, and proprietary information that cannot be shared without special legal instruments.

To manage the complexity, we factor these multi-organization cyber infrastructures into *enclaves*, a set of computer and network hardware and software that is owned by a single organization and administered under a unified policy by a single (possibly separate) organization. An enclave may be very small, consisting of one or two machines, or it may be very large, comprising numerous networks. Enclaves are the building blocks of infrastructures and are the largest unit over which a mixed-initiative system [1] can be fielded without crossing proprietary boundaries between organizations.

Accomplishing concerted cyber defensive action that spans organizational boundaries is difficult. Infrastructures have unique cyber security needs that cannot be adequately addressed by individual enterprise solutions, and the complicated relationships in infrastructures make it possible for defensive actions by one organization to affect the others adversely [2]. Both legal [3, 4] and practical issues require that the consequences of change be managed through coordination across organizations in a near-real-time manner.

Cyber adversaries, on the other hand, are not hindered by central coordination; they can rapidly and concertedly disrupt multi-organizational computer infrastructures. Therefore, enclave defenders need intelligent defenses with the autonomy to adapt in real-time to both internal and external threats. The real-time nature of threats on the Internet requires that humans not become a bottleneck, and the seriousness of proprietary boundaries requires that automated defenses strictly observe the policies of the cooperating

organizations. A mixed-initiative approach solves these problems by allowing not only shared control between humans and software agents but also shared initiative among human stakeholders across the infrastructure.

Current cyber defense systems involve humans at multiple levels, but people are often far down in the control structure, requiring them to make too many time-critical decisions. Information flow between humans is slow and frequently asynchronous. In a crisis, humans may be unable to cooperate because of culture, language, legal, proprietary, availability, or other obstacles. Such systems cannot adapt to the Internet speeds of cyber threats. Consequently, effective cyber defense requires a framework that simultaneously capitalizes on the adaptability of humans and the speed of machines. In other words, humans must be put in the *right* loop to maximize their effectiveness while preserving their legal responsibility for the actions of their autonomic systems [4].

2 Introduction

This paper presents a new mixed-initiative hierarchical framework of humans and agents that is well suited to protecting computational infrastructures. The framework, Cooperative Infrastructure Defense (CID), is designed to rapidly adapt to new cyber attacks via swarming software agents while enabling humans to supervise the system at an appropriate level. We interpose a hierarchy of rational software agents between the swarm and the human supervisors to provide a channel for system guidance and feedback. Using various kinds of rationality actually turns false positives into beneficial forms of positive feedback and improves system performance.

CID represents a revolutionary new way of looking at system control for cyber security. Traditionally, humans control the entire system assisted by automated tools and subsystems. In CID, agents share the decision-making power, handling most of the real-time portion autonomously but enabling human involvement at all levels. The human supervisor does not directly control the system rather humans exert supervisory influence sharing the initiative for action with their software agents. CID is designed to be a scalable, dynamic, and robust framework for securing increasingly complex computational infrastructures. CID makes humans an intrinsic part of the solution, engaging them without requiring them to directly control and enables diverse organizations within an infrastructure to cooperate in an adaptive cyber defense.

In our research, we have created simulations of the entire framework, prototypes of the user interface, and prototypes of the mobile sensor agents. Our initial prototype, built in an agent simulation framework, was demonstrated at the VizSec 2008 conference. We are currently implementing the framework in Java using the JADE agent framework (http://jade.tilab.com/) on a network of virtual machines.

The remainder of this paper is structured as follows. CID is described at a high level in Section 3. Section 4 describes examples of human-agent and agent-agent interaction in the CID framework. Section 5 discusses related work. The conclusion outlines possible directions for future research, summarizes the CID framework, and reviews the expected benefits.

3 System Overview

In CID humans and various types of software agents share the responsibilities of securing an infrastructure comprised of enclaves that belong to member organizations. Figure 1 shows how one human can supervise a multi-enclave system with a few enclave-level agents, a host-level agent at each machine or group of similar machines, and a large swarm of simple mobile agents. Our terminology is as follows:

- Humans function as *Supervisors*. They provide guidance to and receive feedback from one or more enclaves. They must to take action only when the lower-level agents encounter a problem that requires human involvement. Supervisors may take initiative as desired inspecting and guiding any element of the system. However, direct human *control* of the system is discouraged because such involvement would destroy its natural adaptive abilities.

- Enclave-level agents called *Sergeants*, which are each responsible for the security state of an entire enclave. Sergeants may make service agreements with Sergeants of other enclaves. Sergeants dialogue with humans to gain guidance for running the system according to business drivers and human security policies. Sergeants create and enforce executable policies for the entire enclave.

- Host-level agents called *Sentinels*, which are responsible for protecting and configuring a single host or a collection of similarly configured hosts such as a cluster or storage network. Sentinels interact with human supervisors only when they need clarification about how to classify ambiguous evidence from the swarm.

- Swarming agents, called *Sensors*, roam from machine to machine within their enclaves searching for problems and reporting to the appropriate Sentinel. Sensors are diverse; their classifiers are each uniquely derived from the set of known problem indicators. Sensors use stigmergic messages called *digital pheromone* [5] to communicate.

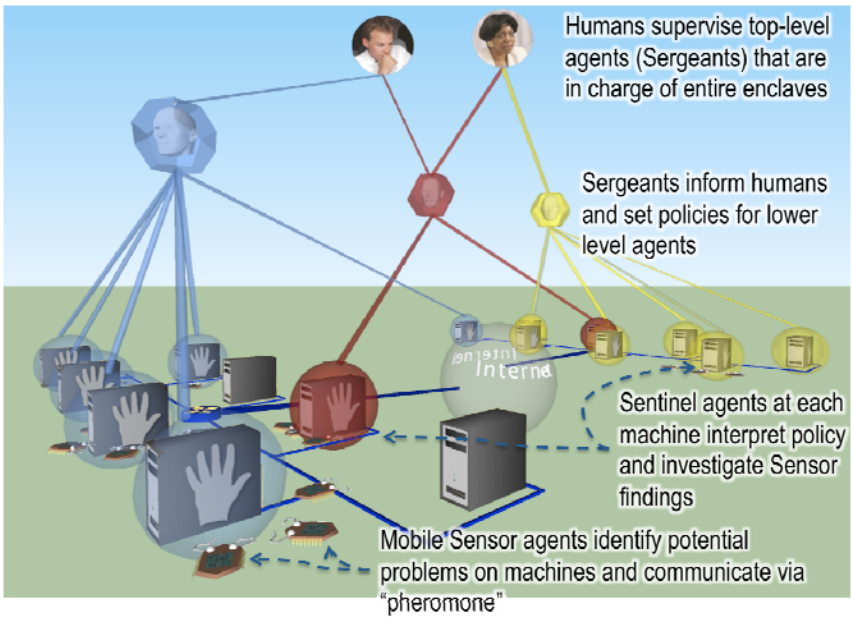

Figure 1: CID is a hierarchical framework of human supervisors, enclave-level rational agents, and swarming agents. A single human may supervise multiple enclaves via the agent hierarchy.

The concept of supervisors and agents of the CID framework operating within a hierarchical structure is supported by the research of Parunak [6], Smieja [7], and Selfridge [8], who each suggested hierarchical arrangements of heterogeneous agents. Interposing logic-based rational agents between the humans and the swarm provides a basis for communication, interaction, and shared initiative. The hierarchical arrangement gives humans a single point of influence that allows multiple points of effect. The following sections describe the roles of each actor in the CID and the relationships between actors are depicted in Figure 2.

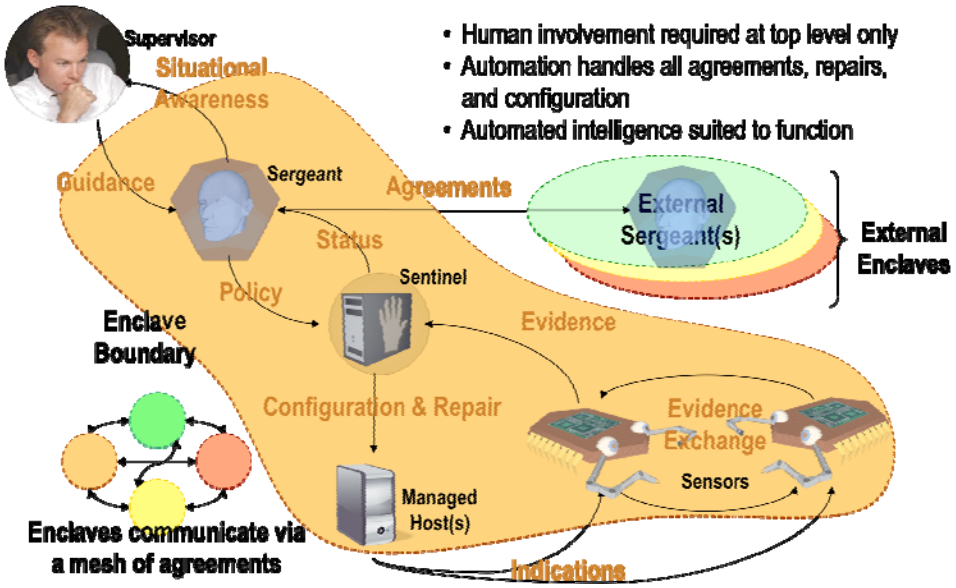

Figure 2: Cooperation among humans and agents in CID.

3.1 Supervisors

At the top layer of Figure 1 and Figure 2 are human supervisors who may direct one or more enclaves. Supervisors may belong to one or more interdependent organizations within the infrastructure, while the cyber assets in every enclave all belong to a single organization by definition. A Supervisor might also be a member of a regulatory organization or law enforcement agency and only monitor the equipment in the enclaves. Human supervisors translate business policy into guidance via natural-language and graphical controls for top-level agents called *Sergeants*.

3.2 Sergeants

Each enclave has a top-level agent called a Sergeant. In autonomic computing [9] terms, the Sergeant corresponds to the orchestrating autonomic manager. Sergeants provide situational awareness to their Supervisor, via an information visualization interface. Sergeants translate the guidance from the human Supervisor into actionable policy across all the machines within an enclave. We believe this will involve a natural-language dialogue that accepts human guidance and feeds back what the translation will be in terms of policy similar to IBM's SPARCLE (Server Privacy ARrchitecture and CapabiLity Enablement) policy workbench [10]. Sergeants will employ supervised learning algorithms so that interactions with them become more efficient over time. Sergeants are "heavy-weight" rational agents that make decisions based on logic. One possible implementation we have considered for Sergeants is Belief-Desire-Intention (BDI) logic [11]. The Sergeant presents the activities of lower-level agents to the human Supervisor and functions as an interface to influence system operation. Supervisors use Sergeants to enact environmental settings and policies that govern the general operation of the lower-level agents without controlling the lower-level agents directly. Sergeants provide a "geography" for the enclave that enables the swarm to operate properly.

Another major function of Sergeants is to broker agreements between CID enclaves on behalf of the Supervisors. To ensure that their actions are properly attributable, Sergeants must have a separate digital identification from the Supervisor they report to. Since they negotiate on behalf of humans, they may incur liability for their owning organization. Thus, there must be a mechanism to describe the types and degrees of authority the Supervisor has delegated to the Sergeant. Often, this authorization can be quantified in terms of maximum dollars that can be spent or types of service contracts that can be negotiated. An example interaction using this mechanism is described in Section 4.65.

3.3 Sentinel

Sentinels are mid-level rational software agents that, together with its managed host(s), correspond to the notion of autonomic elements. In autonomics parlance [9], the Sentinel is the Autonomic Manager and the host is the Managed Element. Each Sentinel is responsible for a single machine or group of machines that are similarly configured. For example, a Sentinel might be responsible for a single server, a router, a storage area network, a group of load-balanced web servers, or even a set of managed user workstations. Sentinels implement the policy they receive from the Sergeant and apply it to the configurations of the machine(s) they manage.

Sentinels also interface with the lowest-level agents: the swarming Sensor agents that gather information on potential problems found on the hosts. Sentinels provide the local geography to Sensors and provide mobility by negotiating with their destination. Sentinels also provide rewards and spawning capabilities to visiting Sensors. The Sentinels combine evidence from the Sensors with their own experience, shared knowledge from other Sentinels, guidance from Sergeants and Supervisors (interpreted by the Sergeant), and contextual host information to determine whether a problem exists and to devise potential solutions. Mechanisms similar to this are used in survivability architecture Willow [12].

Sentinels give feedback on the utility of Sensor findings in the form of “rewards” to the Sensors that visit their nodes. The analogy of foraging ants is used to describe the effect this feedback has on the system. By rewarding visiting Sensors, the Sentinel will attract more of them. A variety of visiting Sensors will provide more information on the potential problems experienced by the Sentinel and enable it to make more informed decisions about how to fix the problems.

3.4 Sensor

Sensors are lightweight, swarming, mobile software agents that roam and detect problems. They are modeled after behaviors of social insects and they employ a form of ant-colony algorithms and swarm intelligence [13]. The Sensors' logic is as simple as possible; their power is in their numbers and their diversity. Sensors wander across the geography superimposed on the enclave by randomly adjusting their current heading similar to the movement of real ants. Each Sensor uses a learning classifier [14] to match a particular set of conditions in the hosts they visit. There are two broad categories of Sensors: Markovian (memoryless) and differential. Markovian Sensors look for static conditions that may either define signatures of known problems or well-known anomalous conditions. Differential Sensors look for differences in conditions between hosts in recent memory and their current host. For example, there may be an unusual rate of network connections, a large number of open files, strange file names in system directories, or unusually high processor utilization.

Sensors communicate with each other stigmergically via trails of digital pheromone [5] messages. Decentralized, pheromone-based systems have been demonstrated to simply and effectively solve highly constrained problems where logic-based, optimizing approaches prove intractable [15]. Pheromone-based techniques have been shown to be robust and therefore appropriate for dynamic applications, such as network routing [15, 16].

Sergeants and Sentinels select successful Sensors (those that are consistently rewarded for useful findings) as templates for spawning new Sensors. This may be done by perturbing the parameters of a single Sensor’s classifier or by combining the classifiers of two or more Sensors.

4 Interaction Examples

CID is a mixed-initiative system that operates in a fundamentally different way from systems where initiative is solely human or solely automated. Initiative for actions, handling of interruptions, and responsibility for various activities are all shared. The purpose of CID’s design is to enable humans to function in the role they believe most appropriate at a given time. Thus, CID’s automation is decentralized and flexible, accommodating many types of interactions. In this section, we will highlight several of the

characteristic interactions designed into CID. These imply many more, but we have selected the following to highlight responsibilities within the system and to underscore the flexibility of human intervention they enable.

4.1 Supervisor-Sensor Interaction

The Supervisor can adjust global target parameters to make the Sensor swarm more or less responsive to intrusion evidence and control Sensor population. Two of these parameters are particularly important: activation and crowding tolerance. Sensors are programmed to try to achieve their target levels of these two quantities. The Supervisor may use these two parameters to regulate the size of the Sensor swarm and adjust its responsiveness to attackers. Activation level measures how successful a Sensor has been in collecting evidence useful to Sentinels. High activation yields larger Sensor populations and diverse classifiers. Crowding measures how often a Sensor encounters other Sensors while foraging. If a foraging Sensor senses crowding in excess of its target level, it is more likely to terminate itself reducing the population by removing Sensors that are no longer effective. Highly activated Sensors and Sensors following pheromone trails do not check their crowding metric because high concentrations of Sensors are desirable where problems are being detected. Thus the swarm adapts to new problems and maintains an appropriate size.

4.2 Sensor-Sentinel Interaction

Sentinels and Sensors cooperate to reduce the interruption false positives cause for human defenders. It is important to understand that even very low false positive rates can yield a tremendous amount of interruption that humans are not well equipped to handle [18]. Each false alarm that reaches a human implies an investigation may be necessary. Interactions between the Sensor swarm and the Sentinels weed out most of those false positives before they become alarms that interrupt the human. However, humans still need visibility into the system (Section 4.5), and the system may need human guidance to classify new types of input (Section 4.3).

Sentinels reward Sensors that find useful evidence by adding to their activation level. At activation levels below their target, Sensors remain in foraging mode, actively following pheromone trails and seeking evidence that will earn them a reward from Sentinels. When a Sensor's activation level exceeds the target, it stops foraging and begins dropping pheromone messages that point to the Sentinel where it was most greatly rewarded. As it drops pheromone, it loses activation until it enters foraging mode again. Consistently high activation levels indicate that a Sensor is successful and a Sentinel should select it for spawning.

As the Sentinel collects more and more information from various Sensors that visit (along with information from other sources, Section 4.3) it will develop either a diagnosis of the problem and a means to fix it or an understanding that this situation is actually an acceptable variation. The Sentinel will use this knowledge to decide how to feed future Sensors that visit. When the Sentinel has "enough information" (the reports from the Sensors fit a model) on the problem, it will make a report to its Sergeant. The report to the Sergeant will contain the classifiers used by the Sensors that helped diagnose the problem and any other evidence it has that may be useful in new classifiers.

False positives from the Sensors are not a problem; the Sentinel should feed them unless what the Sensor reports is *known* not to be a problem. This strategy will attract more Sensors until a clearer picture can be constructed. CID uses false positives to generate useful positive feedback and increase its problem-solving ability.

4.3 Sentinel-Supervisor Interactions

Sentinels use a semi-supervised, reinforcement-learning algorithm to classify evidence received from visiting Sensors. They dialogue with the Supervisor (mediated through the enclave Sergeant) to train themselves on problems that are too difficult to decide without human guidance. The untrained Sentinel classifies most kinds of evidence as unknown, asking for feedback from the human Supervisor on its decision. Asynchronous feedback from the human supervisor provides rewards and punishments to the Sentinel's reinforcement-learning algorithm. Sentinels may classify Sensor findings on a continuous scale (see Figure 3) from normal with high confidence (-1) to suspicious with high confidence (+1). Near zero, the Sentinel's classification is more uncertain. Low confidence metric values (close to zero) between the suspicious and normal certainty threshold values trigger the Sentinel to ask for clarifying feedback from the human. When the confidence metric is outside the certainty thresholds, Sentinels will share their classification information with neighboring Sentinels whose machines have similar architectures. In this way the system avoids having humans manually diagnose the same problem again and again.

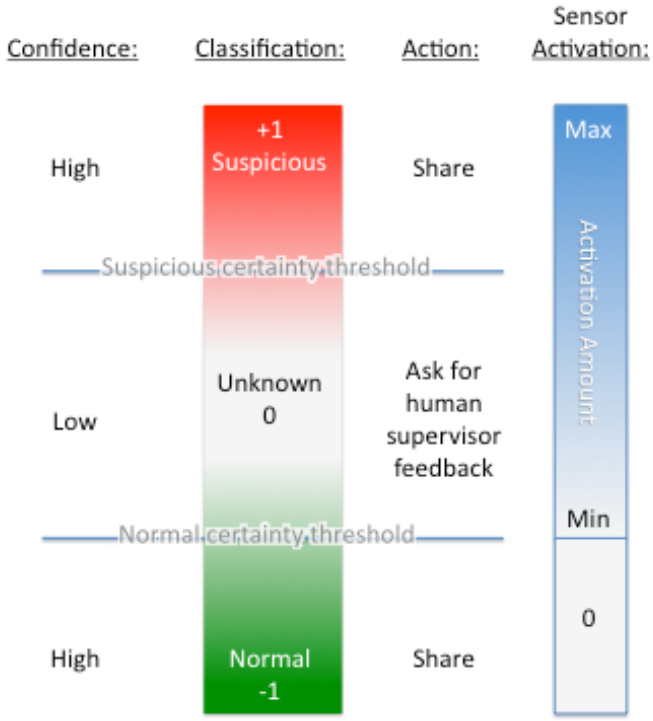

Figure 3: Sentinel classification of and actions based on Sensor data.

Trained Sentinels no longer require direct human feedback. Instead they use information gathered from the Sensors, from the Sergeant, and from other Sentinels to diagnose and repair the problem indicated. First, the Sentinel checks its local database of problems and solutions derived from its own experience and that of its neighbors for a near match. If no match exists, it requests matches from the enclave-level Sergeant agent. If no matching problems exist, the Sergeant will then alert the human Supervisor who can repair the problem manually or dismiss it as a non-problem. A special command shell or interface provided by the Sentinel is instrumented with learning classifiers and can learn how to repair similar problems by monitoring the human's repair activities. When a new known problem and solution are discovered, the Sentinel stores this information in its database and passes the solution up the hierarchy to the enclave-level agent, the Sergeant. The report will contain the classifiers used by the Sensors that helped diagnose the problem and any other evidence it has that may be useful in new classifiers.

4.4 Sergeant-Sentinel Interactions

Sergeants and Sentinels take charge of several areas to reduce human workload in managing large enclaves. Sergeants provide to the Sentinels the enclave security policy and a geographic representation of the enclave that defines the neighbors of each Sentinel. Sentinels provide reports to the Sergeant on problems that have been solved and on Sensor types that have been particularly effective. These interactions allow the human Supervisor to retain situational awareness and supervisory power but do not require the Supervisor to exert direct control to secure the system.

The two-dimensional geography the Sensors move around on is maintained by the Sergeant. The Sergeant builds its geography by periodically conducting a breadth-first search of the enclave and ordering the enclave members along a Peano-Hilbert space-filling curve. This curve preserves network distances as spatial distances on the grid and minimizes the number of hops a Sensor travels on its way to a neighboring Sentinel. As hosts join and leave the network, the Sergeant automatically adjusts and broadcasts the geography to keep it valid.

Sentinels report to their Sergeant about Sensors that have been especially useful for detecting problems on their hosts. The Sergeant uses this list of the most successful Sensors to create anticipatory classifiers based on combinations of highly successful classifiers. Sergeants may also share information from this list of Sensors with the Sergeants of allied enclaves.

4.5 Sergeant-Supervisor Interactions

The Supervisor interacts with the Sergeant through an information visualization and graphical user interface that gives the Supervisor situational awareness of state of the enclave. Utilizing the Netlogo agent simulation toolkit, we have simulated variations of this interface and plan to conduct user studies on which is most effective. Using slider bars, the user can adjust the parameters of the environment to influence the behavior of the agents involved. As mentioned earlier, the Supervisor uses this interface to adjust the activation and crowding tolerance target levels.

In the simulated interface (Figure 4), the Sergeant gives a color indication of the condition of the hosts. Each square in the visualization represents the status of a Sentinel that reports to the Sergeant. The color of the squares could indicate activity levels, security conditions, or any encoding the Supervisor prefers. In the simulation we use the red, green, and blue components of the color to represent file system, memory, and CPU activity levels. Systems performing similar tasks typically have similar colors. In the figure, workstations appear reddish, web servers are shades of blue, and file servers appear gray. A color change would indicate a change in behavior and function and could indicate a problem.

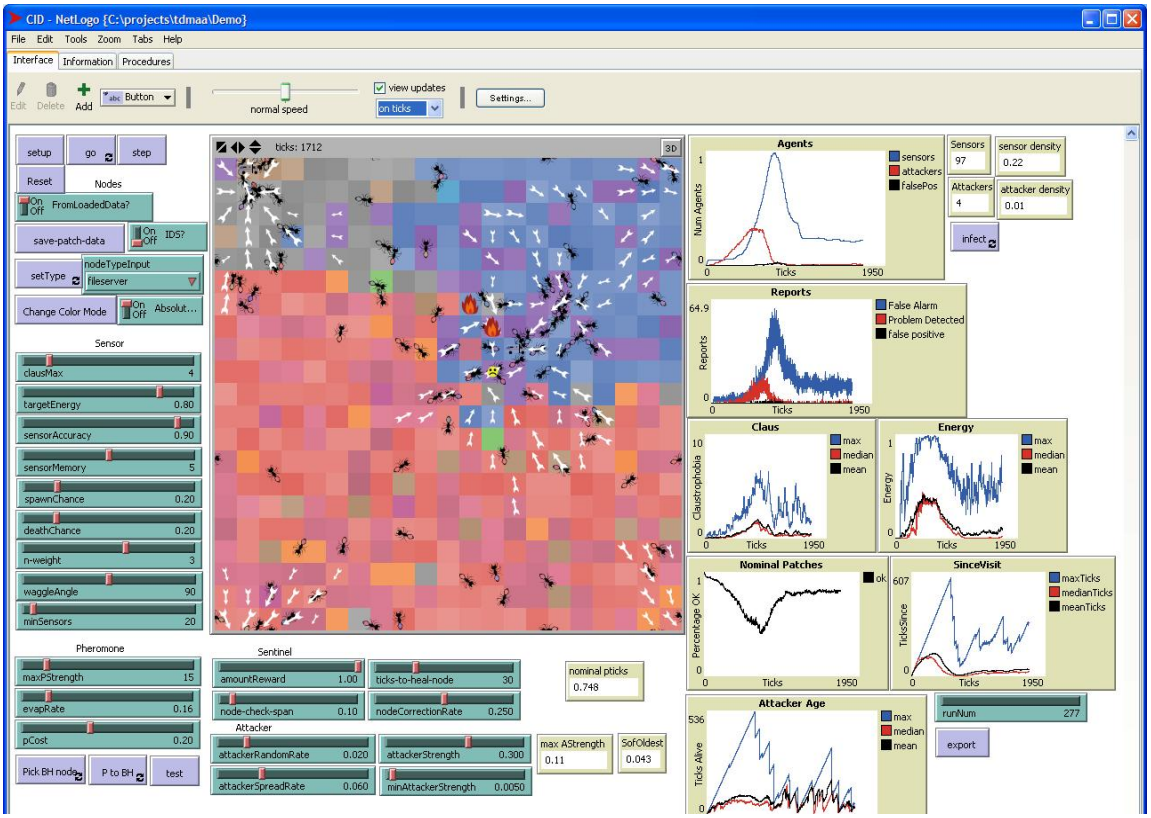

Figure 4: Prototype of Sergeant UI

Alternatively the coloring scheme could show the relative state of each host. A host running in a nominal state would be black. When a Sentinel reports a potential problem that needs human intervention, the Sergeant could color increase the red color in the square. The color could fade as the Sentinel resolved it, but an increasing intensity might indicate that the need for guidance is becoming more urgent. The Sergeant might then enact other means to interrupt the Supervisor and get attention to the troubled machine. It also might contact other neighboring enclaves to ask their Sergeants if similar problems are occurring there.

Another major Sergeant-Supervisor interaction is the establishment of policy for the enclave. We envision this to be a natural language and graphical interaction process where Sergeants and human Supervisors iteratively dialogue to arrive at policies that will enact the business and security objectives desired by the owners of the enclave systems. We anticipate a natural-language dialogue similar to that used by IBM's SPARCLE (Server Privacy ARrchitecture and CapabiLity Enablement) policy workbench [10] which translates natural language requirements into a standard machine readable form. Once the dialogue converges to an acceptable set of policies, the Sergeant will compile the Supervisor's guidance into executable policies that direct the Sentinels in their machine configurations.

Another important Supervisor-Sergeant interaction type involves the Supervisor delegating authority to the Sergeant up to specified authorization limits. Sergeants broker agreements between organizational units on behalf of the Supervisors, so they must have a separate digital identification from the Supervisor they report to. Because they negotiate on behalf of humans they may incur liability for their owning organization. Thus, there must be a mechanism to describe the degree of trust the Supervisor has placed in them. The trust mechanism may be implemented via authorization certificates such as the Simple Public Key Infrastructure (SPKI) certificate (RFC 2692, http://www.ietf.org/rfc/rfc2692.txt) which supports authorization and delegation up to specified limits. For instance, if a Sergeant wishes to negotiate a change in the Service Level Agreement (SLA) its unit has from an Internet Service Provider (ISP) the cost of the

change would be evaluated. The ISP could ask the Sergeant whether it had sufficient trust to negotiate the change autonomously. The Sergeant would present authorization credentials, signed by its Supervisor showing the dollar amount (or whatever unit) it is entrusted with to make decisions of this sort. If the authorization level was sufficient, the ISP would go ahead with the change. If the Sergeant lacked sufficient authorization, the ISP could require the human Supervisor's signature on the change. The Supervisor would probably start out trusting the Sergeant very little, but as the Sergeant gained experience and demonstrated reliability, the Supervisor might increase its delegated authorization limits.

4.6 Sergeant-Sergeant Interactions

Sergeants are the only software agents that communicate with entities external to the enclave such as outside resources and other Sergeants. Outside resources such as virus definitions and attack profiles would assist the Sergeant in formulating guidance for the Sentinels. An updated attack profile would be sent to all Sentinels in the system so they can begin watching for them based on Sensor reports. Likewise, if a new attack is identified and resolved on the system, the Sergeant would pass this information along to allied Sergeants in neighboring enclaves (those it trusts based on the mechanisms in 4.5). The information passed might be a simple description or it might contain parameters and classifiers actually used by the Sensors that helped the system identify the novel attack (or effectively deal with known attacks). Sergeants in charge of other enclaves could then instantiate these Sensors in their own enclaves. A benefit of this approach is that the Sensor definition can be shared without sharing the data from any machine within the enclave and avoiding data exfiltration or violation of proprietary information.

4.7 Sensor Agent Management

Decentralized, pheromone-based systems have been demonstrated to simply and effectively solve highly constrained problems where logic-based, optimizing approaches prove intractable. Figure 4 shows the simulation model of a pheromone based system we built to investigate tradeoffs among pheromone implementations. Central to any system that relies on swarming sensors is the ability to communicate information and influence action. The CID framework uses a pheromone-based system for inter-Sensor communications that has been demonstrated to be simple and efficient [15]. This decentralized communication approach allows Sensors to not only achieve the local goal of finding food, but also collectively achieve the global objective of discovering inflected computers in a dynamic environment.

Pheromone-based techniques have been shown to be robust and therefore appropriate for dynamic applications, such as network routing [15][16]. This is an important feature for CID since the number of computers and their security status (*e.g.* infected or clean) will change over time. As previously discussed, a Sensor that discovers pheromone has a probability of following it that is related to the pheromone strength. The rate of pheromone decay and how pheromone is potentially overwritten will impact the system adaptiveness, as discussed in [15][16].

5 Related Work

The mixed-initiative approach is not common in cyber security applications. CID blends the initiative of humans and a hierarchy of agents to avoid the extremes of either marginalizing the human in favor of black-box machine intelligence or including the human in the wrong loop, too close to the problem so that the system speed is bounded by human reaction time.

In earlier research [20] interviews with operators of computer systems we interviewed agreed that a "five nines" (99.999%) reduction of information was necessary for them to retain situational awareness. System administrators and security officers we interviewed indicated that their preference was to see changes in the visualization only when something that required human attention occurred. Feedback from other clients revealed that operators find even a hundred alerts per day too taxing. Clearly, operational staff need to reduce the amount of workload required by the available data, and teaming with autonomous agents is an excellent potential solution.

The hierarchical arrangement of humans and various types of agents has been proposed in a variety of forms. Ibrahim [17] proposes a network management system that utilizes a hierarchy of mobile agents to manage a distributed system while reducing bandwidth consumption. Parunak [6] proposes a heterogeneous hierarchy to solve highly constrained military movement problems. However, the human mostly serves as an observer in these systems and has very limited interaction with the agents.

The network management system proposed by Ray [19] proposes a similar arrangement of agents as well as keeping the human as a high level policy maker. Our system is further decentralized with no concept of a "home base" for our mobile (and non-cognitive) Sensors.

6 Conclusions and Future Research

Thus far we have focused our research mainly on the Sentinel and Sensor behavior in our simulated environment thoroughly mapping out the behavior of the Sensors and their interactions with the Sentinels. We have created a demonstration system of the Sensor-Sentinel levels in JADE and have created vertical prototypes of interactions at the Supervisor, Sergeant, and Sentinel levels. However, we have several areas of future research we intend to pursue such as:

- Defining the dialogue between Supervisor and Sergeant that translates human guidance into unambiguous policy for the Sentinels
- Experimenting with our practical Sensor-Sentinel demonstration on a computer cluster
- Conducting usability studies on the prototype Sergeant user interface
- Learning to generalize knowledge gained from solutions to problems that share similarities
- Testing the framework against real-world attacks on the systems being monitored

The CID framework keeps the human in the right loop via a hierarchy of software agents that implements high-level policy and handles low-level events. The framework resolves issues in a decentralized manner at the lowest possible level to minimize alerts that interrupt the human user. This builds user trust in the system and improves system efficiency. The efficiency realized by CID's mixed-initiative approach is highly scalable and difficult to disrupt. We believe that by sharing system control with a hierarchy of agents and utilizing the truly adaptive capabilities of swarm intelligence we can greatly enhance the security of our computational infrastructures and the societies that depend on them.

References

[1] Allen, J.E., Guinn, C.I., Horvtz, E.: Mixed-Initiative Interaction. IEEE Intelligent Systems. 14, 14—23 (1999)
[2] Frincke, D., Wespi, A., Zamboni, D.: From Intrusion Detection to Self-Protection. Computer Networks. 51, 1233--1238 (2007)
[3] Dahiyat, E.A.R.: Intelligent Agents and Intentionality: Should We Begin to Think Outside the Box? Comput. Law Secur. Rep. 22, 472--480 (2006)
[4] Scher, M.B.: On Doing "Being Reasonable." ;login: 31, 40--47 (2006)
[5] Brueckner, S.: Return from the Ant: Synthetic Ecosystems for Manufacturing Control. PhD Dissertation. Department of Computer Science, Humboldt University, Berlin, (2000)
[6] Parunak, H.V.D., Nielsen, P., Brueckner, S., Alonso, R.: Hybrid Multi-Agent Systems: Integrating Swarming and BDI Agents. In: Brueckner S.A.., Hassas, S., Jelasity, M, Yamins, D. (eds.) ESOA 2006. LNAI, vol. 4335, pp 1--14. Springer, Heidelberg (2007)
[7] Smieja F.: The Pandemonium System of Reflective Agents. IEEE Transactions on Neural Networks. 7, 97--106 (1996)
[8] Selfridge, O.G.: Pandemonium: a Paradigm for Learning. In: Anderson, J.A.D., Rosenfeld, E.: Neurocomputing: Foundations of Research, pp 115--122. MIT Press, Cambridge, MA, USA (1988)
[9] Kephart, J.O., Chess, D.M.: The Vision of Autonomic Computing. Computer 36, 41--50 (2003)

[10] Karat, J., Karat, C.-M., Brodie, C., Feng, J.J.: Privacy in Information Technology: Designing to Enable Privacy Policy Management in Organizations. Int. J. Human.-Computational. Studies. 63, 153--174 (2005)
[11] Rao, A.S., Georgeff, M.P.: BDI Agents: From Theory to Practice. In: First International Conference on Multi-Agent Systems (ICMAS-95). pp. 312--319. AAAI Press, Menlo Park, CA, USA (1995)
[12] Knight, J., Heimbigner, D., Wolf, E.L., Carzaniga, A., Hill, J., Devanbu, P., Gertz, M.: The Willow Architecture: Comprehensive Survivability for Large-Scale Distributed Applications. In: Distributed Applications: Intrusion Tolerance Workshop, Dependable Systems and Networks (DSN 2002). Washington DC (2002)
[13] Dorigo, M., Gambardella, L.M.: Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. IEEE Trans. Evolutionary Computation. 1, 53--66 (1997)
[14] Holland, J.H., Booker, L.B., Colombetti, M., Dorigo, M., Goldberg, D.E., Forrest, S., Riolo, R.L., Smith, R.E., Lanzi, P.L. Stolzmann, W., Wilson, S.W.: What Is a Learning Classifier System? In: Lanzi, P.L., Stolzmann, W., Wilson, S.W. (eds.) Learning Classifier Systems '99. LNCS, vol. 1813 pp. 3--32 Springer, Heidelberg (2000)
[15] Di Caro, D., Dorigo, M.: AntNet: Distributed Stigmergetic Control for Communications Networks. J. Artifificial Intelligence Research. 9, 317--365 (1998)
[16] Bonabeau, E., Henaux, F., Guérin, S., Snyers, D., Kuntz, P., Theraulaz, G.: Routing in Telecommunications Networks with "Smart" Ant-Like Agents. Working Papers 98-01-003, Santa Fe Institute. (1998)
[17] Ibrahim, M.A.M.: Distributed Network Management with Secured Mobile Agent Support. In: Proceedings – 2006 International Conference on Hybrid Information Technology, ICHIT '06, vol. 1, pp. 244--251. IEEE, Piscataway, New Jersey (2006)
[18] Axelsson, S. The base-rate fallacy and the difficulty of intrusion detection. ACM Trans. Information and System Security, 3(3):186–205, 2000.
[19] Ray, P., Parameswaran, N., Lewis, L., Jakobson, G.: Distributed Autonomic Management: An Approach and Experiment Towards Managing Service-Centric Networks. In: 5th International Conference on Service Systems and Service Management, ICSSSM'08, pp. 1--6. IEEE, Piscataway, New Jersey (2008)
[20] Fink, G.A., et al. Bridging the Host-Network Divide: Survey, Taxonomy, and Solution. in In Proceedings of the 20th Large Installation System Administration Conference (LISA '06). 2006. Washington, DC: USENIX.