MTC2: A Command and Control Framework for Moving Target Defense and Cyber Resilience

Marco Carvalho[1,2]
[1]Intelligent Communication and Information Systems
Harris Institute for Assured Information
Dept. Computer Sciences
Florida Institute of Technology,
Melbourne, FL 32901
mcarvalho@fit.edu

Thomas C. Eskridge, Larry Bunch,
Adam Dalton, Robert Hoffman,
Jeffrey M. Bradshaw, and Paul J. Feltovich
[2]Florida Institute for Human and Machine Cognition
Pensacola, FL 32502 {teskridge,lbunch,adalton,
rhoffman,jbradshaw,pfeltovich}@ihmc.us

Daniel Kidwell and Teresa Shanklin
U.S. Department of Defense
Maryland, USA
{dlkidw2,tashank}@tycho.ncsc.mil

***Abstract*— In this paper we discuss the need for a new command and control (C2) approach for the practical deployment of Moving Target Defenses (MTDs) enterprise networks. We describe some of the requirements and constraints associated with the combined use of multiple moving target defenses, and introduce a human-agent teamwork approach for the command and control of MTDs. We introduce and discuss some of the specific concepts and technologies that could play an important role in the development of this capability, and conclude by describing the implementation details of the human-agent teamwork C2 prototype, called MTC2.**

Keywords—Moving Target, Cyber Security, Cyber Resilience

I. INTRODUCTION

The conventional paradigm in computer network defense is largely based on the notion of protecting a target network infrastructure that is mostly fixed, and has been designed and deployed to provide a set of services to legitimate users. Conventional network security has been traditionally been centered on the notion of creating gates, barriers, or detection mechanisms in the network, hoping to identify and separate legitimate from malicious traffic.

Moving Target Defense (MTD) proposes a conceptual shift in this paradigm. The MTD concept proposes that the target itself does not need to be static, and that a dynamic (or moving) target design can be conceived such that it maintains functionality for legitimate users, while making it difficult for adversaries to identify and exploit system vulnerabilities. Conceptually, a moving target defense relies on sets of tools or mechanisms responsible for monitoring the state of the computer network, and a set of tools or mechanisms responsible for the mobility, or effectively changing the system.

Several kinds of MTD capabilities have been proposed [3][5][6]. *Defense monitoring capabilities* include intrusion detection, server and firewall log analysis, and traffic pattern monitors. *Mobility capabilities* create dynamic changes in the target system, directly affecting its "mobility space." Mobility capabilities focus on four different kinds of changes that can be made to the system. These changes may be associated with: a) the computational platform (i.e., operating systems and architecture); b) the application or service itself; c) the data used by services and applications; or d) the network.

While early MTD results have been encouraging [5], questions about their general practicality and utility are still unresolved. Interdependencies among individual defense capabilities and the functionality of critical applications and services are poorly understood. Moreover, the study of the interactions among different configurations of individual tools, or among whole toolsets for different operational contexts, has been hampered by the dearth of generally available tools and techniques. A better understanding of these interactions is essential for deployment of multiple moving MTDs, and even more so when adaptation to (or co-evolution with) the adversary is being considered. All these reasons motivate our interest in developing tools that address the MTD design requirements for Command and Control (C2), which is the focus of this paper.

A MTD C2 must be capable of managing, combining, and optimizing the use of multiple MTDs under different operational contexts and mission requirements. Additionally, these capabilities must not only enable low-level distributed monitoring, control, and coordination but also the high-level understanding of the successes and failures of MTD strategies and the possible inclusion of human guidance in the selection, modification and execution of these strategies. For this purpose, we advocate an approach to MTD C2 based on principles of human-agent teamwork [1][2].

In this work we will discuss the rationale for the proposed approach, and will introduce the MTC2 prototype framework. We illustrate MTC2 operation and provide a preview of the complexity of the MTD parameter control space as a precursor to developing algorithms for optimal control of MTD

978-1-4799-0503-4/13/$31.00 ©2013 IEEE

interactions. Optimal defense configuration by the MTC2 is left as part of our future work.

II. A Moving Target Defense Infrastructure

Three main components are necessary for a complete MTD system: 1) monitoring capabilities, 2) mobility capabilities, and 3) command and control capabilities.

A classical feedback-loop control formulation of the defense infrastructure can be used to represent the interdependency among these components [2]. In our approach, the human analyst works in concert with the adaptive algorithms that update the defenses (i.e., actuators in mobility space) based on feedback from the state of the network and services (Figure 1). One important difference from a single feedback-loop approach in this formulation is that the sensing components can be configured and deployed at runtime, allowing the C2 to configure both the sensors and the actuators.

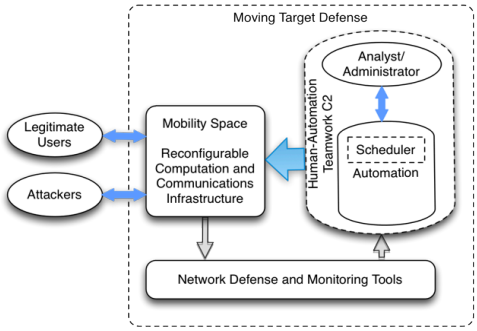

Figure 1. A Human-Agent Teamwork Approach to MTC2

In MTC2, we rely on principles of human-agent teamwork [7], which regards the human as an intrinsic part of the control loop (Figure 1). We first outlined how these principles could be applied to MTD C2 in [9]. Our overall approach incorporates two important concepts: coactive emergence [8] and organic resilience [4], both introduced in section 3. In section 4, we introduce our specific design approach and in section 5 demonstrate its prototype implementation, called MTC2.

III. Toward A Human-Agent Teamwork Approach for MTC2

Most approaches to MTD—and cyber operations in general—tend to focus on specific mechanisms, with the analyst playing the role of compensating for various shortcomings in the opaquely-constructed automated control loop, rather than being part of the perception, decisions, and actions taking place within the loop itself whenever it makes sense to do so. We believe that more resilient performance can be obtained by leveraging the joint capabilities of humans and automation – so long as the work system is designed to support such teamwork (See Figure 1). The crucial role of humans is to: 1) keep the technology aligned to richer situation contexts than what can be modeled within the system itself, 2) verify ongoing progress and effectiveness, 3) use their expertise to shape and reshape system actions, 4) take corrective action as needed, and 5) contribute the human powers of creativity, perception, and decision-making to the work.

A. *Coactive Emergence as an Approach to Human-Agent Teamwork*

We apply the concept of *coactive emergence* in the design of human-agent teamwork methods for cyber operations. The phrase "coactive emergence" captures what happens as secure system configurations, effective responses to threats, and useful interpretations of data are continuously developed through the interplay of sensemaking and decision-making activities jointly undertaken by analysts and software agents. The word “coactive” emphasizes the joint, simultaneous, and interdependent nature of such collaboration among analysts and agents [8].

B. *Organic Resilience*

Our approach to resilience relies on software agents to assure graceful, robust, and adaptive performance in the face of stressors and surprise. Organic resilience [4] proposes a self-organizing approach to collective obligations, building from a biologically inspired analogue to functional differentiation. In MTC2 some of the organic resilience principles are proposed for the self-organization of strategies for defense management and parameterization.

As with many biological systems, the functionality of organic resilience is to avoid static and centralized single-point-of-failure solutions in work system organization. Thus, although groups of agents within the work system may be collectively responsible for jointly executing various tasks, the specific responsibilities assigned to human or machine agents are not completely sorted out in advance. The goal of organic resilience is to promote self-organization of both human and software agents within the constraints of their individual capabilities, current applicable policies, operational context, and availability of resources.

IV. Components of MTC2

A prototype of the MTC2 framework has been developed to demonstrate and evaluate the proposed concepts. The current implementation relies on a set of well-established services and capabilities that can help support the principles of coactive emergence, organic resilience, and the easy integration with existing MTDs. In this section, we provide a brief description of some of the key components in our MTC2 prototype.

A. *Software Agent Framework*

To meet the challenging demands of human-agent teamwork in cyber applications, we have developed a new agent framework called *Luna* [1]. Three reasons dictated our decision to implement Luna: Security requirements, formulation of common agent tasks by end users, and coordination of joint activity within mixed teams of humans and software agents.

In considering the *security requirements* of current software agent platforms, Luna allows policy constraints to govern behavior at every level of the agent system including agent management, communication, and resource utilization. With respect to the need for a platform supporting the interactive *formulation of common agent tasks by end users*, rather than by software developers, we believe that policy management systems may also prove useful, if not necessary. Finally, to

satisfy the need for a platform that would provide built-in support for effective *coordination of joint activity within mixed teams of humans and software agents*, we have implemented methods for producing and obtaining progress appraisals, agent runtime status, and shared data and state among agents. These capabilities enable communication and control between humans, agents, and agent environments.

A modular implementation of the Luna framework called MIRA (My Intelligent Research Agent) is currently underway, and is designed to provide the interface and APIs of Luna, but to optionally run without the policy infrastructure (or other services) for quick prototyping, calibration and agent design. When configured to instantiate the supporting policy services, MIRA is functionally equivalent to Luna. MIRA is designed to be modular and to allow for the easy integration of alternative services for all critical tasks of the agent system, including basic communications, discovery, messaging, pub-sub, policies, and others. MIRA provides an easy framework for the design and evaluation of alternative services and capabilities, without any changes to the software agents. MTC2 agents can be implemented for both the Luna and MIRA frameworks, with functional equivalence.

B. Policy Services Framework

We utilize a policy management and enforcement framework called KAoS to govern the actions of the agents in MTC2. Whereas other policy-based approaches are available, the ontology-based approach proposed in KAoS enables semantically-rich specifications of virtually any kind of constraint on any specific kind of action in richly defined dynamic contexts.

In support of human-agent teamwork, each MTC2 software agent can be governed by policies that are designed to assure the key features of *observability* (e.g., mandatory status updates at an appropriate frequency, or in response to specified events), *directability* (e.g., immediate responsiveness to redirection due to policy changes), *interpredictability* (e.g., obligation policies assuring that required behavior will be executed within a specified time period), and *adaptation* (e.g., policies governing the range of adaptations permitted and the process of propagation to other agents).

An agent-based implementation of context mirroring across different multiple execution environments is also provided as part of the MTC2 support infrastructure. Through policy, the agent execution environment also governs agent progress appraisal [10] — informing human analysts about what tasks and which agents are significantly ahead or behind schedule, and thus re-planning their own efforts on interdependent tasks accordingly.

In network security scenarios, there is always a tradeoff between the level-of-defense and the level-of-service provided to legitimate users. People are well-positioned to evaluate that tradeoff, based on context and mission requirements. Software agents, on the other hand, are well-suited for identifying specific defense configurations that will satisfy user requirements in conformance with system policies and other constraints. High-level tradeoff requirements are defined by users through policies and enforced by the software agents through a configuration to optimize the proposed trade-off. Humans and agents, in this context, collaborate to jointly control the specific MTD at different levels of abstraction.

C. MTC2 APIs

The MTC2 API is designed to facilitate integration of command and control with existing MTDs. It includes four main interfaces: the KnowledgeModel API, the Strategy API, the Registration API, and the Event API.

The *KnowledgeModel API* provides a general method for representing resources protected by, and concepts and controls utilized by a MTD. The API distinguishes among types (or classes) of resources and specific instances of those resources. It is designed to be compatible with ontology-based descriptions of these classes and instances (i.e., OWL) for straightforward integration with semantic reasoners. Using the KnowledgeModel API, we have built convenience classes for registering the types of things that are important to the defense, such as logical services and their execution environments (e.g., applications, operating systems, hardware).

The *Strategy API* provides an interface by which MTC2 monitors and directs the operation of the MTDs used to protect a set of services. *Strategy Templates* are used to specify the parameters and ranges of the controls for each MTD. *Strategy Descriptions* restrict the range of the parameters of a strategy template to a subset of the allowed types and instances, and are the primary mechanism used by the distributed MTC2 to direct defense movements and control changes. *Strategy Instances* describe the possible instantiations that have been selected to satisfy a given strategy description. Finally, the *Strategy State* describes the instantiation currently being used to satisfy the strategy.

The *Registration API* allows MTDs to register with MTC2 services, and provides MTC2 with handles back to the MTDs. MTC2 then uses the Strategy API to observe and direct changes to MTD strategies. The *Event API* provides a generic abstraction for MTC2 services to be notified about changes to the system state. Such events may include strategy events (e.g., strategy changed, strategy state changed), information events (e.g., intrusion alerts received from an IDS), and progress events (e.g., noting the fact that the movement of service X to host Y is 90% complete).

Combined, these APIs allow for the integration of standard MTDs into the MTC2 framework. The APIs have been designed to allow MTDs to describe themselves in terms of both the controls that the C2 will manage and the feedback the MTD will provide. They also support visualizations that facilitate sensemaking and MTD execution activities within the human-agent team. In this paper, we do not describe the specific visualization components of MTC2, but the interested reader could find more details about our principles of interactive visualization design in [8].

V. AN ILLUSTRATIVE SCENARIO

To test the MTC2 and illustrate the capabilities of human-in-the-loop control of MT defenses, we created and examined the performance differences of a series of application scenarios that a network administrator might typically encounter. In these

scenarios, the MTC2 interfaces are used to choose and evaluate different defense configurations and produce data to help understand the impact of the MTD on the legitimate and malicious traffic success rates.

The demonstration environment consists of thirteen virtual machines running a combination of Windows and Linux operating systems. Collectively, the machines emulate a small office environment with two private sub-networks behind a router/firewall (see Figure 2). The environment publicly exposes three services, each of which is protected by a different MT Defense. For the purposes of this paper, we will focus on one of these services and its defense, an FTP service protected by an Application Diversity defense.

The Application Diversity defense is designed to protect against exploits that are specific to a particular brand or implementation of an application providing a service. It operates by periodically modifying a router to redirect service requests to one of several running applications that are functionally equivalent yet use different application implementations and versions. Application Diversity has two controls, the Frequency with which the switching strategy is executed, and the Distribution of time spent in each application.

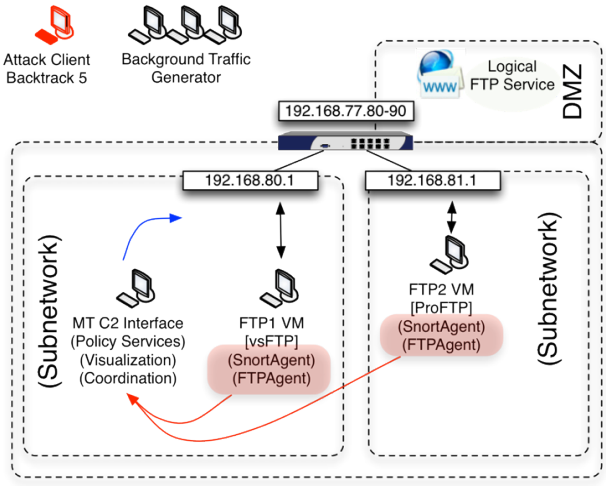

Figure 2. Partial Network topology showing the FTP services involved in the example MTC2 scenario.

In this example, the Application Diversity defense redirects FTP requests to one of two functionally equivalent configurations of the FTP service: the ProFTP application running on one of the subnets, and the vsFTP application running on another subnet.

The traffic scenarios are constructed to produce a combination of legitimate background traffic and attempted attacks against the FTP service. In each scenario, traffic is generated at specified rate and the attacks are continuously submitted to the server. The traffic generator keeps track of the number of requests, successful responses, and failed responses for the scenario. Attacks on the FTP service are designed to exploit a vulnerability that allows root-level control of the VM, and occur at the rate of once every 25 seconds. The attacker uses the root access to shut down the FTP service for approximately 20 seconds, which degrades its overall performance. Legitimate FTP requests are generated at the rate of one every two seconds.

Overall network performance improves when the rate of legitimate request failures decreases. Thus, the objective of the MTC2, in this case, is to maximize the number of successful legitimate requests per unit time, while minimizing the number of successful attacks per unit of time. It is important to note that, in this example, MTC2 is concerned with the aggregate success-failure ratios for both legitimate and malicious traffic, and not with any one specific attack. Thus, we are not concerned with blocking an individual FTP attack, but in reducing the total number of attacks over time.

Our scenario implementation includes the initial version of the MTD APIs, as well as integrated C2 elements based on Luna and KAoS. We are in the process of collecting results for several scenarios designed to evaluate overall network performance when the agents reconfigure the defenses with and without humans as team members. While this paper covers the MTC2 managing a single defense, the full prototype includes four defenses operating simultaneously.

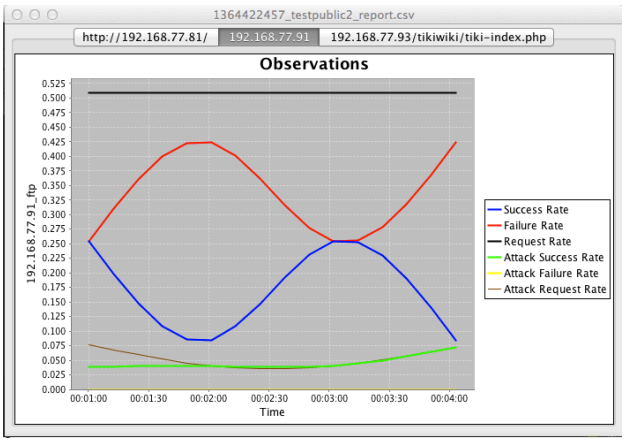

Figure 3. Baseline performance of the vulnerable FTP service subject to fixed rates of legitimate and malicious traffic.

Figure 3 illustrates the performance of the vulnerable FTP server under the demonstration scenario conditions without using the Application Diversity defense, and provides a performance baseline we can use to compare with later experiments. The straight black line at 0.5 shows the rate of background client requests (one every 2 seconds), and the blue and red lines below it indicate the rate of successful and unsuccessful legitimate user requests, respectively. The remaining lines are the rates of attack requests, attack successes, and attack failures. The successful attack line (green) hovers around the 0.04 mark indicating successful attacks at a rate of once every 25 seconds. The high rate of legitimate request failures indicates that the attacks are succeeding. With every attack succeeding, a question arises as to why there are any successful client requests fulfilled. The reason is that the difference between the attack rate of once every 25 seconds and the recovery of the service after 20 seconds leaves a 5 second window, allowing at least two client requests to be handled before the next attack. When the same

attacks are run against the second, non-vulnerable FTP service, no attacks are successful and all client requests receive successful responses (not shown in the graph).

In the next scenario, we instantiate the Application Diversity defense and configure it to switch between alternative implementations of the FTP server every 30 seconds (the Frequency control), and to run a balanced distribution, with implementations running for 15 seconds each (the Distribution control). An illustration of this switching schedule is shown in Figure 4.

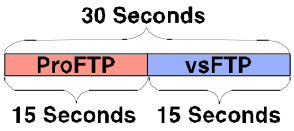

Figure 4. Balanced switching schedule for Application Diversity defense of FTP service.

The results of running the same background and attack scenario as in our baseline are shown in Figure 5.

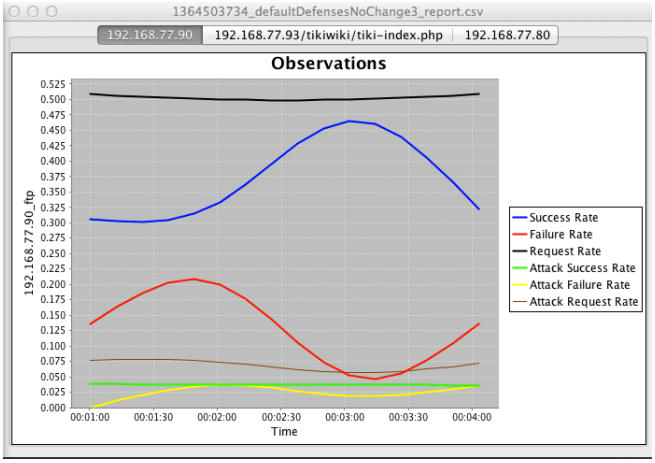

Figure 5. Effects of the Application Diversity Defense configured by the MTC2 for a balanced 30-second interval.

Notice that the red line in Figure 5 indicating failed legitimate FTP requests is now much lower than the number of successfully fulfilled requests, indicating that the Application Diversity defense of the FTP service was performing well. It should be expected that any time we switch between a vulnerable and non-vulnerable service implementation, the overall number of request failures will diminish, simply because there is less time spent in a vulnerable state.

However, it would not be prudent to expect that if the defense spends half of its time running the vulnerable server and half of its time running the non-vulnerable server, it would result in 50% failed legitimate requests. For that to be true, the rate of attacks and the rate of defense application switching would need to be carefully synchronized. If they are not synchronized, there can be situations where the defense switches to the vulnerable application, but the attacker does not send a request to that application before it switches out again. Recall that in this experiment, attacks are launched every 25 seconds, but the time spent using the vulnerable application before the Application Diversity defense switches is only 15 seconds.

The MTC2 is designed to utilize the rates of successful and failed requests as feedback in the defense control strategies that will optimize defense behavior. We are currently investigating several optimization approaches. As a precursor to that investigation, we experimented with biasing the time between switching between applications. Figure 6 shows one example of a switching schedule strategy that biases one application over another. In this case, the distribution of time between the applications is modified to spend 80% of the time on the non-vulnerable application and 20% of the time on the vulnerable application.

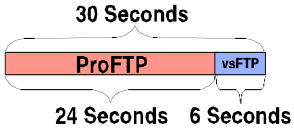

Figure 6. Biased switching schedule for Application Diversity defense of FTP service.

Figure 7 illustrates a variation of the scenario where the Application Diversity defense starts with the initial 50-50% strategy (see Figure 5) for two minutes, then switches to the 80-20% biased strategy.

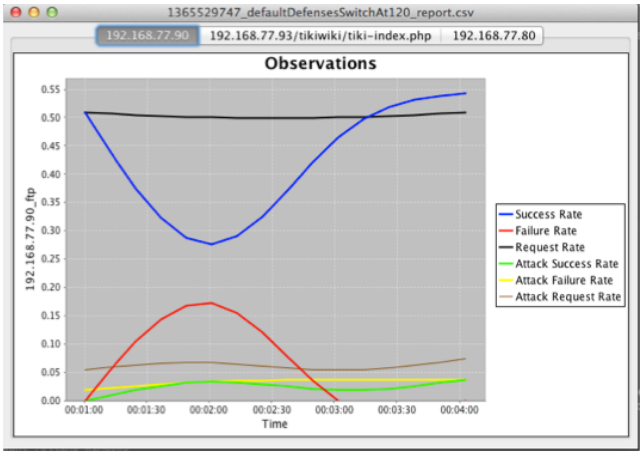

Figure 7. Effects of the Application Diversity Defense configured by the MTC2 for a 80%-20% split, with a 30 second interval

These results show initial performance consistent with that in Figure 5, where the peak of failed legitimate requests occurs around the 2:00 mark. Once the biased switching strategy is instantiated, a significant difference in overall network performance results.

It is interesting to note in Figure 7 that the rate of successfully fulfilled legitimate requests (shown in blue) begins to exceed the line denoting the legitimate request rate (shown in black) at about the 3:15 mark. This can result from request queuing or network transition time due to defense application switching. Further experimentation will measure network variables such as these to determine the exact cause. These measurements will be necessary to ensure that backlogs do not get so large as to cause additional problems. The

conditions illustrated in Figure 5 and Figure 7 represent the effect of only one of the configuration parameters: the distribution of time in each service implementations. The frequency of control strategy execution is another parameter that can be controlled (e.g., 30 seconds in Figure 4 and Figure 6).

Figure 8 illustrates a different configuration for the proposed scenario where the frequency of application switching begins at 30 seconds, and changes to 15 seconds two minutes into the run, running each application for only 7.5 seconds before changing.

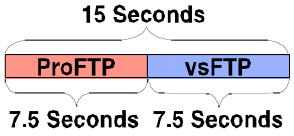

Figure 8. Higher frequency, balanced switching schedule for Application Diversity defense of FTP service.

The results shown in Figure 9 indicate that there is another control strategy for the defense that significantly reduces the effects of the FTP service attacker. However, taken with Figure 5 and Figure 7, this demonstration also provides a glimpse into the complexity of the surface created by the defense control parameter space.

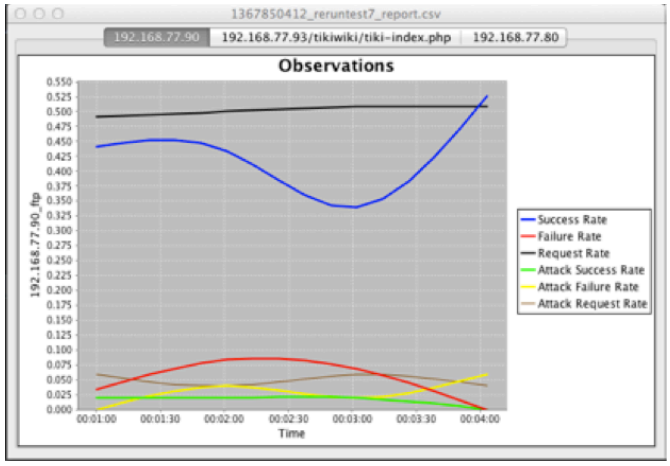

Figure 9. Example output showing the results of modifying the time between application changes

Because of this complexity, the user may not be able to easily navigate the defense control parameter space unaided. Increased computational support in the form of predictive agents and optimization strategies could be used to assist the operator in generating and comparing alternative control strategies.

While fairly simple, the scenarios illustrated in this work are suggestive of the complexity of moving target defenses with multiple configurations parameters and their effects on system performance. The problem is further complicated when multiple defenses are combined and must be jointly coordinated. A human-agent teamwork approach would combine user expertise with feedback from automated search and learning.

VI. Conclusions

In this paper we have discussed requirements for an MTD C2 framework and presented preliminary results obtained from a demonstration implementation that manages MTD defenses and that shows the complexity of the control parameter space. The major components of our MTC2 framework (Luna/MIRA, KAoS, and the MTC2 API) support the human-agent teamwork principles of coactive emergence and organic resilience, and provide a way to integrate existing MTDs into the control loop. Our focus in this paper is on the discussion of specific capabilities for MTC2 support, and we provided several example scenarios that demonstrate the MTC2 API and illustrate multiple control changes that improve performance compared to a baseline. Future tests of the framework will focus on the integration of multiple independent and overlapping defenses. As we progress with the prototype development, new results and MTC2 API's will be made available to the research community.

References

[1] Bunch, L., J.M. Bradshaw, M. Carvalho, T. Eskridge, P. Feltovich, J. Lott and A. Uszok. Human-Agent Teamwork in Cyber Operations: Supporting Co-Evolution of Tasks and Artifacts with Luna. Invited Paper in Ingo J. Timm and Christian Guttmann (eds.), *Multiagent System Technologies: Proceedings of the Tenth German Conference on Multiagent System Technologies (MATES 2012)*, Trier, Germany, 10-12 October 2012. Berlin, Germany: Springer, LNAI 7598, pp. 53-67.

[2] Carvalho, M., Bradshaw, J. M., Bunch, L., Eskridge, T., Feltovich, P., Hoffman, R., Lott, J., and Kidwell, D. A human-agent teamwork approach to moving target defense command and control. Poster presented at the Moving Target Research Workshop, Washington, D.C., June 2012. Online at: http://cps-vo.org/node/3850

[3] Ghosh, A. K., Pendarakis, D., and Sanders, W. H. Moving target defense co-chair's report - National Cyber Leap Year Summit 2009. Tech. rep., Federal Networking and Information Technology Research and Development (NITRD) Program, 2009.

[4] Carvalho, M., Lamkin, T., and Perez, C. Organic resilience for tactical environments. In 5th International ICST Conference on Bio-Inspired Models of Network, Information, and Computing Systems (Bionetics), Boston, MA, December 2010.

[5] Jajodia, S., Ghosh, A. K., Swarup, V., Wang, C., and Wang, X. S. Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats, 1st ed. Springer Pub. Company, Incorporated, 2011

[6] Sheldon, F. T., and Vishik, C. Moving toward trustworthy systems: R&D Essentials. Computer 43, 9, pp. 31-40, September 2010.

[7] Bradshaw, J.M., Virginia Dignum, Catholijn Jonker, and Maarten Sierhuis. Guest editor introduction. Special issue on Human-Agent-Robot Teamwork (HART), *IEEE Intelligent Systems*, March/April 2012 (vol. 27 iss. 2), pp. 8-13.

[8] Bradshaw, J.M., Marco Carvalho, Larry Bunch, Tom Eskridge, Paul Feltovich, Matt Johnson, and Dan Kidwell. Sol: An Agent-Based Framework for Cyber Situation Awareness. *Künstliche Intelligenz*: Volume 26, Issue 2 (2012), pp. 127-140.

[9] Carvalho, Marco, J.M. Bradshaw, Larry Bunch, Tom Eskridge, Paul J. Feltovich, Robert H. Hoffman, and Daniel Kidwell. Command and Control Requirements for Moving Target Defense. *IEEE Intelligent Systems*, May/June 2012 (vol. 27 iss. 3), pp. 79-85.

[10] Feltovich, Paul, Jeffrey M. Bradshaw, William J. Clancey, Matthew Johnson, and Larry Bunch. "Progress appraisal as a challenging element of coordination in human and machine joint activity." In A, Artikis, G. O'Hare, K. Stathis, & G. Vouros (Eds.) Engineering Societies in the Agents World VIII, , 2008.