Automated Effectiveness Evaluation of Moving Target Defenses: Metrics for Missions and Attacks

Joshua Taylor
Siege Technologies
1105 Floyd Avenue
Rome NY USA
joshua.taylor@siegetechnologies.com

Kara Zaffarano
Siege Technologies
1105 Floyd Avenue
Rome NY USA
kara.zaffarano@siegetechnologies.com

Ben Koller
Siege Technologies
1105 Floyd Avenue
Rome NY USA
ben.koller@siegetechnologies.com

Charlie Bancroft
Siege Technologies
540 North Commercial Street
Manchester NH USA
charlie.bancroft@siegetechnologies.com

Jason Syversen
Siege Technologies
540 North Commercial Street
Manchester NH USA
jason.syversen@siegetechnologies.com

ABSTRACT
In this paper, we describe the results of several experiments designed to test two dynamic network moving target defenses against a propagating data exfiltration attack. We designed a collection of metrics to assess the costs to mission activities and the benefits in the face of attacks and evaluated the impacts of the moving target defenses in both areas. Experiments leveraged Siege's Cyber-Quantification Framework to automatically provision the networks used in the experiment, install the two moving target defenses, collect data, and analyze the results. We identify areas in which the costs and benefits of the two moving target defenses differ, and note some of their unique performance characteristics.

Keywords
Moving target defenses; metrics; modeling; cyber-security; virtualization; quantification; experimentation

1. INTRODUCTION
Moving Target Defenses (MTDs) have a large potential to improve cyber-security of individual hosts, as well as at the network level. Quantitatively evaluating the effectiveness of MTDs is a challenging task for a number of reasons. MTDs can increase the complexity of a networked environment, which can make it difficult to apply the same sensors that could be used with the unprotected network. Additionally, sophisticated cyber-attacks progress through a number of stages, and each MTD, by design, protects against some of these stages, but not others. This makes it difficult to quantify the effectiveness of an MTD at stages that occur later than those it is designed to protect against because such stages are less likely to be observed in the wild (since the attack may have been thwarted at an earlier stage). In earlier work, we described methodology for MTD quantification using Siege's Cyber Quantification Framework [1].

MTD'16, October 24 2016, Vienna, Austria
© 2016 ACM. ISBN 978-1-4503-4570-5/16/10…$15.00
DOI: http://dx.doi.org/10.1145/2995272.2995282

In this paper, we present the results of a series of experiments performed using that methodology to assess two MTDs, ARCSYNE, and SDNA.

This paper is organized as follows. In Section 2 we provide an overview of the MTDs under examination and Siege's Cyber Quantification Framework, followed by our testing methodology and definitions of new MTD-specific metrics in Section 3. In Section 4 we describe the specific experiments that were performed according to this methodology, and analyze the results in Section 5. We conclude in Section 6 with discussion of lessons learned and potential future work.

2. BACKGROUND
2.1 Moving Target Defenses
Two MTDs were provided by the Air Force Research Laboratory (AFRL) for this effort, both based on dynamic network techniques. The first, ARCSYNE (Active Repositioning in Cyberspace for SYNchronized Evasion) is based on a network of protected nodes with synchronized IP-hopping. ARCSYNE protects a network through system-wide encryption and an ever-changing network topology. The second, SDNA (Self-shielding Dynamic Network Architecture), is based on cryptographically secure network dynamics on an IPv6 network [2, 3, 4].

2.2 The Cyber Quantification Framework
The process of planning for cyber operations today is more of an art than a science. Planners must rely more on instincts and experience than on experimental data and quantitative analysis. The adoption of cyber-assets is hindered because planners lack clear data to compare assets by complexity, cost, and benefit. There is a need for principled and repeatable experimentation and quantitative analysis of cyber-assets. To that end, we have been conducting research into the fundamental metrics for cyber asset and defense characterization, and constructing the Eprouvette Cyber Quantification Framework (CQF) to support configuration and provisioning of experiments, and data collection and analysis. At a high level, our experimentation process is composed of three steps: (i) define tests and execute experiments on virtualization-based testbeds; (ii) examine the collected raw data and assess and compute metrics values; and (iii) compare and contrast the metric values for different experimental subjects.

The CQF consists of three components roughly corresponding to three steps of our aforementioned experimentation process: the Experimentation Configuration Module; the Cyber Metrics Processor; and the Asset Assignment Engine.

The Experimentation Configuration Module (ECM) is a modular system for specifying and executing experiment configurations. The ECM provides low-level API and a high-level web interface for constructing experiments, varying control parameters, selecting data sensors, and so on, and for executing experiments in experimentation testbed. The current implementation supports VMware (ESXi and vCenter) based virtualization systems as testbeds. Experiment configurations may specify operating system variants, network topologies, software packages, and the ECM automates the construction of the required networks, and provisioning of the virtual machines, launching of tests, and data collection.

The Cyber Metrics Processor (CMP) processes the raw data collected during experimentation by the ECM, determines (based on what types of raw data were collected) what metrics can be computed, and computes metrics values for the entities used in the experiments. Like the ECM, the CMP supports a low-level API for programmatic access, but interaction is typically through a high-level web-interface. The CMP also provides users access to the raw data in tabular form and simple plotting and graphing.

The Asset Assignment Engine (AAE) provides functionality for more sophisticated analysis and comparison of experimental subjects, in terms of the metric values computed by the CMP. The AAE is designed to help planners by assessing and comparing the predicted performance of experimental subjects in specified operating environments. The AAE supports several types of visualizations, and different types of prediction strategies, including decision tree regression, which is particularly suitable for cyber-experimentation wherein many features are categorical or discrete in nature.

3. MTD TESTING METHODOLOGY & METRICS

As much as possible, this work is intended to apply the scientific method, characterized by posing questions and hypotheses, making predictions and performing experiments, and analyzing the results, to the evaluation of moving target defenses.. For this effort, we sought to answer high-level question: how effective are these MTDs, and how costly? Hypotheses included, e.g., that MTDs are effective at some points of the Cyber Kill Chain, but not against all it, or that an MTD might be more costly in network overhead, but less costly in local system resources. Testable predictions were based on the intent of the particular MTDs, such as that attacker reconnaissance will obtain a misleading view of the network structure, of that desktop applications will remain responsive despite increased network usage. Experiment design involved creating instrumented networks running mission and attack tasks with and without MTDs. The analysis process examined the raw data and computed metric values, and sought to create some mathematical models of some aspects of MTD performance, costs, and benefits.

A series of tests were conducted to assess the MTD technologies. For each MTD, network topology and task set, we ran the task set on the network topology with and without the MTD installed and collected measurements.

Each task set was designed such that differences in measurements with and without the MTDs for the task set characterized some higher-level measurement (e.g., cost, benefit) of the MTD. In particular, in this effort we used a "mission activity" task set and an "adversary activity" task set. The differences in measurements for the "adversary activity" task set characterized the defensive benefit of the MTD. The differences in measurements for the "mission activity" task set characterized the performance costs of the MTD on a mission. Additional attributes on tasks made it possible to analyze high-level attributes in greater detail (e.g., MTD performance costs for a specific type of mission task). By distinguishing these parts of an experiment, each experiment test run could be specified as a combination of a task set, a network topology, and an MTD configuration, as shown in Figure 1.

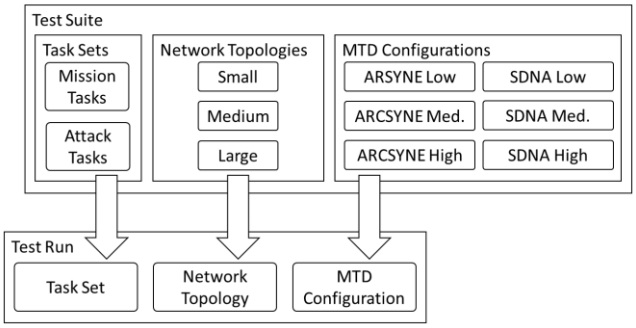

Figure 1: Experiment Configurations

3.1 Metrics

We defined eight MTD "measure of effectiveness" (MOE) metrics, one for each combination of task type (attack and mission) and attribute (success, integrity, productivity, and confidentiality). Mission-based metrics are based on the assumption that in operational networks, continual communication tasks contribute to a mission goal, and attack-based metrics on the assumption that periodic attacks can compromise confidentiality, integrity, and availability.

Evaluation of these metrics, shown in Table 1, is based on performing sets of tasks on networks with and without MTDs deployed. These metrics are broken down into two categories: mission metrics and attack metrics and an associated task attribute (described below). The mission metrics assess an MTD's impact on the underlying mission network and the ability to continue operations. The attack metrics assess an MTD's ability to hinder attackers.

Table 1: MTD Evaluation Metrics

Category	Type	Description	Attribute
Productivity	Mission	The rate at which mission tasks are completed	*duration*
	Attack	How quickly an attacker can perform and complete adversarial tasks	
Success	Mission	How often mission tasks are successful completed	*completed*
	Attack	How successful an attacker may be while attempting to attack a network	
Confidentiality	Mission	How much mission information is exposed to eavesdroppers, can be intercepted, etc.	*exposed*
	Attack	How much attacker activity may be visible to detection mechanisms	
Integrity	Mission	How much mission information is transmitted without modification or corruption	*intact*

Attack	The accuracy of information viewed by an attacker

While we did not directly instrument measurements such as increased attack surface created by the MTD (e.g., SDNA's intermediate nodes, ARCSYNE's rendezvous nodes), we expected that these would be reflected indirectly in metrics such "Attack Productivity" (since there will be more nodes against which an attack can be launched, and more nodes that could be used as pivots) and "Mission Confidentiality" (since Attacker Reconnaissance may be more successful with more nodes on the network).

The metrics defined in Table 1 are based on the assumption that within a given operational network, tasks are continually performed. These tasks may be mission-oriented or adversarial in nature (and each of these categories can be subdivided further). Each individual task has a number of attributes, some of which are observable. These include whether the task complete successfully, whether the tasks' "internal data" remains intact during its lifetime, the duration of the task, the amount of time allotted for the task before a timeout occurs, and whether the tasks's "internal data" is visible to observers.

Each of the attributes is associated with a category, as shown in Table 1, and each task has a value for each of these attributes. For instance, whether task *i* completed successfully is:

$$completed_i = attribute_i^{Completed}$$

During some period of observation of a network, there will be some number of attempted tasks, some of which can be categorized as mission tasks, and some of which can be categorized as adversarial tasks. We denote the number of each type of tasks as *numTasks* with a subscript. For instance the number of adversarial tasks in an experiment is:

$$numTasks_{Attack}$$

The metrics can now be defined as weighted averages of task attribute values over the different types of tasks. (In our testing, uniform weights were used. The intent behind incorporating weights into metric definitions is to support later inquiries of the type "which MTD is most effective at protecting the confidentiality of *one particular type of mission task*?") For each category *C* from Productivity, Success, etc., and task type *T* from Mission and Attack, the corresponding $metric_{T,C}$ is defined by the formula:

$$metric_{T,C} = \sum_{i=1}^{numTasks_T} \frac{w_i \times attribute_i^C}{numTasks_T}$$

where the indexed tasks are those of the appropriate type (that is, just the attack or mission tasks). For instance, *missionConfidentiality* is $metric_{Mission,Confidentiality}$, which is:

$$\sum_{i=1}^{numTasks_{Mission}} \frac{w_i \times confidentitality_i}{numTasks_{Mission}}$$

The questions of "how many tasks are there?" and "how are task attributes observed?" are experimental in nature. In a live setting, sensors would be deployed to both detect tasks and observe the attributes. In the experimental setting of the present effort, the number of tasks is fixed for an experiment, and custom sensors are deployed to observe their attributes. The experimental designs are described in the following section.

4. EXPERIMENTAL DESIGN

This section outlines the parameters used to define the different security levels evaluated for each MTD technology, the configurations used for testing, as well as the terminology (tags) used to describe the data within the experiment. A total of 44 tests were configured. The configurations include permutations on five variables:

1. Which MTD setup is in use (ARCSYNE or SDNA.)
2. Network scale (10 nodes, 20 nodes.)
3. Whether representative attack tasks or the APT are used
4. Whether the MTD is deployed, and if so, at which security level (low, medium, high). Security levels are interpreted differently for each MTD.
5. Whether the attacker is internal or external.

The configurations are listed in Table 2. A white triangle(△) indicates that MTD Effectiveness Metrics were collected; a black circle(•) indicates that only Successful Deployment, indicating the success of the APT, was collected.

Internal attacks were not performed with ARCSYNE, as ARCSYNE only attempts to protect the communications among a network of trusted nodes from external observers. ARCSYNE does not attempt to protect the protected nodes from each other and so does not, by design, protect against internal attacks.

Table 2: Test Configurations

		ARCSYNE				SDNA			
		Scale 10		Scale 20		Scale 10		Scale 20	
		Tasks	APT	Tasks	APT	Tasks	APT	Tasks	APT
No MTD		△	•	△	•	△	•	△	•
Low Security	Internal					△	•	△	•
	External	△	•	△	•	△	•	△	•
Medium Security	Internal					△	•	△	•
	External	△	•	△	•	△	•	△	•
High Security	Internal					△	•	△	•
	External	△	•	△	•	△	•	△	•

The tests measured the network security gains and associated performance implications for the MTD in various configurations but did not act as a "red-team" looking for specific vulnerabilities in the MTD systems.

4.1 Security Levels

ARCSYNE and SDNA do not operate in identical ways, and do not have all of the same kinds of configuration parameters. In order to approximate general security levels for the MTDs, we chose different hop delay (the time between IP address changes) for ARCSYNE, and different access policies for SDNA. These security levels are detailed in Table 3.

Table 3: MTD Security Levels

	ARCSYNE	SDNA
Low	Hop delay 10.0s	Whitelist: Intranet, FTP, mail, LDAP, HTTP, ping, DNS, SSH, MySQL, NTP, SNMP, WebSSL, RDP
Medium	Hop delay 1.0s	Whitelist: FTP, mail, HTTP, ping, DNS, SSH, MySQL, NTP
High	Hop delay 0.1s	Whitelist: FTP, Mail, Web, MySQL, ping

4.2 Network Topologies

The experimental parameters simply specify the network as a having a scale of 10 or 20, but the two MTDs structure networks differently, and require additional nodes on the network. The scale parameter refers to the number of end-user workstations in the network. The differences between the network structures mandated by the two MTDs are described in the following sections.

The SDNA network configuration, shown in Figure 2, consists of one main network, isolated networks for the ACC node, server node, each client node, and, in the Internal Attacker scenario, the attacker node. The Internal Attacker is a node attached to the network in the same manner as a Client and running the same SDNA software as a client. The External Attacker is a standard Fedora node connected directly to the main SDNA network.

The ARCSYNE network configuration, shown in Figure 3, is a simpler structure than that required by SDNA. (Note that Figure 3 includes provisions for both an internal and an external attacker, but the experiments on which we report here did not include the internal attacker.) There are two networks with each node connected to both. The external attacker is a Debian machine attached to the network in the same manner, but is not running ARCSYNE.

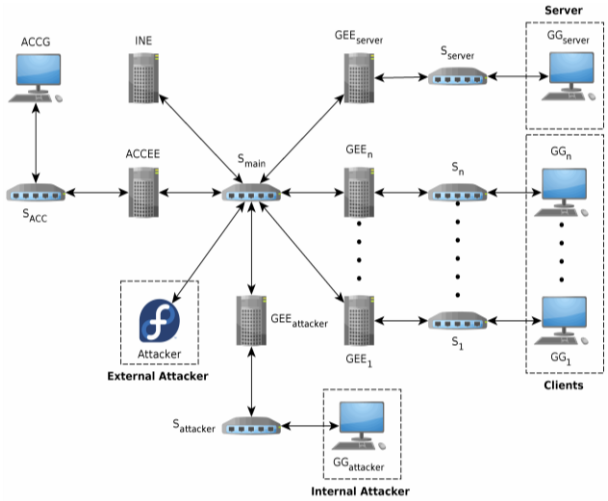

Figure 2: SDNA Attacker Network Diagram

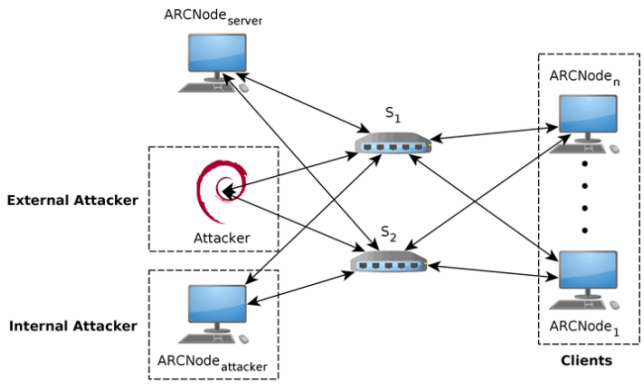

Figure 3: ARCSYNE Attacker Network Diagram

4.3 APT Model

In order to develop a realistic set of adversarial tasks, and in order to sub-categorize adversarial tasks in a manner to better characterize MTD effectiveness against them, we adopted a specific model of Advanced Persistent Threat (APT) behavior. Specifically, we adopted a threat model based on the Cyber Kill Chain™[1] introduced by Lockheed Martin [5]. The Cyber Kill Chain™ includes seven stages: (i) Reconnaissance, target identification; (ii) Weaponization, linking exploitation with deliverable payload(s); (iii) Delivery, transmission of a payload to target environment; (iv) Exploitation, execution of the payload to gain access; (v) Installation, of persistent code; (vi) Command and Control, remote communications; and (vii) Actions on Targets, data collection, exfiltration, propagation, and malicious operations.

MTD defenses are not designed to provide complete security against actions at every stage of the Cyber Kill chain. Instead, different MTD technologies provide varying degrees of protection against actions at each stage. For experimentation, we developed a suite of adversarial tasks representative of an advanced persistent threat through the Cyber Kill Chain (e.g.: nmap for network scanning during the reconnaissance and weaponization phase; Ncrack, SCP, and SSH for delivery and exploitation). By categorizing adversarial tasks according to kill chain stages, it is possible to characterize the defensive benefits of different MTDs against different stages of attacks.

5. RESULTS

In this section we report on some of the results from our experimentation. In the large, we found that both ARCSYNE and SDNA were effective at stopping the types of adversarial behaviors that *they were designed to mitigate*, but each carries its own unique overhead to the network and end-user system. The differences in types of configuration possible for each system makes it impossible to do a direct comparison, but there are some trends that apply to both MTDs, such as that higher security levels, unsurprisingly, can cause higher overhead or impact to missions. During our experimentation, we also discovered that some characteristics of MTDs, such as compatibility with certain software packages, or intended area of protection, should be made known to decision makers, but are not perfectly reflected in the metrics we have set forth here. It is important to recognize that we tested the MTDs against a typical set of attacks that were not intended to bypass or exploit the MTD systems. In a real world scenario it is anticipated that an attacker would evolve their tactics, techniques and procedures (TTP) in an attempt to minimize or circumvent the security gains provided by the MTD system.

5.1 Attack Metrics

Both ARCSYNE and SDNA were effective at stopping the kinds of attacks that they were designed to mitigate. In both cases, there are circumstances that can allow an attacker to make slightly more progress. ARCSYNE does not perform access control between the nodes within the protected network (and does not attempt to), so internal attacks on ARCSYNE were excluded from these experiments. Internal attacks against SDNA were also stopped, but were able to progress slightly farther when the attacker was located within the network. The approximate attack metric scores for the five configurations are shown in Table 4.

[1] Cyber Kill Chain is a registered trademark of Lockheed Martin.

Table 4: Attack Metrics for MTD Configurations

Configuration	Confidentiality	Success	Productivity	Integrity
No ARCSYNE	0.7	1.0	0.5	1.0
ARCSYNE	1.0	0.2	0.7	0.0
No SDNA	0.7	0.9	0.7	0.9
SDNA, External	1.0	0.3	0.7	0.0
SDNA, Internal	1.0	0.5	0.4	0.1

Note that the metrics must be interpreted in light of each other. For instance, while attacker confidentiality increases in the presence of both MTDs, this is because as the attack tasks fail, there is less adversarial information that can ever be exposed. It is also interesting to note that attack productivity in the external case for SDNA (and the only attacker case for ARCSYNE) actually increases. This is due to the fact that attack tasks that fail, end more quickly, which technically means that more tasks could be launched by an attacker during the experiment. The most important metrics that show that the MTDs are stopping the attack tasks are Attack Success and Attack Integrity. Decreasing Attack Success values mean that fewer attack tasks, even those that finished, are finishing successfully. Decreasing Attack Integrity values mean that reconnaissance tasks like an nmap scan are not returning accurate information to the attacker.

Security policies did have noticeable effects on the metric values for different types of attack tasks. For instance, since SDNA's lower security levels allow SSH traffic, the SSHPass attack task makes more progress at the lower security levels, allowing partial success at lower security levels, as shown in Table 5.

Table 5: SSHPass Attack Metrics for SDNA Security Levels

Configuration	Confidentiality	Success	Productivity	Integrity
No SDNA	1.0	1.0	1.0	1.0
SDNA, Low	1.0	0.4	0.6	0.4
SDNA, Med.	1.0	0.2	0.6	0.3
SDNA, High	1.0	0.0	0.5	0.0

5.2 Mission Metrics

Our hypotheses included that MTDs would be reasonably good at their intended tasks, stopping most or all of the attack tasks, and would impose minimal system overhead. For the most part, the MTDs do have a slight overhead that is reflected in the mission metrics (also see Section 5.3 for low-level performance overhead). Some investigation shows that the greatest impacts on mission metrics were due to unusual configurations of the MTDs. Table 6 shows the results of the mission metrics for the five configurations.

Table 6: Mission Metrics for MTD Configurations

Configuration	Confidentiality	Success	Productivity	Integrity
No ARCSYNE	0.2	1.0	1.0	0.7
ARCSYNE	1.0	0.9	0.9	0.6
No SDNA	0.5	1.0	1.0	1.0
SDNA, External	0.2	0.5	0.4	0.5
SDNA, Internal	1.0	0.5	0.4	0.5

For ARCSYNE, confidentiality increases, which is expected because ARCSYNE encrypts all traffic on the protected network. There are also slight decreases in other metrics, reflecting the fact that the MTD imposes some overhead. Further investigation showed that the decreases were almost entirely found in the low security level with the high hop delay (10s). Such high hop delay values would be uncommon in practice (recommended values are between 0.1 and 1.0 seconds), so the observed mission metrics for ARCSYNE at typical hop delay settings seem relatively stable. These latter results are shown in Table 7.

Table 7: ARCSYNE Mission Metrics vs. Hop Delay

Hop Delay	Confidentiality	Integrity	Productivity	Success
0.1s	1.0	0.75	1.0	1.0
1.0s	1.0	0.75	1.0	1.0
10.0s	1.0	0.45	0.8	0.75

At initial investigation, SDNA appears to impose a more drastic overhead. However, further investigation again reveals that the MTD generally has a better result, but that one particular factor pulls down the overall scores. It turns out that the connection mechanisms in FTP are not supported by SDNA's access policy implementation. When FTP tasks are removed from consideration, SDNA's impact on the mission metrics is much less pronounced.

A form of our initial hypothesis, then, is confirmed by these findings. MTDs can operate in a way that is effective against adversarial tasks, and that has little impact on mission tasks. However, operators and administrators who deploy these systems must be aware of their expected ranges of configuration values, and must know in advance which services an MTD is and is not compatible with.

5.3 Performance

The mission metrics provide some guidance for the high level impact that an MTD may impose on the mission tasks that a network is implemented to support. During these experiments, we also collected lower-level performance data, such as percent CPU utilization, to determine the effects of the MTDs on the individual hosts in the network. Each MTD does impose some overhead on individual end-user systems, and both have some unique characteristics.

ARCSYNE is based on IP-hopping, and each node periodically changes its IP address, which requires some amount of computation. Investigating ARCSYNE node's CPU usage under different circumstances indicates higher CPU usage with smaller hop delays (i.e., greater hop frequency), and with larger scale (probably since there are more nodes to synchronize with).

At a hop delay of 1.0s, the CPU utilization of ARCSYNE nodes remains relatively stable, at approximately 3%. When the hop delay is decrease to 0.1s (which is an increase in hop frequency), the average CPU utilization increases to approximately 30%.

Examination of approximate average CPU usage in all six combinations of scale and hop delay are shown in Table 8. Unlike ARCSYNE, which exhibited fairly consistent CPU utilization that changed with scale, SDNA has "spiky" CPU usage, and "normal" CPU usage does not seem to increase with scale.

Table 8: ARCSYNE CPU Usage vs Scale and Hop Delay

		Hop Delay		
		0.1	**1.0**	**10**
Scale	**10**	≈10%	≈3%	≈1%
	20	≈30%	≈5%	≈1%

6. CONCLUSIONS

The experimentation process led to a number of interesting observations. One is that some defenses may be incompatible with certain types of mission tasks, which planners and operators must take into consideration while selecting an MTD for deployment. Another is that an important aspect of an MTD's behavior is its response to errors; experimentation revealed that one MTD that usually provides encryption would transmit data in the clear if it failed to connect to the protected network.

The effectiveness evaluations reveal that both MTDs are relatively successful in deterring the types of attacks that they aim to stop, and that both MTDs stopped the APT. The data reveals that there is a tradeoff between the defenses provided by an MTD and its cost to mission performance and resource overhead.

Experimentation with complex systems and networks is a difficult task. Characterizing network-based MTDs is challenging, for a number of reasons, not the least of which is that because implementations vary in so many ways, there are many aspects of performance that are unique to each defensive system. It is difficult to design metrics in advance that will accurately and informatively capture the behavior of complex systems. When developing these metrics, it was clear that a distinction between "task completed", "task completed successfully", and "task completed successfully, yielding valuable information" was necessary. This is especially true, for agile defenses that may aim to deceive attackers, requiring attacks to appear successful, while not providing attackers with useful information.

This research can continue in a number of directions. It would be quite interesting to test with network-based MTDs to find similarities and differences, and to extend this methodology more broadly. One extension would be to branch out toward other types of non-network-based MTDs, such as rotating software defenses, in which server and client software is periodically rotated or even lower-level defenses such as memory space randomization. Support for automated combinatorial testing could reduce the amount of testing needed, while increasing the number of useful variable combinations. Anti-virus systems may contain dozens of vulnerabilities, many leading to remote code execution (RCE) and raises questions regarding the net security gain they provide [6] [7]. Future MTD quantification work should include an analysis of vulnerabilities in the MTD systems, and quantify how adding an MTD to the network affects the attack surface of the network.

7. ACKNOWLEDGMENTS

This work was funded by the Department of Defense Office of the Assistant Secretary of Defense for Research & Engineering Cyber Research Program and contracted through the Air Force Research Laboratory (Contract no. FA8750-14-C0229).

8. REFERENCES

[1] K. Zaffarano, J. Taylor and S. N. Hamilton, "A Quantitative Frameworkfor Moving Target Defense Effectiveness Evaluation," in *MTD 2015: Second ACM Workshop on Moving Target Defense, 22nd ACM SIGSAC Conference on Computer and Communications Security*, Denver, CO, 2015.

[2] J. Yackoski, H. Bullen, X. Yu and J. Li, "Applying Self-shielding Dynamic to the Network Architecture," in *Moving Target Defense II: Applications of Game Theory and Adversarial Modeling*, Springer, 2013, pp. 97-115.

[3] J. Yackoski, J. Li, S. A. DeLoach and X. Ou, "Mission-oriented Moving Target Defense based on Cryptographically Strong Network Dynamics," in *CSIIRW'13: Proceedings of the Eighth Annual Cyber Security and Information Intelligence Research Workshop*, 2013.

[4] J. Yackoski, P. Xie, H. Bullen, J. Li and K. Sun, "A Self-shielding Dynamic Network Architecture," in *Military Communications Conference: MILCOM 2011*, 2011.

[5] E. M. Hutchins, M. J. Cloppert and R. M. Amin, "Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains," *Leading Issues in Information Warfare & Security Research,* 2011.

[6] K. Askola, R. Puuperä, P. Pietikäinen, J. Eronen, M. Laakso, K. Halunen and J. Röning, "Vulnerability Dependencies in Antivirus Software," in *2008 Second International Conference on Emerging Security Information, Systems and Technologies*, Cap Esterel, France, 2008.

[7] P. Wagenseil, "What?! Antivirus Software Could Make PCs More Vulnerable," 29 February 2016. [Online]. Available: http://www.tomsguide.com/us/antivirus-flaws-bsidessf,news-22331.html. [Accessed 25 July 2016].