# A Quantitative Framework for Moving Target Defense Effectiveness Evaluation

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

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

Samuel Hamilton
Siege Technologies
108 Church Place
Falls Church VA USA
samuel.hamilton@siegetechnologies.com

## ABSTRACT

Static defense has proven to be a brittle mechanism for defending against cyber attack. Despite this, proactive defensive measures have not been widely deployed. This is because flexible proactive defensive measures such as Moving Target Defense (MTD) have as much potential to interfere with a network's ability to support the mission as they do to defend the network. In this paper we introduce an approach to defining and measuring MTD effects applied in a network environment to help guide MTD deployment decisions that successfully balance the potential security benefits of MTD deployment against the potential productivity costs.

## Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*; C.4 [**Performance of Systems**]: Modeling techniques, Measurement techniques, Performance attributes; D.4.8 [**Software Engineering**]: Metrics—*performance measures, process metrics*

## General Terms

Formal Models, Algorithms, Cost/performance, Security and protection

## Keywords

moving target defenses, cyber-security, virtualization, experimentation

## 1. INTRODUCTION

There is a wide range of potential mechanisms for utilizing MTD technologies to improve security, at both the host and network level. In this paper, we will be concentrating on metrics associated with network level defenses. A common approach is to utilize an intelligent modeling algorithm to selectively modify configurations based on circumstance. Carter, Riordan, & Okhravi have utilized game theory to identify prime opportunities to reconfigure the network [3]. Zhu, Hu, & Liu applied reinforcement learning to a training set to sculpt a moving target defense strategy [13]. Control theory is another popular approach [1].

Some approaches to moving target defense make no attempt to tune policy to circumstance, but instead deploy mechanisms to continuously change configurations while enabling valid users to reliably interact with the network, while leaving invalid users the challenge of penetrating the network despite constant reconfiguration [2]. IP hopping is the most common of these approaches, where IP addresses are constantly in motion [10, 11, 12].

In order to compare and evaluate the potential costs and benefits of these various approaches, it is important to be able to quantify the security benefits associated with each approach along with potential productivity costs that may be introduced (either through the overhead associated with deployment of the system, or potential interference such a system may introduce to legitimate network operations). There are several potential mechanisms for doing such a comparison, ranging from pure analytical approaches based on mathematical analysis, coarse grained simulation, data gathered from testbeds or cyber ranges of representational networks with real missions, and experimentation and instrumentation of real operational networks. Each of these approaches represents a different tradeoff between analysis cost and accuracy of results.

The focus of this paper is on metrics and analysis approaches that utilize cyber testbeds, as it is our position that this represents the most realistic data that can be gathered short of operational deployment experiments, which are often implausible without having first gathered strong evidence that such experiments are worthwhile and will cause no harm to ongoing operations.

The remainder of the paper is organized as follows: In Section 2, we present our overall approach, which shows the experimental workflow associated with automatically generating, configuring, and running experiments to gather the necessary data from virtualized environments. In Section 3, we briefly describe Siege's Cyber Quantification Framework (CQF), which we will use to configure and execute tests, as well as our methods for representing network topologies, experimental designs, and the formal representations for later analysis. We describe our metrics in Section 4, and provide mathematical definitions of each, as well as intuitive descriptions of the values they capture. The effort under which this work is funded is still in progress; we describe the intent for the rest of the effort in Section 4.5, and conclude in Section 5.



*MTD'15,*October 12 2015, Denver, CO, USA
© 2015 ACM. ISBN 978-1-4503-3823-3/15/10…$15.00
DOI: http://dx.doi.org/10.1145/2808475.2808476

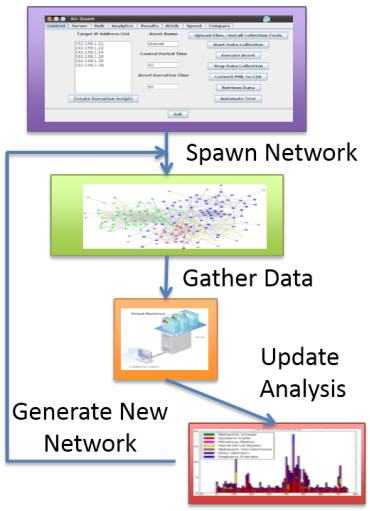


**Figure 1: Experimental Workflow**

## 2. OVERALL APPROACH

In order to measure the effectiveness of network oriented MTD technologies, we must develop two coordinated techniques: (i) mechanisms for gathering data on effectiveness; and (ii) metrics that process that data and extract effectiveness measurements. Security, productivity, and the appropriate tradeoff between the two cannot be statically evaluated in way that is equally applicable to all parties considering the potential deployment of an MTD technology. As such, our approach is to define multiple metrics that measure different areas of potential interest, and to persist and maintain the raw data from which metric results are derived. If new metrics are developed in the future that better represent the needs of an enterprise customer, the raw data can be used to calculate results for the new metric without rerunning experiments.

Figure 1 illustrates our proposed experimental workflow. Our initial launch panel will enable the configuration of an experiment, and allow for viewing and analysis of results while an experiment is in progress. Once an experiment is configured, it will automatically spawn a network of virtual machines configured with custom software to replicate mission oriented network activity, and a set of sensors to gather data. We will execute realistic missions with and without cyber-attacks, gather data, and then update our analysis of what pre-conditions make sense before launching the technology under test, and what post-launch conditions should be expected. As this analysis is being updated, a new topology is configured and an experimental instance is launched, and the process is repeated until the experiment stopping condition is reached (which may be in a time limit, or reaching certain pre-identified statistical significant measurements for certain metric measurements).

The traditional approach to cyber metric design is to try to quantify the effect the system under test has on three aspects of the mission data flowing through the system, Confidentiality, Integrity, and Availability of data [8]. Confidentiality refers to the ability to ensure data only gets exposed to those intended to have it, Integrity refers to ensuring that data is not modified inappropriately, and Availability refers to ensuring the data is delivered to those that legitimately request it.

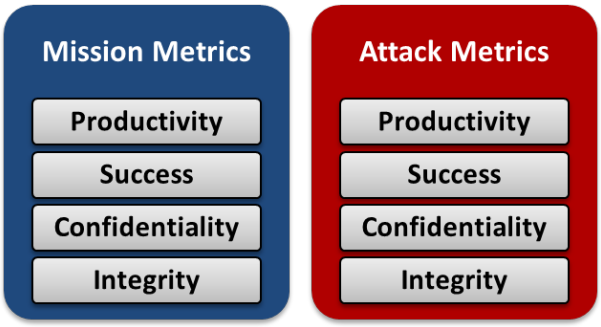


**Figure 2: Mission and Attack Metrics**

Our approach is inspired by this approach, but is modified to address information operation issues that the traditional approach does not directly address. For example, there are conditions under which an MTD might fail to stop an attack, but is still able to monitor and log much more fine grained detail on attack operations that allow for improved attribution or post-attack characterization of the attacker. Such benefits are important, but not well represented in traditional information assurance metrics. In our approach, we explicitly model a range of potential attacker and defender (or mission) objectives, and then run multiple experiments to collect data on the interaction between these objectives and the MTD. Metrics are derived from the statistical differences between these interactions during runs when an MTD is not deployed (the baseline) and when it is deployed. Figure 2 shows our four basic metric categories, applied to both the attacker missions, and defender missions.

## 3. EXPERIMENTAL FRAMEWORK

Siege utilizes an automated testbed for testing and measuring cyber effects as part of its workflow for designing cyber capabilities. In order to measure the effectiveness of MTD technologies, we have adapted and leveraged our Cyber Quantification Framework (CQF). Figure 3 outlines the MTD effectiveness characterization process implemented within the CQF.

Siege's Cyber Quantification Framework (CQF) has been used for experimental quantitative assessment of various types of cyber-assets, including both hardware and software. The CQF combines a user-friendly interface for designing large-scale experiments, sophisticated integration with virtualization servers (currently VMware vSphere, but the workflow can be ported to other virtualization systems, or adapted to physical cyber-ranges), and a workbench for data analysis and visualization.

Figure 4 shows the browser-based experiment configuration interface. The drag-and-drop UI presents categories of "items" on the left hand side that can be dragged into the central workspace. Top level items represent (virtual) hardware elements such as routers or switches, and virtual machine templates. Other non-top level items are dragged into these, and encapsulate additional test parameters, such as software that can be installed on virtual machines, data collectors that can be deployed to virtual machines during experimentation, and virtual networks that can be established on virtual routers.

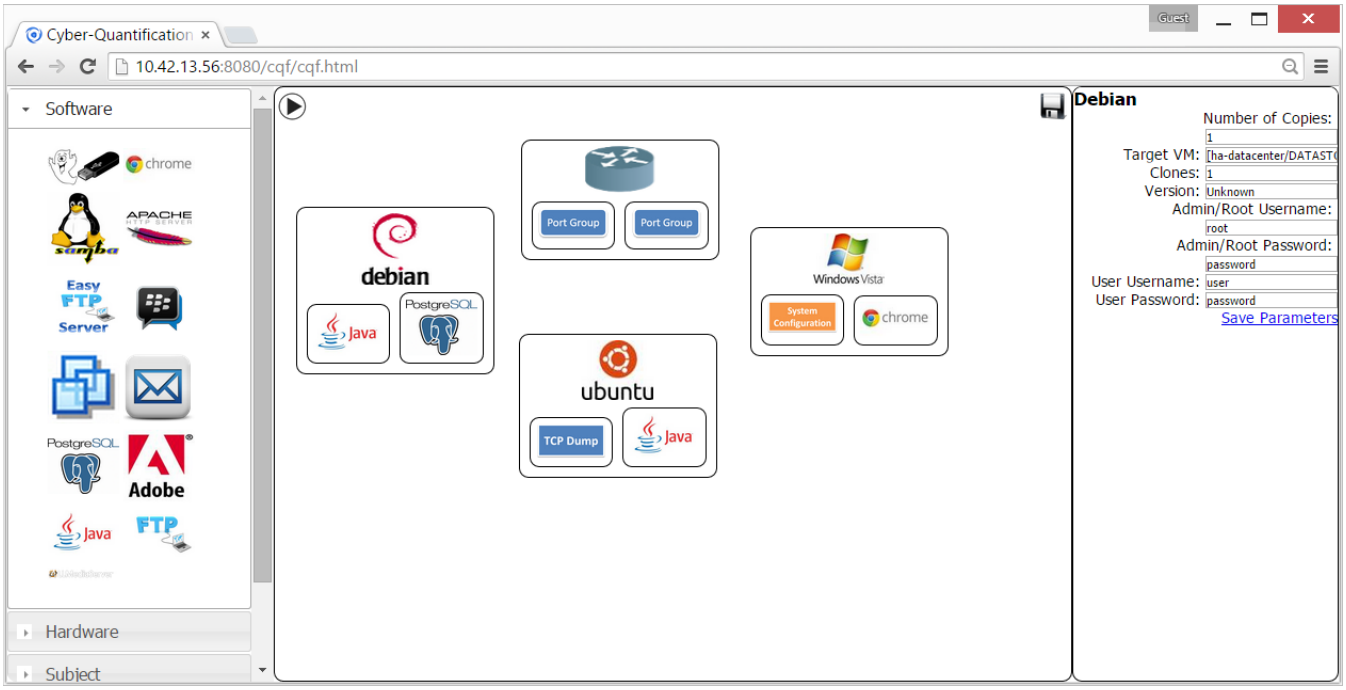


**Figure 4: The Experimental Configuration Interface**

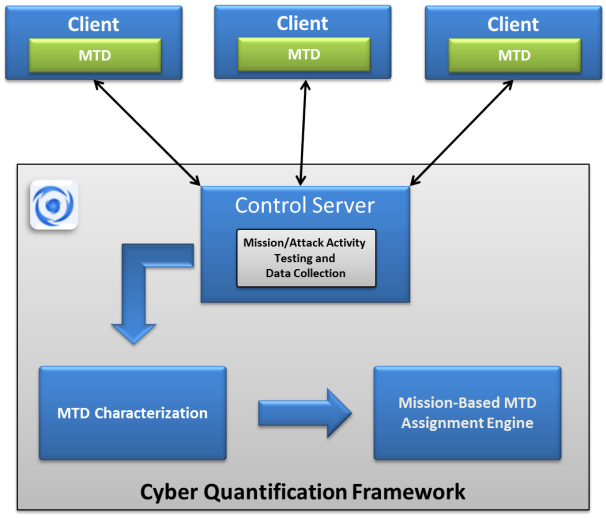


**Figure 3: The Cyber Quantification Framework**

Most of our past work on large scale quantification has focused on creating large numbers of virtual machines and performing individual experiments and data collection on each virtual machine; that is, the experiments were large scale in the sense that a large number of machines were created, configured, and run, all at once. However, this approach is not immediately applicable to the evaluation of network-based MTDs because these require entire networks in order to exhibit realistic and representative behavior.

Thus, experimental processes for network-based MTDs require automated network construction and configuration, as well as automated stimuli that operate on the network. Rather than using network traffic simulators, which are often based on mimicking the load and traffic behaviors of real networks, we are using the concept of activity models, wherein an activity model is a collection of tasks, each of which has a number of observable attributes, such as whether it successfully completes, how long it takes to complete, whether any transmitted data is corrupted, and whether any transmitted data is sent on the wire in plaintext. The activity model approach is not intended to produce traffic with the most realistic dynamics, but rather to be instrumentable, reproduceable, and immediately sensitive to the characteristics of MTDs under consideration, namely, their effectiveness and their cost.

In the following sections, we describe the process of automated network topology generation, the formal representation of activity sets, and the instantiation of activity sets that correspond to mission-oriented and attacker-oriented behavior.

**Table 1: Mission Oriented Network Behaviors**

| Server Type | Operation Simulation |
|---|---|
| Application Server | User & database middleware |
| Message server | Chat server |
| File Server | Connection through FTP |
| Mail Server | Email transmission |
| Database Server | SQL, data retrieval |
| Web Server | Browser usage |

## 3.1 Automated Topology Generation

To properly assess an MTD's applicability to a mission, a variety of mission relevant topologies should be assessed. In order to gather data on a large enough scale to fairly judge the technology, it is necessary to be able to automate the generation of different network topologies and conditions. Our approach is to leverage the existing vSphere and ESXi infrastructures and a browser-based web application that we have developed for test administration. The test administration application orchestrates the automated deployment of a range of varied network topologies containing a heterogeneous collection of hosts running versions of Linux and Microsoft Windows, along with associated routers and other network components necessary for realistic data generation and collection.

Each MTD technology under test has a set of operating systems and configuration options, which are instantiated during topology generation through the dynamic creation of linked clones and network infrastructures needed to automate the process of topology generation. The concept takes base installations of operating systems to create the compatible mission component nodes, as well as arbitrary nodes that may exist on a network, and applies the desired network structure. This generation also includes nodes that apply operation simulation such as the server types listed in Table 1.

To ensure metrics can be effectively calculated that account for scalability effects, we need to be able to generate topologies of different scales and complexities. We approach this by outlining network characteristics on a mission by mission basis. Thus, each mission template represents a range of different possible topologies that would support such a mission, along with associated network mission characteristics. As such, our approach differs from pure random topological generation approaches, such as the Waxman method where all the nodes are created and a probability of an added link is independently generated [9]. The disadvantage of this approach is that the network has an unnaturally even distribution of nodes, with less bottlenecks and other network elements that can affect performance behavior. Instead, we adapt a parameterized approach wherein a hierarchical structure is imposed to generate representative networks in line with observed power law distribution models, as well as other characteristics frequently seen in real networks but not in randomly generated ones [7]. Once the hierarchical structure is imposed we aim for a Heavy-Tailed (skewed) distribution of nodes. We do this by distributing the number of nodes on each subgraph in the hierarchy by picking a number of nodes in accordance to a bounded Pareto distribution, roughly following the techniques suggested by Crovella et al. [4]. The result is a series of realistic representative networks tailored by mission type. This allows us to not only analyze data across all mission types, but also specify effectiveness for a subset of network characteristics and mission types of interest to a potential user of the MTD under test.

## 3.2 Activity Models

We define an activity model as the combination of a set of *tasks* and a set of *task attributes*. Each task represents an individual instance of an activity. For instance, an activity model with three tasks might have the tasks: (i) user A sends an email to user B; (ii) user B sends an email to user A; and (iii) user C views a web page. Each task attribute identifies a property for which each task in the task set has a value and the range of values that the property can assume. For instance, the aforementioned activity model might have two task attributes: (i) task begin time, the value of which must be a timepoint; and (ii) task duration, the value of which must be a temporal duration.

Symbolically, we represent an activity model as a tuple $\langle T, A \rangle$ where $T = \{ \tau_1, \ldots, \tau_n \}$ is a set of $n$ tasks, and $A = \{ \alpha_1, \ldots, \alpha_m \}$ is a set of $m$ attributes. A *run* of a model is a process that produces a dataset, which is simply a mapping function $\nu \colon T \times A \to V$ which takes a task $\tau$ and an attribute $\alpha$ to a value from the permissible values for the attribute $\alpha$.

For instance, a run of the activity model described above could be the process where we:

- Ask a colleague A to send an email to another colleague B and record the time at which A begins composing the email and how much time elapses until hitting the send button.
- Ask a colleague B to send an email to A, after installing a sensor on B's workstation that records the time when the "Write Email" button is pressed and the time when the "Send" button is pressed.
- Visit a web page, and time how long it takes to read it. Later, retrieve the server logs to determine when the web page was requested.

After executing this run, attributes for each task can be determined. This run is somewhat artificial, but illustrates several important points:

- A run may produce attribute values indirectly. For instance, there may be multiple ways of determining when a user begins the process of emailing another user.
- A run need not determine attribute values in the same way for each task.
- A run may synthesize attribute values from data that are easier to collect. For instance, it may be easier to determine when a user begins to perform an activity by information collected by the system with which the user is interacting rather than from the user.

We now define two specific activity models whose tasks have the same set of attributes. The first is a mission activity model, whose activities correspond to legitimate network activities, such as sending email, and retrieving content from a database. The second is an attacker activity model, whose activities correspond to the types of actions an attacker would perform. Our two activity models use the same set of attributes (though the mechanisms for collecting attribute

values may differ for the different types of tasks). These attributes are:

- duration: length of time to complete the task execution, values are non-negative real numbers;
- success: whether the task was successfully completed, values are 0 (task did not complete successfully) and 1 (task completed successfully);
- unexposed: whether task information was exposed, values are 0 (information was exposed) and 1 (information was not exposed);
- intact: whether task information was corrupted, values are 0 (information was corrupted) and 1 (information was not corrupted).

Our attribute list was chosen to be representative enough of real network traffic, while providing concrete, quantifiable data to the metrics subsystem defined in Section 4.

### 3.2.1 Mission Representation

The mission tasks will apply to the mission oriented metrics; mission productivity, mission success, mission confidentiality, and mission integrity as shown in Figure 2. Each test network will have a variable number of clients and an assortment of activity servers configured for use within the enclave, e.g.: Mail Server; File Server; Database Server; and Web Server. This list was chosen to represent multiple communication mechanisms sufficient to identify potential operational issues introduced by an MTD technology under test.

A standard suite of mission task servers is based on monitoring communications between user workstations. MTDs may affect these types of communications in different ways. For instance, some chat protocols are based on establishing peer-to-peer connections, while others route all messages through a central server; some servers depend on privileged ports, while others use unreserved high port numbers), and missions may make use of different selections of these services.

Each client will perform mission tasks such as the following: sending and retrieving email, downloading files with FTP, querying a database with SQL, and retrieving web pages. Each task will be repeated at timed intervals. For instance, the client will send 60 emails, one every second.

### 3.2.2 Attacker Representation

To select characteristic types of attacker activities, we consider the Cyber Kill Chain®[1] illustrated in Figure 5 [5]. While no single attacker model can perfectly capture the workflow of every attacker, the Cyber Kill Chain has been proven to be a useful model for describing the high-level process that most serious attackers will follow. By basing the tasks in our attacker model on the stages of the Cyber Kill Chain, we believe that we obtain reasonable data indicating how effective MTDs are at preventing, deterring, or interrupting attacker behaviors.

It is important to note, however, that 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. Our attacker model will aim to perform activities from several stages of the Cyber Kill Chain, and not necessarily in order. This makes the attacker model less realistic, in some sense, but improves the data that it provides. For instance, if a given MTD is particularly successful at preventing reconnaissance and delivery, it may be very difficult to obtain real-world data about the MTD's effectiveness against later Cyber Kill Chain stages. By including tasks representative of various stages, we can make better assessments of the form "MTD X is good at stopping reconnaissance, but does little against Command and Control," which is more valuable than "MTD X is good at stopping reconnaissance, and we don't know how it handles Command and Control, because no attacker ever made it that far." As such, it will be important to include representative tasks from a variety of the kill chain stages, but also to recognize that an MTD's inability to prevent certain types of activities is not necessarily an indictment of its intended effectiveness.

Because the Cyber Kill Chain may be unfamiliar, we provide a brief summary of its seven stages:

**Reconnaissance** This includes information gathering and target identification. Cyber-reconnaissance can incorporate port scanning, traffic interception, and service probing. Reconnaissance provides an attacker with initial situational awareness of the target environment.

**Weaponization** Cyber-weapons must often be customized for specific targets. The same payload, for instance, might be packaged differently for different exploitable vulnerabilities. Weaponization is the process of selecting payloads that are compatible with exploitable vulnerabilities observed in the target network and packaging them appropriately.

**Delivery** The delivery process is responsible for the actual transmission of the weaponized payload to the targeted environment. Delivery mechanisms could include buffer overflows, social engineering, and direct or indirect access to target systems.

**Exploitation** Once a payload has been delivered to a target, the actual exploitation occurs when the payload is executed though some vulnerability. This may be through a software bug that allows code injection, or through coöpted legitimate means (e.g., tricking a user into executing a file).

**Installation** Simple one-off instances of malware may not be concerned with long term access to compromised systems, but advanced persistent threats will try to use a payload to establish some permanent and reliable access to compromised systems for later use.

**Command and Control** A successful installation procedure will result in functional command and control deploying to the compromised systems.

**Actions on Targets** With one or more compromised systems, an attacker can perform long-term actions on targets at will. These include, but are certainly not limited to data collection, information exfiltration, propagation, and malicious operations.

The Discovery stage information can be collected through the use of a network discovery attack vector such as nmap. Nmap will output the network visibility an attacker has

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

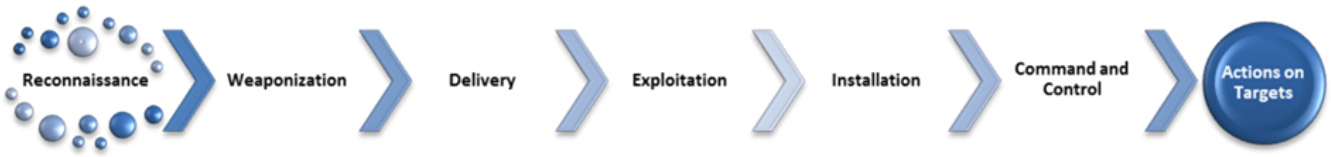


**Figure 5: Stages of the Cyber Kill Chain**

**Table 2: Attack (APT) Components**

| Stage | Potential Sensor | Data Collected |
|---|---|---|
| Discovery | nmap | assesses network visibility |
| Delivery | ncrack | dictionary attack success or failure |
| C2, Exfiltration | ncat | ability to successfully establish a backdoor |
| Action | read, write, execute | ability to run, modify, delete, create; illustrates the ability to perform desired actions (including privilege escalation) |
| Propagation | Discovery and Delivery combination | ability to maneuver within the network |

during the course of an attack. The difference between the network visibility of an attacker without an MTD running can be compared to the network visibility when the MTD is running. This comparison will effectively indicate whether an MTD is making it more difficult for an attacker to (accurately) view the network.

The same conceptual theory applies to the remaining stages. The delivery of a payload is the process of getting the attack on to the target system, whether it be through the use of exploitation or even user initiated (eg. phishing attack) methods. A representative method for remote exploitation is a system with weak credentials in which ncrack can be leveraged to compromise the target system. One would want to know if the use of an MTD will be able to stop this type of attack from occurring.

Attackers want data, making data exfiltration a big concern. A representative data exfiltration tool is ncat (similarly netcat) allowing the attacker to pivot within a network and relay data back to a reachable (Internet or outside connected) system. The ability to stop this attack avenue would be a very valuable feature of an MTD.

Sometimes attackers would like to apply D5 effects (Deceive, Deny, Disrupt, Degrade, Destroy) to a particular system or network. This capability can be measured with a very simple methodology; the ability to read, write, and execute on the target system. If the introduction of an MTD can reduce or eliminate an attacker's ability to affect a target system with these operations, we want to capture that information in our metrics.

The ability for an attacker to maneuver within a network means that the security of the infrastructure is only as secure as the weakest (least hardened interconnected) system. The introduction of an MTD technology can hinder or even mitigate that attack avenue.

**Table 3: Experimental Matrix**

| Activity Model | Without MTD | With MTD |
|---|---|---|
| Mission Model | mission baseline | cost to mission |
| Attacker Model | attacker baseline | effectiveness |

# 4. METRICS

Metrics process data gathered from multiple runs, where each run represents a combination of a mission, topology, adversary model, and MTD deployment. Some of the runs will have no adversary model and/or no MTD deployed. Runs with no MTD deployed represent a baseline run, which can be contrasted to effects measured during identically configured runs with a deployed MTD technology. This contrast drives our metrics.

The primary metric categories we measure are illustrated in Table 3, which includes both the mission and attack models with and without the MTD deployed. In all cases, we will collect values for all four of the attributes we defined in Section 3.2. By comparing the results for each attribute between the test with the MTD deployed and without the MTD deployed, we can assess the cost of the MTD to mission tasks and the effectiveness of the MTD against attacker tasks. Examining four attributes over two activity models gives eight individual metrics that can be partitioned into two sets of four, or four sets of two, shown earlier in Figure 2, and described in more detail in Table 4.

## 4.1 Productivity

Productivity is a measure of how quickly tasks in an activity model can be completed. Given an activity model $\mathcal{M} = \langle T, A \rangle$, where $A$ is the set of task attributes defined in Section 3.2 and a valuation $\nu$, we define the productivity of $\mathcal{M}$ is defined as the average of the duration attribute over the tasks in $\mathcal{M}$. That is,

$$\mathit{Productivity}(\mathcal{M}, \nu) = \frac{1}{|\,T\,|} \sum_{\tau \in T} \nu(\tau, \mathsf{duration})$$

When $\mathcal{M}$ is an instance of the mission model, we call its productivity *mission productivity*. Mission productivity is the rate at which mission tasks are completed. The difference between mission productivities of a valuation for a run with the MTD and a valuation without the MTD is the *cost* of deploying the MTD. Note that it may be possible for the cost to mission productivity of an MTD deployment to be negative in that it is possible that some MTDs decrease the amount of time required to complete mission tasks.

**Table 4: MTD Effectiveness Metrics**

| Metric | Description |
|---|---|
| Productivity | *Mission Productivity* can be measured by the rate at which mission tasks are completed |
| | *Attack Productivity* is a measure of how quickly an attacker can perform and complete adversarial tasks |
| Success | *Mission Success* can be measured by the number of attempted mission tasks that are successfully completed |
| | *Attack Success* is a measurement of how successful an attacker may be while attempting to attack a network |
| Confidentiality | *Mission Confidentiality* is a measure of how much mission information is exposed to eavesdroppers, whether information could be intercepted, etc. |
| | *Attack Confidentiality* is a measure of how much attacker activity may be visible by detection mechanisms |
| Integrity | *Mission Integrity* is a measure of how much mission information is transmitted without modification or corruption |
| | *Attack Integrity* is a measurement of the accuracy of the information viewed by an attacker |

Similarly, when $\mathcal{M}$ is an instance of the attacker model, we call its productivity *attacker productivity*. Attacker productivity is the rate at which attacker tasks are completed. The difference between attacker productivity for a run with the MTD and a run without the MTD is the effectiveness of the MTD with regard to attacker productivity, or the benefit of deploying the MTD.

We have defined productivity in terms of the duration attribute. While there may be other measures that could also be rightly called productivity, we expect that it is uncontroversial to assume that decreased duration is typically a good result for mission tasks, and that increased duration of attacker tasks is typically a good result from a defensive standpoint. However, we recognize that the arithmetic mean of duration may not be the single best indicator of task time: a single outlier could change the average task time significantly, even though the majority of task durations actually change in the other direction. These types of considerations have led us to make a clean distinction between the data that we collect (that is, the task attributes), and the metrics that we define based on this data. If a flaw should be discovered in a metric definition, or an incremental improvement is proposed, it may not be necessary to rerun tests, but rather only to compute new values from the data. This is an important benefit of our approach.

## 4.2 Success

Success is computed similarly to productivity, but using the success attribute rather than duration. The success attribute is Boolean valued, taking on just 0 and 1, but the average over a number of tasks makes mission success and attacker success real-valued numbers in the range $[0, 1]$. Formally, success is defined as:

$$\mathit{Success}(\mathcal{M}, \nu) = \frac{1}{|\,T\,|} \sum_{\tau \in T} \nu(\tau, \mathsf{success})$$

As with productivity, the difference between mission success with the MTD and without the MTD represents the cost (in terms of successful completion of tasks) of deploying the MTD. The difference between attacker success with the MTD and without the MTD represents a benefit of deploying the MTD, and the effectiveness of the MTD at thwarting attacker activities.

At this early stage in our MTD experimentation, we are focusing simply on the success or failure of all tasks in a mission model, but we expect that by assigning additional attributes to tasks, we can characterize the behavior of MTDs much more specifically. For instance, if a valuation also assigns a Cyber Kill Chain phase to each task, then we could identify the phase against which the MTD is most effective. For instance, let $\Phi = \{\, \mathsf{Reconnaissance}, \mathsf{Weaponization}, \ldots \,\}$ be the set of kill chain phases, phase be the task attribute whose value is an element of $\Phi$, and $T_\phi = \{\, \tau | \nu(\tau, \mathsf{phase}) = \phi \,\}$ the set of tasks from the model whose phase is $\phi$. Then the following is the kill chain phase against which the MTD appears to be most effective, when $\nu'$ is a valuation with the MTD and $\nu$ a valuation without it.

$$\arg\max_{\phi \in \Phi} \frac{1}{|\,T_\phi\,|} \sum_{\tau \in T_\phi} \nu'(\tau, \mathsf{success}) - \nu(\tau, \mathsf{success})$$

There are, of course, other measures that could be computed from the same attribute values. For instance, rather than looking at the absolute change in success values, it might be appropriate to look at the proportional change in success values.

## 4.3 Confidentiality

Confidentiality is a measure of how much information is exposed by activity model tasks. For the mission model, exposing information is typically undesirable, whereas an attacker being exposed is desirable. Confidentiality is computed similarly to the metrics above, with the same type of costs and benefits derived from them. For a mission model

$\mathcal{M}$, we have:

$$Confidentiality(\mathcal{M}, \nu) = \frac{1}{|T|} \sum_{\tau \in T} \nu(\tau, \mathsf{unexposed})$$

In principle, there are many ways in which information could be exposed (e.g., being stored in a database in such a way that a web application presents it to users), but in these initial experiments, we simply refer to whether information is visible in plaintext in network traffic.

We have an informal hypothesis that some MTDs that are beneficial in ensuring confidentiality of mission information may also help preserve the confidentiality of attacker information, at least if the attacker already has access to compromised hosts and can generate traffic on the network. Because we will test activities representative of different stages in the kill chain, we hope to be able to confirm or refute this hypothesis.

## 4.4 Integrity

Integrity is a measure of how much information produced by the activity model tasks is preserved (not corrupted). For the mission model, corrupting information is typically undesirable, though the damage it causes may vary, especially depending on the type information (e.g., digital versus analogue), whereas an attacker's transmissions being corrupted is beneficial and will hinder their attacks. Integrity is computed in the now familiar fashion; we have:

$$Integrity(\mathcal{M}, \nu) = \frac{1}{|T|} \sum_{\tau \in T} \nu(\tau, \mathsf{intact})$$

Note that some types of information are much more sensitive to information corruption than others. In encrypted data, even a single bit of corruption may render a transmission useless, but in an analogue audio transmission, static is unpleasant, but may cause no significant loss of functionality. Our activity models are based on digital information where the amount of corruption can be easily measured, but we stress that we are measuring how much an MTD may corrupt data, not how significant that corruption would be in practice.

## 4.5 Overall Metrics

In the previous sections we described a series of metrics designed to measure productivity, success, confidentiality, and integrity from both an attacker and a defender perspective. Each metric is designed to be calculated independently such that overall metrics which blend the potential costs and benefits associated with deploying an MTD can be easily tailored to the needs of an individual customer. Our default configuration provides a simple weighted average of each metric, where the network mission is positively weighted, and the attacker mission is negatively weighted.

In addition to designing each metric to be separable, we preserve all data collected in a database which is dynamically linked to our metrics. This allows us to leverage our metrics to answer questions developed after our experiments. For example, if you wanted to determine if the effectiveness of the MTD was dependent on network policy, you could rerun metrics on the data with different network policies and measure the effects.

# 5. CONCLUSION

In this paper we presented an overall approach to metric design for Moving Target Defense technology for network defense. Our approach utilizes a cyber testbed which can be rapidly configured and reconfigured to run a series of tests such that large quantities of data can be gathered and analyzed. We also presented details on our test design, cyber testbed, and Siege's Cyber Quantification Framework.

# 6. 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-C-0229).

# 7. REFERENCES

[1] M. D. Adams, S. D. Hitefield, B. Hoy, M. C. Fowler, and T. C. Clancy. Application of cybernetics and control theory for a new paradigm in cybersecurity. *CoRR*, abs/1311.0257, 2013.

[2] S. Antonatos, P. Akritidis, E. P. Markatos, and K. G. Anagnostakis. Defending against hitlist worms using network address space randomization. *Computer Networks*, 51(12):3471–3490, 2007.

[3] K. M. Carter, J. F. Riordan, and H. Okhravi. A game theoretic approach to strategy determination for dynamic platform defenses. In Jajodia and Sun [6], pages 21–30.

[4] M. E. Crovella, M. Harchol-Balter, and C. D. Murta. Task assignment in a distributed system (extended abstract): improving performance by unbalancing load. In *ACM SIGMETRICS Performance Evaluation Review*, volume 26, pages 268–269. ACM, 1998.

[5] E. M. Hutchins, M. J. Cloppert, and R. M. A. and. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. In J. Ryan, editor, *Leading Issues in Information Warfare and Security Research*, volume 1, pages 80–106. Academic Publishing International, Reading, United Kingdom, 2011.

[6] S. Jajodia and K. Sun, editors. *Proceedings of MTD'14: The First ACM Workshop on Moving Target Defense*, New York, 2014. ACM.

[7] A. Medina, I. Matta, and J. Byers. On the origin of power laws in internet topologies. *SIGCOMM Comput. Commun. Rev.*, 30(2):18–28, Apr. 2000.

[8] G. Stoneburner. Underlying technical models for information technology security. NIST Special Publication 800-33, National Institute of Standards and Technology, Dec. 2001.

[9] B. M. Waxman. Routing of multipoint connections. *Selected Areas in Communications, IEEE Journal on*, 6(9):1617–1622, 1988.

[10] J. Yackoski, H. Bullen, X. Yu, and J. Li. Applying self-shielding dynamic to the network architecture. In S. Jajodia, A. K. Ghosh, V. S. Subrahmanian, V. Swarup, C. Wang, and X. S. Wang, editors, *Moving Target Defense II: Applications of Game Theory and Adversarial Modeling*, volume 100 of *Advances in Information Security*, pages 97–115. Springer, 2013.

[11] 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.

[12] J. Yackoski, P. Xie, H. Bullen, J. Li, and K. Sun. A Self-shielding Dynamic Network Architecture. In *MILCOM'2011: Military Communications Conference 2011*, pages 1381–1386, 2011.

[13] M. Zhu, Z. Hu, and P. Liu. Reinforcement learning algorithms for adaptive cyber defense against Heartbleed. In Jajodia and Sun [6], pages 51–58.