SAND2016-11029C

Empirical Assessment of Network-based Moving Target Defense Approaches

Brian P. Van Leeuwen, William M.S. Stout, and Vincent E. Urias
Sandia National Laboratories
Albuquerque, New Mexico, USA
{bpvanle, wmstout, veuria}@sandia.gov

Abstract— Moving Target Defense (MTD) is based on the notion of controlling change across various system attributes with the objective of increasing uncertainty and complexity for attackers; the promise of MTD is that this increased uncertainty and complexity will increase the costs of attack efforts and thus prevent or limit network intrusions. As MTD increases complexity of the system for the attacker, the MTD also increases complexity and cost in the desired operation of the system. This introduced complexity may result in more difficult network troubleshooting and cause network degradation or longer network outages, and may not provide an adequate defense against an adversary in the end.

In this work, the authors continue MTD assessment and evaluation, this time focusing on application performance monitoring (APM) under the umbrella of Defensive Work Factors, as well as the empirical assessment of a network-based MTD under Red Team (RT) attack. APM provides the impact of the MTD from the perspective of the user, whilst the RT element provides a means to test the defense under a series of attack steps based on the LM Cyber Kill Chain.

Keywords—moving target defense; metrics; defensive work factors

I. INTRODUCTION

Since the inception of computing and computer networking, there has existed the threat of exploitation, attack, degradation, breach and compromise by malicious parties. Much research has been devoted to finding ways to secure against the ever advancing threat to various information systems. Since information system budgets are limited, system operators must decide which security solutions provide the desired security cost most effectively. A minimum of two factors is necessary to assess the value of a security solution for a networked information system. The two factors are:

1. Effectiveness at thwarting specific cyber-attacks, and
2. Total cost of deploying the security solution.

Clearly the two factors noted above will have additional detail associated with deriving answers to each of the questions. Ideally it would be nice if mechanisms were available to produce quantitative answers to each of the questions so a security solution can be selected and the system operator will know system security posture and the costs to obtain and maintain that posture. Unfortunately, in most cases, no such mechanism exists to produce a reliable quantitative answer to the effectiveness of the security solution to thwarting cyber-attacks. However, the second factor noted above, referencing cost of deploying the security solution is much more reliable in deriving a quantitative solution. The authors of this paper produced a previous paper that began the definition and means to derive an answer to the second factor [1]. Previous work focused on costs associated with impacts to network performance. Additionally, previous work delved into costs associated with deployment, operation, and impact to in-place security. The research described in this paper expands on the capability to measure costs of an MTD deployment further by measuring platform-specific impacts of MTD.

Typically any type of security method or apparatus deployed on a networked information system comes at a cost. This cost can be a financial cost occurring during deployment and operation of the system or the cost may be associated with reducing system performance. A system performance cost describes the additional performance overhead necessary to achieve system objective while deploying security techniques and security systems.

A. Defensive Work Factors through APM

Moving Target Defenses can be host-based and/or network-based. Host-based MTD techniques are those that vary some aspect of the computing platform itself. The technique may vary aspects of OS, application code, or properties of the platform [2].

Network-based MTD techniques class includes defensive techniques dynamically varying network aspects of a distributed information system. The various aspects provide network connectivity and enable system transactions across multiple computing platforms and are candidates for MTD techniques. Example techniques included in this class are dynamic IP address and/or port randomization [3], routing path randomization [4], and proxy-location randomization [5]. The fundamental idea for these techniques is periodically changing the structure of the network an adversary must use to access resources or data in the protected information system.

Ultimately our objective is to understand how any security approaches, such as MTD, impacts the user of the system. The instrumentation used should provide a lightweight end-user experience (EUE) monitoring of all transactions. Detailed tracing of transactions through the application platform and various application tiers supported on multiple platforms or virtual machines. The tracing will reveal the location of performance impacts potentially introduced by the MTD. The impacts, which manifest as end-user experience issues, may result from any component of distributed applications, database access (e.g., SQL) or remote web

service calls. The instrumentation must accomplish the monitoring in a lightweight manner and not impact the running application, server, or database.

In our analysis, we leverage instrumentation used in application performance management (APM) tools. The APM tool can monitor operating system and infrastructure. A large number of specific metrics are collected periodically from the various components supporting the application. The tool's measuring time granularity can capture impacts from MTDs with very fast movement periods. A typical APM tool will generate transaction maps describing transaction paths and exposing transactions impacted by an MTD deployed in the architecture. The computing architecture can be complex and isolating the impact to a specific component in the architecture is difficult. Mapping transactions through system architectures is helpful in characterizing performance costs of MTD. Transaction maps are created for every transaction and accurately reflect changes in, for example, configurations, load balancing of application components, and remote service dependencies.

B. Red Team Assessment Against MTD Approaches

The second key factor noted in Section I describes the effectiveness at thwarting specific cyber-attacks. In numerous research papers, authors have identified and describe methods to define the level of protection provided by various MTD solutions [6, 7, 8]. Some methods include probabilities derived by a subject matter expert knowledgeable of the system and cyber-attacks. However, it remains a challenge how the probability values can be used to provide an apples-to-apples comparison when determining which security solution provides necessary security at the most desirable cost.

In evaluating effectiveness of an MTD security solution our research team focuses on a red team approach. A red team approach is an all-out attempt to gain access to a system by any means necessary, and usually includes cyber penetration testing. In our research the red team approach is limited to a cyber red team analysis in that the focus is to assess the effectiveness of an MTD approach to thwarting the attack. Non cyber-attack paths will not contribute to the description of the capability of MTD to thwart attacks. Furthermore, in our research the focus is to perform a red team analysis on an actual system running actual software, versus a more limited red team paper study. However the actual system is instantiated using extensive virtualization (described in a following section).

A red team analysis can be described as a real life exercise carried out by a highly-skilled small team of trained professionals that are tasked with evaluating the effectiveness of the cyber security by accessing the actual system or some replica of the actual system. In general, since all it takes is the weakest link for a cyber security breach to occur, a red team will assess a broad range of cyber features to gain access and thus identify if and where the breaking point exists. The red team approach intends to duplicate the same process that a motivated attacker would follow to map out an organization's infrastructure, perform reconnaissance at key physical installations, and then test the cyber defenses that are in place.

In our red team approach to assessing the effectiveness of MTDs to thwart specific cyber-security attacks the approach is to perform red team analysis on a realistic networked information system without an MTD and then repeat the red team assessment on the same system with the deployed MTD. In cases of evaluating numerous security approach mechanisms, a method to automate the cyber-attacks is necessary for consistency in the deployment of the attacks. An additional benefit of automating the deployment is the reducing the amount of time required from a highly-skilled team of trained professionals. In most cases this additional benefit can be key in the number of systems and depth of analysis available for the purpose of evaluating security solutions.

As noted above, an attacker typically follows a systematic approach to attacking a system. An attacker of a system typically follows seven steps to accomplish their mission and thus a red team will also attempt to follow the same seven steps as described by the Lockheed Martin Cyber Kill Chain® Methodology [9]. The seven steps described by the Kill Chain Methodology are:

- Reconnaissance,
- Weaponization,
- Delivery,
- Exploitation,
- Installation,
- Command and Control (C2), and
- Actions on Objectives.

In the description of the Kill Chain the authors describe the capability to disrupt attack on a networked information system by preventing an attacker from progressing through all seven steps. Thus the Kill Chain describes areas of opportunity to protect the system of interest and enterprise.

In the case of evaluating the effectiveness of an MTD to thwart attack, the Kill Chain Methodology is applicable since MTD as a security approach is very different than traditional cyber security approaches. MTD obtain their resistance to attack by a periodic change of the attack surface and thus obtain their effectiveness during and following the reconnaissance step. An attacker's objective is, during the initial intrusion into a system, to perform reconnaissance of the system for the purpose of identifying the possible next attack steps that lead to the objective of the attacker. Applying the Kill Chain steps to MTD, the attacker is disrupted at the reconnaissance step since the attacker's knowledge of the system must be continuously reset. This forces the attacker to either substantially increase the speed of the attack or continuously perform reconnaissance both of which increase the "noisiness" of the attack and thus increase possibility of detection by other security mechanisms.

The remainder of the paper discusses the considerations for developing a testing infrastructure for our two evaluation goals (Section II), implementing support for an MTD in such an infrastructure (Section III), the experiment itself and results (Section IV), challenges and further considerations (Section V) and finally the conclusions (Section VI).

II. Description of the Test Environment

A. Emulation Platform

Cyber modeling and simulation platforms are beginning to reach a maturity level where they can accurately represent entire enterprises, and in some cases, small countries. Multiple research areas are converging to create more faithful representations of candidate networks by more accurately representing routing configurations, network artifacts, and application activity. While the activity on emulated endpoints and the representative networks is becoming increasingly realistic and dynamic, the machines and physical/virtual topologies are inherently static. Sandia's approach to network simulation is based on the methodology of Emulytics™ [10, 11]. Through Emulytics, we may increase the realism and volatility of testbeds by providing the following features:

- Simulate network disconnects,
- Move endpoints between segments,
- Alter core routing,
- Dynamically instantiate/destroy network segments,
- Dynamically extend network (endpoint becomes NAT and shares connection),
- Attach and detach removable storage devices,
- Dynamically map devices (e.g., USB) to endpoints,
- Record physical topology activity and replay actions.

Some of these features (such as network disconnect and moving network segments) exist in some form in modern cloud architectures. OpenStack, VMware vSphere, and Deter are capable of starting and stopping virtual machines. However, they lack the integration required to reconfigure running routers or endpoints to accommodate the network changes. One could conceivably use the OpenStack API to destroy a network segment and rewire the endpoints to another, however the platform does not have the ability to reach into the endpoints and reconfigure them for the new topology, nor update routing tables to accommodate new links in the network, which is key to allowing these events to be scripted and/or replayed. To meet the above goals, we leverage the Sandia-developed minimega platform [12].

The minimega emulation platform uses the open source Open vSwitch [13] software switch as the networking component for experiments. Open vSwitch can trunk network data to physical networks, allowing multiple Open vSwitch instances on multiple nodes to federate into one or more large, isolated, experiment networks. Additional capability in the minimega tool suite allows human interactions with endpoints via VNC, a platform agnostic remote desktop protocol. Minimega provides this capability via a VNC gateway that allows the user to browse through and interact with any VMs on a running experiment. To provide the capability to record and playback user interactions with VMs, minimega intercepts mouse and keyboard traffic generated by the user, and stores it, along with timing information, to file. These events can later be played back as if the user were again interacting with the VM. VNC recordings can be played back to any VM, not just the one on which the recording was made. In order to support playback, a partial Remote Frame Buffer (RFB) protocol emulator is implemented in minimega, such that minimega can act like a VNC client to any endpoints. This implementation scales well, and can handle at least dozens of live user interactions being recorded and later played back.

Through the flexibility of the minimega platform we are able to generate the enterprise networks for this test repeatably, so as to minimize variation of environmental variables. Leveraging the VNC-replay capability we are able to script actions (e.g., attacks) within the environment, against target machines, to identify pass/fail points in kill chains. Finally, with the use of Open vSwitch, we are able to simply *plug-in* the software defined network (SDN)-based MTD under study in this research (as described in Section III below).

B. Enterprise Emulation

Our objective is to determine both defender work factor costs and ability to thwart attack on network information systems models that reflect levels of realism necessary to provide sufficient guidance to decision makers. In our assessment of MTDs we target application in enterprise systems. Thus in measuring end-user experience we attempt to construct system models that incorporate all of the components that impact or are impacted by an MTD deployment. Examples of the various system components that will be impacted by MTD (as deployed in our test network) include the following:

1. Microsoft Windows Domain

Windows domain is a logical computer network in which user accounts, computers, and printers are registered with a central database known as a domain controller. A primary function of a domain controller, which Microsoft names Active Directory Domain Services (AD,DS), is user authentication.

2. Example System Application

An example client-server application is deployed on the experiment to measure the performance impacts of the MTD to the user. In our experiment, an application that is distributed over multiple servers and remote data base is used. The selected application performs securities trading (i.e., stocks, bonds) and has components distributed across an enterprise system. The application is built using the .NET software framework and provides an excellent surrogate for many applications deployed in today's information systems.

3. Example Enterprise Local Area Network (LAN)

Our experimental deployment is built on a Layer-3 notional enterprise network. The enterprise network includes core, distribution, and access layers to represent the enterprise routed architecture. The route architecture represents the various regions in an enterprise that host the various distributed functions associated with a securities and exchange entity. In our example we deploy a network based MTD on the enterprise system using SDN implementations. Additionally, the experiment network model includes connectivity to a modeled Internet. The connectivity is through a DMZ. The experiment deployment model topology is shown in Figure 1.

A model of the network system under test is shown in Figure 1. The model follows the access, distribution and core network infrastructure model. Three subnetworks (VLAN) were designed to contain endpoints/workstations and services.

A DMZ is included to provide passage from the internal enterprise assets to external entities, where RT entities would reside.

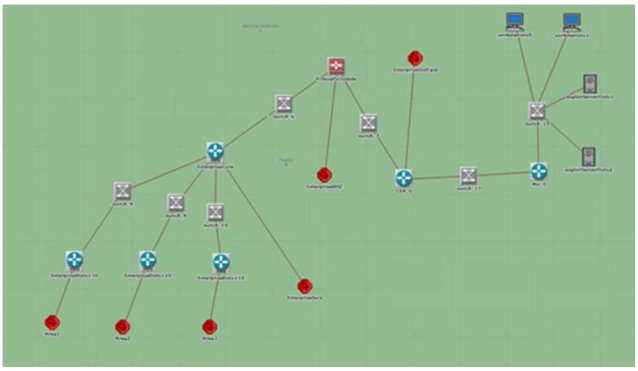

Figure 1. Experiment Deployment Topology
Note: Each red icon represents a subnet with multiple end-points.

III. Considerations for IP-Randomization in an Emulated Enterprise

In the previous work [1], a network performance monitoring (NPM) assessment was conducted on a network running an IP-randomization in a moving target defense context. The network upon which the defense the deployed was a comparable to a SCADA network, wherein low bandwidth links connected endpoints, such as remote terminal units (RTUs), with collection point(s) in a star-like topology. Measurements were taken between linked endpoints to capture Layer 3-4 metrics. For the current effort, an enterprise network was selected to deploy the IP-Randomization MTD. This topology was chosen to accommodate the execution of cyber-attacks on network and endpoints in accordance with the LM Cyber Kill Chain methodology. Several considerations and changes were made to the IP-Randomization mechanism and emulation environment to support this evaluation.

A. Open vSwitch Connections to VMs

To support Openflow compatible switching in the former paper, Open vSwitch processes were started on the endpoint virtual machines. These Open vSwitches were then connected to the SDN controller via an out-of-band network. For the enterprise deployment, access layer or "last-mile" switches (again, Open vSwitches) were placed before the endpoint virtual machines. These last-mile switches were then connected to the SDN controller on the virtualization platform environment's management plane.

B. Open vSwitch Deployment on Hosts

Each last-mile switch was essentially treated as a "building" switch, identified by a virtual LAN (VLAN) identifier. Each switch was implemented as an Open vSwitch bridge and deployed to a *specific* physical host. All virtual machines within that particular VLAN were then deployed to that physical host.

C. Interconnectivity of Logical Topology

Each VLAN switch, or Open vSwitch "building" bridge, was a connected to a virtual router deployed in the experiment network. This router was deemed as part of the distribution layer of the hierarchical topology. Connections from the distributed layer elements were connected to an Open vSwitch bridge named "experiment." The experiment bridge connected virtual machine taps from the distribution layer elements as well as the core router elements. Also included on the experiment bridge was a physical interface on the physical host (accordingly connected to the experiment backplane switch) to provide connectivity between the hosts - and subsequently the logical topology core and distribution routing devices.

D. Permitted Communication Pairs

In the previous deployment, two approaches to IP-Randomization were utilized. The first of which was based on reactive flow installation. For this implementation, as endpoints required communication, first packets of flows were sent to the controller to do IP/port validation. Following validation, flow rules were installed on source and destination Open vSwitches to do matching and IP address swaps/translations. The second implementation was based on proactive flow installation. For this approach, flow rules were installed on source/destination Open vSwitches to do matching and IP-address swaps/translations. While the former method worked for the testing of network Layer 3-4 attributes on a small scale, in a true network the amount of traffic passed to a controller reactively may bog the controller process as well as fill switch buffers. The latter method of proactively installing flows would provide communication and IP swap/translations when communications were required, without the need to send flow-rule misses to the controller to handle.

A naive approach for testing used an enumeration of all communication pairs in the network (bidirectional), for which flows would then be generated to do IP-address swap/translation. Thus, for a network of n endpoints, the number of flows required would be $2n^2$. Again, for a small network, the space requirement is not infeasible; however, the approach does not scale. So, for the testing of the enterprise deployment, "permitted communication pairs" were generated to proactively install flow-rules for. These pairs would be from general endpoints, such as workstations, to servers in the network, e.g., north-south communications. East-west communications, e.g., between workstations, would be denied. For the timeout period between flow-rule removals, the random IPs would be regenerated and redeployed to last-mile switches, with an interval of overlap between old flows and new flows (to account for transmission delays in the network). Because of this regeneration process, the ability to change the mapping of permitted communication pairs would be possible. For example, temporarily permitting an east-west communication or adding/removing devices from the network.

E. Collection of Flow Log Data

With a reactive flow-rule installation approach - all rule-misses would be sent to the controller to deal with. Such an approach provides immediate visibility of the communication requests from and between endpoints. With the implementation of proactive flow-rule installation, this particular aspect may be lost (since flow rules are installed before communications are initiated). Although flow counters may be queried to determine how many packets were transmitted for a particular communication session, visibility into "other" communication requests was needed (e.g., red team actions). To meet this, rule misses were forwarded to the controller for logging. This would provide insight into unpermitted communication request from endpoints (permitted on the network or not).

IV. Experiment Setup and Analsys

To assess the MTD's impact on the end user through APM tools, and the effectiveness of the MTD against an adversarial attack, the modeled network described in Section II – Part 3 was deployed in the minimega emulation platform. A multitier, distributed file-service was deployed against multiple servers to acquire APM metrics during transactions. For the RT testing, an attack chain crafted through Metasploit (Kali) was executed on the network. Both of these actions were carried out on the network without MTD to provide a baseline ground truth, and with the MTD running to assess its impact and effectiveness. Further details and experiment results follow below.

A. Results 1: APM Assessment

The experimental system was instrumented extensively using APM tools. A key aspect provided by APM is the decomposition of application components with its capability to automatically generate transaction and server maps. The transaction and server maps expose issues related to specific paths transactions and enable measuring impacts of MTD through each transaction. An example transaction and server map is shown in Figure 2.

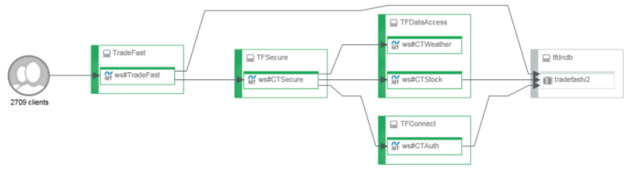

Figure 2. Transaction and Server Map

Using APM, the individual transactions executing on specific servers can be traced. This provides metrics such as server response time for specific functions. With host based MTD deployments, the resulting impact of the MTD on specific transactions can be identified. Network delays between the application components or supporting services are measured and thus can be attributed to network-based MTD. Additionally, APM monitors key operating system resources metrics, such as CPU, memory, and networking, on all components of the system being monitored. Measured network delays resulting from the experimental system shown in Figure 2 are shown in Figure 3.

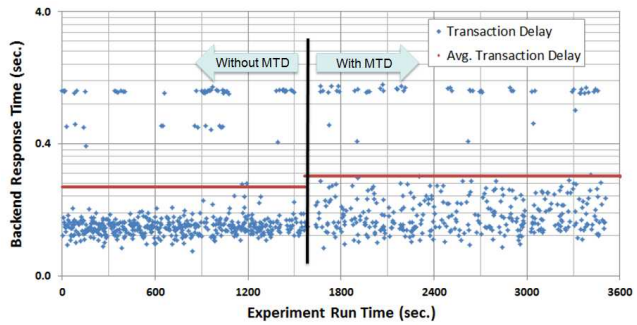

Figure 3. APM Results of Transaction Delays

B. Results 2: Red Team Assessment

As noted in Section I – Part B, cyber-attacks against the system under test would follow the Lockheed Martin Cyber Kill Chain methodology. The Kill Chain methodology is broken down into 7 steps, specifically [9]:

- Reconnaissance: Harvesting email addresses, conference information, etc.
- Weaponization: Coupling exploit with backdoor into deliverable payload.
- Delivery: Delivering weaponization bundle to the victim via email, web, USB, etc.
- Exploitation: Exploiting a vulnerability to execute code on victim's system.
- Installation: Install malware on the asset.
- Command and Control (C2): Command channel for remote manipulation of victim.
- Actions on Objectives: With 'Hands on Keyboard' access, intruders accomplish their goal.

To qualify each step in the Kill Chain, a matrix was created to describe four states as shown in Table 1. A binary set of states {Thwarted | Completed} account for an attack action that was either averted or successful. Ambiguity within either of the binary states is captured in either {Partially Thwarted | Partially Completed}. For the former, an attack action might be thwarted if additional information or actions taken by the defender were taken using previously acquired knowledge from the defense system. Partially Completed describes a success for the attacker if additional information were obtained by the attack about the defense system or if the defense system (MTD) does not address the attack scenario specifically.

Leveraging the VNC-replay capability described in Section II, a multistage attack was scripted to execute within the seven Kill Chain steps. This attack was played out on the system under test without the MTD running, and with the MTD running. The results of these attacks are displayed in Table 2.

Table 1. Kill Chain Evaluation Matrix

Thwarted Action blocked.	**Partially Thwarted** Action thwarted IF additional information provided.
Partially Completed Action completed IF security posture changed/relaxed.	**Completed** Action achieved.

Table 2. Attack Results

	Recon	***Weapon***	***Delivery***
Attack method	Scan/sniff network	Craft exploit	Drive-by download
Without MTD	Completed	N/A	Completed
With MTD	Thwarted	N/A	Partially Completed

Table 2. Attack Results, Cont.

	Exploit	***Install***	***C2***	***Actions***
Attack method	Execute on guest	Worm	Remote shell	Exfiltration
Without MTD	Completed	Completed	Completed	Completed
With MTD	Completed	Partially Thwarted	Partially Thwarted	Partially Thwarted

As the results show, without the MTD running, the scripted attack was able to complete all seven steps of the LM Cyber Kill Chain. With the MTD running, IP scans were thwarted, as the endpoints would not response to ICMP or TCP SYN packets from unknown or unauthorized entities. As for Delivery, since the attack was accomplished through drive-by download (via phishing); theoretically the attack could ensue. However, if the IP for the external server were not part of the allowable communication pairs, this step would have been stopped (this would not work with a proxy web-server). For the Exploit step, the MTD would not provide protections on the endpoint, hence this step would go off unhampered. For the Install step, the MTD would not be able to prevent the installation on the guest; however, the worm would not propagate through the network, as unauthorized connections would be discovered and the guest would then be removed from the overlay network. The same principle applies to the C2 and Action steps. Communications to the external (or internal) C2 server would be identified, and subsequently shutdown. The same consequence translates to the exfiltration Action step.

V. Challenges and Further Considerations

A. APM Assessment

As with any assessment executed on virtualized/emulated devices, the question arises as to whether the system adequately reflects a real system. For our Emulytic approach, we attempted to craft a multitier network with virtual devices that resemble physical enterprise networks, employing an access, distribution, and core model. To truly implement physical device latencies would be an undertaking unto itself; we feel the primary aims of this research were achieved through the comparison of the treated and untreated networks through "objective" end-user quality assessment. More rigorous examples of metric collection may be found in the authors' previous work.

B. Red Team Assessment

The MTD under test in this research was a network-based MTD. Hence, it was unreasonable to expect it to thwart attacks on host, specifically exploits and malware executions. Furthermore, the types of attacks selected were based on "in-the-wild" attacks regenerated by Sandia Red Teamers. Due to the constraints of repeatability and re-testability, the scripted attacks were largely static in nature. Reponses from certain attacks would likely prompt other actions for a live red teamer to undertake. The enumerations for this could be endless and vastly varied based on the methodologies, tools, and intelligence available to the attacker.

Additional considerations were also made as to placement of the attacker, to include start points. For this exercise, the attacker was primarily situated outside of the network. Other attack vectors might originate from insiders, placing them directly on endpoints or with access to network infrastructure (switches, routers) to accommodate man-in-the-middle attacks. Based on the MTD, there may be markedly different results between tests.

VI. Conclusions

In this paper we have conducted an empirical assessment of a network-based moving target defense. The end-user assessment was accomplished using Application Performance Monitoring techniques against a multitier, distributed service in an enterprise network. The results of this test show that this particular MTD resulted in an approximately 30ms delay increase, however, larger server delays are not significantly increased resulting in minimal impact from the user perspective. Additionally, a red team evaluation was conducted against the network, using the Lockheed Martin Cyber Kill Chain methodology as a template. The results of this test showed that a network-based MTD is effective at limiting or thwarting some network-based attacks, but fails at preventing host attacks.

These results are intuitive and almost expected given the context, network topology and attack methodologies. What is novel in this research is the instrumentation and replay mechanisms in the emulation environment to provide the data empirically. Given varied constraints, MTD techniques and attack factors, the ability to modify the testing environment to extract similar, relevant data is completely realizable.

References

[1] B. Van Leeuwen, W. M. S. Stout and V. Urias, "*Operational cost of deploying Moving Target Defenses defensive work factors*," Military Communications Conference, MILCOM 2015 - 2015 IEEE, Tampa, FL, 2015, pp. 966-971.

[2] H. Okhravi, M.A. Rabe, et.al., "*Survey of Cyber Moving Targets,"* Lincoln Laboratory – MIT Technical Report, September 2013.

[3] M. Dunlop, S. Groat, W. Urbanski, R. Marchany, J. Tront, *"MT6D: A Moving Target IPv6 Defense,"* MILITARY COMMUNICATIONS CONFERENCE, 2011 - MILCOM 2011 , vol., no., pp.1321,1326, 7-10 Nov. 2011.

[4] E. Al-Shaer and Q. D. J. Jafarian, "*On the Random Route Mutation Moving Target Defense,*" National Symposium on Moving Target Research, June 2012.

[5] K. S. Quan Jia and A. Stavrou, "*Motag: Moving target defense against internet denial of service attack,.*" International Conference on Computer Communications and Networks (ICCCN), 2013.

[6] W. Peng, F. Li, C. T. Huang and X. Zou, "*A moving-target defense strategy for Cloud-based services with heterogeneous and dynamic attack surfaces,*" 2014 IEEE International Conference on Communications (ICC), Sydney, NSW, 2014, pp. 804-809.

[7] H. Okhravi, J. F. Riordan, and K. M. Carter; "*Quantitative Evaluation of Dynamic Platform Techniques as a Defensive Mechanism*," Research in Attacks, Intrusions, and Defenses. 2014.

[8] R. Zhuang, S. Zhang, S. A. DeLoach, X. Ou, and A. Singhal; "*Simulation-based Approaches to Studying Effectiveness of Moving-Target Network Defense*," National Symposium on Moving Target Research, 2012.

[9] 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, vol. 1, pp. 80, 2011.

[10] V. Urias, B. Van Leeuwen, B. Richardson, "*Supervisory Command and Data Acquisition (SCADA) system cyber security analysis using a live, virtual, and constructive (LVC) testbed*," MILITARY COMMUNICATIONS CONFERENCE, 2012 - MILCOM 2012 , vol., no., pp.1,8, Oct. 29 2012-Nov. 1 2012.

[11] V. Urias, B. Van Leeuwen, W. Stout, B. Wright, "*Emulytics at Sandia National Laboratories*" 2015, MODSIM World 2015.

[12] Minimega: a distributed VM management tool, http://minimega.org/

[13] Open vSwitch, http://openvswitch.org