Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/cose

Computers & Security

Dynamic security metrics for measuring the effectiveness of moving target defense techniques

Jin B. Hong[a,*], *Simon Yusuf Enoch*[b], *Dong Seong Kim*[b], *Armstrong Nhlabatsi*[c], *Noora Fetais*[c], *Khaled M. Khan*[c]

[a] *Department of Computer Science and Software Engineering, University of Western Australia, Australia*
[b] *Department of Computer Science and Software Engineering, University of Canterbury, New Zealand*
[c] *Department of Computer Science and Engineering, KINDI Computing Lab, Qatar University, Qatar*

ARTICLE INFO

Article history:
Received 27 April 2018
Revised 31 July 2018
Accepted 14 August 2018
Available online 23 August 2018

Keywords:
Emerging networking technology
Moving target defense
Security analysis
Security metric
Security model

ABSTRACT

Moving Target Defense (MTD) utilizes granularity, flexibility and elasticity properties of emerging networking technologies in order to continuously change the attack surface. There are many different MTD techniques proposed in the past decade to thwart cyberattacks. Due to the diverse range of different MTD techniques, it is of paramount importance to assess and compare their effectiveness. However, each technique causes distinct (dynamic) changes in the network, making an objective comparison difficult. In this paper, we incorporate MTD techniques into a temporal graph-based graphical security model, and develop a new set of dynamic security metrics to assess and compare their effectiveness. To this end, we first categorize and compare different attack and defense efforts. Second, we describe the temporal graph-based graphical security model to capture dynamic changes made by various MTD techniques in the network. We then develop a new set of security metrics for attack and defense efforts to evaluate the effectiveness of the MTD techniques. We implement two different MTD techniques, namely network topology shuffle and software diversity, and show their effectiveness against a targeted attack scenario in our experimental analysis. The results demonstrate that the proposed dynamic security metrics can capture different properties of MTD techniques, permitting a more fine-grained comparison and offering guidance for selecting the most effective MTD technique.

1. Introduction

Emerging networking technologies (ENTs), such as cloud computing (Mell and Grance, 2011) and Software-Defined Networking (SDN) (Akhunzada et al., 2016; Masoudi and Ghaffari, 2016), are shifting from static hardware-based networks to dynamic programmable software-based networks. The granularity, flexibility and elasticity properties of ENTs provide new and innovative approaches to enhance the network security. However, those ENTs also bring forward a new set of attack vectors that are targeted by the attackers (Kreutz et al., 2015). Moving Target Defense (MTD) is one of the novel proactive defense mechanisms that benefits from the ENTs and their properties, with an objective to continuously change the attack surface to thwart cyberattacks (Evans et al., 2011; Zhuang et al., 2012).

* Corresponding author.
E-mail addresses: jin.hong@uwa.edu.au (J.B. Hong), simon.yusuf@pg.canterbury.ac.nz (S.Y. Enoch), dongseong.kim@canterbury.ac.nz (D.S. Kim), armstrong.nhlabatsi@qu.edu.qa (A. Nhlabatsi), n.almarri@qu.edu.au (N. Fetais), k.khan@qu.edu.qa (K.M. Khan).

https://doi.org/10.1016/j.cose.2018.08.003
0167-4048/

There are many different MTD proposed such as Zhuang et al. (2013), Zhang et al. (2012), Okhravi et al. (2011) and Vikram et al. (2013). Since there is a large number of MTD techniques available, it is important to assess the effectiveness of MTD techniques to optimally select which ones to deploy. Various assessment frameworks have been proposed to compare the effectiveness of MTD techniques (Lei et al., 2016; Maleki et al., 2016; Zaffarano et al., 2015). However, comparing the effectiveness of MTD techniques requires an in-depth knowledge of both the attacker and the defender capabilities. In practice, it is infeasible to specify all the capabilities of attackers and defenders due to uncertainties (Jafarian et al., 2014). Moreover, the attack surface of the network changes frequently as MTD techniques change the network configurations periodically and spontaneously to mitigate cyberattacks. As a result, a transition from one network state to another shifts the attack surface accordingly, and such change should be captured when assessing the effectiveness of MTD techniques.

It is infeasible in practice to enhance the security all the time for consequent network states, resulting in some network states with a negative impact on security. Here, the term *network state* refers to the particular configuration of the network, which may be changed to a different one when MTD techniques are triggered.[1] Hence, comparing the MTD techniques should not only evaluate their effectiveness against cyberattacks, but also evaluate the shifting attack surface between the network states. In order to evaluate the shifting attack surface between the network states, the security metrics must be able to capture the changing security information. However, existing security metrics do not capture the changing security information as the attack surface shifts.

In this paper, we address the aforementioned problems by assessing the effectiveness of MTD techniques using newly developed dynamic security metrics based on attack and defense efforts. To do this, we utilize a graphical security model, namely the T-HARM (Yusuf et al., 2016), and formalize the changes made by MTD techniques. Moreover, we develop a set of new dynamic security metrics that captures the shifting attack surface of the network when MTD techniques are used. The first step is to categorize attack efforts based on their characteristics and also categorize defense efforts based on the cost and constraints associated with the network components, services and communications. Then, we use T-HARM to incorporate and capture MTD techniques based on their characteristics. Further, we develop a new set of dynamic security metrics that captures the security changes in the network. These metrics are used to measure the effectiveness of MTD techniques. Lastly, we demonstrate the capabilities of our proposed approach through a comparative analysis. Our contributions are summarized as follows.

- Categorize attack and defense efforts.
- Formalize the changes made by MTD techniques.
- Develop a new set of dynamic security metrics to evaluate the effectiveness of MTD techniques.
- Experiment an in-depth comparison of MTD techniques via comparative analysis using the proposed security metrics, and demonstrate their applicability.

The rest of the paper is organized as follows. Related work is presented in Section 2. Categorization of attack and defense efforts is proposed in Section 3. Incorporating the MTD techniques into the temporal graph-based graphical security model is presented in Section 4. The new set of dynamic security metrics is formulated in Section 5. Comparative analysis of MTD techniques is presented in Section 6. Our findings and limitations are critically discussed in Section 7, and finally Section 8 concludes this paper.

2. Related work

Prior to MTD, the general field of intrusion response systems (IRS) provided an automated defense selection, providing both reactive and proactive defense approaches. Kreidl and Frazier (2004) introduced a host-based autonomic defense system by applying a feedback control. They utilized a partially-observable Markov Decision Process (POMDP), and computed the trade-offs between the failure cost and the maintenance cost. Holgado et al. (2017) proposed a multi-step attack prediction method using a Hidden Markov Model, and demonstrated the effectiveness against DDoS attacks. Miehling et al. (2018) used a Condition Dependency Graph, an extension to an Attack Graph (Sheyner et al., 2002), which embedded a state space to quantify the progression of the attacker over time. Another application of graphical security model is proposed by Zonouz et al. (2009), where they presented a response and recovery engine using the model named Attack Response Tree, which is an extension from the Attack Tree (Schneier, 1999), that uses a competitive Markov decision process. The IRS solutions provide effective defense selection against identified attacks, but their effectiveness in the dynamic network environment has not been studied. Also, the IRS solutions do not periodically adapt to the changing dynamic network environments. Although decision-theoretic approaches can be used to address this issue, their applicability is restricted to the defined threat model, as the evaluation results are dictated by the threat model used. To address this issue, MTD techniques provide capabilities to periodically adapt their solutions to the changing network environment.

New MTD techniques are evaluated by analyzing the changes in the security posture of the network (Kampanakis et al., 2014; Luo et al., 2014; MacFarland and Shue, 2015). However, there is only a few security metrics available to measure and compare the effectiveness of MTD techniques. Evans et al. (2011) proposed an approach to evaluate the effectiveness of MTD techniques by evaluating the probability of an attack success. However, they only focused on the randomization-based MTD techniques, which are difficult to be compared with non-probabilistic MTD techniques (i.e., outcomes are either true or false). Zhuang et al. (2012) proposed to use simulation-based approaches, but they investigated only the rate of attack success for proactively changing the network without

[1] Since we are interested in comparing different MTD techniques, we only consider changes in the network states with respect to MTD techniques only (i.e., other network changing events, such as updates, are not considered).

taking into account the cost associated with the defender. Xu et al. (2014) proposed a three-layer model to evaluate and compare the effectiveness of MTD techniques. The model evaluates the MTD techniques from a mission-oriented security point of view, but they do not provide how to formulate and interpret the mission, as well as the general approach to incorporate the mission into the model domain. Zaffarano et al. (2015) proposed a quantitative framework considering the mission-oriented security view, and developed new metrics to evaluate the attacker and defender missions. However, the scope of the mission is limited to *productivity, success, confidentiality* and *integrity*, and the proposed metrics only provide a high-level security overview without capturing the changes made between the network states when MTD techniques are deployed (i.e., the metrics do not take into account the low level changes made in the network). Maleki et al. (2016) proposed to use a Markov-model-based framework to measure the effectiveness of MTD techniques based on security capacity, which is derived from the MTD technique and its deployed system's parameters. However, they only considered the probability of a successful attack to defeat the MTD technique in relation to the attackers time and cost spent. As a result, they lack multiple security views of different MTD techniques. Lei et al. (2016) proposed to use a multi-layer network resource graph (MNRG) and a change-point detection in real time to capture changes in the network. However, this approach lacks the availability of metrics to conduct a comparative analysis for MTD techniques (i.e., cannot provide different security perspectives). Zhu and Başar (2013) proposed to use a two-person zero-sum game approach to design a multi-stage MTD system, where an MTD technique is used to change the defense strategy by adapting to the attack pattern. However, the game is based on qualitative damage metric only, which does not provide information to evaluate and compare the effectiveness of the MTD system. There are many other game theory-based MTD system designs, but they do not take into account various security metrics to assess their effectiveness. Our previous work in Hong and Kim (2016) used graphical security models to capture the properties of MTD techniques, but the overall security changes are not provided as existing security metrics used cannot capture and represent changes in the network.

Many evaluation methodologies have been proposed to compare the effectiveness of MTD techniques, but they lack appropriate security metrics to capture the details of changes made to the network when those techniques are deployed. Pendleton et al. (2016) surveyed security metrics for the security assessment, and they reported that more direct measurements to assess the effectiveness of MTD techniques are needed, such as the security gained and/or the attack efforts by deploying MTD techniques. This paper aims to address some of the key issues identified from the previous work, namely; (1) to capture and categorize attack and defense efforts required when MTD techniques are deployed, (2) to propose a new set of security metrics reflecting attack and defense efforts, and (3) to investigate the usability of the proposed metrics taking into account the trade-off between attack and defense efforts, and the overall effectiveness of different MTD techniques.

3. Categorization of attack and defense efforts

Attack and defense efforts vary depending on the imposed threat. For example, a route randomization may be used against DDoS attacks (Jia et al., 2014), while IP shuffling may be used against probing/reconnaissance types of attacks (Jafarian et al., 2014). Hence, only the MTD techniques designed to mitigate the same type of threats (i.e., have the same security objectives) should be compared. Another approach to match MTD technique instances to individual attack/threat instances can be used to identify which cyberattacks can be mitigated (Xu et al., 2014). However, this approach is impractical as it requires enlisting and matching all MTD techniques and attack instances, which can be time-consuming and error-prone due to a lack of attack instance classification, as well as new attack instances that are being discovered. Moreover, certain MTD techniques are probabilistic (Huang and Ghosh, 2011; Maleki et al., 2016; Manadhata, 2013), resulting in a more complex relationship between the MTD techniques and attack instances to evaluate the effectiveness. Instead, we take into account effort-based attacker and defender actions that would benefit or obstruct the attacker. We assume that increasing the attack efforts will deter cyberattacks, based on the definition of the MTD that continuously changing the attack surface thwarts cyberattacks. To compare MTD techniques, relevant attack and defense effort categories are selected and evaluated. This simplifies the matching MTD technique instances to individual attack instances. These attack and defense efforts will also form the basis of new dynamic security metrics for assessing the effectiveness of MTD techniques, which are presented in Section 5. First, we categorize attack efforts in Section 3.1, followed by the categorization of defense efforts in Section 3.2.

3.1. *Categorization of attack efforts*

Attack efforts vary depending on the complexity of the attack scenarios, which are dependent on the security characteristics of the network configuration. Hence, identifying security characteristics of the network configuration can be used to evaluate the attack efforts. The security characteristics of using MTD techniques can be specified by the changes made in the network configuration, which do not require the complex relationships between MTD technique and attack instances. This approach simplifies the attack model that only requires their characteristics, in comparison to enlisting all possible attack instances. Moreover, attack instances are related to changes made in the network rather than directly to the individual MTD technique instances. As a result, MTD techniques are compared based on their characteristics to make cyberattacks more difficult, rather than checking individually which cyberattacks they can mitigate. That is, MTD techniques are evaluated based on maximizing attack efforts, instead of against individual attack instances.

Fig. 1 shows the categorization of attack efforts. It shows two subcategories of the attack efforts, based on *Reconnaissance* and *Resource*. Reconnaissance is an action taken by the attacker to gather information. This is further divided into *Scanning* and *Frequency*, where *Scanning* observes the network

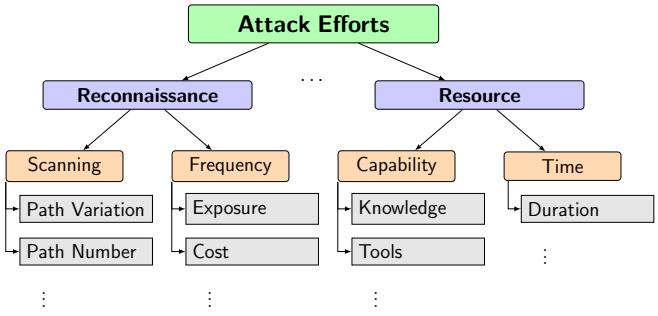


Fig. 1 – Categorizing attack efforts.

configurations, and *Frequency* specifies the amount of scanning. *Resource* specifies the properties of the attacker, which divides into *Capability* and *Time*. *Capability* specifies the ability of the attacker (e.g., knowledge of attacks, tool availability etc.), and *Time* specifies how much time would be/has taken for the cyberattack. New dynamic security metrics are proposed to capture those attack efforts in Section 5.1. Although the categorization of attack efforts does not capture all possible ways, our module-based approach allows flexible changes to the categorization (i.e., subcategories can be added, updated and removed without disrupting other subcategories). Individual and different selections of the attack efforts subcategories can then be evaluated to compare the effectiveness of the MTD techniques.

3.2. *Categorization of defense efforts*

When deploying MTD techniques, network administrators also need to consider the cost associated with them and the trade-offs (e.g., security versus performance). Similar to attack efforts, we consider the defender's efforts made when deploying MTD techniques. Fig. 2 shows the categorization of defense efforts. It shows three subcategories on the basis of network components; network nodes, services, and communications. All MTD techniques make changes, either directly or indirectly, within those network component categories. *Nodes* (e.g., hosts) can be changed using MTD techniques such as software and application diversity (Cox et al., 2006; Jackson et al., 2011; Nguyen-Tuong et al., 2008), which require time and resources to do so. *Services* form a platform for interaction (e.g., applications), which requires adoption and deployment of new services and maintenance. This can be changed using MTD techniques such as network level diversity (Newell et al., 2013; Williams et al., 2009). *Communications* (e.g., connections between network nodes) can be changed using MTD techniques such as topology shuffle (Hong et al., 2017; MacFarland and Shue, 2015), virtual machine migration (Jia et al., 2014), and IP shuffling (Jafarian et al., 2012; 2014). Similarly with attack efforts, new dynamic security metrics capturing each subcategory of defense efforts are presented in Section 5.2. Defense efforts can also be represented as modules, which can be updated where necessary.

4. Temporal graphical security model

We use a temporal graphical security model, namely a Temporal Hierarchical Attack Representation Model (T-HARM) (Yusuf et al., 2016), and incorporate MTD techniques in order to capture the properties of attack and defense efforts. We implement dynamic security metrics in the T-HARM to assess the effectiveness of MTD techniques, which are captured as described below. An SDN shown in Fig. 3 is used as a toy example. We assume each host in the example has one vulnerability v_i, where i corresponds to the host number. In this section, we extend the T-HARM capabilities to incorporate MTD techniques by capturing the changes made to the network. First, we describe the network states in Section 4.1. Then, Section 4.2 presents the formalism of the T-HARM, and we describe the MTD technique characteristics and definitions to incorporate them into the T-HARM in Section 4.3.

4.1. *Network states*

In this paper, we use the term *network states* to represent different network configurations and settings at various times. This includes, but not limited to, applications and operating system changes, network topology changes, vulnerability patches and other possible network configuration changes. The network state is changed when MTD techniques are deployed onto the network, in order to shift the attack surface. Hence, we capture different network states arising from MTD techniques modifying the network at different times. Of course, the network configuration may be changed by the system configuration settings and policies other than the MTD techniques. However, we are only concerned with comparing MTD techniques, and therefore, such changes are not considered in this paper. By capturing the changes made by MTD techniques, we can evaluate the changes in the attack surface and the security posture of the network, which forms the foundation of comparing different MTD techniques. An attacker may learn the deployed MTD technique over time, and evolves the attack strategy in order to bypass it. But even if the attacker changes the attack pattern, it does not change the network states. Even if the attacker learns how to evolve the attack pattern, there are still relative efforts associated with the change. For example, if an attacker increases the number of port scanning to discover all hosts in a network that has deployed IP shuffling, then the attacker is increasing the attack efforts (here, the probing rate) in order to bypass the IP shuffling. On the other hand, if the attacker changes the network configuration in order to bypass the deployed MTD technique, it will reflect the ineffectiveness of the deployed MTD technique as it did not protect the network by allowing the attacker to bypass its defense by modifying the configurations. However, this changes in the configuration will be captured using the T-HARM as a new network state, which will then be evaluated and reflect the shortcomings of the MTD technique exploited. Hence, an evolving attacker will either increase the attack efforts, or introduce vulnerable network states, which can be assessed using the proposed approach.

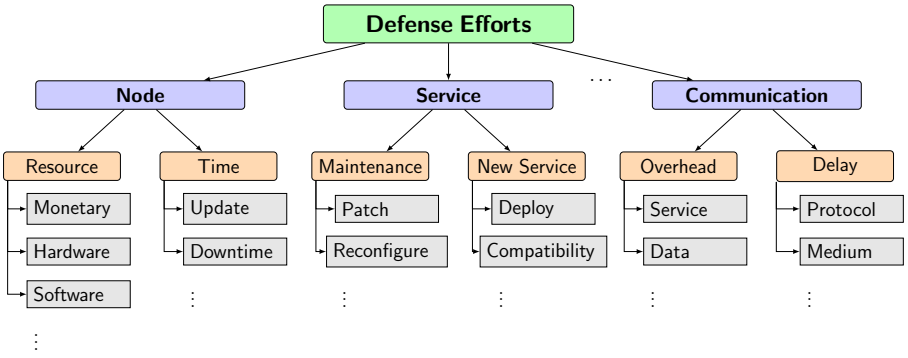
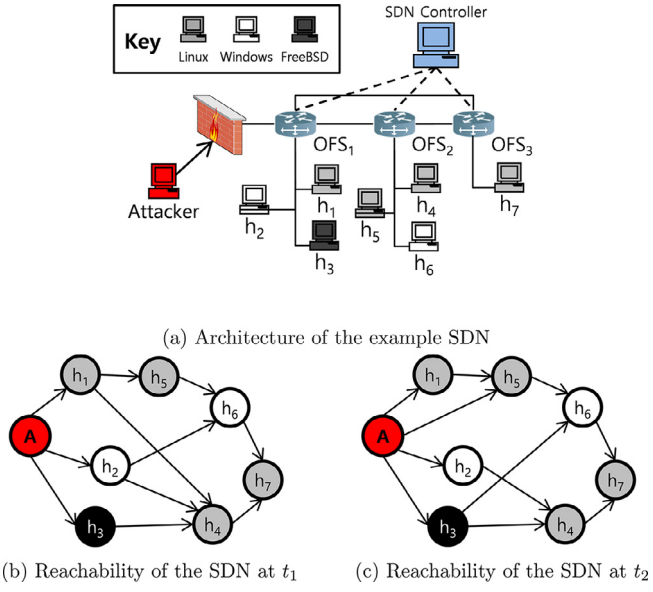


Fig. 2 – Categorizing defense efforts.



(a) Architecture of the example SDN

(b) Reachability of the SDN at t_1

(c) Reachability of the SDN at t_2

Fig. 3 – An example SDN.

4.2. T-HARM

Given a set of network states S, we first formally define the T-HARM as follows.

Definition 1 (Yusuf et al., 2016). The T-HARM is a 3-tuple (N, H, V), where $N = \{n_{t_0}, n_{t_1}, \ldots, n_{t_n}\}$ is a set of HARM for each network state s_{t_i} at time t_i for $i \in \mathbb{N}$, and each n_{t_i} corresponds to s_{t_i}. $H = \{h_0, h_1, \ldots, h_x\}$ is a finite set of all hosts in the network, and $V = \{v_0, v_1, \ldots, v_y\}$ is a finite set of all vulnerabilities found in the network. Further, $H_{t_i} \subseteq H$ is a set of hosts that are in the network state at time t_i, and similarly $V_{t_i} \subseteq V$ is a set of vulnerabilities that are in the network state at time t_i.

The HARM for each network state is 2-layered consisting of an Attack Graph (Sheyner et al., 2002) at the upper layer and Attack Trees (Schneier, 1999; 2000) at the lower layer that models the reachability of hosts and vulnerability information, respectively. The individual HARM captured at each time t_i then can be defined as follows.

Definition 2 (Yusuf et al., 2016). A 2-layered HARM is a 3-tuple $n_{t_i} = (U_{t_i}, L_{t_i}, C_{t_i})$, where U_{t_i} is an upper layer using an Attack Graph that captures only the reachability of hosts that establishes attack paths, L_{t_i} is a lower layer that is a set of Attack Trees that only captures the vulnerability information of each host in the upper layer, and C_{t_i} is the mapping between the upper layer hosts to each lower layer Attack Tree model (i.e., for every host, there is an associated Attack Tree).

Finally, the upper and the lower layers of the HARM are defined respectively as follows.

Definition 3 (Yusuf et al., 2016). The upper layer of the HARM is a 2-tuple $U_{t_i} = (H_{t_i}, E_{t_i})$ at time t_i, where H_{t_i} is a finite set of hosts in the network, and $E_{t_i} \subseteq H_{t_i} \times H_{t_i}$ is a set of edges.

Definition 4 (Yusuf et al., 2016). The lower layer of the HARM is a set of Attack Trees $L_{t_i} = \{l_{h_1}, l_{h_2}, \ldots, l_{h_j}\}$ at time t_i, where an Attack Tree instance associated with an upper layer host $h_j \in H_{t_i}$ is a 4-tuple $l_{h_j} = (A_{t_i}, B_{t_i}, c_{t_i}, root_{t_i})$ with $l_{h_j} \in AT_{t_i}$, where $A_{t_i} \subseteq V_{t_i}$ is a set of vulnerabilities, $B_{t_i} = \{b^j_{t_i} \mid b^j_{t_i} \in \{AND, OR\}\}$ is a set of logical gates, $c_{t_i} \subseteq \{b^j_{t_i} \rightarrow e_k\} \forall b^j_{t_i}, e_k \in A_{t_i} \cup B_{t_i}$ is a mapping of gates to vulnerabilities and other gates, and $root_{t_i} \in A_{t_i} \cup B_{t_i}$ is the root (i.e., the root node either a vulnerability or a gate connecting vulnerabilities and other gates).

Given the definitions above, the example SDN shown in Fig. 3 is denoted as follows.

Example 1. The T-HARM of the example is $T - HARM_{eg} = (N_{eg}, H_{eg}, V_{eg})$, where $N_{eg} = \{n_{eg,t_1}, n_{eg,t_2}\}$, $H_{eg} = \{A, h_1, h_2, \ldots h_7\}$, and $V_{eg} = \{v_1, v_2, \ldots, v_7\}$. Further, $H_{eg,t_1} = H_{eg,t_2} = H_{eg}$ (i.e., all hosts appear in both time t_1 and t_2, and similarly $V_{t_1} = V_{t_2} = V_{eg}$.

Example 2. The 2-layered HARM for the example at time t_1 is $n_{eg,t_1} = (U_{eg,t_1}, L_{eg,t_1}, C_{eg,t_1})$.

Example 3. The upper layer of the HARM n_{eg,t_1} is $U_{eg,t_1} = (H_{eg,t_1}, E_{eg,t_1})$, where $H_{eg,t_1} = H$, and $E_{eg,t_1} = \{\{A, h_1\}, \{A, h_2\}, \{A, h_3\}, \{h_1, h_4\}, \{h_1, h_5\}, \{h_2, h_4\}, \{h_2, h_6\}, \{h_3, h_4\}, \{h_4, h_7\}, \{h_5, h_6\}, \{h_6, h_7\}\}$ is a set of edges.

Example 4. The lower layer of the HARM n_{eg,t_1} is a set of Attack Trees $L_{eg,t_1} = \{l_{h_1}, l_{h_2}, \ldots, l_{h_7}\}$. For instance, an attack tree of a host h_1 is $l_{h_1} = (A_{h_1,t_1}, B_{h_1,t_1}, c_{h_1,t_1}, root_{h_1,t_1})$, where $A_{h_1,t_1} = v_1$, $B_{h_1,t_t} = \{OR_1\}$, $c_{h_1,t_t} = \{(OR_1, v_1)\}$, and $root_{h_1,t_1} = OR_1$.

The HARM models the security posture of each network state, which can be used to compute attack paths and incorporate exploitability for each vulnerability using the CVSS BS (Yusuf et al., 2016). Given the definition of the T-HARM above, we incorporate MTD techniques based on their characteristics in the next section.

4.3. *MTD technique characteristics and definitions*

MTD techniques are mainly classified into three categories (Hong and Kim, 2016); (i) Shuffle, (ii) Diversity, and (iii) Redundancy. Each category of MTD techniques changes the network configuration in a different way. We extend our previous work in Hong and Kim (2016), to incorporate changes by MTD techniques in the T-HARM as follows.

4.3.1. *Shuffle*

Shuffle-based MTD techniques rearrange the existing network configurations, such as VM migration (Jia et al., 2014), network path reconfiguration (Hong et al., 2017; Rohrer et al., 2014), host mutation (Jafarian et al., 2012) at the network level and address space randomization (Evans et al., 2011) at the application level. The network level shuffle-based MTD techniques change the network topology and/or the connectivity of hosts. Hence, we can capture the topological changes to create new instances of the HARM at different time intervals t_i. This is defined in the T-HARM as follows.

Definition 5. Given $\{h_x, h_y\}$ (where $h_x \in H$, $h_y \in H$) to be updated (i.e., add or remove) at time t_{i+1} as a result of the shuffle-based MTD technique, the upper layer of the HARM in $n_{t_{i+1}}$ changes to $(H_{t_{i+1}}, E_{t_{i+1}})$, where there are no changes in the set of hosts (i.e., $H_{t_{i+1}} = H_{t_i}$), and $E_{t_{i+1}} E_{t_i} \oplus \{(h_x, h_y)\}$.

Example 5. For the path reconfiguration shown in the example SDN between n_{eg,t_1} and n_{eg,t_2}, the transition is captured from U_{eg,t_1} to U_{eg,t_2} such that $H_{eg,t_1} = H_{eg,t_2} = H$ (i.e., no changes in the host as defined above), and $E_{t_2} = E_{t_1} \oplus \{(A, h_5), (h_1, h_4), (h_2, h_6), (h_3, h_6)\}$.

On the other hand, shuffle in the application level only affects individual hosts in order to withstand attacks. Hence, the lower layer of the T-HARM captures the changes in the security posture of the host. Consequently, the resulting effects are as same as deploying the diversity-based MTD techniques. Therefore, it is defined in Definition 6.

4.3.2. *Diversity*

Diversity-based MTD techniques have different network and system component configurations while maintaining the identical operations and functions. This includes network routing node diversification (Newell et al., 2013), virtual machine diversity (Williams et al., 2009) at the network level and data diversity (Nguyen-Tuong et al., 2008), software diversity (Cox et al., 2006; Jackson et al., 2011; O'Donnell and Sethu, 2004) and web application diversity (Taguinod et al., 2015) at the application level. The diversity-based MTD techniques do not alter the connectivity of the network components in both the network and the application levels. Hence, only the lower layer of the T-HARM is changing through different time t_i. As a result, however, it will change the set of attack paths when the set of vulnerabilities is changed between the network states (i.e., attackers need a different set of exploits to carry out a multi-stage attack). This can be defined in the lower layer of the HARM only, and used in each state of the T-HARM as follows.

Definition 6. Using the diversity-based MTD technique, let a set of vulnerabilities changed (i.e., added or removed) as a result is given by a function $d(h_{t_{i+1}}) \subseteq V$. Then, the lower layer of the HARM in $n_{t_{i+1}}$ is updated with $A_{t_{i+1}} = A_{t_i} - d(h_{t_{i+1}})$, $B_{t_{i+1}} = \{b^j_{t_{i+1}} \mid b^j_{t_{i+1}} \in \{AND, OR\}\}$, $c_{t_{i+1}} \subseteq \{b^j_{t_{i+1}} \rightarrow e_k\} \forall b^j_{t_{i+1}}, e_k \in A_{t_{i+1}} \cup B_{t_{i+1}}$, and $root_{t_{i+1}} \in A_{t_{i+1}} \cup B_{t_{i+1}}$.

This is not applicable in the example SDN, but any diversity technique will result in changes to the lower layer of the HARM at t_{i+1}.

4.3.3. *Redundancy*

Redundancy-based MTD techniques replicate existing network components in order to provide high availability, mainly concerned with denial of service types of attacks. This includes routing path redundancy (Al-Wakeel and Al-Swailemm, 2007), virtual machine redundancy (Jia et al., 2014) at the network level and data redundancy (Nguyen-Tuong et al., 2008), server redundancy (Gorbenko et al., 2009; Huang et al., 2006) at the application level. The network level redundancy-based MTD techniques change the network topology by adding new connections with the replicas. The upper layer HARM is extended with replicated nodes, and the corresponding lower layer nodes are created. This can be captured in the T-HARM as follows.

Definition 7. Let the replicated host in the network be $h^r_{t_{i+1}}$, then the upper layer of the HARM in $n_{t_{i+1}}$ changes to $(H_{t_{i+1}}, E_{t_{i+1}})$, where $H_{t_{i+1}} = H_{t_i} \cup \{h^r_{t_{i+1}}\}$, and $E_{t_{i+1}} \subseteq H_{t_{i+1}} \times H_{t_{i+1}}$.

Redundancy in the application level separates into two; (1) functional redundancy (e.g., servers), and (2) resource redundancy (e.g., data). Only in case of (1), vulnerabilities are duplicated. Even though they are the same set of vulnerabilities, the multiplicity may decrease the attack efforts. This can be defined in the T-HARM as follows.

Definition 8. Let a set of vulnerabilities added by the replication be $V^r_{t_{i+1}} \subseteq \{v_{t_{i+1}} \mid v_{t_{i+1}} \in V\}$. Then, the lower layer of the HARM in $n_{t_{i+1}}$ is updated with $A_{t_{i+1}} = A_{t_i} \cup V^r_{t_{i+1}}$, $B_{t_{i+1}} = \{b^j_{t_{i+1}} \mid b^j_{t_{i+1}} \in \{AND, OR\}\}$, $c_{t_{i+1}} \subseteq \{b^j_{t_{i+1}} \rightarrow e_k\} \forall b^j_{t_{i+1}}, e_k \in A_{t_{i+1}} \cup B_{t_{i+1}}$, and $root_{t_{i+1}} \in A_{t_{i+1}} \cup B_{t_{i+1}}$.

Similarly with the diversity techniques, redundancy will result in changes to either the upper layer or the lower layer of the HARM at time t_{i+1} depending on which type of redundancy is used.

The given definition for the T-HARM can be used to capture different network states generated when deploying MTD techniques. The captured network states and security information (e.g., attack paths, vulnerability exploitability) are used to compute a new set of security metrics in the next section.

Table 1 – Network and characteristics terms.

Terms	Descriptions
S	A set of network states
H	A set of hosts in the network
V	A set of vulnerabilities in the network
AP	A set of attack paths
VAR	A set of variants

Table 2 – Network and characteristics functions.

Functions	Descriptions
$t(s_i)$	The time duration of an ith network state
$t(ap_i)$	The time duration of an attack exploiting the attack path ap_i
$t(v_k)$	The time taken to exploit a vulnerability v_k
$et(s_i)$	The time duration of the edge pair changes in s_i
$et(h_j)$	The time taken to update the edge pairs of a host h_j
$path(s_i)$	The set of AP at the ith network state
$vuls(ap_i)$	The set of vulnerabilities associated with an attack path ap_i
$ES(s_i)$	The set of edges in the ith network state
$Ep(v_k)$	The exploitability of a vulnerability v_k using the CVSS
$vc(var_k, h_j)$	The cost of assigning a variant var_k to a host h_j
$vh(h_j, i)$	The variant of the host h_j in the ith network state
$dt(var_k, h_j)$	The downtime assigning the variant var_k to the host h_j

5. Dynamic security metrics

Existing security metrics lack the capabilities to represent the shift in the security posture of networks as the network components change over time. It is of paramount importance to understand the shifting security posture in order to provide effective security solutions. To address this issue, we take into account attack and defense efforts to develop a set of new security metrics to capture the effects of MTD techniques with respect to changes in the network using the T-HARM. First, Section 5.1 presents the attack efforts metrics, and Section 5.2 presents the defense efforts metrics. Then, we describe the use of the metric modules in combinations in Section 5.3. Lastly, we describe the quantification of these metrics in Section 5.4. Because the proposed dynamic security metrics are modularized, this approach provides flexibility to add/modify/remove metrics as necessary. In this section, we present a few selected metrics for attack and defense efforts as a demonstration, and a full suite of dynamic security metrics will be developed in our future work.

The terms and functions used for the network and its characteristics are shown in Tables 1 and 2, respectively. The cardinality (i.e., the number of elements) of sets is represented by the vertical lines (i.e., $|S|$ represents the cardinality of the set of network states).

5.1. *Attack efforts metrics*

The static networks allow attackers to discover vulnerabilities and plan their attack indefinitely. To increase the attack efforts, the attack surface can be changed continuously. The objectives for security metrics evaluating MTD techniques are to capture the change in the security posture as the attack surface changes. One aspect of the attack surface is the attack paths. By observing the changes to attack paths, we can evaluate the increase in the attack efforts.

5.1.1. *Scanning: path variation*

If the network is static, then the set of possible attack paths remains the same unless the vulnerabilities are changed. When an attacker scans for vulnerabilities, the attacker can also discover the static nature of the network if such attack paths are not changed. Therefore, it is critical that attack paths are changed in order to change the attack surface. As the attack surface shifts, the dependencies between the vulnerabilities change so do the attack paths. However, if some attack paths remain (i.e., vulnerability dependencies are unchanged), then they can still be exploited as their visibility is the same. Hence, the attack path variation (APV) measures the shift in attack paths as the network changes when MTD techniques are deployed. We assume that replacing the existing attack paths with new ones increases the attack effort, because when the intended attack path changes, the attacker must redesign the attack strategy and may also need a new set of exploits. That is, APV captures the change in the set of attack paths between the network states.

Given the definition above, the difference in attack paths, $\Delta AP_{(AP_i, AP_{i-1})}$, between the two network states, i and $i-1$, can be described as shown in Eq. (1). Here, AP_i represents the set of attack paths found in the i^{th} network state. The variation in the current set of attack paths is the difference from the previous set of attack paths. That is, the MTD technique is less effective when the proportion of attack paths from the previous network state is large. The set of difference from the previous network state to the current one reveals the new attack paths which were not in the previous network state.

$$\Delta AP_{i,i-1} = \frac{|AP_i - AP_{i-1}|}{|AP_i|} \tag{1}$$

We can then compute the APV metric for all consecutive network states as shown in Eq. (2). The metric has been normalized by the number of consecutive network state pairs. This represents the overall changes in the attack paths and their variations over the observed network states S. This metric is quantifiable such that it measures how the attack paths are changed when MTD techniques are deployed. Hence, there are no arbitrary values that need to be assigned to compute APV.

$$APV = \frac{\sum_{i=1}^{|S|} \Delta AP_{i,i-1}}{|S| - 1} \tag{2}$$

5.1.2. *Scanning: path number*

Increasing the number of attack paths can negatively impact the network security, as it reveals more choices to be taken by the attacker when the scanning is carried out. Therefore, changes in the number of attack paths, $\Delta|AP_{(AP_i, AP_{i-1})}|$, also need to be taken into account. Eq. (3) shows the proportion of attack paths increased. If the number of attack paths stays the

same, then it equates to value 1. But if the previous network state had no attack paths, then it equates to 0.

$$\Delta\overline{AP}_{i,i-1} = 1 - \frac{|AP_i| - |AP_{i-1}|}{|AP_i|} \tag{3}$$

Consider now a reduced number of attack paths in the current network state. Assuming that the attack efforts stay the same (i.e., there is no advantage), then the effectiveness can be calculated as shown in equation below.

$$\Delta\overline{AP}_{i,i-1} = 1 - \frac{max(|AP_i| - |AP_{i-1}|, 0)}{|AP_i|} \tag{4}$$

Similarly with *APV*, we compute the difference between the number of attack paths for all network states, and then normalize it. Eq. (5) shows the computation of APN metric that measures the differences between the numbers of attack paths for all network states. The APN metric also does not need arbitrary assignment of values, it is calculated only based on the observed number of attack paths from all network states *S*.

$$APN = \frac{\sum_{i=1}^{|S|} \Delta\overline{AP}_{i,i-1}}{|S| - 1} \tag{5}$$

5.1.3. *Frequency: exposure*

The attacker is likely to prepare and launch an attack successfully if the exposure of an attack path is long enough, similar to the attack lifetime described in Evans et al. (2011). Hence, the duration of an attack path should be minimized to enhance security. However, estimating the amount of time needed for the attacker to prepare and launch an attack is difficult. Hence, the best case is to minimize the duration of an attack path exposure. That is, the goal of the metric is to compute the duration of each attack path exposed. We use the function $t(ap_i)$ defined in Table 2 to compute the attack path exposure as shown in Eq. (6), which is normalized by the number of attack paths and the total number of network states.

$$APE = 1 - \frac{\sum_{i=0}^{|S|} t(ap_j)}{|AP| \times \sum_{i=1}^{|S|} t(s_i)} \forall ap_j \in AP_i \tag{6}$$

5.1.4. *Capability: knowledge*

Costs are one of the important decision constraints for both attackers and defenders. We estimate the cost of an attack based on the difficulty of exploiting vulnerabilities using the Common Vulnerability Scoring System (CVSS) (Schiffman et al., 2004), particularly the exploitability score (from the Base Equation of the CVSS) that determines the difficulty of exploiting the vulnerability (i.e., the knowledge of the attacker will determine the ability to exploit vulnerabilities with lower exploitability scores etc). We use the CVSS version 2, as many of the legacy vulnerabilities do not have the version 3 available yet, but new vulnerabilities still have version 2 available, which is more practical (both versions have the exploitability (sub) score which is used in this subsection). However, other means of cost metrics can be used to further categorize the knowledge category of the attacker efforts in addition to the CVSS exploitability scores.

The attack cost associated with exploiting vulnerability can be calculated taking into account all possible attack paths. We assume that exploiting the same variant (e.g., application, operating system and others) does not reduce the attack efforts. Then, the exploitability of each attack path becomes the cumulative product of all the vulnerabilities required. The attack cost of exploitation, AC_i, is shown in equation below.

$$AC_i = 1 - \prod_{j=1}^{|AP_i|} \left(1 - \prod_{k=1}^{|ap_j|} Ep(v_k)\right), \quad \text{where } v_k \in vuls(ap_j) \forall ap_j \in AP_i \tag{7}$$

Using the above equation, we can compute the attack cost associated with the exploitation of vulnerabilities as shown in Eq. (8) taking into account all network states.

$$ACE = 1 - \frac{\sum_{i=0}^{|S|} AC_i}{|S|} \tag{8}$$

The inner product computes the exploitability of each attack path and combines them using the disjoint set theory. This is processed for all network states which are then normalized (i.e., transformed into the range [0, 1]).

5.1.5. *Time: duration*

The amount of time taken for the attacker to compromise each stepping stone in an attack path is another significant factor, as the longer the attack takes, the more likely it will be detected. Hence, increasing the amount of time to attack can negatively affect the attacker. Also, we assume the attacker will minimize the time taken for an attack. Hence, we compute the minimum time taken by the attacker to compromise the target in each network state based on the time taken to exploit each vulnerability in the attack path, which is presented as a function $t(ap_i)$. In practice, we can approximate the minimum time to exploit (i.e., the function $t(ap_i)$) through empirical studies (McQueen et al., 2009; Zhang et al., 2014), as well as using other timing models as appropriate. Assuming that exploiting a vulnerability has a specific time frame, we do not have to consider the skills of different attackers. Eq. (9) shows the normalized metric representing the time taken to compromise the target in a given network state. The time has been normalized by the maximum amount of time to compromise the target by the attacker.

$$ACD = \sum_{i=0}^{|S|} \frac{\frac{min(t(ap_j))}{max(t(ap_j))}}{|S|} \forall ap_j \in AP_i \tag{9}$$

5.2. *Defense efforts metrics*

There are costs associated with deploying MTD techniques in the network. In this section, we capture the cost associated with MTD techniques with respect to the defense efforts presented in Section 3.2.

5.2.1. *Resource: monetary*

The network nodes may be changed with another variant, and each variant has its associated cost. For example, generating a software-based diversity may be done automatically by a compiler (Jackson et al., 2011) that may have low cost, but diversifying the whole virtual machine (Williams et al., 2009) may

have a higher cost to populate all of its functionalities. Such actions can be converted to monetary values based on the system mission (e.g., converting downtime to loss of revenue for mission systems). First, the variant assignment cost, VC_i, associated with a given network state s_i is shown in below equation.

$$VC_i = 1 - \sum_{j=1}^{|H_i|} vc(var_k, h_j) \mid vh(h_j, i) \neq vh(h_j, i-1) \quad \forall h_j \in H_i, \forall var_k \in VAR \tag{10}$$

We can compute the normalized variant assignment cost $\overline{VC_i}$ at the ith network state by dividing vc_i by the maximum variant assignment cost allowed for that particular network state as shown in below equation.

$$\overline{VC_i} = 1 - \sum_{j=1}^{|H_i|} \frac{vc(var_j) \mid vh(h_j, i) \neq vh(h_j, i-1)}{|H_i| \times max(vc(var_k))} \quad \forall h_j \in H_i \forall \{var_j, var_k\} \in VAR \tag{11}$$

The total cost associated with different diversity when MTD techniques deployed for all network states is shown in eq. (12). The cost has been normalized by the number of network states.

$$NVC = 1 - \sum_{i=1}^{|S|} \frac{\overline{VC_i}}{|S| - 1} \tag{12}$$

5.2.2. *Time: downtime*

The time required to replace the node with a different variant causes a downtime in the network. The aim is to minimize the downtime in order to continuously provide the network services. Downtimes can be estimated and measured, which can be used for input to this metric calculations. First, Eq. (13) shows the downtime, DT_i, calculation of a given network state s_i.

$$DT_i = 1 - max(dt(var_k, h_j) \mid vh(h_j, i) \neq vh(j_j, i-1) \forall h_j \in H_i \tag{13}$$

Eq. (14) shows the downtime experienced when assigning new variants as a result of deploying the MTD techniques in the network. The downtime has been normalized by the maximum downtime experienced.

$$NVDT = 1 - \sum_{i=1}^{|S|} \frac{max(dt(var_k, h_j) \mid vh(h_j, i) \neq vh(j_j, i-1)}{(|S| - 1) \times max(dt(var_k, h_j))} \quad \forall h_j \in H_i, \forall var_k \in VAR \tag{14}$$

5.2.3. *Overhead: service*

Communication maintenance has some cost associated with it. Restructuring the SDN topology (Hong et al., 2017) may have low overhead, while maintaining a set of virtual IP address for IP shuffling (Jafarian et al., 2014) may have more. Such costs depend on which communication service is in use, and how the communication paths are changed. For the basics, we define the difference between the edge set as the edge variation cost. We assume the cost of changing an edge is the same. Then, we only have to count the number of edge changes between the network states. Networking technologies deploy edge changes in parallel (e.g., SDN), but the amount of change still affects the network performance of the affected region. Hence, the more edge changes, the higher cost is observed (e.g., delay or downtime). The edge variation cost can be computed as shown in Eq. (15), where the cost of each network state is normalized by the maximum edge variation cost, and the total cost normalized by the number of network states.

$$EVC = 1 - \sum_{i=1}^{|S|} \frac{\frac{|ES(s_i) \ominus ES(s_{i-1})|}{|ES(s_i) \cup ES(s_{i-1})|}}{|S| - 1} \tag{15}$$

5.2.4. *Delay: medium*

Similar to the downtime under *Node* category, the communication medium when changed attracts a delay and loss of data between the communications. Hence, it is important that MTD techniques do not affect the system performance by limiting the edge variation time. This can be measured or approximated based on the communication protocol being used with its specification. Using this metric, the effect in time by deploying different MTD techniques is captured and compared. Eq. (16) shows the computation of the time duration of the edge pair changes in s_i.

$$et(s_i) = max(et(h_j)) \forall h_j \in H_i \tag{16}$$

Based on Eq. (16), we can calculate the edge variation time for all network states as shown in Eq. (17). The edge variation time is normalized by the maximum amount of time taken for edge set changes.

$$EVT = 1 - \frac{\sum_{i=1}^{|S|} et(s_i)}{max(et(s_i)) \times (|S| - 1)} \forall s_i \in S \tag{17}$$

5.3. *Application of dynamic security metric modules*

We have described some of the attack and defense effort modules for capturing changes in the security posture when MTD techniques are deployed. This section describes the use of the proposed metrics using an example SDN, and further, demonstrates the use of compound dynamic security metrics to evaluate the effectiveness of MTD techniques. In this section, we use the example SDN shown in Fig. 3.

The reachability of hosts and the running operating system (OS) on each host at the given network state are shown in Table 3. The network state s_0 is the initial network state, and s_1 changes the host reachability only, while s_2 assigns different operating systems to hosts only. We also consider the transition between the network states as follows: (1) At time t_0, the network state $state_0$ is used for 3 h, (2) at time t_1, the network state is changed to $state_1$ with the duration of 2 h, (3) at time t_2, the network state is changed to $state_0$ for 1 h, and (4) at time t_3, the network state is changed to $state_2$ for 2 h. Hence, there are four different network states observed. For instance, $S = \{s_0, s_1, s_2, s_3\}$, where $s_0 = s_2 = state_0$, $s_1 = state_1$, and $s_3 = state_2$ and the cardinality of S is $|S| = 4$. The number of hours is chosen arbitrarily for demonstration only, the actual duration of each network state can be measured based on reconfiguration schedules. We consider these network state transitions in the following examples.

Table 3 – Example SDN network states.

Network state	Host reachability	Running OS
$state_0$	$A \rightarrow \{h_1, h_2, h_3\}$	h_1: Linux
	$h_1 \rightarrow \{h_4, h_5\}$	h_2: Windows
	$h_2 \rightarrow \{h_4, h_6\}$	h_3: FreeBSD
	$h_3 \rightarrow \{h_4\}$	h_4: Linux
	$h_4 \rightarrow \{h_7\}$	h_5: Linux
	$h_5 \rightarrow \{h_6\}$	h_6: Windows
	$h_6 \rightarrow \{h_7\}$	h_7: Linux
	any host $\rightarrow$ Outside SDN	
$state_1$	$A \rightarrow \{h_1, h_2, h_3, h_5\}$	h_1: Linux
	$h_1 \rightarrow \{h_5\}$	h_2: Windows
	$h_2 \rightarrow \{h_4\}$	h_3: FreeBSD
	$h_3 \rightarrow \{h_4, h_6\}$	h_4: Linux
	$h_4 \rightarrow \{h_7\}$	h_5: Linux
	$h_5 \rightarrow \{h_6\}$	h_6: Windows
	$h_6 \rightarrow \{h_7\}$	h_7: Linux
	any host $\rightarrow$ Outside SDN	
$state_2$	$A \rightarrow \{h_1, h_2, h_3\}$	h_1: Linux
	$h_1 \rightarrow \{h_4, h_5\}$	h_2: Linux
	$h_2 \rightarrow \{h_4, h_6\}$	h_3: FreeBSD
	$h_3 \rightarrow \{h_4\}$	h_4: Windows
	$h_4 \rightarrow \{h_7\}$	h_5: Linux
	$h_5 \rightarrow \{h_6\}$	h_6: Windows
	$h_6 \rightarrow \{h_7\}$	h_7: Linux
	any host $\rightarrow$ Outside SDN	

Table 4 – Basic metric values associated with variants.

Variant (OS)	Exploitability	Timetaken to exploit (h)
Linux	0.15	2
Windows	0.2	1
FreeBSD	0.1	4
	Var assign. cost	Var assign. downtime (h)
Linux	1	0.5
Windows	1	0.2
FreeBSD	1	0.8

The attack goal is to reach the target host H_7 through an elevation of privilege. We assume that each host has a remote-to-root vulnerability, and different operating systems have different exploitability values, the time taken to exploit, the variant assignment cost and the variant assignment downtime as summarized in Table 4. We used random but reasonably assigned values for demonstration. In practice, these values can be retrieved from empirical studies (McQueen et al., 2009; Zhang et al., 2014), various vulnerability databases (Gallon and Bascou, 2011; National Institute of Standards and Technology, 2018), and system and network configuration details. For simplicity, the term *Variant Assignment* is written in short as *Var Assign*. We also assume the time taken to update the set of edges for each host is determined by the number of edges updated (including addition and removal) from the previous network state (i.e., the sum of the number of updated edges). Given the scenario, we describe the computation of each metric below, as well as the combinations of those metrics to demonstrate the compound metric.

5.3.1. *APV computations*

The difference between the attack path sets from the example SDN is captured using the APV metric. For instance, $AP_0 = \{(A, h_1, h_4, h_7), (A, h_1, h_5, h_6, h_7), (A, h_2, h_4, h_7), (A, h_2, h_6, h_7), (A, h_3, h_4, h_7)\}$ is the set of attack paths in s_0 with the cardinality value of $AP_0 = 5$. Then, the APV can be computed as shown in below equation.

$$\begin{aligned} APV &= \frac{\sum_{i=1}^{|S|} \frac{|AP_i - AP_{i-1}|}{|AP_i|}}{|S| - 1} \\ &= \frac{\frac{2}{5} + \frac{4}{5} + \frac{4}{5}}{3} \\ &= 0.6667 \end{aligned} \tag{18}$$

The APV value represents the variations in the attack paths for all states in the SDN. Lower APV value means the set of attack paths tends to be more static. The APV value 0.6667 represents the expected proportion of changes to the set of attack paths for all the observed network states.

5.3.2. *APN computations*

Similarly with the APV computation, Eq. (19) shows the APN computation for the example SDN.

$$\begin{aligned} APN &= \frac{\sum_{i=1}^{|S|} 1 - \frac{max(|AP_i| - |AP_{i-1}|, 0)}{|AP_i|}}{|S| - 1} \\ &= \frac{(1 - -0) + (1 - -0) + (1 - -0)}{3} \\ &= 1 \end{aligned} \tag{19}$$

There is no change in the number of attack paths as the number of attack paths for each network state is five, giving the APN value of one. Not increasing the number of attack paths (or at least maintaining it) is better than increasing it. On the other hand, if the APN value tends towards zero, then the number of attack paths is increasing as the network transits to other network states.

5.3.3. *APE computations*

The APE metric measures the amount of attack path exposure for all network states. Eq. (20) shows the APE computation for the example SDN.

$$\begin{aligned} APE &= 1 - \frac{\sum_{i=0}^{|S|} t(ap_j)}{|AP| \times \sum_{i=1}^{|S|} t(s_i)} \forall ap_j \in AP_i \\ &= 1 - \frac{40}{11 \times 8} \\ &= 0.5455 \end{aligned} \tag{20}$$

If the initial attack paths are exposed in all the network states without any new attack paths, then the APE value tends toward zero. Hence, it is better to achieve a high value of APE which represents that an attack path is only exposed in one network state. In the best case scenario, the APE value will be the same as $1 - \frac{1}{|S|}$, and as the number of network states tends toward infinity, it becomes one.

5.3.4. ACE computations

The ACE metric computes the exploitability for the attacker to reach the target. We can compute the attack cost associated with exploiting vulnerabilities in each of the example SDN state as shown in below equation.

$$\begin{aligned} ACE &= 1 - \frac{\sum_{i=0}^{|S|}\left(1 - \prod_{j=1}^{|AP_i|}\left(1 - \prod_{k=1}^{|ap_j|} Ep(v_k)\right)\right)}{|S|} \\ &= 1 - \frac{0.0177 + 0.0151 + 0.0177 + 0.0171}{4} \\ &= 0.9831 \end{aligned} \tag{21}$$

If vulnerabilities have high exploitability value (e.g., 1), the ACE value tends toward zero. Given the assumed exploitability of the assigned operating systems is low, the ACE metric value for the example SDN was computed near one. This means the example SDN configuration is done such that it increases the attack efforts in terms of the exploitation difficulty.

5.3.5. ACD computations

The ACD metric computes the ratio of the shortest and longest times taken for the attacker to exploit the target in each network state. Hence, lower ACD value represents that a significant proportion of the network states can be exploited in a shorter time than the expected maximum amount of time. Eq. (22) shows the calculation steps for the example SDN.

$$\begin{aligned} ACD &= \sum_{i=0}^{|S|} \frac{\frac{min(t(ap_j))}{max(t(ap_j))}}{|S|} \forall ap_j \in AP_i \\ &= \frac{\frac{4}{8} + \frac{5}{8} + \frac{4}{8} + \frac{5}{7}}{4} \\ &= 0.5848 \end{aligned} \tag{22}$$

5.3.6. NVC computations

The total cost associated with different diversity when MTD techniques deployed for all network states is shown in Eq. (23). The NVC value has been normalized by the number of network states.

$$\begin{aligned} NVC &= 1 - \sum_{i=1}^{|S|} \frac{\sum_{j=1}^{|H_i|} \frac{vc(var_j)|vh(h_j,i) \neq vh(h_j,i-1)}{|H_i| \times max(vc(var_k))}}{|S| - 1} \forall \{var_j, var_k\} \in VAR \\ &= 1 - \frac{0 + 0 + \frac{2}{7}}{3} \\ &= 0.9048 \end{aligned} \tag{23}$$

As only the topology configurations change during the first three network states, the overall NVC value is high. If the NVC value is low, then the network state transitions are reassigning variants to the hosts in the network frequently, which also increases the variant assignment cost.

5.3.7. NVDT computations

Similarly as the NVC calculations, the NVDT can be calculated as shown in below equation.

$$\begin{aligned} NVDT &= 1 - \sum_{i=1}^{|S|} \frac{max(dt(var_k, h_j) \mid vh(h_j, i) \neq vh(j_j, i-1)}{(|S| - 1) \times max(dt(var_k, h_j))} \forall h_j \in H_i \\ &= 1 - \frac{0}{3 \times 0.8} + \frac{0}{3 \times 0.8} + \frac{0.5}{3 \times 0.8} \\ &= 0.7917 \end{aligned} \tag{24}$$

The higher NVDT value represents less downtime observed when assigning variants in the network. As the NVDT value converges to zero, it represents the maximum downtime for all network states.

5.3.8. EVC computations

The EVC for the example SDN can be calculated as shown in Eq. (25). The higher value represents fewer changes to the set of edges between the network states which results in lower edge assignment cost.

$$\begin{aligned} EVC &= 1 - \sum_{i=1}^{|S|} \frac{\frac{|ES(s_i) \ominus ES(s_{i-1})|}{|ES(s_i) \cup ES(s_{i-1})|}}{|S| - 1} \\ &= 1 - \frac{\frac{4}{13} + \frac{4}{13} + \frac{0}{13}}{3} \\ &= 0.7949 \end{aligned} \tag{25}$$

5.3.9. EVT computations

The EVT measures the time taken to assign the edge sets in the network state, which determines the delay observed in the network. The time taken to update the edge pairs of a host is assumed to be the number of updated edges. Then, the EVT calculation for the example SDN is as shown in equation below.

$$\begin{aligned} EVT &= 1 - \frac{\sum_{i=1}^{|S|} et(s_i)}{max(et(s_i)) \times (|S| - 1)} \forall s_i \in S \\ &= 1 - \frac{1}{1 \times 3} + \frac{1}{1 \times 3} + \frac{0}{1 \times 3} \\ &= 0.3333 \end{aligned} \tag{26}$$

Although there are only a small number of edge changes, the maximum edge change for any state is one. Hence, the EVT value for the example SDN is relatively low. If the EVT value converges to zero, then the maximum delay for changing the edge set is observed for all network states.

5.3.10. Compound securitymetrics

Security decision makers may have different security requirements and objectives, in which the interested metrics may change. Our proposed metrics are modules, which can be used in any combinations. In general, Eq. (27) shows the calculation of the compound security metrics using the proposed security metric modules. Here, W is a set of weight values with $w_i \in W$, and M is a set of metric modules to be computed with $m_i \in M$, where $|M| = |W|$ and the weight values add up to one (i.e., $\sum w_i = 1$).

$$output_{M,W} = \sum_{i=1}^{|M|} w_i \times m_i \forall m_i \in M, w_i \in W \tag{27}$$

For example, we wish to compute the effectiveness of the changes made in the example SDN with respect to the attack cost of exploitation (i.e., ACE) taking into account the cost of variant assignment cost (i.e., NVC) and the edge variation cost (i.e., EVC). Then, the compound security metric for the scenario can be calculated as shown in Eq. (28), assuming the weight values are equal for each metric (i.e., $M_{eg}^1 =$

$\{ACE, NVC, EVC\}$ and $W^1_{eg} = \left\{\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right\}$).

$$\begin{aligned} output1_{M^1_{eg}, W^1_{eg}} &= \sum_{i=1}^{|M_{eg}|} w_i \times m_i \\ &= 1/3 \times ACE + 1/3 \times NVC + 1/3 \times EVC \\ &= 0.9831/3 + 0.9048/3 + 0.7949/3 \\ &= 0.8943 \end{aligned} \quad (28)$$

The given network state scenario for the example achieved 0.9831 taking into account the attack cost of exploitation ACE, and taking into account the defense efforts NVC and EVC. On the other hand, if other network states are generated with a higher cost (i.e., NVC and EVC values were less), then the compound metric value will consequently be lower.

For comparison, we now compute the effectiveness of the changes made in the example SDN with respect to the attack path variation (i.e., APV) taking into account the cost of variant downtime (i.e., NVDT) and the edge variation downtime (i.e., EVT). Then, the compound security metric for the scenario can be calculated as shown in Eq. (29), similarly assuming the weight values are equal for each metric (i.e., $M^2_{eg} = \{APV, NVDT, EVT\}$ and $W^2_{eg} = \left\{\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right\}$).

$$\begin{aligned} output2_{M^2_{eg}, W^2_{eg}} &= \sum_{i=1}^{|M_{eg}|} w_i \times m_i \\ &= 1/3 \times APV + 1/3 \times NVDT + 1/3 \times EVT \\ &= 0.6667/3 + 0.7917/3 + 0.3333/3 \\ &= 0.5972 \end{aligned} \quad (29)$$

The two examples above using the compound metrics clearly show that taking into account the different security objectives of using MTD techniques (i.e., different security goals) can result in different effectiveness observed.

In a similar way, other metrics can be combined to produce a compound security metrics, as well as to assigning different weight values to highlight certain metrics. One approach to assigning weight values to different dynamic security metrics is to collect information regarding the operational requirements to evaluate the importance of different services of the network. For example, the network used by a hospital requires high confidentiality and integrity. In order to achieve this using MTD techniques, higher weight values are assigned to the dynamic security metrics measuring the *Reconnaissance* (e.g., path variation and number, exposure, and cost). On the other hand, a network used by a media firm requires high availability and less concerned with the overall security. Then, higher weight can be assigned to minimize the *Communication* subcategory of the defense efforts to ensure the operational requirements. Hence, we can formulate weight templates for different network systems with different security objectives. However, optimizing the weight value is out of scope in this paper. We explore the changes to the new metrics proposed and the trade-off between the attack and defense efforts in Section 6.

5.4. *Dynamic metric quantification*

In this section, we describe methods to quantify metric values for MTD techniques. In general, we need to specify three categories of characteristics, which are: (1) network, (2) defense techniques (e.g., MTD techniques), and (3) attacker capabilities.

5.4.1. *Network-based characteristics*

The network-based characteristics can be quantified based on the system information (e.g., vendor provided performance charts, protocol specifications etc). For the attack efforts, many of the *Reconnaissance* subcategories can be quantified using the network-based characteristics, as this stage of an attack gathers the system information. Path-based metrics can be computed using various security models (Kordy et al., 2014) using the system configuration as the input. Similarly, the exposure of the network components can be quantified from the system descriptions. For the defense efforts, All the categories can be quantified using the network-based characteristics in conjunction with the properties of the defense techniques, as described in the next subsection.

5.4.2. *Defense technique characteristics*

Quantifying the defense technique characteristics varies due to the diverse and embedded randomness nature of the current defense techniques. For the case of MTD techniques, they can be deployed onto various layers of the systems and networks for mitigating different threats. Defense technique characteristics only affect the defense efforts, and all MTD techniques change the network configurations in some ways. Hence, they can be quantified in terms of the system resources. When an MTD technique is deployed, we can quantify the associated efforts by identifying which subcategories are affected. For example, if we deploy a host-to-host path variation (i.e., a shuffle technique), then this would mostly affect the *Service* and *Communication* subcategories of the defense efforts. We can measure the defense efforts based on the specification of the path variation settings (e.g., overhead traffic load (in bytes, also in percentage), such as a new header payload, for deploying this MTD technique). But if we deploy an operating system diversity (i.e., a diversity technique), then this affects mostly the *Node* subcategory (e.g., downtime (in seconds) when changing the OS, cost (in dollars) of purchasing the OS, cost (in dollars) of required hardware to support the OS etc).

5.4.3. *Attack capability characteristics*

The most difficult characteristic to quantify is the unknown nature of the attackers (e.g., attacker's knowledge, tools available, the time resource available etc). However, we can still utilize some of the existing data collected to specify certain metrics based on empirical studies (e.g., vulnerability scoring systems, McQueen et al., 2009; National Institute of Standards and Technology, 2018; Zhang et al., 2014). We can also adopt zero-day vulnerability analysis approaches (Ingols et al., 2009; Wang et al., 2014) to specify the attacker capabilities. However, this is out of scope in this paper.

6. Comparative analysis

We conducted an experimental analysis via simulations to assess the effectiveness of MTD techniques and compare them. This analysis also demonstrates the functionalities of the proposed metrics based on attack and defense efforts (as

shown in Section 3). To generalize our proposed approach, we conduct experimental analysis using a generic network that randomly connects the hosts, which includes any possible network configurations. By doing so, the subset of practical network configurations is included in the analysis. In Section 6.1, we describe the simulation setup including the system and threat models used for the comparative analysis, and the chosen MTD techniques. Then, in section 6.2, we present the comparative analysis results using the proposed dynamic security metrics based on simulations.

6.1. Simulation setup

6.1.1. System model

To compare different MTD techniques, we set up a generic dynamic network that could implement various MTD techniques (i.e., host and network configurations can be dynamically updated). For example, in an SDN cloud (Azodolmolky et al., 2013; Banikazemi et al., 2013), you can virtualize multiple VMs with software variants enabled, as well as control the data flow using the SDN functions. There are three network variables: (1) the number of hosts, (2) the number of software variants, and (3) the number of network states. The initial connections between hosts are randomly assigned with a density value $0 > d > 1$ (i.e., a host is connected to d^*N number of other hosts). The system configuration is then changed based on the MTD techniques. The software variants specify different types of vulnerabilities on the host, while providing an equivalent service (e.g., a database host using an Oracle SQL server can be replaced with an MS SQL server with appropriate interface implementation, MS Office can be replaced with Libre Office for document edition, etc). The number of network states is governed by the number of iterations made by the MTD techniques.

6.1.2. Threat model

There are many MTD techniques designed for various threats. For the purpose of demonstrating the MTD technique comparison, we scope the threat specific to a privilege escalation based on a STRIDE threat model (LeBlanc and Howard, 2002). We further assume that the attacker is located outside the network, and aims to compromise a specific target host inside the network. The attacker must carry out a reconnaissance and execute exploitations in a sequence in order to compromise the target host. We also assume that if the chain of privilege escalation is broken (e.g., a shuffle technique redirecting service routes), then the attacker loses the privilege gained back to the last reachable host in the chain.

6.1.3. MTD techniques

For the above attack scenario, two representative MTD techniques are considered for the comparative analysis; (1) topology reconfiguration using the shuffle-based MTD technique (Hong et al., 2017) (*shuffle-based* for short), and (2) assigning software variants (Cox et al., 2006) to hosts using the diversity-based MTD technique similar to the one shown in Newell et al. (2013) (*diversity-based* for short), but focusing on the security perspective instead of satisfying the routing goal described in the paper. Both techniques are designed to thwart attacks that are penetrating into the networks via privilege escalations.

To have an in-depth view of the chosen MTD techniques and their effectiveness when network configurations change, we investigate how those attack and defense efforts change with respect to the numbers of hosts, software variants and network states generated. However, other network properties (e.g., protocols, IP addresses, port number *etc*) can be modeled in the T-HARM to capture different types of attack and defense efforts as well. We considered the following defense scenarios: (1) no MTD techniques are used (base case), (2) shuffle-based MTD technique only, (3) diversity-based MTD technique only, and (4) combination of shuffle and diversity. For the case of (4), shuffle was applied to the network topology first and then diversified software on hosts. There are different MTD techniques that can be combined together, but their effectiveness may vary. Our previous work in Hong and Kim (2016) and Alavizadeh et al. (2018, 2017) provides approaches to model combinations of different MTD techniques. However, optimizing the combinations of MTD techniques for maximizing security is out of scope in this paper.

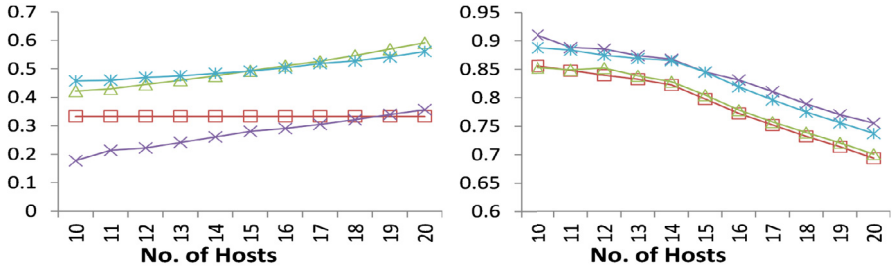
The exploitability value for each software variant is randomly assigned uniformly between values 0.1 and 1 inclusive. The downtime, variant assignment cost, edge update time are all assigned with the value of one unit. These values can be populated from empirical studies or other statistical data to be more accurate, but our analysis did not take those into account as we are focusing on comparing different MTD techniques when such parameters are changed. The following sections present results with respect to the number of hosts, available software variants and the number of network states. For attack efforts, a metric value toward one is making the attack more difficult, and for defense efforts, a metric value toward zero is making the defense more difficult, vice versa. For the overall combination of attack and defense efforts, a metric value towards one is beneficial to the defender (i.e., attack efforts increased while defense efforts decreased).

6.2. Simulation results

6.2.1. Varying the number of hosts

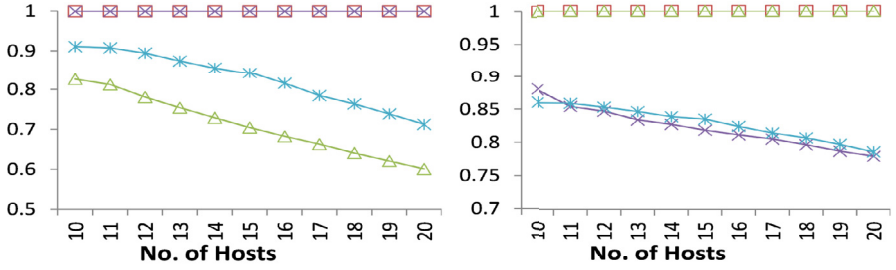
As the number of hosts increases, the management of the network becomes more complex in order to satisfy various constraints (e.g., performance, security). Similarly, understanding the attack and defense efforts is also difficult without examining and collecting security changes made. We examine how the two chosen MTD techniques change the attack and defense efforts when the number of hosts changes. Fig. 4 shows the result. For this analysis, we fix the number of software variants to be three, and the number of network states to be five. Also, the weight distribution among selected metrics is equal (i.e., each dynamic security metric is equivalently important).

Fig. 4a shows the attacker's reconnaissance-related metrics (i.e., APV, APN and APE). As the number of hosts increases, all MTD techniques increase the attack efforts. This is because a large proportion of the attack paths is changed using MTD techniques. However, shuffling the network topology with a small number of hosts may have a negative impact as it may increase the number of attack paths. As expected, not deploying any MTD techniques is not effective in increasing the attack efforts, but it keeps the number of attack paths constant, which does not increase the APN module. Moreover,



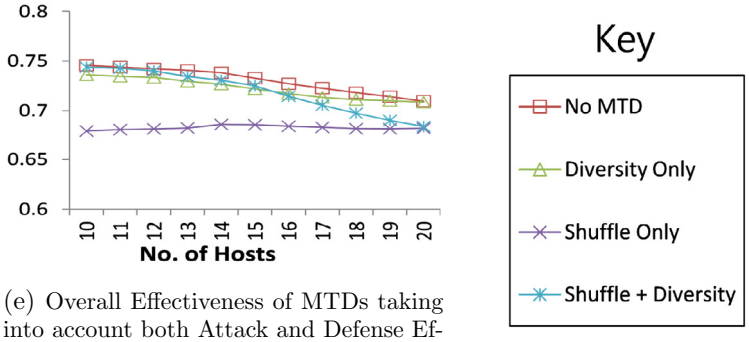
(a) Measuring the Reconnaissance Attack Efforts subcategory using a compound security metric with APV, APN and APE

(b) Measuring the Resource Attack Efforts subcategory using a compound security metric with ACE and ACD



(c) Measuring the Node Defense Efforts subcategory using a compound security metric with NVC and NVDT

(d) Measuring the Communication Defense Efforts subcategory using a compound security metric with EVC and EVT



(e) Overall Effectiveness of MTDs taking into account both Attack and Defense Efforts

Fig. 4 – Comparative analysis of Shuffle and Diversity MTD techniques with respect to the changing number of Hosts.

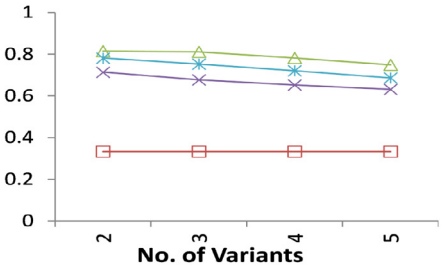
combining software diversity with topology shuffle does not necessarily benefit increasing the attack efforts with respect to the attack paths.

Fig. 4b shows the attacker's resource-related metrics (i.e., ACE and ACD). It clearly shows that as the number of hosts increases, the normalized attack cost decreases. With a larger number of hosts, the number of possible attack paths as well as the attack surface is likely to increase, which decreases the attack efforts in a sense that it provides more options for attacks. It also shows that the shuffle-based approach is slightly better than the diversity-based approach.

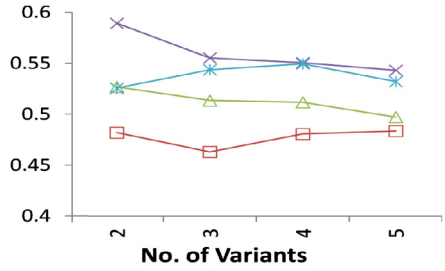
Fig. 4c shows the defense efforts related to the network nodes (i.e., NVC and NVDT). There is no network node related costs for No MTD and shuffle-based MTD technique scenarios as they do not modify the network nodes. As the number of hosts increases, the metric value decreases for diversity-based MTD techniques. This is expected as more hosts would require new software variant assignments to satisfy the MTD condition.

Fig. 4d shows the defense efforts related to the network edges (i.e., EVC and EVT). Similarly with network nodes, diversity-based MTD technique scenarios do not modify the network edges resulting in metric values of one. Shuffle-based MTD techniques show a similar trend, which increases the defense efforts (i.e., the metric values decrease) as the number of hosts increases. There are more edges needed to be changed to satisfy the MTD condition.

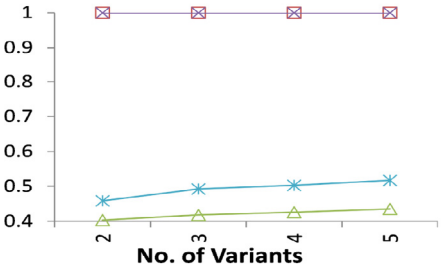
Finally, Fig. 4e shows the overall effectiveness of the two chosen MTD techniques and their combinations with respect to the changing number of hosts. The result shows that not deploying any MTD technique is the best solution, because there are no defense efforts needed for this approach. However, if we increase the weight value of the attack efforts, then not deploying any MTD technique quickly becomes the worst



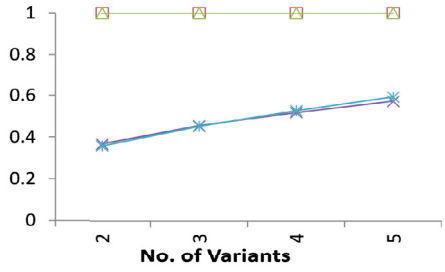
(a) Measuring the Reconnaissance Attack Efforts subcategory using a compound security metric with APV, APN and APE



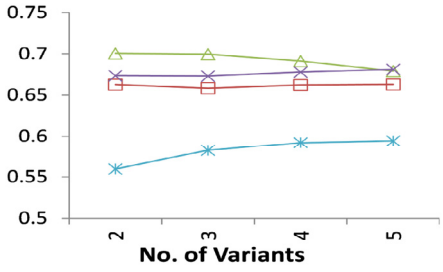
(b) Measuring the Resource Attack Efforts subcategory using a compound security metric with ACE and ACD



(c) Measuring the Node Defense Efforts subcategory using a compound security metric with NVC and NVDT



(d) Measuring the Communication Defense Efforts subcategory using a compound security metric with EVC and EVT



(e) Overall Effectiveness of MTDs taking into account both Attack and Defense Efforts

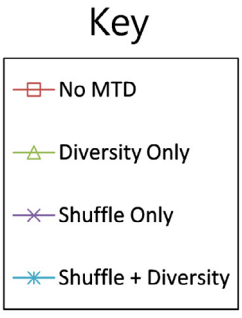


Fig. 5 – Comparative analysis of shuffle and diversity MTD techniques with respect to the changing number of software variants.

scenario. This is also apparent when we compute the effort to security gain ratio. Finding a better distribution of the weight values between the individual metric modules is needed to take into account different attack and defense efforts proportionally, but this optimization is out of scope in this paper. Taking into account only the MTD techniques, it shows that the diversity-based MTD technique would be most suitable when the network has the same number of software variants and network states, but the number of hosts increases.

6.2.2. *Varying the number of software variants*

The number of available software variants is a significant factor, as if the number of variants equals to the number of hosts, no such diversity-based MTD techniques are necessary. However, it is infeasible in practice, and the network only has a limited number of variants to assign with. We examine how the two chosen MTD techniques cope with the number of software variants changes. Fig. 5 shows the result, given the numbers of hosts and network states are fixed with values 20 and 5, respectively.

Fig. 5a shows the attack reconnaissance-related metrics as the number of variants is increased. It shows that increasing the number of available variants does not affect significantly to the defense scenarios. The software diversity provided the highest attack efforts, followed by the combination of the shuffle and diversity. Without MTD techniques deployed, there are no changes to the attack paths information.

Fig. 5b shows the attack resource-related metrics. There are slight variations due to the exploitability of variants being assigned randomly. However, the trend is still observed. Shuffle-only or diversity-only scenarios gradually decrease. This implies that those techniques are less effective as the number of variants increases. However, their combinations

may increase attack efforts if the deployment configuration can be optimized. However, it is out of scope in this paper.

Fig. 5c and Fig. 5d show the defense efforts. They show that as the number of variants increases, the metric values increase as well. Since there are more variants already in the network to work with, both shuffle and diversity-based MTD techniques change less number of network components to satisfy the MTD goal (i.e., to continuously change the attack surface). As the number of variants converges to the number of hosts, the defense effort metric tends toward value one.

Finally, Fig. 5e shows the overall effectiveness of the two chosen MTD techniques and their combinations with respect to the changing number of software variants. In contrast to the changing number of hosts, both shuffle and diversity-based MTD techniques are better than no MTD techniques deployed, even with equal weights to the defense efforts. On the other hand, the combined shuffle and diversity MTD technique performed poorly, because it does not benefit when the number of variants is increased, while defense efforts are increased as both shuffle and diversity-based MTD techniques are implemented.

6.2.3. *Number of network states*

MTD techniques can generate multiple network states that hold certain security properties at different times. For this experiment, we vary the number of network states generated by the chosen MTD techniques (while no MTD technique is static). We examine the effectiveness of MTD techniques as the number of network states generated grows large. In practice, there are tens of thousands of network states. But for comparing our chosen MTD techniques, we did not need so many observations to identify the changing attack and defense effort trends. For more complex MTD techniques, more network states generated can be observed to identify the changing attack and defense efforts using the proposed approach. Fig. 6 shows the result, given the numbers of hosts and software variants are fixed with values 20 and 3, respectively.

Fig. 6a shows the attack reconnaissance-related metrics. It shows that increasing the number of network states does not have a large impact on the path-based attack efforts. Specifically, the software diversity increases the APE value when the number of network states increases, but at the same time the APV value decreases. APE is increased as software variants are assigned to decrease the exploitability, but APV is decreased as the diversity does not significantly change the set of attack paths. Similarly, the topology shuffle approach diversifies the network that decreases the exploitability, but this is achieved by shuffling only a small subset of edges in the network that results in decreasing APV value. Hence, within path-based attack efforts, there were trade-offs that balanced the effectiveness of the chosen MTD techniques.

Fig. 6b shows the attack resource-related metrics. Only the shuffle-based MTD techniques benefited as the number of network states increased. In detail, shuffling the network topology increased both ACE and ACD, while software diversity remained consistent. Shuffling the network topology can increase the length of attack paths, while diversifying software cannot. Hence, more network states can increase the ACD value, as well as the ACE by increasing the mean attack path length.

Fig. 6c shows the defense efforts related to the network nodes. Although not very effective, the defense efforts related to diversity are increasing (i.e., metric values are decreasing). On the other hand, Fig. 6d shows the defense efforts related to the network communication that capture the defense efforts related to shuffle. The defense efforts are reduced as the number of network states increased (i.e., metric values are increasing), because a similar proportion of edges is changed which is distributed over multiple network states. However, there is a limit to the number of network states that can be generated which hold MTD conditions required by the shuffle-based MTD techniques, which will set the upper-bound of the defense efforts.

Finally, Fig. 6e shows the overall effectiveness of the two chosen MTD techniques and their combinations with respect to the changing number of network states. Since other defense scenarios other than using the shuffle-based MTD techniques did not show significant changes, only the shuffle-based MTD techniques benefited with an increased number of network states. Even if the weight values are changed, the effect will be similar as there are no other significant changes observed.

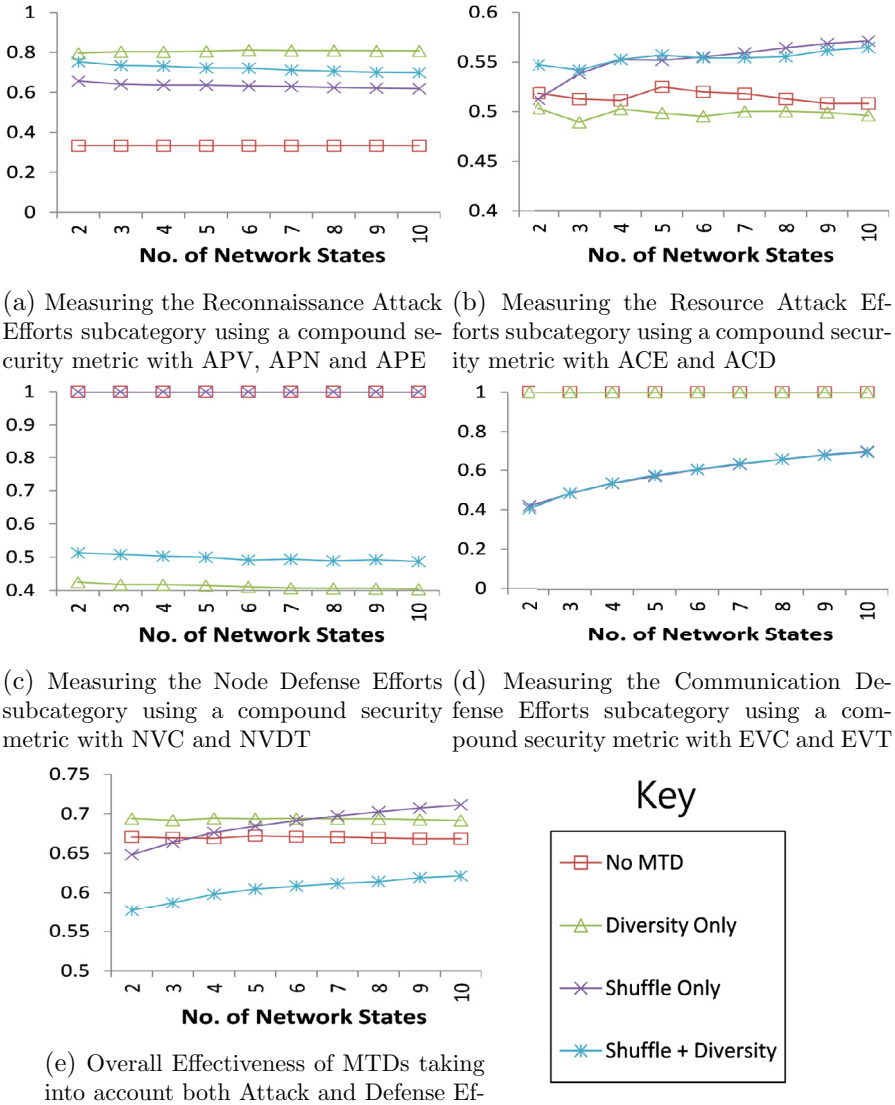
7. Discussion

Although many methodologies and methods are proposed to evaluate the effectiveness of MTD techniques, there are lacks of metrics to capture the details of security impact when the network changes (Pendleton et al., 2016). In this paper, we have categorized attack and defense efforts based on the shift in the attack surface when the network changes. Using the attack and defense efforts, we proposed a new set of security metrics in order to evaluate the effectiveness of MTD techniques. Furthermore, a comparative analysis via simulations demonstrated the usability of our proposed metrics in order to compare MTD techniques, as well as to observe the trade-offs between attack and defense efforts. In this section, we discuss our findings and limitations of our work.

7.1. *Comparing MTD techniques*

We have proposed a new set of security metrics that capture the properties of attack and defense efforts as the network changes over time. We demonstrated this through a comparative analysis via simulations, where the numbers of hosts, variants, and network states are taken into account. The results showed that there is a trade-off between the attack and defense efforts, as increasing the attack efforts also increases the defense efforts (i.e., reducing the security risk increases the security cost). We have observed that in most cases, not using any MTD techniques showed the worst case for the attack efforts even though there are no defense efforts needed (i.e., the benefit of deploying MTD techniques has a better trade-off compared to not deploying any). These results showed that MTD techniques can enhance the security posture of the network.

Depending on what changes are likely to be made to the network, one can maximize the attack efforts while



(a) Measuring the Reconnaissance Attack Efforts subcategory using a compound security metric with APV, APN and APE

(b) Measuring the Resource Attack Efforts subcategory using a compound security metric with ACE and ACD

(c) Measuring the Node Defense Efforts subcategory using a compound security metric with NVC and NVDT

(d) Measuring the Communication Defense Efforts subcategory using a compound security metric with EVC and EVT

(e) Overall Effectiveness of MTDs taking into account both Attack and Defense Efforts

Fig. 6 – Comparative analysis of shuffle and diversity MTD techniques with respect to the changing number of network States.

maintaining the minimum defense efforts made. For example, the shuffle-based MTD technique is more effective than the diversity-based MTD technique when considering multiple network states to be generated, while it is the opposite if the network is expected to be expanding with a growing number of hosts. In our future work, we will consider incorporating more MTD techniques to evaluate their effectiveness to review their characteristics in more details.

7.2. *MTD techniques and threats*

We have mainly focused on the effectiveness of MTD techniques against the privilege escalation-based attacks in our comparative analysis. By focusing on a specific threat, we have classified the properties of attack and defense efforts related to that threat more precisely. However, it also scoped the ability to evaluate various MTD techniques that are designed for mitigating other threats. To address this issue, the effort-based categories also need to be expanded to evaluate other threats. The modular design of the effort-based categories allows flexible additions and updates of the current category model. Hence, we can add effort categories (and subcategories) for other threats as necessary without altering the existing ones. Moreover, some of the existing categories (e.g., cost-related) may be applicable to multiple threats, which can be used to compare various MTD techniques. We will further expand the effort-based categories to include various threats, as well as developing a set of security metrics to measure them in our future work.

7.3. *Security metric modules and weights*

One of the main contributions of this paper is the proposal for a new set of security metrics that can evaluate the

effectiveness of MTD techniques. These metrics can take into account attack and defense efforts as the network state changes over a period of time. By modulating those metrics, it is more convenient to compare, as well as to apply different weighting factors to different components of the compound metrics based on the user's perceived importance of different attack and defense efforts (i.e., the weight assigned to individual dynamic security metrics can be dictated by the user requirements). Some security metrics may not be normalized due to the boundary-less nature (e.g., system risk), but the focus was to observe effort-based metrics. However, incorporating non-effort-based metrics will also be considered in our future work.

Weight values assigned to each modularized metric is important, as it determines how much attack or defense efforts are made with regards to the selected metric. In our experiment, we assigned the same weight values to all metrics, but in practice, they could be different. For example, assigning a variant will have a significant cost to the defense node category, while SDN shuffling will have less cost to the communication category with respect to variant assignment. We will further investigate the practical weight value assignment strategy to further extend our findings in this paper.

7.4. *Parameter measurement*

One of the major issues is to identify which subcategories of the attack and defense efforts are needed in order to characterize an MTD technique, and the relative efforts between them. For example, the value 0.5 computed for the downtime subcategory may be more significant compared to the value 0.8 computed for the service overhead in the defense efforts. To address this issue, we have proposed a compound security metrics to assign different weights (i.e., the importance) between different dynamic security metrics. However, the method to assign those weights is currently done manually. In the future, we will investigate how to systematically assign the importance between the dynamic metrics.

To measure attack and defense efforts, properties of the network and MTD techniques were quantified. Some parameters, however, were more difficult to measure. For example, we can collect the number of attack paths and attack paths variations by evaluating different network states captured, but relying on the impact scores from the vulnerability databases may not fully reflect the practical aspects of exploiting the vulnerability. However, there are various empirical studies as well as many efforts to make the databases more practical, and we expect the accuracy of those data would increase over time.

8. Conclusion

In this paper, we have incorporated MTD techniques into a T-HARM to model changes in the security of the dynamic networks and categorized attack and defense efforts for capturing the effectiveness of MTD techniques. Moreover, a new set of dynamic security metrics has been developed in order to provide comprehensive evaluation methodologies to compare different MTD techniques. Our experimental analysis for two concrete examples of MTD techniques has demonstrated the comparisons of them, where the proposed metrics provided an in-depth detail of their effectiveness in terms of attack and defense efforts.

Acknowledgment

This work was made possible by the support of a grant (NPRP 8-531-1-111) from the Qatar National Research Fund (QNRF). The statements made herein are solely the responsibility of the authors.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.cose.2018.08.003 .

REFERENCES

Akhunzada A, Gani A, Anuar N, Abdelaziz A, Khan M, Hayat A, Khan S. Secure and dependable software defined networks. J Netw Comput Appl 2016;61:199–221. doi:10.1016/j.jnca.2015.11.012.

Al-Wakeel S, Al-Swailemm S. PRSA: a path redundancy based security algorithm for wireless sensor networks. In: Proceedings of IEEE wireless communications and networking conference (WCNC 2007); 2007. p. 4156–60. doi:10.1109/WCNC.2007.759.

Alavizadeh H, Jang-Jaccard J, Kim D. Evaluation for combination of shuffle and diversity on moving target defense strategy for cloud computing. In: Proceedings of 17th IEEE international conference on trust, security and privacy in computing and communications (TrustCom 2018); 2018. p. 1–10. (to appear)

Alavizadeh H, Kim D, Hong J, Jang-Jaccard J. Effective security analysis for combinations of mtd techniques on cloud computing. In: Proceedings of the 14th international conference on information security practice and experience (ISPEC 2017); 2017. p. 539–48. doi:10.1007/978-3-319-72359-4_32.

Azodolmolky S, Wieder P, Yahyapour R. SDN-based cloud computing networking. In: Proceedings of the 15th international conference on transparent optical networks (ICTON 2013); 2013. p. 1–4. doi:10.1109/ICTON.2013.6602678.

Banikazemi M, Olshefski D, Shaikh A, Tracey J, Wang G. Meridian: an SDN platform for cloud network services. IEEE Commun Mag 2013;51(2):120–7. doi:10.1109/MCOM.2013.6461196.

Cox B, Evans D, Filipi A, Rowanhill J, Hu W, Davidson J, Knight J, Nguyen-Tuong A, Hiser J. N-variant systems: a secretless framework for security through diversity. In: Proceedings of USENIX security symposium (USENIX security 2006); 2006. p. 105–20.

Evans D, Nguyen-Tuong A, Knight J. Effectiveness of moving target defenses. In: Jajodia S, Ghosh A, Swarup V, Wang C, Wang X, editors. Moving target defense. Advances in information security, vol. 54. New York, NY: Springer, 2011. doi: 10.1007/978-1-4614-0977-9_2.

Gallon L, Bascou J. Using CVSS in attack graphs. In: Proceedings of the 6th international conference on availability, reliability and security (ARES 2011); 2011. p. 59–66. doi:10.1109/ARES.2011.18.

Gorbenko A, Kharchenko V, Romanovsky A. Using Inherent Service Redundancy and Diversity to Ensure Web Services Dependability. In: Butler M, Jones C, Romanovsky A, Troubitsyna E, editors. Methods, Models and Tools for Fault

Tolerance. Lecture Notes in Computer Science, vol. 5454. Berlin, Heidelberg: Springer, 2009. doi: 10.1007/978-3-642-00867-2_15.
Holgado P, Villagra V, Vazquez L. Real-time multistep attack prediction based on hidden Markov models. IEEE Trans Depend Secure Comput 2017;Early Access(1):1–14. doi:10.1109/TDSC.2017.2751478.
Hong J, Kim D. Assessing the effectiveness of moving target defenses using security models. IEEE Trans Depend Secure Comput 2016;13(2):163–77. doi:10.1109/TDSC.2015.2443790.
Hong J, Yoon S, Lim H, Kim D. Optimal network reconfiguration for software defined networks using shuffle-based online MTD. In: Proceedings of the 36th international symposium on reliable distributed systems (SRDS 2017); 2017. p. 234–43. doi:10.1109/SRDS.2017.32.
Huang Y, Arsenault D, Sood A. Closing cluster attack windows through server redundancy and rotations. In: Proceedings of the 6th IEEE international symposium on cluster computing and the grid (CCGRID 2006); 2006. p. 12–21. doi:10.1109/CCGRID.2006.1630916.
Huang Y, Ghosh AK. Introducing Diversity and Uncertainty to Create Moving Attack Surfaces for Web Services. In: Jajodia S, Ghosh A, Swarup V, Wang C, Wang X, editors. Moving Target Defense. Advances in Information Security, vol 54. New York, NY: Springer, 2011. doi: 10.1007/978-1-4614-0977-9_8.
Ingols K, Chu M, Lippmann R, Webster S, Boyer S. Modeling modern network attacks and countermeasures using attack graphs. In: Proceedings of the 25th annual computer security applications conference (ACSAC 2009); 2009. p. 117–26. doi:10.1109/ACSAC.2009.21.
Jackson T, Salamat B, Homescu A, Manivannan K, Wagner G, Gal A, Brunthaler S, Wimmer C, Franz M. Compiler-Generated Software Diversity. In: Jajodia S, Ghosh A, Swarup V, Wang C, Wang x, editors. Moving Target Defense. Advances in Information Security, vol 54. New York, NY: Springer, 2011. doi: 10.1007/978-1-4614-0977-9_4.
Jafarian J, Al-Shaer E, Duan Q. Openflow random host mutation: transparent moving target defense using software defined networking. In: Proceedings of the 1st workshop on hot topics in software defined networks (HotSDN 2012); 2012. p. 127–32. doi:10.1145/2342441.2342467.
Jafarian J, Al-Shaer E, Duan Q. Spatio-temporal address mutation for proactive cyber agility against sophisticated attackers. In: Proceedings of the 1st ACM workshop on moving target defense (MTD 2014); 2014. p. 69–78. doi:10.1145/2663474.2663483.
Jia Q, Wang H, Fleck D, Li F, Stavrou A, Powell W. Catch me if you can: a cloud-enabled DDoS defense. In: Proceedings of the 44th annual IEEE/IFIP international conference on dependable systems and networks (DSN 2014); 2014. p. 264–75. doi:10.1109/DSN.2014.35.
Kampanakis P, Perros H, Beyene T. SDN-based solutions for moving target defense network protection. In: Proceedings of the 15th IEEE international symposium onworld of wireless, mobile and multimedia networks (WoWMoM 2014); 2014. p. 1–6. doi:10.1109/WoWMoM.2014.6918979.
Kordy B, Pietre-Cambacedes L, Schweitzer P. DAG-based attack and defense modeling: don't miss the forest for the attack trees. Comput Sci Rev 2014;14:1–38. doi:10.1016/j.cosrev.2014.07.001.
Kreidl O, Frazier T. Feedback control applied to survivability: a host-based autonomic defense system. IEEE Trans Reliab 2004;53(1):148–66. doi:10.1109/TR.2004.824833.
Kreutz D, Ramos F, Verissimo P, Rothenberg C, Azodolmolky S, Uhlig S. Software-defined networking: a comprehensive survey. Proc IEEE 2015;103(1):14–76. doi:10.1109/JPROC.2014.2371999.
LeBlanc D, Howard M. Writing secure code. 2nd ed. Pearson Education; 2002.
Lei C, Ma D, Zhang H, Wang L. Moving target network defense effectiveness evaluation based on change-point detection. Hindawi Math Prob Eng 2016;2016:1–11. doi:10.1155/2016/6391502.
Luo Y, Wang B, Cai G. Effectiveness of port hopping as a moving target defense. In: Proceedings of the 7th international conference on security technology (SecTech 2014); 2014. p. 7–10. doi:10.1109/SecTech.2014.9.
MacFarland D, Shue C. The SDN shuffle: creating a moving-target defense using host-based software-defined networking. In: Proceedings of the 2nd ACM workshop on moving target defense (MTD 2015); 2015. p. 37–41. doi:10.1145/2808475.2808485.
Maleki H, Valizadeh S, Koch W, Bestavros A, van Dijk M. Markov modeling of moving target defense games. In: Proceedings of the ACM workshop on moving target defense (MTD 2016); 2016. p. 81–92. doi:10.1145/2995272.2995273.
Manadhata PK. Game Theoretic Approaches to Attack Surface Shifting. In: Jajodia S, Ghosh A, Subrahmanian V, Swarup V, Wang C, Wang x, editors. Moving Target Defense II. Advances in Information Security, vol 100. New York, NY: Springer, 2013. doi: 10.1007/978-1-4614-5416-8_1.
Masoudi R, Ghaffari A. Software defined networks: a survey. J Netw Comput Appl 2016;67:1–25. doi:10.1016/j.jnca.2016.03.016.
McQueen M, McQueen T, Boyer W, Chaffin M. Empirical estimates and observations of 0day vulnerabilities. In: Proceedings of the 42nd Hawaii international conference on system sciences (HICSS 2009); 2009. p. 1–12. doi:10.1109/HICSS.2009.186.
Mell P, Grance T. In: Technical Report. SP 800–145. The NIST definition of cloud computing. Gaithersburg, MD, United States: NIST; 2011.
Miehling E, Rasouli M, Teneketzis D. A POMDP approach to the dynamic defense of large-scale cyber networks. IEEE Trans Inf Forensics Secur 2018;13(10):2490–505. doi:10.1109/TIFS.2018.2819967.
National Institute of Standards and Technology. National vulnerability database. 2018.
Newell A, Obenshain D, Tantillo T, Nita-Rotaru C, Amir Y. Increasing network resiliency by optimally assigning diverse variants to routing nodes. In: Proceedings of the 43rd annual IEEE/IFIP international conference on dependable systems and networks (DSN 2013); 2013. p. 602–14. doi:10.1109/DSN.2013.6575305.
Nguyen-Tuong A, Evans D, Knight J, Cox B, Davidson J. Security through redundant data diversity. In: Proceedings of the 38th annual IEEE/IFIP international conference on dependable systems and networks (DSN 2008); 2008. p. 187–96. doi:10.1109/DSN.2008.4630087.
O'Donnell A, Sethu H. On achieving software diversity for improved network security using distributed coloring algorithms. In: Proceedings of the 11th ACM conference on computer and communications security (CCS 2004); 2004. p. 121–31. doi:10.1145/1030083.1030101.
Okhravi H, Comella A, Robinson E, Yannalfo S, Michaleas P, Haines J. Creating a cyber moving target for critical infrastructure applications. Critical infrastructure protection V. IFIP advances in information and communication technology; 2011. p. 107–23.
Pendleton M, Garcia-Lebron R, Cho J, Xu S. A survey on systems security metrics. ACM Comput Surv 2016;49(4):1–35. doi:10.1145/3005714.
Rohrer J, Jabbar A, Sterbenz J. Path diversification for future internet end-to-end resilience and survivability. Telecommun Syst 2014;56(1):49–67. doi:10.1007/s11235-013-9818-7.

Schiffman M, Eschelbeck G, Ahmad D, Wright A, Romanosky S. CVSS: a common vulnerability scoring system. National Infrastructure Advisory Council (NIAC); 2004.
Schneier B. Attack trees. Dr Dobb's J 1999;24(12):21–9.
Schneier B. Secrets and lies: digital security in a networked world. John Wiley and Sons Inc.; 2000.
Sheyner O, Haines J, Jha S, Lippmann R, Wing J. In: Technical Report. Automated generation and analysis of attack graphs. CMU; 2002.
Taguinod M, Doupe A, Zhao Z, Ahn G. Toward a moving target defense for web applications. In: Proceedings of the IEEE international conference on information reuse and integration (IRI 2015); 2015. p. 510–17. doi:10.1109/IRI.2015.84.
Vikram S, Yang C, Gu G. NOMAD: towards non-intrusive moving-target defense against web bots. In: Proceedings of the 1st IEEE conference on communications and network security (CNS 2013); 2013. p. 55–63. doi:10.1109/CNS.2013.6682692.
Wang L, Jajodia S, Singhal A, P C, Noel S. k-zero day safety: a network security metric for measuring the risk of unknown vulnerabilities. IEEE Trans Depend Secure Comput 2014;11(1):30–44. doi:10.1109/TDSC.2013.24.
Williams D, Hu W, Davidson J, Hiser J, Knight J, Nguyen-Tuong A. Security through diversity: leveraging virtual machine technology. IEEE Secur Privacy 2009;7(1):26–33. doi:10.1109/MSP.2009.18.
Xu J, Guo P, Zhao M, Erbacher R, Zhu M, Liu P. Comparing different moving target defense techniques. In: Proceedings of the 1st ACM workshop on moving target defense (MTD 2014); 2014. p. 97–107. doi:10.1145/2663474.2663486.
Yusuf S, Ge M, Hong J, Kim H, Kim P, Kim D. Security modelling and analysis of dynamic enterprise networks. In: Proceedings of the IEEE international conference on computer and information technology (CIT 2016); 2016. p. 249–56. doi:10.1109/CIT.2016.88.
Zaffarano K, Taylor J, Hamilton S. A quantitative framework for moving target defense effectiveness evaluation. In: Proceedings of the second ACM workshop on moving target defense (MTD 2015); 2015. p. 3–10. doi:10.1145/2808475.2808476.
Zhang S, Zhang X, Ou X. After we knew it: empirical study and modeling of cost-effectiveness of exploiting prevalent known vulnerabilities across IaaS cloud. In: Proceedings of the 9th ACM symposium on information, computer and communications security (ASIA CCS 2014); 2014. p. 317–28. doi:10.1145/2590296.2590300.
Zhang Y, Li M, Bai K, Yu M, Zang W. Incentive Compatible Moving Target Defense against VM-Colocation Attacks in Clouds. In: Gritzalis D, Furnell S, Theoharidou M, editors. Information Security and Privacy Research. SEC 2012. IFIP Advances in Information and Communication Technology, vol 376. Berlin, Heidelberg: Springer, 2012. doi: 10.1007/978-3-642-30436-1_32.
Zhu Q., Başar T.. Game-theoretic approach to feedback-driven multi-stage moving target defense; Springer International Publishing. p. 246–263. 10.1007/978-3-319-02786-9_15
Zhuang R, Zhang S, Bardas A, DeLoach S, Ou X, Singhal A. Investigating the application of moving target defenses to network security. In: Proceedings of the 6th international symposium on resilient control systems (ISRCS 2013); 2013. p. 162–9. doi:10.1109/ISRCS.2013.6623770.
Zhuang R, Zhang S, DeLoach S, Ou X, Singhal A. Simulation-based approaches to studying effectiveness of moving-target network defense. In: Proceedings of national symposium on moving target research; 2012. p. 1–12.
Zonouz S, Khurana H, Sanders W, Yardley T. RRE: a game-theoretic intrusion response and recovery engine. In: Proceedings of the 39th IEEE/IFIP international conference on dependable systems networks (DSN 2009); 2009. p. 439–48. doi:10.1109/DSN.2009.5270307.

Jin B. Hong is a Lecturer in the Department of Computer Science and Software Engineering at the University of Western Australia, Australia. He received his Ph.D. degree in Computer Science from the University of Canterbury, New Zealand. His research interests include security modeling and analysis of computer and networks such as cloud computing, SDN and IoT, and Moving Target Defense. He is a member of the IEEE.

Simon Enoch Yusuf is a staff of the Department of Computer Science, Federal University Kashere, Gombe, Nigeria. He received M.Sc. degree in Computer Science from the University of Ibadan, Nigeria. He is currently a Ph.D. student at the University of Canterbury, New Zealand under the supervision of Dr. Dong Seong Kim. His research interests include security metrics and graphical security modeling.

Dong Seong Kim is currently a Senior Lecturer in Cybersecurity in the Department of Computer Science and Software Engineering at the University of Canterbury, New Zealand. His research interests include dependability, cybersecurity and resilience modeling and analysis. He is a senior member of the IEEE.

Armstrong Nhlabatsi is a Post-doctoral researcher at Qatar University, Qatar. He received his Ph.D. degree in Computer Science from the Open University, United Kingdom. His research interests include security requirements engineering, security risk evaluation, requirements traceability, and the feature interaction problem for information security. He previously worked at the University of Swaziland as a lecturer in programming techniques and digital electronics.

Noora Fetais is the director of KINDI Computing Research Centre at Qatar University. She received her Ph.D. in computer engineering from the University of Sussex. She is currently holding various positions such as Vice Chair of IEEE-Qatar Section, Qatar Ambassador of Women in Data Science (WiDS) at Stanford University, among others. She was the first Women to chair the Faculty Senate of Qatar University. She is a member of the IEEE.

Khaled M. Khan is an Associate Professor in the department of Computer Science and Engineering and the Manager of KINDI Computing Research Lab at Qatar University. Prior to these, he served Western Sydney University (Australia) as a Senior Lecturer, and was the Head of postgraduate programs. He received his Ph.D. in computing from Monash University, and BS and MS degrees both in computer science from the Norwegian University of Science and Technology. He is a member of IEEE.