Frontiers of Information Technology & Electronic Engineering
www.zju.edu.cn/jzus; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Review:

Moving target defense: state of the art and characteristics*

Gui-lin CAI†, Bao-sheng WANG, Wei HU, Tian-zuo WANG

(*College of Computer, National University of Defense Technology, Changsha 410073, China*)

†E-mail: cc_cai@163.com

Received June 11, 2016; Revision accepted Aug. 14, 2016; Crosschecked Oct. 9, 2016

Abstract: Moving target defense (MTD) has emerged as one of the game-changing themes to alter the asymmetric situation between attacks and defenses in cyber-security. Numerous related works involving several facets of MTD have been published. However, comprehensive analyses and research on MTD are still absent. In this paper, we present a survey on MTD technologies to scientifically and systematically introduce, categorize, and summarize the existing research works in this field. First, a new security model is introduced to describe the changes in the traditional defense paradigm and security model caused by the introduction of MTD. A function-and-movement model is provided to give a panoramic overview on different perspectives for understanding the existing MTD research works. Then a systematic interpretation of published literature is presented to describe the state of the art of the three main areas in the MTD field, namely, MTD theory, MTD strategy, and MTD evaluation. Specifically, in the area of MTD strategy, the common characteristics shared by the MTD strategies to improve system security and effectiveness are identified and extrapolated. Thereafter, the methods to implement these characteristics are concluded. Moreover, the MTD strategies are classified into three types according to their specific goals, and the necessary and sufficient conditions of each type to create effective MTD strategies are then summarized, which are typically one or more of the aforementioned characteristics. Finally, we provide a number of observations for the future direction in this field, which can be helpful for subsequent researchers.

Key words: Moving target defense, Security model, Function-and-movement model, Characteristics
http://dx.doi.org/10.1631/FITEE.1601321 **CLC number:** TP393

1 Introduction

With the rapid growth of information technology, the Internet has become a national key infrastructure that ensures the normal function and operation of many important areas, such as transportation, economy, and energy. Furthermore, the Internet has been profoundly changing our work habits and daily life, and influencing the normal running of society. Therefore, network security, especially Internet security, has become one of the most pressing problems for governments, enterprises, and network users. Nevertheless, the cyber-attacks, such as IP prefix hijacking (Liu *et al.*, 2014), botnet (Wang *et al.*, 2012), and distributed denial-of-service (DDoS) attack (Zhang *et al.*, 2011), can still be found everywhere. Also, the major security incidents have been frequently reported in recent years, such as PRISM (http://www.zdnet.com/article/prism-heres-how-the-nsa-wiretapped-the-internet/), Heartbleed bug (http://heartbleed.com/), and eBay data leakage. Such security disasters are repeatedly showing that the Internet security is challenged all the time.

The root source of the severe network security problem is the asymmetric situation between attacks and defenses. First, the attackers have the advantage of time, because they can perform vulnerability analysis and penetration testing for a specific target repeatedly until they achieve the final goal. Second, the attackers have an asymmetric advantage of

* Project supported by the National Basic Research Program (973) of China (No. 2012CB315906)
ORCID: Gui-lin CAI, http://orcid.org/0000-0002-9322-2539

information, because the attackers can initiate and launch an attack as long as there is a usable vulnerability, while the defenders have to secure all the potential vulnerabilities and prevent all attacking means that can be used by the attackers. Third, the attackers have the advantage of cost to expand the attack, because the homogeneity in network configurations enables the attackers to carry out a large-scale attack easily and at a low cost once a small-scale attack succeeds. The network configurations nowadays are typically deterministic, static, and homogeneous (NITRD, 2010; http://www.whitehouse.gov/sites/default/files/microsites/ostp/fed_cybersecurity_rd_strategic_plan_2011.pdf). The deterministic and static nature endows the attackers with the advantages of time and information, and the homogeneity endows the attackers with the advantage of cost. In other words, these features reduce the difficulties faced by cyber attackers in scanning the network, identifying specific targets, and gathering essential information. This gives the attackers the advantages of building up, launching, and spreading attacks. For this reason, in the combat between cyber network attack and defense, the attackers typically have asymmetric advantages and the defenders are at the disadvantaged position by being passive. Unfortunately, traditional defense mechanisms and approaches before the moving target defense (MTD) era could (and can) barely do anything to change this situation (Jajodia *et al.*, 2011; 2013).

MTD (NITRD, 2009) is a novel way to reverse this asymmetric situation between attacks and defenses. It keeps moving the attack surface of a protected system through dynamic shifting, which can be controlled and managed by the administrator. In this way, the attack surface exposed to attackers appears chaotic and changes over time. Therefore, the work effort, i.e., the cost and the complexity, for the attackers to launch a successful attack, will be greatly increased. As a result, the probability of successful attacks will be decreased, and the resiliency and security of a protected system will be enhanced effectively. It is important to note that MTD is not a specific approach, but an active defense principle. It can be applied to different system attributes, such as IP address, service port number, protocol, and running platform, which leads to a variety of MTD mechanisms. For example, if MTD is applied to the IP address, then a variety of IP address mutation approaches come into being; when it is applied to the running platform, a variety of dynamic platform techniques come into being. It can also be applied to existing security evaluation or defense approaches to improve the effect of that approach. For example, Rahman *et al.* (2014) applied it to the process of the power system state estimation to harden the security and increase the correctness of the measurement.

Currently, a lot of related works are available. These studies involve several facets of MTD, including theory, strategy, and evaluation. Okhravi *et al.* (2013) have analyzed some strategies by focusing on identifying the overhead, cost, and weakness of these MTDs qualitatively. The work is meaningful, but the strategies they covered are not comprehensive and do not involve the other facets in the MTD field. Comprehensive analyses and research on this field are still absent, and we attempt to fill the gap in this study. We focus on the scope and area of MTD, provide fundamental insights, and propose future directions. Existing literature is systematically categorized and analyzed to identify the function, essence, and characteristics of MTD (in consideration of space, we describe only the MTDs proposed after the notion of MTD being viewed as a security theme to reverse the asymmetric situation between attacks and defenses). Several new perspectives and elucidations on MTD are rendered.

The reminder of this paper is organized as follows. Section 2 proposes a new security model and a function-and-movement model. The new security model is used to describe the changes in the traditional security model caused by the introduction of MTD. The function-and-movement model provides a common framework for understanding the state of the art in the field of MTD. Furthermore, in the function-and-movement model, the related works are divided into three main research areas based on their research content, the MTD theory, MTD strategy, and MTD evaluation. Then the three main research areas are presented respectively. Section 3 presents the theory on studying the common principles to create an effective MTD strategy. Section 4 presents the three main schools of thought of MTD strategies, which use their own methodologies to produce a large number of MTD strategies to defend the protected target. The common characteristics shared by these strategies are extracted. A summary of these

strategies is presented in this section to provide the necessary and sufficient conditions to create an effective MTD. Section 5 presents existing models and approaches for evaluating MTD strategies. After the systematic review, several future research directions in this field are highlighted in Section 6, and the whole article is concluded in Section 7.

2 Models for moving target defense

2.1 A new security model

System and network administrators are currently in a reactive state of patching and upgrading to secure vulnerable systems (NITRD, 2010), and the general defense process of traditional defense mechanisms and approaches conforms to the policy, protection, detection, response, and recovery (PPDRR) model shown in Fig. 1 (Liu *et al.*, 2011).

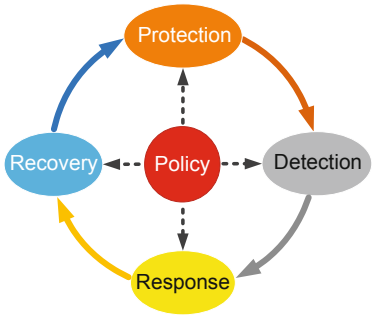

Fig. 1 The policy, protection, detection, response, and recovery (PPDRR) model

In the traditional PPDRR security model, policy is the core of a defense system. The implementation of all the protection, detection, response, and recovery processes is based on a policy. Protection is usually achieved through traditional static security technologies, including firewall, cryptography, and authentication. Detection can be used to discover new threats and vulnerabilities, which is the basis of response. Response is the most important link in the security cycle and the most effective way to solve the potential threat. Recovery is the last link in a security cycle. The system would be restored to its pristine state or a more secure state than its past state after recovery. Under the guidance of policy, protection, detection, response, and recovery constitute a complete and dynamic security cycle.

MTD is an active defense technique because it keeps changing one or more attributes automatically in a way to increase the work effort needed for attacking (NITRD, 2009). This active ability of an MTD system is independent of the state of the environment it resides on. To be more effective and practical, an MTD system should also be equipped with the reactive ability, which responds to an anomalous event observed or perceived (Carvalho *et al.*, 2012). Currently, some existing mechanisms have been designed with active and reactive abilities simultaneously, such as ChameleonSoft (Azab *et al.*, 2011) and moving attack surface (MAS) (Huang and Ghosh, 2011). In other words, the operation mode of MTDs is no longer consistent with the traditional PPDRR security model; i.e., the introduction of MTD changes the general defense process of traditional defense mechanisms and approaches, and produces a new security model accordingly. The new security model that incorporates MTD is shown in Fig. 2. This model includes both the processes of active defense and reactive defense. In the active mode, the defense process is independent of the network status and shifts the attack surface periodically or erratically; thus, it does not need the detection and recovery links. In the reactive mode, the defense process is triggered by security alerts, and thus it conforms to the PPDRR model. Moreover, the ratio of active defense is x, while the ratio of reactive defense is $(1-x)$, where $0.5 < x \leq 1$ is used to express that MTD is mainly an active technique. The value of x is determined by the defender/administrator as a security-cost trade-off.

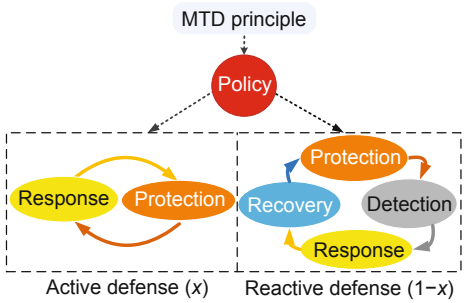

Fig. 2 New security model with moving target defense (MTD)

2.2 Function-and-movement model

In this subsection, we present a generalized model for the existing MTD research as a three-dimensional model (Fig. 3). The model shows

a traditional networking system, which has normal functionality and the ability of MTD. The functionality is expressed by a function model and the ability of MTD is expressed by a movement model.

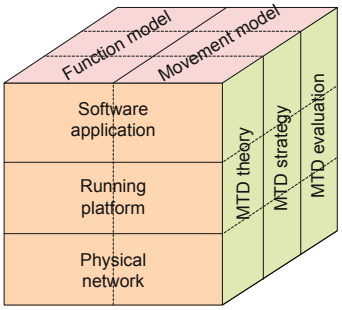

Fig. 3 Function-and-movement model

MTD was introduced as one of the game-changing themes of network security defense for the first time in 2009, and thereafter, a multitude of related works on how to create MTD emerged rapidly. These works involve three main facets in this field, and we call them MTD theory, MTD strategy, and MTD evaluation, respectively. All the three areas in this field share the common goal of creating an effective MTD, which can increase the work effort for attackers, limit the exposure of vulnerabilities and opportunities for attacks, and increase the resiliency and security of the protected system. MTD theory attempts to find the answers to some fundamental questions, such as how to create an effective MTD system (Hobson *et al.*, 2014; Zhuang *et al.*, 2014b) and what capabilities and features an MTD system should have (Carvalho *et al.*, 2013; Green *et al.*, 2015). These studies can help researchers better understand what MTD is and provide some basic design principles for the MTD strategy. MTD strategy aims at designing different moving mechanisms for the target system to enhance its security and resiliency. It is the core of MTD technology to provide the expected defense. Currently, more attention has been paid to this area in the MTD field. MTD evaluation provides appropriate models and approaches to measure the defense effect and the cost of different MTD mechanisms. Evaluation must be done once the design is completed to provide reference for the improvement of theory and strategy. In this sense, the three facets are necessary for an effective MTD.

A conventional networking system can be roughly divided into three layers, respectively termed software application, running platform, and connections to the physical network. In addition, the running platform includes the hardware platform and operation system. For a static node, all the three layers are vulnerable to attack. To improve the resiliency and security of the networking system, we can equip it with the capability of MTD.

MTD is distinguished from the traditional reactive defense by the fact that it can move one or more system attributes continually. The ability of MTD can be implemented in one of the three layers (software, running platform, and physical network) or more. Simultaneously, the research of movement involves all the three main areas (MTD theory, MTD strategy, and MTD evaluation). Each facet may involve one or more layers that are necessary to compose a networking system. For each networking system, it has its own normal functionality, such as communicating or providing web service. The achievement of normal functionality needs the support of the three layers that are necessary to form a networking system. Collectively, the function-and-movement model makes up an MTD system.

3 Moving target defense theory

According to the research content, the research on MTD can be classified into three areas, which are MTD theory, MTD strategy, and MTD evaluation (Fig. 3). In this section, we summarize the studies in the area of MTD theory.

The concept of MTD has been proposed for many years; however, 'what is MTD' and 'how to create an effective MTD' are still unclear. Researchers have been trying to give answers to these questions (Carvalho *et al.*, 2013; Carvalho and Ford, 2014; Hobson *et al.*, 2014; Zhuang *et al.*, 2014b), yet still have some arguments on these research. In this study, the term 'MTD theory' aims at describing the common design principles as well as the capabilities and features that an MTD system should have to provide guidance for creating an MTD.

We introduce the concepts of attack surface and attack surface shifting because the research on MTD is incomplete without talking about the notions of attack surface and attack surface shifting. Actually, the concept of attack surface had been proposed before MTD was proposed as a game-changing theme, and it is usually used as an important indicator to

measure the security of software (Manadhata and Wing, 2011a). After the introduction of MTD, attack surface shifting is considered an effective way to achieve MTD (Clark *et al.*, 2013; Manadhata, 2013; Crouse *et al.*, 2015), and some researchers have attempted to find a common way for shifting the attack surface to provide the optimal MTD using the game theory (Manadhata, 2013; Zhu and Başar, 2013). We now introduce the existing definitions for attack surface and attack surface shifting.

In fact, there is no standard definition for what is meant by attack surface (Zhuang *et al.*, 2014b), and existing definitions of attack surface are associated with the scenarios considered by the researchers. Manadhata and Wing (2011b) treated the attack surface as the subset of the system's resources (i.e., methods, channels, and data) used in attacks on the system. Zhuang *et al.* (2012) considered that the attack surface consists of the system resources exposed to attackers (e.g., the software residing on the hosts, the ports open for the communication between hosts, and vulnerabilities in the various components) as well as compromised network resources that can be used to further penetrate the system. Zhu and Başar (2013) defined the attack surface of a system as the set of vulnerabilities exhibited by the system, which can potentially be exploited by the attacker. Peng *et al.* (2014) defined the attack surface of an active virtual machine (VM) instance, which is used to deploy the target service, as the totality of its externally accessible resources.

Huang and Ghosh (2011) described the process of shifting the attack surface to illustrate the moving attack surface of the target using the graphic form. However, they did not define the process formally. Manadhata (2013) defined the notion of attack surface shifting as follows: Given a system, s, its environment, E, s's old attack surface, R_{o}, and s's new attack surface, R_{n}, s's attack surface has shifted if there exists at least one resource, r, such that (1) $r \in (R_{\mathrm{o}} \setminus R_{\mathrm{n}})$ or (2) $r \in (R_{\mathrm{o}} \cap R_{\mathrm{n}}) \wedge (r_{\mathrm{o}} \succ r_{\mathrm{n}})$. The meaning of $r \in (R_{\mathrm{o}} \setminus R_{\mathrm{n}})$ is that resource r belongs to R_{o} but no longer belongs to R_{n}. Therefore, the attack that is on the basis of r would not work anymore on R_{n}. Let r_{o} denote r's contribution to R_{o}, and r_{n} denote r's contribution to R_{n}. The meaning of $r_{\mathrm{o}} \succ r_{\mathrm{n}}$ is that resource r makes a larger contribution to the attack surface R_{o} than to the attack surface R_{n}, and $r \in (R_{\mathrm{o}} \cap R_{\mathrm{n}}) \wedge (r_{\mathrm{o}} \succ r_{\mathrm{n}})$ means that resource r still belongs to R_{n}, but makes a smaller contribution to the attack surface R_{n}. However, the measurement of a resource's contribution to the attack surface is a new question.

From existing research, we know that the attack surface is composed of multiple parameters, each of which is equipped with a set of values. Attack surface shifting means that at least one parameter (or its value) is replaced. For simplicity and without loss of generality, we treat the attack surface of a system as the set of the system's properties that can be used for attack, and it consists of the vulnerabilities (including the software and hardware vulnerabilities), IP address, and port number. We treat the IP address as a part of the attack surface for the reason that it is the premise of the attacker to exploit and analyze vulnerabilities and finally launch an attack. Similarly, the port number is a premise for some attacks, such as DoS/DDoS. Accordingly, we can treat attack surface shifting as replacing a software/hardware entity with certain vulnerabilities, or replacing the value(s) of the IP address and(or) the port number.

Next, let us focus on the common principles for MTD design and the capabilities and features that an MTD system should have.

An initial theory of MTD has been proposed, which consists of the MTD systems theory (Zhuang *et al.*, 2014b) and the attacker theory (Zhuang *et al.*, 2015). In MTD system theory, Zhuang *et al.* (2014b) put forward numerous basic definitions and summarized three essential problems that must be addressed to carry out the formal process of an MTD system: first, how to select the next configuration state of the MTD system; second, how to select an adaptation to carry out to reach the next configuration state, which can be seen as a strategy; third, when to carry out the adaptation to actually change the state of the MTD system. In the attacker theory, Zhuang *et al.* (2015) also defined key concepts that support a precise discussion of attacker knowledge, attack types, and attack instances, to help researchers understand the interaction between MTD systems and the attacks they want to thwart. In addition, they proposed some MTD design principles and a basic design schema of MTD for computer networks, to improve the resiliency of the system under attack (Zhuang *et al.*, 2012), which can integrate with another MTD mechanism called self-shielding dynamic

network architecture (SDNA) (Yackoski *et al.*, 2011).

Hobson *et al.* (2014) have concluded that there are three primary challenges in developing MTD defenses: coverage, unpredictability, and timeliness. Coverage can be simply defined as the extent in which all elements of a defended attack surface are subject to movement. Unpredictability reflects the amount of the key information (about the coverage and the rule of movement) obtained or guessed by the attacker. Timeliness in a movement is required to ensure that it happens at the correct time with respect to the attacker. In other words, an effective MTD technique should address the next three issues: the right pieces to be moved, a large enough space for the movement to take place in, and the correct time for the movement to take place. In addition, the investigations of Hobson *et al.* (2014) and Zhuang *et al.* (2014b) are interlinked. The goal of configuration selection is to improve the unpredictability of the protected system, and the selection involves the coverage.

From our own perspective, we believe that the essence of MTD is not only 'changing' but also 'moving', which means that the vulnerabilities and attacked attributes of the protected target remain the same before and after deploying the MTD technique, but the static nature of the target is broken. In other words, the vulnerabilities and attacked attributes can move continuously and try to be one step ahead the attacker to achieve the defense goals. From the existing research (NITRD, 2009; Hobson *et al.*, 2014; Zhuang *et al.*, 2014b), we conclude that to achieve the goal, there are three elements for an MTD technique: WHAT to move, HOW to move, and WHEN to move.

1. 'WHAT to move' refers to the moving parameter. As mentioned earlier, the attack surface of a system consists of one or more parameters, which can be the one or more attributes of the protected target that is essential for an attack, such as the IP address and the service port; or the running entities, such as the operating system (OS), hardware, and software, on which the vulnerabilities that can be attacked by the attacker reside. For each moving parameter, there is a domain of values for selection. All the domains of the parameters form the configuration space.

2. 'HOW to move' refers to the way to move. It implies two operations, selection and replacement. Selection means choosing a new parameter with its value or assigning a new value to the previous moving parameter(s) from its(their) domain(s) through various ways, such as choosing randomly (Roeder and Schneider, 2010; Azab *et al.*, 2011; Jafarian *et al.*, 2012), choosing according to the game theory (Manadhata, 2013; Zhu and Başar, 2013; Carter *et al.*, 2014), and choosing according to the attack behavior observed or the situation of cyber-security (Azab *et al.*, 2011; Huang and Ghosh, 2011; Jia *et al.*, 2013). Replacement is using the selected new parameter with its value or the selected new value to replace the old one. Currently, the general method is only choosing a new value to replace the old one while maintaining the moving parameter.

3. 'WHEN to move' refers to the time series defined by the defender to replace the current value of the moving parameter, i.e., the frequency of moving. It is a critical problem that can influence the performance (or even availability) of the protected target. If the frequency is too low, while the attacker is fast enough, there is most likely a successful attack. On the other hand, if the frequency is too high, although it can provide a high degree of security, it would introduce high overhead and reduce the system performance and availability of the services. A perfect solution should be with high frequency when there is an attack and should be unchanged when there are no attacks. Now, it is usually achieved by presetting either fixed or varying intervals, or triggering by an anomalous event.

Besides the works mentioned above, there are some other related works on MTD theory.

Carvalho and Ford (2014) described the background of MTD and some basic important issues in MTD (such as building resiliency with MTDs, as well as the promises and challenges for moving ahead). Furthermore, they analyzed the requirements for command and control mechanisms that implement the logic for the system mobility and adaptive response to failures and attacks (Carvalho *et al.*, 2012), and proposed a command and control framework based on the principles of human-agent teamwork for the practical deployment of MTD (Carvalho *et al.*, 2013). Moreover, Beraud *et al.* (2010; 2011) developed a prototype cyber command and control (C2) system called network maneuver commander (NMC), to improve the network resiliency in a compromised cyber environment. The prototype

focuses on maneuvering network-based elements preemptively on the basis of the threat information and the predictions based on the historical data. The elements maneuvered include path, port, hardware platform, application with data and context, protocol, link, OS, address, etc. The NMC implies the design of a hybrid MTD and is meaningful for subsequent researchers. However, it is mainly the prototype of the decision framework, and further related work is needed to make it practical.

Crosby *et al.* (2013) believed that the design of an MTD mobility mechanism must consider the network dependencies, i.e., the information used in an attack. Thus, the following three aspects should be focused on: first, identifying the dependencies that an attacker has on protocols, services, and applications; second, breaking the dependencies; third, designing supplement mechanisms to mitigate the broken dependencies for legitimate users. In addition, Green *et al.* (2015) identified and defined seven properties common to network-based MTDs (NMTDs), which are keys to ensure the effectiveness of the MTDs. Our previous work (Cai *et al.*, 2016) characterized the running patterns of MTD mechanisms to help us understand the running behaviors of MTDs in a better and easier way.

Torrieri *et al.* (2013) identified the challenges and research issues arising from jamming and other attacks by external sources and insiders, and then proposed a general framework to deal with such issues based on the notion of cyber maneuver. However, further research is required on the designing of each component as well as the integration of these components seamlessly. Taguinod *et al.* (2015) explored the feasibility of applying MTD concepts to web applications. They analyzed the web application stack first to find where and how MTD can be applied. Then they proposed two diversification approaches, changing the language implementation or the database implementation while retaining functionality, to achieve their goals.

These studies are very useful for understanding the MTDs and can provide guidance to developers for designing new MTD systems. To be more intuitive, we summarize the viewpoints and/or contributions of those reviewed studies in the area of MTD theory in Table 1.

4 Moving target defense strategy

An MTD is achieved by designing various strategies for the selected moving parameter(s) to make it/them move under the guidance of the MTD theory to interrupt the reconnaissance from attackers at an appropriate time continually, thus enhancing the resiliency and security of the protected target. In this section, we summarize the studies in the area of MTD strategy. We first classify the existing MTD strategies into three categories and present detailed interpretations of their running modes. Then some common characteristics shared by these strategies are extracted, and the ways to provide the characteristics are concluded. Finally, a summary about the necessary and sufficient conditions for an effective MTD strategy is presented.

4.1 Categories of moving target defense strategies

A great number of MTD strategies have been presented. According to the moving parameter (i.e., 'WHAT to move'), they can be categorized into three main categories, i.e., software transformations (Jajodia *et al.*, 2011), dynamic platform techniques (Okhravi *et al.*, 2014a), and network address shuffling (Carroll *et al.*, 2014). The major difference among them is the manner of moving (i.e., 'HOW to move').

4.1.1 Software transformations

The MTD approaches based on software transformations choose the software/application as the moving parameter, and they usually apply diversity transformations to transform the protected software application, thus generating more variants that provide the same function but with different behaviors and features in several ways. Then the variants are shuffled under a strategy (including 'HOW to move' and 'WHEN to move') to make the target uncertain and unpredictable for the attackers, which makes it hard for the attackers to achieve their malicious intent, thereby increasing the resiliency of the software.

In ChameleonSoft (Azab *et al.*, 2011), large missions of a huge software program are divided into small tasks, and each task is assigned to one or more cells that would manually or automatically generate several variants for the task. The variants have different objectives targeting different

Table 1 A brief summary of the studies mentioned in the area of MTD theory

Reference	Viewpoint and/or contribution
[a]Manadhata and Wing (2011b)	The system's resources (i.e., methods, channels, and data) used in attacks
[a]Zhuang *et al.* (2012)	The system resources exposed to attackers (e.g., software, ports) as well as the compromised network resources
[a]Zhu and Başar (2013)	The set of vulnerabilities exhibited by the system
[a]Peng *et al.* (2014)	The totality of the target system's externally accessible resources
[b]Huang and Ghosh (2011)	Taking a virtual server pool with diversity as an example to graphically show the meaning and process of attack surface shifting
[b]Manadhata (2013)	Defining the notion of attack surface shifting graphically and formally, in which the resources' contribution to the attack surface or the resources' shifting is crucial
[c]Zhuang *et al.* (2014b)	(1) Putting forward a vast number of basic definitions to represent what is MTD; (2) summarizing three essential problems that must be addressed for MTD: selection for the next configuration, selection for an adaptation to carry out to reach the next configuration state, and when to carry out the adaptation
[c]Zhuang *et al.* (2015)	Defining key concepts that support a precise discussion of attacker knowledge, attack types, and attack instances, to help researchers understand the interaction between MTD systems and the attacks
[c]Zhuang *et al.* (2012)	Providing a preliminary design of a network MTD system, which can be a good example for the subsequent researchers
[c]Hobson *et al.* (2014)	Summarizing three primary challenges for developing MTD defenses: coverage, unpredictability, and timeliness
[c]Carvalho and Ford (2014)	Describing the background of MTD, and some basic important issues in MTD (e.g., the promises and challenges for moving ahead)
[c]Carvalho *et al.* (2012; 2013)	Analyzing the requirements for command and control (C2) mechanisms for MTD, and designing a resilient human-agent teamwork C2 prototype to provide a way to integrate MTD into the control loop
[c]Beraud *et al.* (2010; 2011)	Developing a research prototype cyber C2 system called network maneuver commander (NMC), which focuses on maneuvering network-based elements (including path, port, protocol, OS, hardware platform, etc.) to improve the resiliency. It is a good example for subsequent researchers
[c]Crosby *et al.* (2013)	Proposing that the design of an MTD must consider the network dependencies. Moreover, three aspects should be focused on: identifying the dependencies of an attacker, breaking the dependencies, and designing supplement mechanisms to mitigate the influence introduced by breaking the dependencies for the user
[c]Green *et al.* (2015)	Identifying seven properties common to network-based MTDs (NMTDs), namely, unpredictability, vastness, periodicity, uniqueness, availability, revocability, and distinguishability. Moreover, not all the properties are included in an NMTD
[c]Cai *et al.* (2016)	Characterizing the running patterns of MTD mechanisms. The two main fundamental patterns are 'hidden' and 'variation', and the assisted pattern is called 'improvement'
[c]Torrieri *et al.* (2013)	Identifying the challenges and research issues arising from jamming and other attacks, and proposing a general framework to deal with such issues
[c]Taguinod *et al.* (2015)	Exploring the feasibility and providing the way of applying MTD concepts to web applications, which can be considered as a use case for subsequent researchers

[a] About definition of attack surface; [b] about definition of attack surface shifting; [c] about basic theory and design principles

quality attributes, such as reliability, performance, robustness, and mobility. Therefore, there are multiple variants with diversity in each cell. In addition, the variants can shuffle by randomly choosing the next variant, based on a randomly adjusted timer in each cell, to make the software change over time. For the design principle, the situational awareness unit in ChameleonSoft can capture the attacker's scanning and penetration attempts. The variants shuffling at runtime would make it more difficult for an attacker to generate a profile with the possible flaws of the executing variant. Even if an attack succeeds, it can cause only a variant crash, and the recovery mechanism would act rapidly to replace the compromised one with another variant. Therefore, ChameleonSoft can enhance the software's ability against attacks. For the implementation, ChameleonSoft requires the software to be divided into small tasks and to be assigned to cells to generate variants. Then the variants are shuf-

fled at runtime; however, the coupling among the variants would be a problem, in addition to the high complexity.

In compiler-generated software diversity (Jackson *et al.*, 2011), the compiler automatically creates multiple variants that have the same in-specification behavior, but different out-of-specification behaviors when it is translating a high-level source code to a low-level machine code. These variants are created using multiple variation techniques, such as instruction set randomization (ISR), register randomization, no-operation (NOP) insertion, and stack base randomization. In addition, this mechanism presents two orthogonal compiler-based techniques to all users. One is massive-scale software diversity (MSSD), which is suitable for common home/office users. A diversification engine (a 'multicompiler') can be deployed at App Store to make sure that every user gets his/her own diversified software variant. Therefore, the attacker has no knowledge about the internal structure of that software and cannot construct an attack. The other one is a multi-variant execution environment (MVEE), which is suitable for users who have higher security requirements. In an MVEE, multiple variants run at the same time, and the input to the system is simultaneously fed to all variants. A monitoring layer examines their behaviors to determine whether there is an attack, and thus detecting exploitation of vulnerabilities at runtime before the attacker has the opportunity to compromise the system. In this manner, both MSSD and MVEE can make it harder for an attacker to run a successful attack. In addition, one way to achieve compiler-generated software diversity is by inserting non-alignment NOP insertions randomly (Jackson *et al.*, 2013). The MSSD approach can increase the difficulty for a large-scale attack. However, if a single user is attacked, the approach cannot help mitigate the attack. The MVEE approach can interrupt an attack when detected, and thus mitigates the influence of the attack effectively. However, the user needs to prepare multiple variants initially, which increases the cost and inconvenience.

In end-to-end software diversity (Christodorescu *et al.*, 2011), the static component in this mechanism uses a semantics-preserving method to rewrite the application to diversify the entities referenced in subprograms. Once the static component completes its run, the rewritten application would call the runtime component to generate a unique diversification strategy (such as address-space layout randomization and structured query language (SQL) diversification), which is applied to each rewritten subprogram. Actually, every software system has a multitude of aspects that are open to diversification; hence, it can select different aspects of the application to transform and generate multiple variants. Furthermore, the diversification transformations can be repeatedly applied to the program in this approach; from the view of time, the software changes over time. This approach is designed for all applications, in particular, Internet-facing applications; hence, it plays an important and significant role. However, the implementation of this approach would influence the development, deployment, and operation of the applications; in other words, its actual deployment cost is high.

In practical software diversification (Pappas *et al.*, 2013), a novel in-place code randomization method has been presented to help third-party applications against return-oriented programming (ROP) attacks. Through randomly choosing and applying different transformations, such as atomic instruction substitution, instruction reordering, and register reassignment, to each instance of the protected application, multiple variants of the application can be created. In other words, a new variant is randomly chosen and it replaces the current one at random intervals. From the view of time, the software changes over time. This approach can be directly applied to third-party executables to raise the bar of attacks with little overhead introduced. However, the objective of this approach is to eliminate or probabilistically modify as many of the gadgets as possible, and thus it can provide only probabilistic protection against ROP attacks. Therefore, this software diversification technique should cooperate with other existing techniques to improve its defense effect.

Symbiotic embedded machine (SEM) (Cui and Stolfo, 2011) is a new poly-culture architecture that provides complete uniqueness for each distinct device and would inject into the protected program and reside within its host executable to thwart many remote attacks. When an SEM is created and prepared for injection into a host program, both the SEM and the protected program are analyzed, randomized, and mutated into a unique instantiation, which is functionally equivalent to the original code.

Some existing techniques such as ISR, address space randomization (ASR), and polymorphic mutation are used to increase the randomness and diversity at the same time. Moreover, one or more SEMs can be injected into a piece of arbitrary executable code. Thus, multiple variants of the SEM and the host program are created, and the protected program changes over time. SEM provides a new paradigm to protect device drivers, kernel, and userland applications. It can be applied to legacy devices that have been widely deployed; thus, the application of SEM can be widespread. However, the performance of the protected target would be affected because SEM should extract computational resources to execute its own payloads.

Proactive obfuscation (Roeder and Schneider, 2010) uses semantics-preserving code transformations to create multiple server replicas and periodically restarts the server with a fresh version randomly chosen. These multiple server replicas are likely to have fewer shared vulnerabilities; i.e., they are diverse executables. The periodic restarting strategy used in this mechanism would help bound the number of comprised replicas, thus making it harder for an adversary to attack. However, it would introduce additional overhead for the creation and management of these replicas.

In the helix metamorphic shield (HMS) approach (le Goues *et al.*, 2013), the spatio-temporal diversity engine uses dynamic instruction set randomization on the variant selected in the previous generation (the first generation consists of the original input software only) to re-randomize binaries and create multiple variants to form the current generation and shuffle. Therefore, the multiple variants are diverse in each generation and between generations. The rate of re-randomization is preset as a constant value. In addition, when an attack is detected, the GenProg engine uses evolutionary algorithms to create and vet candidate repair patches to repair both security-critical and non-security critical vulnerabilities of the variants. In HMS, the interaction between variant shuffling and vulnerability repairing would generate software variants with growing resistance to attacks. However, the running of this approach would introduce high costs for calculation, detection, and repair.

In NOMAD (Vikram *et al.*, 2013), the name/ID parameter values are randomized to create multiple variants. Each name/ID corresponds only to one of the input HyperText Markup Language (HTML) elements (such as textbox, checkbox, and submit button) in an HTML form. Currently, the name/ID parameters in an HTML element are generally designed as constant and can be used by web bots to automatically fabricate massive requests. In NOMAD, each name/ID parameter used by users to submit data can be randomized; thus, the values of the name/ID parameters are diverse. The name/ID parameter is part of the software, and thus randomizing its value is equivalent to transforming the software. This approach is used as a complement for existing solutions that defend against web bots, by randomizing HTML elements to prevent web bots from identifying these elements and then fabricating and sending bulk automated requests with customized content. In addition, it can add some decoy elements to increase the entropy of randomization space and decrease the likelihood of success for brute force attacks. However, it adds the operations of randomization and de-randomization in the normal process of service, and thus the server's response speed would be affected. Furthermore, if it introduces the decoy elements, the user is required to identify them, which would affect the availability of the service.

In adaptive just-in-time (JIT) code diversification (Jangda *et al.*, 2015), the code is recompiled frequently by a Java bytecode JIT compiler during the execution of the program to generate multiple diversified variants for the target program. Its goal is to reduce the time frame between the two critical timestamps: time of leakage of useful address space information and time of use of the leaked information in an exploit. The approach to achieve this goal is using adaptive NOP insertion, which is performed by a diversification component in the recompilation component. Through this way, it can create diversity and generate a different variant of the binary at these timestamps. In addition, the diversification interval is adaptive. From the time perspective, the software changes over time. This approach significantly reduces the time frame for the attacker to gain useful address space information and subsequently uses the information in memory exploits. The approach is built on profile information generation and needs to identify the hot/cold blocks and then insert NOPs in cold blocks as much as possible. This modifies the behaviors of the compiler and introduces some

overhead.

4.1.2 Dynamic platform techniques

Dynamic platform techniques (DPTs) dynamically change the properties of a computing platform in order to complicate attacks. In other words, they choose the execution environment as the moving parameter, including running platform and configuration. Platform properties refer to hardware and OS attributes such as instruction set architecture (ISA), stack direction, calling convention, kernel version, OS distribution, and machine instance (Okhravi *et al.*, 2014b). Similar to the mechanisms in the category of software transformations, the mechanisms in the category of dynamic platform techniques are equipped with multiple instances, specifically, multiple execution environments or multiple configurations for the same execution environment (John *et al.*, 2014; Lucas *et al.*, 2014). In the case of multiple execution environments, there are two sub-cases; i.e., each execution environment is with a different configuration for the selected properties (Huang and Ghosh, 2011; Okhravi *et al.*, 2011a; Peng *et al.*, 2014; Thompson *et al.*, 2014; Debroy *et al.*, 2016), or they are with the same configuration (Bangalore and Sood, 2009). Subsequently, the instances shuffle under a defined strategy (including 'HOW to move' and 'WHEN to move').

In TALENT (Okhravi *et al.*, 2011a; 2011b; 2012), there are multiple heterogeneous physical hardware platforms and OSs for the running critical application to migrate. OS-level virtualization is used to migrate the environment of the critical application across different platforms. The environment includes the filesystem, open files, and network connections. A portable checkpoint compiler is also used in TALENT to preserve the state of the running application and provide application migration, thus achieving both transparency and scalability. Moreover, the migration is conducted as a result of a security alert or a periodic migration. In TALENT, migration among the diverse platforms would increase the resiliency of the critical application. However, when taking the deployment cost into consideration, the size of the set of the heterogeneous physical hardware platforms and OSs would not be large. Furthermore, the vulnerabilities of each platform are relatively fixed. Therefore, if the attacker is targeting the platform but not the critical application, the performance of the application would still be decreased.

MAS (Huang and Ghosh, 2011) can be viewed as an extension of self-cleansing intrusion tolerance (SCIT) (Bangalore and Sood, 2009). Both in SCIT and MAS, virtualization technology is used to create multiple virtual servers (VSs). The differences between them are twofold. One is that the VSs in SCIT possess the same properties, whereas each VS in MAS is configured with a unique software mix and thus diversity is produced. The other is that the offline VSs in SCIT are rotated to be online for providing service on a regular fixed time interval, while offline VSs in MAS are rotated to replace the current online VSs on a fixed interval or an event-driven basis. In addition, both in SCIT and MAS, the offline servers would be restored to their pristine states. Both in SCIT and MAS, the defense provided by the rotation would increase the survivability of the web server. However, there are multiple VSs, and the rotation interval is small; thus, the resource redundancy is quite high and the management overhead is large. Furthermore, rotation has certain regularity, and it may be broken by the attacker.

In the MTD strategy for cloud-based services (Peng *et al.*, 2014), virtualization technology is also used to create multiple VMs to deploy a service in the cloud, and enough diversity between configurations is introduced to ensure the effectiveness of MTD when the VMs are created. The service can be migrated among active VMs (the VM instances on which the service is currently deployed). At any given moment, some VM instances are active while others are inactive, and they can be converted into each other. The migration is under the following strategy: during each unit of discrete time, an active VM instance makes a decision of whether to migrate to one of its replacements to minimize the unnecessary migration through a probabilistic and thresholded strategy. The replacements of a VM instance are a subset of the whole VM pool, which are both diverse (in configuration) and similar (within technical feasibility). The proposed strategy is risk-aware, which can help improve the security. The suitable condition(s) for this strategy is/are that the cloud-based service is dense or/and the attacker is strong. When the service is sparse while the attacker is weak, the strategy does not provide significant advantage over a static service (i.e., without the MTD strategy)

in terms of resiliency.

In the approach of evolving computer configuration (John *et al.*, 2014; Lucas *et al.*, 2014), evolution-inspired techniques (such as genetic algorithms) are used to create multiple functional and secure configurations based on the previous configurations. The crossover operation ensures that the configurations of the new generation are different from the old ones, and that the mutation operation is carried out to maintain diversity across the configurations of the current generation. All the configurations generated would be assessed by an assessment component to determine their security levels. The new and more secured configurations would be periodically implemented to change the system's attack surface. This approach would render the system's attack surface variable, in addition to increasing the system's capacity for defending against attacks. Hence, it can effectively make it harder for the attacker to attack. However, the cost of this approach would be high for the following two reasons. One is that it needs to create VMs and make them alive to deploy new configurations for testing their feasibilities, performances, and security levels. The other reason is that it needs to collect and update the information about any security event for assessing the new generated configurations. Furthermore, its application scope is limited, and currently it is suitable only for RedHat® installed Apache™[1] web servers.

Multiple operating system rotation environment (MORE) MTD (Thompson *et al.*, 2014) aims to offer improved security through platform diversity and frequent OS rotation. In MORE, there are several VMs equipped with different distributions of Linux. A periodic rotation of the various VM hosts is performed to provide dynamic defense, while reducing the likelihood and the impact of a successful exploit and ensuring the application availability during the OS rotations. This environment can make it harder for the attacker to exploit the vulnerabilities of the OS and then initiate the attack. Furthermore, it is based on the existing technology and is easily deployable. However, its current implementation is focused on only OS diversity; if the attacker attempts to attack the platforms that the diverse OSs reside on, this environment would not work.

In the software-defined networking (SDN) based frequency-minimal MTD approach (Debroy *et al.*, 2016), there is a heterogeneous VM pool for the cloud application to migrate to. The VMs are connected with the OpenFlow controller and periodically share their status information (such as residual compute/storage capacity) with the controller. The frequency of the VM migration is adaptive to the statistical DoS attack pattern and probability. The ideal migration location is chosen based on the candidate VM's capacity, available network bandwidth, and VM reputation in terms of attack history. This approach is proposed to counter the loss of availability (LOA) attacks for protecting critical cloud-hosted applications. Furthermore, the approach aims to reduce the cost and resource wastage through minimizing the frequency of migration, which is associated with the statistical DoS attack pattern and probability. Therefore, the effectiveness of this approach depends on the fitness between the attack model considered by the authors and the real attack behaviors. When the attack model that is based on previous statistical data cannot reflect the attack behavior in the near future, this approach may not reduce the cost or may even lose the effectiveness.

4.1.3 Network address shuffling

Network address shuffling is a dynamic reconnaissance defense that periodically permutes the mappings between addresses and devices. In other words, the network address is chosen as the moving parameter. The network addresses are a combination of IP and transport layer information (protocol and port numbers) and either or both types of information can be used for shuffling (Carroll *et al.*, 2014). For each mechanism in the category of network address shuffling, there is a set of candidate network addresses. However, the methods of using the range of addresses (i.e., 'HOW to move') and the decision on 'WHEN to move' are diverse.

In SDNA (Yackoski *et al.*, 2011; 2013a), the SDNA entity within each node can rewrite the addresses of the packets entering and exiting the OS to prevent each guest from knowing the identity of other nodes within the enclave. When a domain name system (DNS) response comes to the guest, the SDNA entity would replace the real IP with a token IP, which is generated by the SDNA entity (the token IPs consist of a set of candidate addresses). When the guest initiates a connection to a token IP, the SDNA entity would rewrite the packets by replacing the token IP with the real IP. In other words, one

side of the communication does not know the other's real address, and the token IP is obtained from the other's SDNA entity when it requests a DNS resolution. SDNA is transparent to OS and is compatible with existing network infrastructure. Moreover, it can cooperate with existing security technologies to enhance the total security. Furthermore, without the cryptographically secure challenge/response from the user to the server via SDNA, the OS would be unable to access the service, which can effectively limit the attacker's ability to attack. However, the traffic between the communication endpoints must flow through one or more intermediate nodes to be rewritten for concealing the endpoints' identities, and it requires multiple key exchanges and authentications in the paths' establishment process; thus, the complexity and cost of implementation are high.

In moving target IPv6 defense (MT6D) (Dunlop *et al.*, 2011), the source and destination network- and transport-layer addresses for both communicating hosts are rotated. At each time increment, MT6D computes the next IIDs (interface identifier in an IPv6 address, which can identify a particular node) for both the sender and receiver of each communicating pair, using a hash function and three parameters: the current IID, a shared symmetric key, and the system timestamp. In this manner, the two sides of the communication can compute their own and the other's next IID. In addition, the time intervals vary for each communicating address pair to increase the security and privacy. MT6D is transparent to the user, and it has no side effect on the normal communication because the obscuration of an address can be made in the middle of ongoing sessions without breaking the connection or requiring a new handshake. It can increase the work effort to attack because the address space of IPv6 is large and the IIDs of a session keep changing. However, MT6D devices must maintain multiple IPv6 addresses for each node at any given time, which may increase storage resource and cause additional cost for routing update.

A series of IP address mutation approaches have been proposed, including OpenFlow random host mutation (OF-RHM) (Jafarian *et al.*, 2012), random host mutation (RHM) (Al-Shaer *et al.*, 2013), and spatio-temporal address mutation (Jafarian *et al.*, 2014).

In OF-RHM (Jafarian *et al.*, 2012), each host is associated with an unused address range (i.e., the set of virtual IPs (vIPs)) that is assigned by the OpenFlow controller based on its specific requirement using satisfiability modulo theories (SMTs). A new vIP is chosen from the range and assigned to the host after each regular mutation interval, and the new vIP is selected by two methods: (1) chosen with uniform probability; (2) a weight is associated with each vIP based on a certain criterion fixed by the administrator, and the selection probability is directly related to the weight. OF-RHM is an IP address mutation approach that should be deployed under the SDN architecture and cannot be deployed in the traditional network. To address this question, RHM has been proposed.

The design principles and implementation of RHM (Al-Shaer *et al.*, 2013) are similar to that of OF-RHM. The main differences relative to OF-RHM are the vIP allocation mechanism and the components for distribution. RHM uses a two-phase mutation approach, which consists of low-frequency mutation (LFM) and high-frequency mutation (HFM), to assign a vIP. An LFM interval contains multiple HFM intervals, and the LFM interval is fixed while the HFM interval is customized based on the required mutation speed of each moving target (MT) host. In each LFM interval, a random network address range denoted as a virtual address range (VAR) is selected for each MT host using SMT. Then in each HFM interval, a random vIP within the VAR assigned during the previous LFM is selected for the MT host, and the selection is based on a hash function.

To further improve the applicability against coordinated attack and the degree of security provided by the mutation approach, Jafarian *et al.* (2014) presented the spatio-temporal address mutation, which incorporates temporal mutation with spatial mutation. In this approach, each host is associated with a unique set of IP addresses, called the ephemeral IP addresses (eIPs, which are similar to the vIP in OF-RHM and RHM), to reach other hosts, and it would be frequently changed after random intervals. The computation of a new eIP is based on the source (requestor) identity (spatial randomization) as well as time (temporal randomization), and the spatial host-IP binding (i.e., eIP) is determined based on two strategies: (1) random mutation (eIP is chosen randomly from the unused address space

based on a uniform distribution); (2) deceptive mutation (the goal is to assign potentially unattractive IPs to potentially attractive targets). In addition, a time-to-live (TTL) value, which is determined based on the identities of both endpoints, is set to indicate the expiration time of the real IP (rIP)-eIP mapping, and each such TTL value is chosen based on a Poisson arrival process with mean λ in this mechanism. Hence, a host must use different eIPs to communicate with another host at various time intervals.

All the three IP address mutation approaches (OF-RHM, RHM, and spatio-temporal address mutation) can keep the real IP addresses of hosts unchanged, as well as associate each host with a short-lived vIP address for communication. This feature can make them transparent for users. Furthermore, a user cannot know the actual address of the other user that he/she communicates with, which can effectively increase the difficulty for the attacker to connect to a specific host. However, the complexity and cost for implementing the three mutation approaches are high because matched gateways must be deployed to perform rIP-vIP translation, and it also incurs a large amount of work for translation, storage, update, and searching of the address information. Furthermore, the gateways can be the new important target that should be protected.

Morphing network assets to restrict adversarial reconnaissance (MORPHINATOR) (http://defense-update.com/tag/morphinator) aims to build a prototype network capable of morphing over time to confuse and thwart potential attackers on military networks. We know that the prototype focuses on the IP address hopping and port hopping techniques to constantly change the characteristics of the network it resides on. However, there is no more detail about it currently.

MOTAG (Jia *et al.*, 2013) was proposed to help online application servers against network flooding attacks. The mechanism requires a group of proxy nodes to be deployed around the server node, and allocates an active working proxy for each certified user to forward data traffic between the user and the server. The mechanism can protect the online server strongly because the address of the server is private for any user; each client is aware of only his/her own working proxy's IP address. When a working proxy is under attack, it would start up a process called client-to-proxy shuffling; i.e., the attacked proxy is replaced by a new proxy, and the associated users are also migrated to the alternative proxy. In other words, the client is passive in accepting the address shuffling of its working proxy and knows nothing about the address of the server. It can also gradually concentrate the attacker to a particular proxy to minimize the impact of an attack. However, it must work together with the attack detection mechanism because there is no shuffling if there are no attacks.

In MTD-MANETs (Albanese *et al.*, 2013), the identity of a legitimate node that presents to other nodes is changed in order to increase the uncertainty for the attacker. Each legitimate node is assigned an ID pool, which includes multiple virtual IDs associated with its real ID. These virtual IDs are generated by a hash chain at runtime and are used for communication, while the real IDs are never publicly used. The validity interval for each virtual ID is randomly selected from a time range defined by the authors. In this mechanism, the virtual identity of a legitimate node, rather than the IP address, changes dynamically. However, as the virtual identity is used for communication between nodes, we still take the moving parameter in it as the IP address. This approach is robust to multiple types of attacks, such as blackhole attack, wormhole attack, and routing message flooding attack. However, it needs to make substantial modifications in the network layer, including introducing a translation service for mapping the virtual IDs to the real IDs, a protocol to propagate the updates of the nodes' virtual identities, and a mechanism that enables the legitimate nodes to securely join the network. As a result, the complexity and cost of implementation are high.

The SDN shuffle approach (MacFarland and Shue, 2015) uses synthetic addressing information to replace the real addressing information for defending against reconnaissance. Each time a client requests a DNS resolution of the server, the DNS server would notice the SDN controller. Then the SDN controller generates a synthetic IP address and a media access control (MAC) address for the application server, sets a short TTL value, and sends them to the DNS server to reply to the client. In addition, the SDN controller orders the server to install network address translation (NAT) rules that translate the synthetic IP and MAC addresses into the real addresses. In

this manner, each client receives synthetic addresses for the server, which can be rotated by the SDN controller for each new DNS resolution. This allows the SDN controller to provide a moving IP address for each client. Specifically, the short TTL set by the SDN controller ensures that the client will re-issue DNS requests for new connections, which allows the server to again change the addresses. As a result, it can defend against reconnaissance effectively. However, it needs to be deployed under the SDN architecture and modify the OS of the DNS server to cooperate with the SDN controller.

In the random port and address hopping (RPAH) approach (Luo *et al.*, 2015), each server host is associated with a VAR, which is the unused address space of the server domain, and each application running on the server host is associated with a virtual port range (VPR), which is the unused port space of the host. After a fixed hopping interval, the server would mutate its IP addresses and communication ports based on a pseudo-random function with shared secret keys, source identity (srcID), service identity (svcID), and time. Therefore, clients must use different vIP:vPort pairs to obtain services in different time intervals, which can effectively defeat various internal and external reconnaissance. However, similar to OF-RHM, RHM, and spatio-temporal address mutation, RPAH must modify the traditional gateways to provide address hopping gateway (AHG), port hopping engine, and port and address hopping gateway (PAHG) for rIP to/from vIP translations, rPort to/from vPort translations, and rIP:rPort to/from vIP:vPort translations, respectively. In addition to the cost for modification, it introduces large overhead for translation, storage, update, and searching of the address information. The gateways can be the new important target to be protected.

4.2 Characteristics of moving target defense strategies

As mentioned earlier, there are three main categories to actively design and develop MTD. Although each of these attempts uses its own methodology for designing various mechanisms, there are some common characteristics that govern the construction of these MTD mechanisms, and we call them multi-candidate, diversity, randomness, and limited timeliness. In addition to the four common characteristics, there is a minor characteristic named 'attack surface reduction'.

4.2.1 Multi-candidate

From the existing research, we know that there is usually a large configuration space for a specific MTD strategy. As mentioned earlier, the configuration space should comprise all the domains of the moving parameters. In other words, it is related to the problem about 'WHAT to move'. Therefore, here the term 'multi-candidate' is not strict or general, and it has different meanings for the MTD strategies in different categories. Moreover, regardless of the category, all the mechanisms are multi-candidate. Each MTD strategy has its own method of generating multi-candidate, which can be initialized at the beginning, or can be generated gradually in the process of defense.

For the mechanisms in the category of software transformations, the multi-candidate usually means the multiple variants of software. In addition, the multi-candidate can be created by four methods, namely, based on specific architecture, compiler, semantics-preserving rewriting, and randomization.

For the mechanisms in the category of DPT, the multi-candidate usually means the multiple execution environments or configurations for shifting. They can be created through three routes, namely, inherence, virtualization technology, and evolution-inspired techniques.

For the mechanisms in the category of network address shuffling, the multi-candidate usually means the multiple values of the IP address or the port for shifting. The multi-candidate can be generated through two routes. One is by assigning a set of candidate IP addresses and/or port numbers to the protected host directly. The other is gradually producing a set of candidate IP addresses and/or port numbers through calculation (e.g., using the current address and a hash function to generate the next address).

For better understanding, we summarize them in Table 2.

4.2.2 Diversity

Diversity takes multi-candidate as the foundation, and it means that there must be much differentiation in terms of properties between these can-

Table 2 Methods to create multi-candidate

Method	Mechanism
[a]Generating based on a specific architecture	ChameleonSoft (Azab *et al.*, 2011)
[a]Compiler-generating	Compiler-generated software diversity (Jackson *et al.*, 2011) and adaptive just-in-time code diversification (Jangda *et al.*, 2015)
[a]Semantics-preserving rewriting	End-to-end software diversity (Christodorescu *et al.*, 2011) and proactive obfuscation (Roeder and Schneider, 2010)
[a]Randomization	Practical software diversification (Pappas *et al.*, 2013), SEM (Cui and Stolfo, 2011), HMS (le Goues *et al.*, 2013), and NOMAD (Vikram *et al.*, 2013)
[b]Inherence	TALENT (Okhravi *et al.*, 2011a), MORE (Thompson *et al.*, 2014), and the SDN-based frequency-minimal MTD approach (Debroy *et al.*, 2016)
[b]Virtualization technology	SCIT (Bangalore and Sood, 2009), MAS (Huang and Ghosh, 2011), and the MTD strategy for cloud-based services (Peng *et al.*, 2014)
[b]Evolution-inspired techniques	The approach of evolving computer configuration (John *et al.*, 2014; Lucas *et al.*, 2014)
[c]Assigning a set of candidate IP addresses and/or port numbers directly	OF-RHM (Jafarian *et al.*, 2012), RHM (Al-Shaer *et al.*, 2013), spatio-temporal address mutation (Jafarian *et al.*, 2014), MOTAG (Jia *et al.*, 2013), and RPAH (Luo *et al.*, 2015)
[c]Producing a set of candidate IP addresses and/or port numbers gradually	SDNA (Yackoski *et al.*, 2011), MT6D (Dunlop *et al.*, 2011), MTD-MANETs (Albanese *et al.*, 2013), and the SDN shuffle approach (MacFarland and Shue, 2015)

[a] Belonging to the category of software transformations; [b] belonging to the category of dynamic platform techniques; [c] belonging to the category of network address shuffling

didates. It is also an important feature of MTD. Intuitively, shifting among the multiple candidates with diversity would make the target more secure than shifting among the multiple candidates with the same properties, because it increases the attacker's exploration surface (Zhuang *et al.*, 2014b) and the difficulty of targeting the correct attack attribute of the system in the right period.

For all the mechanisms in the school of software transformations, diversity is introduced when software variants are generated.

For the mechanisms in the school of dynamic platform techniques, except for SCIT (Bangalore and Sood, 2009), in which each VS is with the same pristine state and the identical properties, and thus there is no diversity, all the other mechanisms exhibit diversity. Moreover, diversity is introduced when the multiple candidates are created.

For all the mechanisms in the category of network address shuffling, regardless of the manner in which the multiple IP addresses and/or port numbers are generated, the multiple IP addresses and/or port numbers must be different from one another; i.e., the set of addresses/port numbers are diverse, to make the mechanisms effective.

Generally, when producing the multi-candidate for each MTD strategy, diversity is usually generated concurrently. Moreover, each MTD strategy has its own way to generate diversity (Table 3).

4.2.3 Randomness

Unpredictability is an important goal of MTD techniques, which can make it hard for the attacker to predict the precise information of the target in the next period, thus increasing the work effort to attack, decreasing the probability of successful attacks, and increasing the target's resiliency. In the achieved MTD mechanisms, unpredictability is usually in the form of randomness, which is the external manifestation of the way to move (i.e., 'HOW to move', and exactly speaking, the way to choose the next candidate). Randomness means that the defender chooses the next configuration pseudo-randomly from the attacker's perspective, which leads to the unpredictability of MTD techniques. Therefore, randomness is a key factor to ensure the effectiveness of MTD strategies. Almost all the MTD strategies have this feature, and the difference among them is the approach to generating the randomness, or more accurately, the way to choose the next candidate. The typical methods are summarized in Table 4.

4.2.4 Limited timeliness

A technique has the feature of limited timeliness if it can change the value of the moving param-

Table 3 Methods to introduce diversity

Method	Mechanism
[a]Targeting different quality attributes, such as reliability, performance, robustness, and mobility	ChameleonSoft (Azab *et al.*, 2011)
[a]Using different variation techniques, such as ISR, register randomization, NOP insertion, and stack base randomization	Compiler-generated software diversity (Jackson *et al.*, 2011)
[a]Diversifying the entities referenced in subprograms	End-to-end software diversity (Christodorescu *et al.*, 2011)
[a]Choosing and applying different transformations, such as atomic instruction substitution, instruction reordering, and register reassignment, to each instance of the protected application	Practical software diversification (Pappas *et al.*, 2013)
[a]Using existing techniques such as ISR, ASR, and polymorphic mutation	SEM (Cui and Stolfo, 2011)
[a]Making the multiple server replicas to have few shared vulnerabilities when created (without details)	Proactive obfuscation (Roeder and Schneider, 2010)
[a]Using dynamic instruction set randomization	HMS (le Goues *et al.*, 2013)
[a]Randomizing each name/ID parameter used by users to submit data	NOMAD (Vikram *et al.*, 2013)
[a]Using adaptive NOP insertion	Adaptive JIT code diversification (Jangda *et al.*, 2015)
[b]There are multiple heterogeneous physical hardware platforms and operating systems inherently	TALENT (Okhravi *et al.*, 2011a)
[b]No diversity	SCIT (Bangalore and Sood, 2009)
[b]Equipping each VS with a unique software mix	MAS (Huang and Ghosh, 2011)
[b]No details	MTD strategy for cloud-based services (Peng *et al.*, 2014)
[b]Using crossover and mutation operations to create and maintain diversity	The approach of evolving computer configuration (John *et al.*, 2014; Lucas *et al.*, 2014)
[b]The VMs are equipped with different distributions of Linux	MORE (Thompson *et al.*, 2014)
[b]There is a heterogeneous VM pool inherently	The SDN-based frequency-minimal MTD approach (Debroy *et al.*, 2016)
[c]Diversity is inherent when a set of candidate IP addresses and/or port numbers is assigned to the host directly	OF-RHM (Jafarian *et al.*, 2012), RHM (Al-Shaer *et al.*, 2013), spatio-temporal address mutation (Jafarian *et al.*, 2014), MOTAG (Jia *et al.*, 2013), and RPAH (Luo *et al.*, 2015)
[c]Diversity is progressively introduced when the set of candidate IP addresses and/or port numbers is produced gradually	SDNA (Yackoski *et al.*, 2011), MT6D (Dunlop *et al.*, 2011), MTD-MANETs (Albanese *et al.*, 2013), and the SDN shuffle approach (MacFarland and Shue, 2015)

[a] Belonging to the category of software transformations; [b] belonging to the category of dynamic platform techniques; [c] belonging to the category of network address shuffling

eter (i.e., using a new value to replace the old one) periodically or erratically. Limited timeliness is the external manifestation of the problem on 'WHEN to move'. It is another key factor to ensure the effectiveness of MTD strategies, and almost all the MTD strategies have this feature. It is the right attribute that makes the information collected by adversaries have short validity period, and thus effectively degrades the attacker's capacity to identify specific targets, gathers more essential information, and then initiates and launches an attack, thus increasing the work effort for attackers to exploit the vulnerabilities for the desired target. The interval of limited timeliness is determined by the mutation frequency, which is critical for the MTD techniques. Now, limited timeliness is usually achieved by presetting a fixed interval, using an adjustable interval, triggered by anomalous events, or in some other ways (Table 5).

4.2.5 Attack surface reduction

In addition to the four common characteristics, attack surface reduction is a minor characteristic that exists in parts of MTD mechanisms and that can enhance the security of the target. Intuitively, the larger the attack surface, the more insecure the system (Manadhata and Wing, 2011b). As mentioned in Section 3, attack surface can be viewed as a set of all the properties of the system that can be used for attack. Attack surface reduction means reducing some of the properties, and the usual way to achieve attack surface reduction is to repair some vulnera-

Table 4 Methods to create randomness

Method (Description)	Mechanism
Choosing randomly (the next candidate is randomly chosen to replace the current candidate)	ChameleonSoft (Azab *et al.*, 2011), end-to-end software diversity (Christodorescu *et al.*, 2011), practical software diversification (Pappas *et al.*, 2013), SEM (Cui and Stolfo, 2011), proactive obfuscation (Roeder and Schneider, 2010), HMS (le Goues *et al.*, 2013), NOMAD (Vikram *et al.*, 2013), adaptive JIT code diversification (Jangda *et al.*, 2015), OF-RHM (Jafarian *et al.*, 2012), spatio-temporal address mutation (Jafarian *et al.*, 2014), and MOTAG (Jia *et al.*, 2013)
Choosing by rotation (the next candidate is chosen by turns)	SCIT (Bangalore and Sood, 2009), MAS (Huang and Ghosh, 2011), and MORE (Thompson *et al.*, 2014)
Choosing according to function (the next candidate is calculated by a function)	MT6D (Dunlop *et al.*, 2011), RHM (Al-Shaer *et al.*, 2013), MTD-MANETs (Albanese *et al.*, 2013), the SDN shuffle approach (MacFarland and Shue, 2015), the SDN-based frequency-minimal MTD approach (Debroy *et al.*, 2016), and RPAH (Luo *et al.*, 2015)
Choosing according to game theory (this method is used to provide a general method for attack surface shifting, and there can be multiple specific strategies under the guidance of this method)	Manadhata (2013), Zhu and Başar (2013), and Carter *et al.* (2014)
Choosing according to the attack behavior observed or perceived (the next candidate is chosen to meet some specific security requirements)	Spatio-temporal address mutation (Jafarian *et al.*, 2014), and TALENT (Okhravi *et al.*, 2011b)
*(The next candidate is chosen based on an MTD service deployment strategy proposed by authors)	MTD strategy for cloud-based services (Peng *et al.*, 2014)
*(The selection probability for each candidate directly relates to the weight that is associated with each candidate based on a certain criterion)	OF-RHM (Jafarian *et al.*, 2012)
*(The next candidate (new configuration) is chosen according to the security level assessed by the assessment component)	The approach of evolving computer configuration (John *et al.*, 2014; Lucas *et al.*, 2014)
*(No details)	SDNA (Yackoski *et al.*, 2011)

* Belonging to some other ways

bilities of the target. It can be seen as an equivalent transformation technique, as it can create another functional equivalence but fewer attack surface variants of the target software/running platform.

As mentioned earlier, attack surface shifting can be considered an effective way of MTD. In addition, attack surface reduction can be another way to improve the security of the system to some extent, and it can work together with attack surface shifting or be used alone.

There are two related works about the combination of attack surface shifting and attack surface reduction. In the approach of evolving computer configuration (John *et al.*, 2014; Lucas *et al.*, 2014), evolutionary algorithms (such as genetic algorithms) are used to create a set of candidate configurations that consist of a group of parameters; then the security score is arrived at based on the security information on the system to choose a new and more secure configuration periodically for deployment. In this way, the system's attack surface changes over time and the security vulnerabilities become smaller. HMS (le Goues *et al.*, 2013) also achieves attack surface reduction using evolutionary algorithms to automatically repair vulnerabilities while shifting the system's attack surface continuously. Although both the types of work use evolutionary algorithms, the main difference between them is embodied in three aspects. The first aspect is the way of attack surface shifting. In John *et al.* (2014) and Lucas *et al.* (2014), attack surface shifting was achieved by replacing the current configuration with a new configuration that was generated based on the old configurations; le Goues *et al.* (2013) used a spatio-temporal diversity engine to imbue software with a dynamically shifting attack surface. The second aspect is the timing of attack surface reduction. In John *et al.* (2014) and Lucas *et al.* (2014), it is the time of configuration replace-

Table 5 Methods to produce limited timeliness

Method (Description)	Mechanism
Presetting a fixed interval (the mechanism changes its candidate periodically, i.e., at a regular/fixed interval)	Proactive obfuscation (Roeder and Schneider, 2010), HMS (le Goues *et al.*, 2013), SCIT (Bangalore and Sood, 2009), MAS (Huang and Ghosh, 2011), TALENT (Okhravi *et al.*, 2011a), MORE (Thompson *et al.*, 2014), OF-RHM (Jafarian *et al.*, 2012), spatio-temporal address mutation (Jafarian *et al.*, 2014), RPAH (Luo *et al.*, 2015), and the approach of evolving computer configuration (John *et al.*, 2014; Lucas *et al.*, 2014)
Using an adjustable interval (the mechanism changes its candidate according to an adjusted timer, or a strategy determined by the administrator)	ChameleonSoft (Azab *et al.*, 2011), end-to-end software diversity (Christodorescu *et al.*, 2011), practical software diversification (Pappas *et al.*, 2013), SEM (Cui and Stolfo, 2011), adaptive JIT code diversification (Jangda *et al.*, 2015), MTD strategy for cloud-based services (Peng *et al.*, 2014), MTD-MANETs (Albanese *et al.*, 2013), MT6D (Dunlop *et al.*, 2011), and RHM (Al-Shaer *et al.*, 2013)
Triggering by anomalous events (the mechanism changes its candidate when there is an attack or when the security level is decreased)	ChameleonSoft (Azab *et al.*, 2011), TALENT (Okhravi *et al.*, 2011a), MAS (Huang and Ghosh, 2011), and MOTAG (Jia *et al.*, 2013)
*(Limited timeliness is triggered by user request)	NOMAD (Vikram *et al.*, 2013)
*(Limited timeliness is achieved by the occurrence of address rewriting when packets enter or exit the OS)	SDNA (Yackoski *et al.*, 2011)
*(Limited timeliness is achieved by setting the TTL value)	SDN shuffle approach (MacFarland and Shue, 2015)
*(Limited timeliness is achieved based on the statistical attack pattern and probability)	SDN-based frequency-minimal MTD approach (Debroy *et al.*, 2016)

* Belonging to some other ways

ment, while in le Goues *et al.* (2013) it is the time when attack attempts are thwarted and detected. The third aspect is the vulnerabilities repaired by them respectively. In John *et al.* (2014) and Lucas *et al.* (2014), the security-critical vulnerabilities are removed gradually, while in le Goues *et al.* (2013), both security-critical and non-security critical vulnerabilities are repaired.

In addition, there are two works focusing on only attack surface reduction. Rinard (2011) suggested applying equivalent transformation to the application, in other words, cutting some functionalities that are beyond users' needs because most of the attack surfaces often come from functionalities that users do not need and may not even be aware of. Also, he believed that some automatic techniques, such as input rectification, functionality excision, functionality replacement, loop perforation, and cyclic memory allocation, can remove the superfluous functionality of a program and thus can eliminate security vulnerabilities (i.e., reduce the attack surface) while enabling the system to provide the normal service. Andel *et al.* (2014) proposed that the protected program with known vulnerabilities would be divided into sections of invulnerable and vulnerable parts, and that the invulnerable sections would run on a traditional processor as usual while the vulnerable sections run on a field-programmable gate array (FPGA), which can make sure that the attacks on the vulnerable sections are no longer carried out. In this sense, it seems that the attack surface of the protected program is reduced. In the two approaches, there is only attack surface reduction but no attack surface shifting. Although they transform the software and can increase the overall security of the protected system to some extent, they cannot be classified as an MTD technology in the strict sense, because they do not conform to the vision of MTD described in the literature (NITRD, 2010).

4.3 Summary of moving target defense strategies

It is well known that the entire goal of MTD is to increase the work effort for attackers, to limit the exposure of vulnerabilities and opportunities for attacks, and to increase the resiliency and security of the protected system (NITRD, 2010). However, there is yet no standard definition of what an MTD

is so far (Zhuang *et al.*, 2014b). NITRD (2010) and Okhravi *et al.* (2013) proposed that a cyber moving target technique refers to any technique that attempts to defend a system and increase the complexity and cost for attackers by making the system less static, less deterministic, and less homogeneous.

In other words, the goal of MTD can be decomposed into three specific sub-goals, namely, less static, less deterministic, and less homogeneous. From our own perspective, the three sub-goals do not need to be reached at the same time. For a single system, if the static nature is broken, the deterministic nature is broken simultaneously, and thus the work effort of the attacker can be increased effectively. For a network/compound system that contains more than one system, if defenders make one or more systems in the network less static and less deterministic, the work effort for attacking the system can be increased. If defenders make the network only less homogeneous, the work effort of large-scale attacks would also increase. No matter what the case is, the work effort of attackers can be increased.

According to the sub-goals, we divide MTD techniques into three types:

1. HETE-type

The MTD techniques can make the target only less homogeneous, in other words, more heterogeneous. Until now, only the compiler-generated software diversity approach (Jackson *et al.*, 2011) falls within this type.

2. DYNA-type

The MTD techniques can make the target only less static and less deterministic, in other words, more dynamic. Until now, only the SCIT (Bangalore and Sood, 2009) belongs to this type.

3. Mixed-type

It is the combination of HETE-type and DYNA-type. Moreover, from the existing works mentioned in Section 4.1, we know that the mechanisms with this type are the most in number.

We have extracted some characteristics shared by the MTD strategies in Section 4.2. For the five characteristics discussed above, we have the following facts:

First, for each proper MTD, multi-candidate is the necessary condition for all the three types of MTD. For the MTD techniques that can make the target only less homogeneous, there must be multiple candidate instances with diversity. For the MTD techniques that can make the target less static and less deterministic, there is a large configuration space provided for moving, which consists of multiple candidate instances.

Second, for an MTD strategy with HETE-type, such as compiler-generated software diversity (Jackson *et al.*, 2011), it is obvious that multi-candidate and diversity are the two necessary and sufficient conditions, and the other conditions are not needed. As mentioned in Jackson *et al.* (2011), the compiler-based technique MSSD creates large-scale software diversity in the network by making every user obtain a unique diversified variant of the target software. Therefore, it is harder for the attacker to expand attack, and the probability of successful large-scale attack would decrease. However, there is no instance replacement and thus it is not equipped with the feature of randomness or limited timeliness.

Third, for an MTD strategy with DYNA-type, such as SCIT (Bangalore and Sood, 2009), multi-candidate, randomness, and limited timeliness are the three necessary and sufficient conditions. On the one hand, if an instance is online only for limited timeliness, it can always interrupt the process of an attack (including the reconnaissance), and the randomness makes it hard for an attacker to follow the change of the target or even makes the attacker lose the target. Hence, an MTD mechanism with the three characteristics can usually increase the work effort of attack and conform to the vision and goal of MTD. On the other hand, MTD tries to create, evaluate, and deploy mechanisms and strategies that are diverse, continually shifting, and changing over time. The shifting of multiple configurations/instances means that an instance is characterized by limited timeliness, and the shifting needs to possess randomness to confuse the attacker effectively. Diversity is only an assistant way to improve the effectiveness for this MTD because it can make the multi-candidate less homogeneous. Although it exists in most MTD techniques used to make the single protected system less static and deterministic, it is neither the sufficient condition nor the necessary condition for this type of MTD. We can take MAS (Huang and Ghosh, 2011) and SCIT (Bangalore and Sood, 2009) as examples. Each VS in MAS is equipped with a unique software mix and thus the VSs are diverse, while each server in SCIT has the same pristine state and thus the

VSs do not possess diversity. However, both MAS and SCIT can be classified into MTD technology. Therefore, diversity is neither the sufficient condition nor the necessary condition for this type of MTD.

Fourth, for an MTD strategy with the mixed-type, multi-candidate, diversity, randomness, and limited timeliness are the four necessary and sufficient conditions. The reason can be seen in the second and third considerations.

Fifth, attack surface reduction is an assistant method used in the MTD technique, and it can be an assistant only for the strategies falling in the categories of software transformations and dynamic platform techniques. As mentioned earlier, the usual way to achieve attack surface reduction is to repair the vulnerabilities of the target. For the strategies in the category of network address shuffling, the moving parameter is the network address, which is an attackable property but not a vulnerability, and it cannot be reduced. For the strategies in the category of software transformations, the moving parameter is the target software, which may include some vulnerabilities that can be reduced. For the strategies in the category of dynamic platform techniques, the moving parameters are the running platform and configuration, which may also include some vulnerabilities that can be reduced.

We use Table 6 to present our five conclusions.

5 Moving target defense evaluation

In this section, we summarize the studies in the area of MTD evaluation. MTD evaluation aims at measuring the effectiveness and efficiency of existing mechanisms to get some insights and provide reference for a new design. To achieve this goal, it needs approaches and metrics. Many evaluation approaches have been proposed and they have been used to measure the preliminary design of an MTD system or a category of MTD techniques. There have also been many evaluation metrics. Some are common metrics that can be used for major evaluation approaches (Sandoval and Hassell, 2010; van Leeuwen *et al.*, 2015; Zaffarano *et al.*, 2015; Zhang *et al.*, 2016). Some are specific metrics to fit with one approach (Clark *et al.*, 2013; Han *et al.*, 2014; Zhuang *et al.*, 2014a). In this study, we focus on the evaluation approaches. Existing evaluation approaches can be typically divided into four groups: experiment-based evaluation, theoretical analysis based evaluation, model analysis based evaluation, and mixed method based evaluation that mixes the various approaches mentioned.

1. Experiment-based evaluation

Zhuang *et al.* (2012; 2013) used a simulation testbed, which was built on NeSSi2, to evaluate the effectiveness of their preliminary design of a network MTD system that can adopt SDNA (Yackoski *et al.*, 2011) as the security policy enforcement unit for each role/VM (Yackoski *et al.*, 2013b). In the experiments, VM refreshing was used as an MTD technique, and conservative attack graph was used to guide the attacks. VM refreshing is an adaptation in each simulation time interval, by which the configuration manager randomly selects a role, and then shuts down a VM that performs the role in one host and restarts a new VM that plays the same role in a random host with a new VM ID and IP address. The goal of the experiments is to investigate the effects of randomly changing one aspect of the system (role to VM mapping) in decreasing an adversary's chance for success. However, the work does not have universality because the evaluation is just based on the proposed design schema.

2. Theoretical analysis based evaluation

Han *et al.* (2014) characterized the power of MTDs through cyber epidemic dynamics. They first classified existing MTD techniques into three groups, network-based, host-based, and instrument-based MTD techniques. Each of these can be correspondingly accommodated within the cyber epidemic dynamics models that with dynamic attack-defense structures, with dynamic parameters, or with dynamic attack-defense structures and dynamic parameters. Then the characterization is done based on two measures defined by the authors, named $(\mu_1, \mu_2, ..., \mu_J, \pi_1^*)$-powerful and $(\mu_1, \mu_2, ..., \mu_J, \pi_1, \gamma)$-powerful. However, the two measures can evaluate only the effectiveness of the MTDs as a whole; in other words, using this approach can identify only whether MTDs can induce the system to be secure but cannot know the degree of improvement in security brought about by the MTD techniques.

3. Model analysis based evaluation

Evans *et al.* (2011) proposed a game model to analyze the effectiveness of dynamic diversity de-

Table 6 A brief summary of the five characteristics

Mechanism	Type	Multi-candidate	Diversity	Randomness	Limited timeliness	Attack surface reduction
[a]Compiler-generated software diversity (Jackson *et al.*, 2011)	HETE	√	√			
[a]ChameleonSoft (Azab *et al.*, 2011)	Mixed	√	√	√	√	
[a]End-to-end software diversity (Christodorescu *et al.*, 2011)	Mixed	√	√	√	√	
[a]Practical software diversification (Pappas *et al.*, 2013)	Mixed	√	√	√	√	
[a]SEM (Cui and Stolfo, 2011)	Mixed	√	√	√	√	
[a]Proactive obfuscation (Roeder and Schneider, 2010)	Mixed	√	√	√	√	
[a]HMS (le Goues *et al.*, 2013)	Mixed	√	√	√	√	√
[a]NOMAD (Vikram *et al.*, 2013)	Mixed	√	√	√	√	
[a]Adaptive just-in-time code diversification (Jangda *et al.*, 2015)	Mixed	√	√	√	√	
[b]TALENT (Okhravi *et al.*, 2012)	Mixed	√	√	√	√	
[b]SCIT (Bangalore and Sood, 2009)	DYNA	√		√	√	
[b]MAS (Huang and Ghosh, 2011)	Mixed	√	√	√	√	
[b]MTD strategy for cloud-based services (Peng *et al.*, 2014)	Mixed	√	√	√	√	
[b]Approach of evolving computer configuration (John *et al.*, 2014; Lucas *et al.*, 2014)	Mixed	√	√	√	√	√
[b]MORE (Thompson *et al.*, 2014)	Mixed	√	√	√	√	
[b]SDN-based frequency-minimal MTD approach (Debroy *et al.*, 2016)	Mixed	√	√	√	√	
[c]SDNA (Yackoski *et al.*, 2011)	Mixed	√	√	√	√	
[c]MT6D (Dunlop *et al.*, 2011)	Mixed	√	√	√	√	
[c]OF-RHM (Jafarian *et al.*, 2012)	Mixed	√	√	√	√	
[c]RHM (Al-Shaer *et al.*, 2013)	Mixed	√	√	√	√	
[c]Spatio-temporal address mutation (Jafarian *et al.*, 2014)	Mixed	√	√	√	√	
[c]MORPHINATOR (http://defense-update.com/tag/morphinator)	Mixed	√	√	√	√	
[c]MOTAG (Jia *et al.*, 2013)	Mixed	√	√	√	√	
[c]MTD-MANETs (Albanese *et al.*, 2013)	Mixed	√	√	√	√	
[c]SDN shuffle approach (MacFarland and Shue, 2015)	Mixed	√	√	√	√	
[c]RPAH (Luo *et al.*, 2015)	Mixed	√	√	√	√	

[a] Belonging to the category of software transformations; [b] belonging to the category of dynamic platform techniques; [c] belonging to the category of network address shuffling

fenses against sophisticated attackers. The proposed model involves an attacker and a defender, and the defender's goal is to provide a service with high reliability and performance, while the attacker's goal is to exploit the server. With considering five attack strategies used by an attacker against diversity defense, they used the model to analyze the effectiveness of dynamic diversity defenses, and to identify scenarios where MTDs (here, the term represents the diversity defenses) are or are not effective and how the rate of re-diversification affects the attacker's success probability.

Xu *et al.* (2014) proposed a three-layer model to evaluate and compare the effectiveness of different MTD techniques. The first layer is the program state machine (PSM), which is created for each program and is used to capture the low-level context information from the individual programs to express how an attack instance and an MTD instance would affect resources in the program. An MTD instance refers to the set of MTDs deployed in a system, in which each MTD is deployed to protect one or multiple programs. The second layer is the system state machine (SSM), which is designed upon the first layer. It is

expected to model the interaction between programs, and different states of SSM represent damage to different components in the system. There are multiple different SSM variants to be switched to evaluate the effectiveness of different MTD techniques. The third layer is the evaluation state machine (ESM), which is used to describe the damage to the whole system in an explicit manner and to evaluate the different defenses against the different attacks provided. It is the first approach to evaluate and compare the effectiveness of different MTD techniques; however, it is too complicated for users.

Moody *et al.* (2014) used stochastic Petri nets (SPNs) to model and evaluate a defensive maneuver cyber platform, which uses MTD and deceptive defense tactics. Through the use of SPNs to model each node comprising the platform and the whole platform system, they discussed the trade-offs between security and operations in the defensive maneuver cyber platform, specifically identifying the impact of the transition firing rate.

4. Mixed method based evaluation

Carroll *et al.* (2014) used urn models (probabilistic models) and simulations to investigate the performance of network address shuffling. By considering static addresses (no address changed) and perfect shuffling (addresses being changed after each and every connection), they identified four factors that influence the probability of attacker success, and the theoretical analysis shows that address shuffling is an acceptable defense if there is a small population of vulnerable systems within a large network address space. Moreover, they used simulations based on actual network traces collected from Dartmouth University's CRAWDAD Project to investigate the relationship between shuffling frequency and connection loss and have obtained an upper bound of the defense performance. Motivated by their work, Luo *et al.* (2014) used the urn models and simulations to analyze the effectiveness of port hopping. The scenarios they used and the conclusions they obtained are similar to those in Carroll *et al.* (2014). Crouse *et al.* (2015) used urn models and simulations to compare the effectiveness and performance of network address shuffling (an example of movement), honeypots (an example of deception), and a combination of both for defending against reconnaissance. The simulations are based on the 2008 SIGCOMM Conference Packet Traces, and they obtained the conclusion that the defense performance of honeypots (deception) is typically better than that of network shuffling (movement) alone, but worse than that of the combination of deception and movement.

Zhuang *et al.* (2014a) presented an analytical model to calculate the likelihood of intrusion success from outside the network to a specific node in a network, in which partial nodes are deployed with MTD techniques such as VM refreshing or replacement. Through the simulations, they arrived at the conclusion that the designer can tune the three key parameters, network node configuration, adaptation (i.e., VM refreshing/replacement) interval, and the number of nodes adapted in each interval, to find better protection for a critical node. However, the model is not practical, as it is based on the hypothesis that each node in the attack path must know the probabilities that the attackers can compromise it from a previous node and from it to its next nodes when there is no adaptation.

Okhravi *et al.* (2014b) proposed a generalized model of DPT to quantify the effectiveness of DPT and validated the model through simulations. They described the four features of existing DPT first and then analyzed and measured the impact of these features on the security provided by DPT on the testbed developed by them, called TALENT (Okhravi *et al.*, 2012), through experiments. Based on the knowledge of the experiments and the analysis results, they presented a generalized model of DPT, which can be used to quantify their effectiveness. They also simulated the generalized model and showed that the testbed measurements and the simulations match with a small error, which validates the fact that they have indeed captured at least the major effect that contributes to the effectiveness of dynamic platform techniques.

Clark *et al.* (2013) proposed an analytical approach to study the effectiveness of the decoy-based MTD and presented a simulation via MATLAB to verify the analysis. In the decoy-based MTD, a large number of decoy nodes were introduced, each equipped with a valid IP address and simplified common network protocols to be like a valid node. Then the IP addresses of all nodes including the decoys were randomized to reduce the probability of the real node being identified and attacked. They presented a simulation study via MATLAB and the real-world data from Network Mapper (NMAP) to

determine whether a node is real or a decoy by observing the timing of the node responses to probe packets, in other words, to characterize the effectiveness of the decoys in mimicking the real nodes on the system performance, which reflects the effectiveness of the decoy-based MTD.

Lu *et al.* (2015) proposed a generic framework to model and evaluate the effectiveness of proactive cyber maneuvers to protect the critical path in MANETs. The example scenario is that an attacker tries to continuously compromise or infect nodes, while the defender attempts to patch vulnerable nodes or heal infected nodes. The authors first proposed the framework to analytically model cyber maneuvers that can be adopted by the defender and their utilities. Based on the framework, they developed an optimal solution to maximize the lifetime of the critical path. Then through analysis and numerical simulations, they validated the effectiveness of the solution.

Hong and Kim (2016) used security models to evaluate the effectiveness of MTDs. They first divided the MTD techniques into three groups, shuffle, diversity, and redundancy, and then incorporated them into a two-layer hierarchical attack representation model (HARM), with the attack graph in the upper layer and the attack tree in the lower layer. Thereafter, they assessed the effectiveness of the MTDs and measured the changes in performance and security after deploying MTD techniques through simulations. Moreover, based on the measurement, the MTDs can be compared to choose the most effective one.

From the existing methods, we find that they can be divided into two groups according to their goals. One group aims to evaluate the effectiveness of one type of MTD strategies, such as the evaluation of dynamic diversity defense (Evans *et al.*, 2011). The other group not only evaluates the effectiveness of one type of MTD strategies, but also identifies the parameters that can influence the defense effect of these MTD strategies, such as the the evaluation of port hopping (Luo *et al.*, 2014). The contributions of these works are summarized in Table 7.

Apart from the methods above, Zaffarano *et al.* (2015) proposed an approach based on the siege's cyber quantification framework (CQF) to measure the effectiveness of network-based. Actually, the emphasis is more on the testbed than on the evaluation approach. In addition, Eskridge *et al.* (2015) proposed an emulation environment for MTD experimentation, which provides a set of tools for cybersecurity researchers to perform experiments for evaluating MTD techniques.

6 Discussion

In this study, we have presented a lot of related works that have important practical significance and that involve three main areas in the MTD field. However, there are still some directions that need to be researched in depth in this field.

1. Practicality

Being practical is the foundation and basis for a mechanism to be widely applied in the real world, and the field of MTD is no exception. There are three problems that need to be addressed for practicability. The first and foremost one is that the technique is needed; i.e., the functionality is needed by users. The second one is the transparency to users, which means that the deployment of the proposed mechanism must not change or influence the normal user's operation greatly. In addition, the performance loss of the protected target induced by the deployment must be acceptable to users. The third one is the deployment cost. If the cost of deploying an MTD technique is too high to accept, the user will refuse to deploy it. Currently, related work is poor. It is well known that inserting garbage instructions is an effective defense against code-reuse attacks. However, it slows down the program execution simultaneously. Murphy *et al.* (2014) proposed the use of software profiling methods to reduce the runtime overhead through optimizations, which were performed by varying the probability for the insertion of a garbage instruction at any particular location in the binary code.

2. Hybrid MTD

Currently, the major MTD mechanisms are at a single layer, which means that there is only one moving parameter or that multiple moving parameters belong to one layer. The hybrid MTD mechanism is a mechanism that has multiple moving parameters at different layers. In addition, the 'layer' here can be seen as the three layers forming a networking system as mentioned earlier or other layers proposed through other criteria. Existing works include MUTE (Al-Shaer, 2011), multi-layer

Table 7 A brief summary of the evaluation approaches

Reference	Approach	Goal	Contribution
[a]Zhuang *et al.* (2012)	Simulation	E-F	Evaluate the effectiveness of a network MTD proposed by the authors, and identify that the frequency of changes can influence the likelihood of attack success
[b]Han *et al.* (2014)	Theoretical analysis	E	Provide a general approach for using cyber epidemic dynamics and two metrics to characterize the power of MTDs
[c]Evans *et al.* (2011)	A game model	E	Consider five attacking scenarios (circumvention attacks, deputy attacks, brute force and entropy reduction attacks, probing attacks, and incremental attacks) to evaluate the effectiveness of dynamic diversity defenses and conclude that MTD is not always effective
[c]Xu *et al.* (2014)	A three-layer model	E	Provide a general model to evaluate the effectiveness of MTDs. Furthermore, this model can be used to compare the effects of different MTDs
[c]Moody *et al.* (2014)	Stochastic Petri nets model	E-F	Evaluate the benefits of the maneuverability for the defensive maneuver cyber platform, which uses MTD and deceptive defense. Furthermore, identify that the ratio of deceptive to operational nodes and the transition firing rate can influence the defense effect
[d]Carroll *et al.* (2014)	Urn models and simulation	E-F	Analyze the effectiveness of network address shuffling. Furthermore, identify the four characteristics that influence the attack success probability, namely, network address space, percentage of address space probed, number of vulnerable computers, and address shuffle frequency
[d]Luo *et al.* (2014)	Urn models and simulation	E-F	Analyze the effectiveness of port hopping. Furthermore, identify the four characteristics that influence the defense effectiveness, namely, port pool size, the number of probes, the number of vulnerable services, and hopping frequency
[d]Crouse *et al.* (2015)	Urn models and simulation	E-F	Compare the effectiveness of network address shuffling, honeypots, and the combination of the two. Furthermore, quantify the influence of some factors, such as network size, the number of scans, deployment of honeypots, and the number of vulnerable computers
[d]Zhuang *et al.* (2014a)	An analytical model and simulation	E-F	Analyze the effect of MTD (specifically, virtual machine refreshing/replacement) in an enterprise network. Furthermore, identify that the three key factors are network node configuration, refreshing/replacement interval, and the number of nodes refreshed/replaced in each interval
[d]Okhravi *et al.* (2014b)	A generalized model for DPT, experiments, and simulations	E	Propose a generalized model of DPT to evaluate the protection provided by dynamic platforms and validate the model
[d]Clark *et al.* (2013)	Analytical approach and simulation	E	Evaluate the effectiveness of the decoy-based MTD, where deception and IP address randomization are deployed
[d]Lu *et al.* (2015)	A generic framework model and simulation	E	Evaluate the effectiveness of the proactive cyber maneuvers to protect the critical paths in MANETs
[d]Hong and Kim (2016)	Security model and simulation	E	Provide a general approach to evaluate the effectiveness of MTDs. Furthermore, the approach can be used to compare different MTDs

'E' represents that the work just evaluates the effectiveness; 'E-F' represents that the work aims not only to evaluate the effectiveness but also to identify the key factors that can influence the defense effect of MTD. [a] Belonging to the experiment-based category; [b] belonging to the theoretics-based category; [c] belonging to the model-based category; [d] belonging to the mixed category

MTD for protecting resource-constrained distributed devices (Casola *et al.*, 2014), the MTD system that enables full protocol stack agility (Corbett *et al.*, 2014), and the MTD with dynamic randomization of network attributes (Chavez *et al.*, 2015). MUTE integrates random address hopping and random

finger printing to randomly modify the routing IP, OS, and application identity that are independent of the actual information, to hinder the adversary's capabilities in scanning or discovering network targets, launching DoS attacks, and creating botnet structures. The multi-layer MTD for protecting resource-constrained distributed devices attempts to protect resource-constrained devices through reconfiguring them at the security level or the physical layer at runtime based on current requirements. The security layer reconfiguration can be performed by switching among different cryptosystems that satisfy specific security requirements, and the physical layer reconfiguration is done by switching among different versions of firmware. The MTD system that enables full protocol stack agility uses the following three MTD approaches, server diversity, modulation and coding scheme obfuscation, and modulation remapping, to counter intelligent jamming attacks. The MTD with dynamic randomization of network attributes contains two types of randomizations, network address (including the IP address and TCP/UDP port) randomization and route randomization, to thwart adversaries. The network address randomization is achieved on the network layer, and the route randomization is achieved on the overlay layer. Compared to a single MTD mechanism, hybrid MTD mechanisms can make the movement on the protected target more complex, thus rendering the attack more difficult and costlier. However, these processes involve greater cost for management and resource maintenance, and can also influence the system performance greatly.

3. Network monitoring

It is clear that defense is an activity aiming at the activities of the attacker; thus, it must know exactly the attack behavior at the present time to make a better defense decision. Prakash and Wellman (2015) validated this opinion. They used the empirical game-theoretic techniques to examine the interaction between attackers and defenders and found that reliable probe detection is very important for effective deterrence. The reliable probe detection is closely related to the ability of monitoring. However, most of the current research is on the mobility capabilities and command-and-control capabilities, which must be linked together tightly (expect Zhu *et al.* (2014), in which the authors proposed two iterative reinforcement learning algorithms to help the defender get more information about the attack, thus making the optimal defenses). The research on network monitoring in MTD is very little; thus, the follow-up research should extend in this direction.

4. Application on existing approaches

As mentioned earlier, MTD is a design principle. It can be applied not only to different system attributes to generate various MTD mechanisms but also to existing security evaluation or defense approaches to improve their effects. Several works have been reported in this area. Rahman *et al.* (2014) applied MTD to the power system state estimation to harden the security and precision of this process. Oehmen *et al.* (2013) applied MTD to enable the means of detection moving away from the static signature-based detection, which is often probed by adversaries, to discover what patterns are being searched for, thus increasing the cost of attacks and the effect of the detection. Colbaugh and Glass (2012) applied MTD to the process of predictability-oriented defense against adaptive adversaries to increase the difficulty of the adversary's reverse-engineering task. Although, strictly speaking, these applications cannot be classified into MTD technology, they have practical significance.

5. Evaluation

All the aforesaid evaluation approaches have two features: (1) The main goal is to evaluate the effectiveness of the approaches and to find the major factors that contribute to the security provided by the MTD approach. It still lacks the evaluation of performance and efficiency. (2) They evaluate mainly one category of MTDs but lack the comparison among different MTD techniques, except Xu *et al.* (2014) and Hong and Kim (2016). Therefore, the follow-up evaluation should extend along these two aspects.

6. Application of game theory and adversarial model

In the MTD area, there are two types of research on the application of game theory and adversarial model. One type is the research on suitability. Bilar *et al.* (2013) presented a detailed study of the coevolution of the Conficker worm and the associated defenses against it, which demonstrates that attackers and defenders present moving targets to each other because advances by one side are countered by the other. Jain *et al.* (2013) described some key challenges in applying the game theory for security and proposed key ideas and algorithms for

solving and understanding the characteristics of large-scale real-world security games. Gonzalez (2013) presented the challenges and potentials for extending computational models of human behaviors at the individual level to address predictions in two-player (i.e., a defender and an attacker) non-cooperative dynamic cyber-security situations. The other type is security scheduling strategy design. Manadhata (2013) introduced a game theoretic attack surface shifting and reduction approach to explore the optimal MTD strategy, in which a complete and perfect information stochastic game model was used. Zhu and Başar (2013) proposed a feedback multi-stage defense strategy based on a two-person zero-sum game theory model to secure the system by shifting its attack surface. Carter *et al.* (2014) used a two-player, incomplete, iterative leader-follower game model to determine the destination platform for dynamic platform defenses. In the context of suitability research, Gonzalez (2013) and Jain *et al.* (2013) proposed some future works. In the context of strategy design, since the interaction between attackers and defenders quite fits the game theory theme, it would be quite interesting to investigate strategy design in the direction of using the game theory.

7. Application of SDN architecture

Currently, some existing MTD mechanisms were presented on the basis of SDN architecture, including the SDN-based frequency-minimal MTD approach (Debroy *et al.*, 2016), OF-RHM (Jafarian *et al.*, 2012), the SDN shuffle approach (MacFarland and Shue, 2015), and the MTD with dynamic randomization of network attributes (Chavez *et al.*, 2015). Kampanakis *et al.* (2014) investigated how SDN can be used to support network-based MTD techniques, and insisted that SDN can make the implementation of MTD techniques more practical, more customizable, and easier to deploy. It is well known that SDN is a flexible architecture, based on open standards, and can be directly programmable. This feature can indeed produce some common benefits, such as separating the network control and forwarding planes, thereby making the design and implementation of MTD more customizable. However, there are still some open research challenges for the application of SDN architecture in the MTD area, such as integration of SDN architecture with the traditional network architecture, security of the controller, and trade-off between the cost and benefits.

7 Conclusions

We have presented an inclusive overview on MTD technology based on the research areas in the MTD field. MTD is a promising approach to continually move the entity or properties that are vulnerable to attacks, thus increasing the work effort of an attacker. This active feature leads to a defense paradigm shift that can alter the asymmetric situation of attacks and defenses in the cyber-security field. The major important literature in this field has been covered in this systematic review, and the foremost contribution of this work is the systematic description of the results of research in this field. Moreover, we have proposed a new security model and a function-and-movement model, extracted some characteristics common to the MTD strategies, and summarized the ways to ensure the effectiveness of MTD strategies.

We first introduced a new security model, as well as a function-and-movement model. The new security model is used to describe the change in the traditional defense paradigm due to the introduction of MTD. From this model, readers can easily understand the operation paradigm of MTD techniques, as well as the difference between MTD and traditional defense. The function-and-movement model provides a new perspective for understanding MTD research, including facets of this field, target for researching, functionality and defense, and the interrelationship among them. We have divided the existing research into three main areas according to their research content, namely, MTD theory, MTD strategy, and MTD evaluation.

Then we systematically presented the state of the art of the three research areas in the MTD field, respectively.

In the area of MTD theory, we described the main research in this area and concluded that there are three elements for an MTD technique, i.e., 'WHAT to move', 'HOW to move', and 'WHEN to move'. 'WHAT to move' refers to the moving parameter. 'HOW to move' refers to the way to move the selected moving parameter, which implies two operations, selection (choosing a new value for the moving parameter(s) from its/their domain through various ways) and replacement (using the selected new value to replace the old one). 'WHEN to move' refers to the time series defined by the defender for replace-

ment, in other words, the frequency of moving. The three elements collectively describe a complete MTD strategy.

In the area of MTD strategy, we have categorized the existing strategies into three main categories, named software transformations, dynamic platform techniques, and network address shuffling according to the moving parameter and presented detailed interpretations of their running mode at first. Then we summarized a set of characteristics for MTD strategies and the ways to ensure the effectiveness of MTD strategies. Moreover, the ways to ensure the effectiveness of MTD strategies include the methods of generating these characteristics, as well as the necessary and sufficient conditions for creating effective MTD strategies. The set of characteristics includes four main characteristics and a minor characteristic. The four main characteristics are multi-candidate, diversity, randomness, and limited timeliness. The minor characteristic is attack surface reduction. The MTD techniques can also be divided into three types according to their specific goals, and the characteristics as necessary and sufficient conditions for strategies were analyzed in each type. The analysis shows that, for the MTD techniques that can make the target only less homogeneous, multi-candidate and diversity are the two necessary and sufficient conditions. The other conditions are not essential. For the MTD techniques that can make the target only less static and less deterministic, multi-candidate, randomness, and limited timeliness are the three necessary and sufficient conditions. For the MTD techniques that can make the target less static, less deterministic, and less homogeneous, multi-candidate, diversity, randomness, and limited timeliness are the four necessary and sufficient conditions. Attack surface reduction is an assistant that can be used only in the techniques of the categories of software transformations and dynamic platform techniques to improve their defense effect.

In the area of MTD evaluation, we have introduced the existing evaluation approaches in detail and concluded their features and shortcomings. We found that some works aim at identifying the parameters that can influence the defense effect of a specific kind/category of MTD strategies, and they are very helpful for designing new MTD systems. We also found that evaluation is mainly for effectiveness but not for performance and efficiency, and there still lacks comparison among different categories of MTD strategies. Thus, they can be the future directions of research. All these summaries in the three areas are very useful for understanding MTDs and can provide guidance to follow-up researchers.

We hope that this exhaustive and informative review article will serve as the taxonomy for navigating and invigorating subsequent research in the MTD field.

References

Albanese, M., de Benedictis, A., Jajodia, S., *et al.*, 2013. A moving target defense mechanism for MANETs based on identity virtualization. Proc. IEEE Conf. on Communications and Network Security, p.278-286. http://dx.doi.org/10.1109/CNS.2013.6682717

Al-Shaer, E., 2011. Toward network configuration randomization for moving target defense. *In*: Jajodia, S., Ghosh, A.K., Swarup, V., *et al.* (Eds.), Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats. Springer New York, USA, p.153-159. http://dx.doi.org/10.1007/978-1-4614-0977-9_9

Al-Shaer, E., Duan, Q., Jafarian, J.H., 2013. Random host mutation for moving target defense. *In*: Keromytis, A.D., di Pietro, R. (Eds.), Security and Privacy in Communication Networks. Springer Berlin Heidelberg, Germany, p.310-327. http://dx.doi.org/10.1007/978-3-642-36883-7_19

Andel, T.R., Whitehurst, L.N., McDonald, J.T., 2014. Software security and randomization through program partitioning and circuit variation. Proc. 1st ACM Workshop on Moving Target Defense, p.79-86. http://dx.doi.org/10.1145/2663474.2663484

Azab, M., Hassan, R., Eltoweissy, M., 2011. ChameleonSoft: a moving target defense system. Proc. 7th Int. Conf. on Collaborative Computing: Networking, Applications and Worksharing, p.241-250.

Bangalore, A.K., Sood, A.K., 2009. Securing web servers using self cleansing intrusion tolerance (SCIT). Proc. 2nd Int. Conf. on Dependability, p.60-65. http://dx.doi.org/10.1109/DEPEND.2009.15

Beraud, P., Cruz, A., Hassell, S., *et al.*, 2010. Cyber defense network maneuver commander. Proc. IEEE Int. Carnahan Conf. on Security Technology, p.112-120. http://dx.doi.org/10.1109/CCST.2010.5678724

Beraud, P., Cruz, A., Hassell, S., *et al.*, 2011. Using cyber maneuver to improve network resiliency. Proc. Military Communications Conf., p.1121-1126. http://dx.doi.org/10.1109/MILCOM.2011.6127449

Bilar, D., Cybenko, G., Murphy, J., 2013. Adversarial dynamics: the conficker case study. *In*: Jajodia, S., Ghosh, A.K., Subrahmanian, V.S., *et al.* (Eds.), Moving Target Defense II: Application of Game Theory and Adversarial Modeling. Springer New York, USA, p.41-71. http://dx.doi.org/10.1007/978-1-4614-5416-8_3

Cai, G.L., Wang, B.S., Luo, Y.B., *et al.*, 2016. Characterizing the running patterns of moving target defense mechanisms. Proc. 18th Int. Conf. on Advanced

Communication Technology, p.191-196. http://dx.doi.org/10.1109/ICACT.2016.7423324

Carroll, T.E., Crouse, M., Fulp, E.W., *et al.*, 2014. Analysis of network address shuffling as a moving target defense. Proc. IEEE Int. Conf. on Communications, p.701-706. http://dx.doi.org/10.1109/ICC.2014.6883401

Carter, K.M., Riordan, J.F., Okhravi, H., 2014. A game theoretic approach to strategy determination for dynamic platform defenses. Proc. 1st ACM Workshop on Moving Target Defense, p.21-30. http://dx.doi.org/10.1145/2663474.2663478

Carvalho, M., Ford, R., 2014. Moving-target defenses for computer networks. *IEEE Sec. Priv.*, **12**(2):73-76. http://dx.doi.org/10.1109/MSP.2014.30

Carvalho, M., Bradshaw, J.M., Bunch, L., *et al.*, 2012. Command and control requirements for moving-target defense. *IEEE Intell. Syst.*, **27**(3):79-85. http://dx.doi.org/10.1109/MIS.2012.45

Carvalho, M., Eskridge, T.C., Bunch, L., *et al.*, 2013. MTC2: a command and control framework for moving target defense and cyber resilience. Proc. 6th Int. Symp. on Resilient Control Systems, p.175-180. http://dx.doi.org/10.1109/ISRCS.2013.6623772

Casola, V., de Benedictis, A., Albanese, M., 2014. A multi-layer moving target defense approach for protecting resource-constrained distributed devices. *In*: Bouabana-Tebibel, T., Rubin, S.H. (Eds.), Integration of Reusable Systems. Springer International Publishing, Switzerland, p.299-324. http://dx.doi.org/10.1007/978-3-319-04717-1_14

Chavez, A.R., Stout, W.M.S., Peisert, S., 2015. Techniques for the dynamic randomization of network attributes. Proc. Int. Carnahan Conf. on Security Technology, p.1-6. http://dx.doi.org/10.1109/CCST.2015.7389661

Christodorescu, M., Fredrikson, M., Jha, S., *et al.*, 2011. End-to-end software diversification of Internet services. *In*: Jajodia, S., Ghosh, A.K., Swarup, V., *et al.* (Eds.), Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats. Springer New York, USA, p.117-130. http://dx.doi.org/10.1007/978-1-4614-0977-9_7

Clark, A., Sun, K., Poovendran, R., 2013. Effectiveness of IP address randomization in decoy-based moving target defense. Proc. 52nd IEEE Conf. on Decision and Control, p.678-685. http://dx.doi.org/10.1109/CDC.2013.6759960

Colbaugh, R., Glass, K., 2012. Predictability-oriented defense against adaptive adversaries. Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics, p.2721-2727. http://dx.doi.org/10.1109/ICSMC.2012.6378159

Corbett, C., Uher, J., Cook, J., *et al.*, 2014. Countering intelligent jamming with full protocol stack agility. *IEEE Sec. Priv.*, **12**(2):44-50. http://dx.doi.org/10.1109/MSP.2013.136

Crosby, S., Carvalho, M., Kidwell, D., 2013. A layered approach to understanding network dependencies on moving target defense mechanisms. Proc. 8th Annual Cyber Security and Information Intelligence Research Workshop, Article 36. http://dx.doi.org/10.1145/2459976.2460017

Crouse, M., Prosser, B., Fulp, E.W., 2015. Probabilistic performance analysis of moving target and deception reconnaissance defenses. Proc. 2nd ACM Workshop on Moving Target Defense, p.21-29. http://dx.doi.org/10.1145/2808475.2808480

Cui, A., Stolfo, S.J., 2011. Symbiotes and defensive mutualism: moving target defense. *In*: Jajodia, S., Ghosh, A.K., Swarup, V., *et al.* (Eds.), Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats. Springer New York, USA, p.99-108. http://dx.doi.org/10.1007/978-1-4614-0977-9_5

Debroy, S., Calyam, P., Nguyen, M., *et al.*, 2016. Frequency-minimal moving target defense using software-defined networking. Proc. Int. Conf. on Computing, Networking and Communications, p.1-6. http://dx.doi.org/10.1109/ICCNC.2016.7440635

Dunlop, M., Groat, S., Urbanski, W., *et al.*, 2011. MT6D: a moving target IPv6 defense. Proc. Military Communications Conf., p.1321-1326. http://dx.doi.org/10.1109/MILCOM.2011.6127486

Eskridge, T.C., Carvalho, M.M., Stoner, E., *et al.*, 2015. VINE: a cyber emulation environment for MTD experimentation. Proc. 2nd ACM Workshop on Moving Target Defense, p.43-47. http://dx.doi.org/10.1145/2808475.2808486

Evans, D., Nguyen-Tuong, A., Knight, J., 2011. Effectiveness of moving target defenses. *In*: Jajodia, S., Ghosh, A.K., Swarup, V., *et al.* (Eds.), Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats. Springer New York, USA, p.29-48. http://dx.doi.org/10.1007/978-1-4614-0977-9_2

Gonzalez, C., 2013. From individual decisions from experience to behavioral game theory: lessons for cybersecurity. *In*: Jajodia, S., Ghosh, A.K., Subrahmanian, V.S., *et al.* (Eds.), Moving Target Defense II: Application of Game Theory and Adversarial Modeling. Springer New York, USA, p.73-86. http://dx.doi.org/10.1007/978-1-4614-5416-8_4

Green, M., MacFarland, D.C., Smestad, D.R., *et al.*, 2015. Characterizing network-based moving target defenses. Proc. 2nd ACM Workshop on Moving Target Defense, p.31-35. http://dx.doi.org/10.1145/2808475.2808484

Han, Y.J., Lu, W.L., Xu, S.H., 2014. Characterizing the power of moving target defense via cyber epidemic dynamics. Proc. Symp. and Bootcamp on the Science of Security, Article 10. http://dx.doi.org/10.1145/2600176.2600180

Hobson, T., Okhravi, H., Bigelow, D., *et al.*, 2014. On the challenges of effective movement. Proc. 1st ACM Workshop on Moving Target Defense, p.41-50. http://dx.doi.org/10.1145/2663474.2663480

Hong, J.B., Kim, D.S., 2016. Assessing the effectiveness of moving target defenses using security models. *IEEE Trans. Depend. Secur. Comput.*, **13**(2):163-177. http://dx.doi.org/10.1109/TDSC.2015.2443790

Huang, Y., Ghosh, A.K., 2011. Introducing diversity and uncertainty to create moving attack surfaces for web services. *In*: Jajodia, S., Ghosh, A.K., Swarup, V., *et al.* (Eds.), Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats. Springer New York, USA, p.131-151. http://dx.doi.org/10.1007/978-1-4614-0977-9_8

Jackson, T., Salamat, B., Homescu, A., *et al.*, 2011. Compiler-generated software diversity. *In*: Jajodia, S., Ghosh, A.K., Swarup, V., *et al.* (Eds.), Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats. Springer New York, USA, p.77-98. http://dx.doi.org/10.1007/978-1-4614-0977-9_4

Jackson, T., Homescu, A., Crane, S., *et al.*, 2013. Diversifying the software stack using randomized NOP insertion. *In*: Jajodia, S., Ghosh, A.K., Subrahmanian, V.S., *et al.* (Eds.), Moving Target Defense II: Application of Game Theory and Adversarial Modeling. Springer New York, USA, p.151-173. http://dx.doi.org/10.1007/978-1-4614-5416-8_8

Jafarian, J.H., Al-Shaer, E., Duan, Q., 2012. OpenFlow random host mutation: transparent moving target defense using software defined networking. Proc. 1st Workshop on Hot Topics in Software Defined Networks, p.127-132. http://dx.doi.org/10.1145/2342441.2342467

Jafarian, J.H., Al-Shaer, E., Duan, Q., 2014. Spatio-temporal address mutation for proactive cyber agility against sophisticated attackers. Proc. 1st ACM Workshop on Moving Target Defense, p.69-78. http://dx.doi.org/10.1145/2663474.2663483

Jain, M., An, B., Tambe, M., 2013. Security games applied to real-world: research contributions and challenges. *In*: Jajodia, S., Ghosh, A.K., Subrahmanian, V.S., *et al.* (Eds.), Moving Target Defense II: Application of Game Theory and Adversarial Modeling. Springer New York, USA, p.15-39. http://dx.doi.org/10.1007/978-1-4614-5416-8_2

Jajodia, S., Ghosh, A.K., Swarup, V., *et al.*, 2011. Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats. Springer New York, USA.

Jajodia, S., Ghosh, A.K., Subrahmanian, V.S., *et al.*, 2013. Moving Target Defense II: Application of Game Theory and Adversarial Modeling. Springer New York, USA.

Jangda, A., Mishra, M., de Sutter, B., 2015. Adaptive just-in-time code diversification. Proc. 2nd ACM Workshop on Moving Target Defense, p.49-53. http://dx.doi.org/10.1145/2808475.2808487

Jia, Q., Sun, K., Stavrou, A., 2013. MOTAG: moving target defense against Internet denial of service attacks. Proc. 22nd Int. Conf. on Computer Communication and Networks, p.1-9. http://dx.doi.org/10.1109/ICCCN.2013.6614155

John, D.J., Smith, R.W., Turkett, W.H., *et al.*, 2014. Evolutionary based moving target cyber defense. Proc. Annual Conf. on Genetic and Evolutionary Computation, p.1261-1268. http://dx.doi.org/10.1145/2598394.2605437

Kampanakis, P., Perros, H., Beyene, T., 2014. SDN-based solutions for moving target defense network protection. Proc. 15th Int. Symp. on a World of Wireless, Mobile and Multimedia Networks, p.1-6. http://dx.doi.org/10.1109/WoWMoM.2014.6918979

le Goues, C., Nguyen-Tuong, A., Chen, H., *et al.*, 2013. Moving target defenses in the helix self-regenerative architecture. *In*: Jajodia, S., Ghosh, A.K., Subrahmanian, V.S., *et al.* (Eds.), Moving Target Defense II: Application of Game Theory and Adversarial Modeling. Springer New York, USA, p.117-149. http://dx.doi.org/10.1007/978-1-4614-5416-8_7

Liu, C.M., Zhang, Y., Chen, R., 2011. Research on dynamic model for network security based on artificial immunity. *Int. J. Knowl. Lang. Process.*, **2**(3):21-35.

Liu, Y.J., Peng, W., Su, J.S., 2014. A study of IP prefix hijacking in cloud computing networks. *Secur. Commun. Netw.*, **7**(11):2201-2210. http://dx.doi.org/10.1002/sec.738

Lu, Z., Marvel, L., Wang, C., 2015. To be proactive or not: a framework to model cyber maneuvers for critical path protection in MANETs. Proc. 2nd ACM Workshop on Moving Target Defense, p.85-93. http://dx.doi.org/10.1145/2808475.2808479

Lucas, B., Fulp, E.W., John, D.J., *et al.*, 2014. An initial framework for evolving computer configurations as a moving target defense. Proc. 9th Annual Cyber and Information Security Research Conf., p.69-72. http://dx.doi.org/10.1145/2602087.2602100

Luo, Y.B., Wang, B.S., Cai, G.L., 2014. Effectiveness of port hopping as a moving target defense. Proc. 7th Int. Conf. on Security Technology, p.7-10. http://dx.doi.org/10.1109/SecTech.2014.9

Luo, Y.B., Wang, B.S., Wang, X.F., *et al.*, 2015. RPAH: random port and address hopping for thwarting internal and external adversaries. Proc. IEEE Trustcom/BigDataSE/ISPA, p.263-270. http://dx.doi.org/10.1109/Trustcom.2015.383

MacFarland, D.C., Shue, C.A., 2015. The SDN shuffle: creating a moving-target defense using host-based software-defined networking. Proc. 2nd ACM Workshop on Moving Target Defense, p.37-41. http://dx.doi.org/10.1145/2808475.2808485

Manadhata, P.K., 2013. Game theoretic approaches to attack surface shifting. *In*: Jajodia, S., Ghosh, A.K., Subrahmanian, V.S., *et al.* (Eds.), Moving Target Defense II: Application of Game Theory and Adversarial Modeling. Springer New York, USA, p.1-13. http://dx.doi.org/10.1007/978-1-4614-5416-8_1

Manadhata, P.K., Wing, J.M., 2011a. An attack surface metric. *IEEE Trans. Softw. Eng.*, **37**(3):371-386. http://dx.doi.org/10.1109/TSE.2010.60

Manadhata, P.K., Wing, J.M., 2011b. A formal model for a system's attack surface. *In*: Jajodia, S., Ghosh, A.K., Swarup, V., *et al.* (Eds.), Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats. Springer New York, USA, p.1-28. http://dx.doi.org/10.1007/978-1-4614-0977-9_1

Moody, W.C., Hu, H., Apon, A., 2014. Defensive maneuver cyber platform modeling with stochastic Petri nets. Proc. Int. Conf. on Collaborative Computing: Networking, Applications and Worksharing, p.531-538.

Murphy, M., Larsen, P., Brunthaler, S., *et al.*, 2014. Software profiling options and their effects on security based diversification. Proc. 1st ACM Workshop on Moving Target Defense, p.87-96. http://dx.doi.org/10.1145/2663474.2663485

NITRD, 2009. National Cyber Leap Year Summit 2009 Co-chairs' Report. Available from https://www.nitrd.gov/nitrdgroups/index.php?title=Category:National_Cyber_Leap_Year_Summit_2009 [Accessed on Jan. 1, 2014].

NITRD, 2010. Cybersecurity Game-Change Research & Development Recommendations. Available from http://www.nitrd.gov/pubs/CSIA_IWG_%20Cybersecurity_%20GameChange_RD_%20Recommendations_20100513.pdf [Accessed on Aug. 20, 2013].

Oehmen, C., Peterson, E., Teuton, J., 2013. Evolutionary drift models for moving target defense. Proc. 8th Annual Cyber Security and Information Intelligence Research Workshop, Article 37. http://dx.doi.org/10.1145/2459976.2460018

Okhravi, H., Comella, A., Robinson, E., *et al.*, 2011a. Creating a cyber moving target for critical infrastructure applications. *In*: Butts, J., Shenoi, S. (Eds.), Critical Infrastructure Protection V. Springer Berlin Heidelberg, Germany, p.107-123. http://dx.doi.org/10.1007/978-3-642-24864-1_8

Okhravi, H., Haines, J.W., Ingols, K., 2011b. Achieving cyber survivability in a contested environment using a cyber moving target. *High Front. J.*, **7**(3):9-13.

Okhravi, H., Comella, A., Robinson, E., *et al.*, 2012. Creating a cyber moving target for critical infrastructure applications using platform diversity. *Int. J. Crit. Infrast. Protect.*, **5**(1):30-39. http://dx.doi.org/10.1016/j.ijcip.2012.01.002

Okhravi, H., Rabe, M.A., Mayberry, T.J., *et al.*, 2013. Survey of Cyber Moving Targets. Technical Report, No. MIT/LL-TR-1166. Lincoln Laboratory, Massachusetts Institute of Technology, USA.

Okhravi, H., Hobson, T., Bigelow, D., *et al.*, 2014a. Finding focus in the blur of moving-target techniques. *IEEE Sec. Priv.*, **12**(2):16-26. http://dx.doi.org/10.1109/MSP.2013.137

Okhravi, H., Riordan, J., Carter, K., 2014b. Quantitative evaluation of dynamic platform techniques as a defensive mechanism. *In*: Stavrou, A., Bos, H., Portokalidis, G. (Eds.), Research in Attacks, Intrusions and Defenses. Springer International Publishing, Switzerland, p.405-425. http://dx.doi.org/10.1007/978-3-319-11379-1_20

Pappas, V., Polychronakis, M., Keromytis, A.D., 2013. Practical software diversification using in-place code randomization. *In*: Jajodia, S., Ghosh, A.K., Subrahmanian, V.S., *et al.* (Eds.), Moving Target Defense II: Application of Game Theory and Adversarial Modeling. Springer New York, USA, p.175-202. http://dx.doi.org/10.1007/978-1-4614-5416-8_9

Peng, W., Li, F., Huang, C.T., *et al.*, 2014. A moving-target defense strategy for cloud-based services with heterogeneous and dynamic attack surfaces. Proc. IEEE Int. Conf. on Communications, p.804-809. http://dx.doi.org/10.1109/ICC.2014.6883418

Prakash, A., Wellman, M.P., 2015. Empirical game-theoretic analysis for moving target defense. Proc. 2nd ACM Workshop on Moving Target Defense, p.57-65. http://dx.doi.org/10.1145/2808475.2808483

Rahman, M.A., Al-Shaer, E., Bobba, R.B., 2014. Moving target defense for hardening the security of the power system state estimation. Proc. 1st ACM Workshop on Moving Target Defense, p.59-68. http://dx.doi.org/10.1145/2663474.2663482

Rinard, M., 2011. Manipulating program functionality to eliminate security vulnerabilities. *In*: Jajodia, S., Ghosh, A.K., Swarup, V., *et al.* (Eds.), Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats. Springer New York, USA, p.109-115. http://dx.doi.org/10.1007/978-1-4614-0977-9_6

Roeder, T., Schneider, F.B., 2010. Proactive obfuscation. *ACM Trans. Comput. Syst.*, **28**(2):Article 4. http://dx.doi.org/10.1145/1813654.1813655

Sandoval, J.E., Hassell, S.P., 2010. Measurement, identification and calculation of cyber defense metrics. Proc. Military Communications Conf., p.2174-2179. http://dx.doi.org/10.1109/MILCOM.2010.5680489

Taguinod, M., Doupé, A., Zhao, Z., *et al.*, 2015. Toward a moving target defense for web applications. Proc. IEEE Int. Conf. on Information Reuse and Integration, p.510-517. http://dx.doi.org/10.1109/IRI.2015.84

Thompson, M., Evans, N., Kisekka, V., 2014. Multiple OS rotational environment an implemented moving target defense. Proc. 7th Int. Symp. on Resilient Control Systems, p.1-6. http://dx.doi.org/10.1109/ISRCS.2014.6900086

Torrieri, D., Zhu, S.C., Jajodia, S., 2013. Cyber maneuver against external adversaries and compromised nodes. *In*: Jajodia, S., Ghosh, A.K., Subrahmanian, V.S., *et al.* (Eds.), Moving Target Defense II: Application of Game Theory and Adversarial Modeling. Springer New York, USA, p.87-96. http://dx.doi.org/10.1007/978-1-4614-5416-8_5

van Leeuwen, B., Stout, W.M.S., Urias, V., 2015. Operational cost of deploying Moving Target Defenses defensive work factors. Proc. Military Communications Conf., p.966-971. http://dx.doi.org/10.1109/MILCOM.2015.7357570

Vikram, S., Yang, C., Gu, G., 2013. NOMAD: towards non-intrusive moving-target defense against web bots. Proc. IEEE Conf. on Communications and Network Security, p.55-63. http://dx.doi.org/10.1109/CNS.2013.6682692

Wang, T.Z., Wang, H.M., Liu, B., *et al.*, 2012. Further analyzing the sybil attack in mitigating peer-to-peer botnets. *KSII Trans. Internet Inform. Syst.*, **6**(10):2731-2749.

Xu, J., Guo, P.Y., Zhao, M.Y., *et al.*, 2014. Comparing different moving target defense techniques. Proc. 1st ACM Workshop on Moving Target Defense, p.97-107. http://dx.doi.org/10.1145/2663474.2663486

Yackoski, J., Xie, P., Bullen, H., *et al.*, 2011. A self-shielding dynamic network architecture. Proc. Military Communications Conf., p.1381-1386. http://dx.doi.org/10.1109/MILCOM.2011.6127498

Yackoski, J., Bullen, H., Yu, X., *et al.*, 2013a. Applying self-shielding dynamics to the network architecture. *In*: Jajodia, S., Ghosh, A.K., Subrahmanian, V.S., *et al.* (Eds.), Moving Target Defense II: Application of Game Theory and Adversarial Modeling. Springer New York, USA, p.97-115. http://dx.doi.org/10.1007/978-1-4614-5416-8_6

Yackoski, J., Li, J., DeLoach, S.A., *et al.*, 2013b. Mission-oriented moving target defense based on cryptographically strong network dynamics. Proc. 8th Annual Cyber Security and Information Intelligence Research Workshop, Article 57. http://dx.doi.org/10.1145/2459976.2460040

Zaffarano, K., Taylor, J., Hamilton, S., 2015. A quantitative framework for moving target defense effectiveness evaluation. Proc. 2nd ACM Workshop on Moving Target Defense, p.3-10. http://dx.doi.org/10.1145/2808475.2808476

Zhang, J., Hu, H.P., Liu, B., 2011. Robustness of RED in mitigating LDoS attack. *KSII Trans. Internet Inform. Syst.*, **5**(5):1085-1100. http://dx.doi.org/10.3837/tiis.2011.05.012

Zhang, M., Wang, L., Jajodia, S., *et al.*, 2016. Network diversity: a security metric for evaluating the resilience of networks against zero-day attacks. *IEEE Trans. Inform. Foren. Sec.*, **11**(5):1071-1086. http://dx.doi.org/10.1109/TIFS.2016.2516916

Zhu, M.H., Hu, Z.S., Liu, P., 2014. Reinforcement learning algorithms for adaptive cyber defense against Heartbleed. Proc. 1st ACM Workshop on Moving Target Defense, p.51-58. http://dx.doi.org/10.1145/2663474.2663481

Zhu, Q.Y., Başar, T., 2013. Game-theoretic approach to feedback-driven multi-stage moving target defense. Proc. 4th Int. Conf. on Decision and Game Theory for Security, p.246-263. http://dx.doi.org/10.1007/978-3-319-02786-9_15

Zhuang, R., Zhang, S., DeLoach, S.A., *et al.*, 2012. Simulation-based approaches to studying effectiveness of moving-target network defense. Proc. National Symp. on Moving Target Research, p.1-12.

Zhuang, R., Zhang, S., Bardas, A., *et al.*, 2013. Investigating the application of moving target defenses to network security. Proc. 6th Int. Symp. on Resilient Control Systems, p.162-169. http://dx.doi.org/10.1109/ISRCS.2013.6623770

Zhuang, R., DeLoach, S.A., Ou, X.M., 2014a. A model for analyzing the effect of moving target defenses on enterprise networks. Proc. 9th Annual Cyber and Information Security Research Conf., p.73-76. http://dx.doi.org/10.1145/2602087.2602088

Zhuang, R., DeLoach, S.A, Ou, X.M., 2014b. Towards a theory of moving target defense. Proc. 1st ACM Workshop on Moving Target Defense, p.31-40. http://dx.doi.org/10.1145/2663474.2663479

Zhuang, R., Bardas, A.G., DeLoach, S.A., *et al.*, 2015. A theory of cyber attacks: a step towards analyzing MTD systems. Proc. 2nd ACM Workshop on Moving Target Defense, p.11-20. http://dx.doi.org/10.1145/2808475.2808478