Towards a Theory of Moving Target Defense

Rui Zhuang Scott A. DeLoach Xinming Ou
Kansas State University
Manhattan, KS USA
{zrui, sdeloach, xou}@ksu.edu

ABSTRACT

The static nature of cyber systems gives attackers the advantage of time. Fortunately, a new approach, called the Moving Target Defense (MTD) has emerged as a potential solution to this problem. While promising, there is currently little research to show that MTD systems can work effectively in real systems. In fact, there is no standard definition of what an MTD is, what is meant by attack surface, or metrics to define the effectiveness of such systems. In this paper, we propose an initial theory that will begin to answer some of those questions. The paper defines the key concepts required to formally talk about MTD systems and their basic properties. It also discusses three essential problems of MTD systems, which include the MTD Problem (or how to select the next system configuration), the Adaptation Selection Problem, and the Timing Problem. We then formalize the MTD Entropy Hypothesis, which states that the greater the entropy of the system's configuration, the more effective the MTD system.

Categories and Subject Descriptors

K.6.5 [**Security and Protection**]: Unauthorized access—*Management of computing and information system*

Keywords

moving target defense; computer security; network security

General Terms

Science of Security

1. INTRODUCTION

Cyber security tends to be implemented in an ad hoc and inconsistent fashion as cyber systems are implemented to satisfy critical business needs, while security is left as an afterthought. As a result, system administrators fight the continual and generally losing battle of monitoring their systems for possible intrusions and compromises, patching potential vulnerabilities, maintaining user access lists, modifying firewall rules, etc. Due to the time and complexity of maintaining such systems, once deployed, their configurations tend to remain unchanged for a long period of time.

While the static nature of cyber systems makes it easier to keep them running, it also gives attackers an extremely valuable and asymmetric advantage – time. Attackers can spend as much *time* as necessary to perform reconnaissance of the target system, study and determine its potential vulnerabilities, and choose the best *time* to launch an attack. Once compromised, this static nature also makes it easier to maintain back doors without being discovered for a long period of *time* [3]. The bottom line is simple. The static nature of modern cyber systems have made them easy to attack and compromise.

Fortunately, a promising new approach to cyber security, called the *Moving Target Defense* or MTD [19], has emerged as a potential solution to the challenge of static systems. While there are many facets of MTD, we can broadly interpret MTD as constantly changing a system to reduce or move the attack surface available for exploitation by attackers. We view the attack surface of a system as the resources accessible to attackers (e.g., software, open ports, component vulnerabilities, and other resources made available via compromises) that can be used to further penetrate the system. While it sounds promising, there is currently little research to show that MTD systems can work effectively in realistic systems. In fact, the approach is so new that there is no standard definition of what an MTD is, what is meant by attack surface, or metrics to define the effectiveness of such systems. In this paper, we propose an initial theory that we hope will begin to answer some of those questions.

1.1 Examples From the Real World

The concept of a moving target defense is not limited to the cyber world. Movement has, and always will be, a major part of military strategy and tactics. Perhaps the most obvious example of using movement as a defense measure comes in the area of air-to-air combat. In air-to-air combat, two opposing aircraft often end up in a one-on-one situation, which is typically called a *dogfight*. In dogfights, one aircraft is the attacker while the other becomes the defender. It is up to the pilot of the defending aircraft to maneuver his or her aircraft to avoid being shot down by the attacker.

In a dogfight, tactical maneuver gives the defender a greater chance of surviving as opposed to simply doing nothing. The defender attempts to maneuver to dodge incoming missiles. In this case, the pilot maneuvers in three dimensional

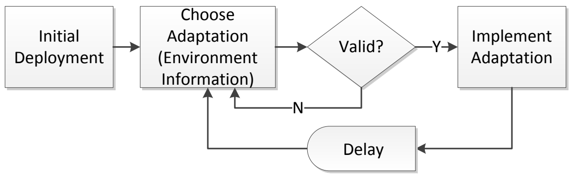

Figure 1: General MTD Process

space using tactics of defensive aircraft maneuvers (such as a high yo-yo defense, unloaded extension, high-g barrel roll, defensive spiral, etc.) to change the state of the two aircraft (location, speed, yaw, pitch, roll, etc.) while not exceeding the physical limitations of the pilot. The goal is to increase the attacker's uncertainty regarding the defender's location. The defender chooses the time of maneuver either proactively, based on training and intuition, or reactively after detecting an incoming missile.

If we compare cyber security with military combat, we look nothing like the defender in a dogfight. Instead of trying to out maneuver our attackers, we are simply content to wait and rely on outmoded techniques to survive the onslaught. We actually look much more like the defenders of the "impregnable" Maginot Line on the eve of World War II who waited in their bunkers relying on static defenses and "defense in depth." It did not work well for the Allies in 1940 either.

1.2 Moving Target Defense System Overview

The general flow of an MTD system is shown in Figure 1. The first step is the initial deployment of the system in its operational setting. Once the system is executing, an MTD system will choose an adaptation to make to its configuration. As discussed above, the adaptation choice may include environment information such as IDS alerts or execution status data. The timing of this choice can be fixed, random, or triggered by external information fed into the system. Since all operational systems have constraints and resource limitations, the configuration resulting for the chosen adaptation must be checked against these constraints to ensure that the new configuration will be valid. If it is not valid, a new adaptation will have to be chosen. Once an adaptation has passed the validity test, it can be implemented. As will be discussed later, there are several key problems that must be solved in order to make an MTD system work. As part of our theory, we will define the basic MTD problems and define the key concepts involved.

1.3 Overview and Related Work

The goal of an MTD system is to *eliminate* the attacker's asymmetric advantage of time [19]. A high-level illustration of the goals of MTD systems is shown in Figure 2. Manadhata et al. [18] introduced the *attack surface* concept to indicate the exploitable components in a computer system. The current approach to creating a more secure system is to *harden* it by reducing the attack surface, which can be accomplished by removing unneeded software, ports, etc. or by using the most current versions of software with all known vulnerabilities patched. Unfortunately, in complex systems such approaches quickly lead to convoluted firewall rules, inadequate authentication mechanisms and fragmented policies, and significant access control and credential maintenance efforts. These issues all basically guarantee a relatively long period of static configuration. As described

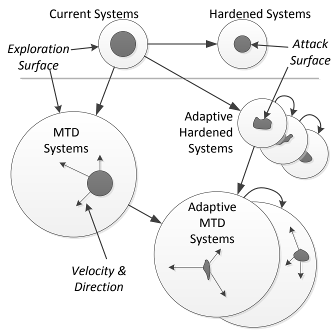

Figure 2: MTD High Level Intuition

above, this approach has proved to be less than a resounding success.

The goal of MTD systems is to adapt the exploitable aspects in the system. Adaptive hardening approaches capture input from intrusion detection systems (IDS) and reactively launch automated responses to patch or block services to thwart ongoing attacks [9, 22, 16, 26]. These automated responses change the *attack surface* at runtime, which eases the maintenance overhead of administrator. However, such approaches require significant effort to develop and maintain large numbers of signatures for identifying intrusions/malware and are ineffective against new and zero-day attacks. In addition, the effectiveness of these approaches is limited by the large number of false alarms from the IDS that can disrupt normal operations.

The goal of MTD is to use two types of motion to increase security. The first is to move the system's configuration over time while the second is to transform the configuration over time. We equate *movement* with modifying the value of a particular configuration unit of the system. For example, we could change a computer's IP address from 192.168.0.34 to 192.168.0.123. Alternatively, we define a *transformation* as changing the number or type of configuration units within a system. For instance, instead of simply changing the IP address of a computer at some point in time, we could transform the system by actually turning computers on and off within the system based on whether or not they were currently required for the system to carry out its intended purpose. We can view such a system as a graph, where the computers are the nodes and the edges are communication paths. If we perform movement within system by modifying the computer IP addresses and the port numbers, then the shape of the graph does not change. On the other hand, if we transform the system by turning on computers only when needed, the size and shape of the graph actually changes as nodes and edges are added/deleted. Of course, both should be considered as viable approaches for MTD systems. Therefore, throughout this paper we refer to movements and transformations as *adaptations*, which can refer to either of the concepts described above.

A key part of a cyber attack is performing exploration or reconnaissance to determine the configuration of the target system before launching the actual attack. Thus, we introduce the concept of the *exploration surface* to represent this space. Using again the IP address example, the exploration surface of a computer running in a typical C class subnet is the set of IPv4 addresses an attacker must search to find the computer. In this case, the exploration surface is $\{192.168.0.1, 192.168.0.2, \ldots 192.168.0.254\}$ and the size of the exploration surface is 254. However, since that computer has only one active IP address when running, the *attack surface* size is 1. A discussion of how we plan to incorporate both *exploration surface* and *attack surface* into our theory is given in Section 5.1.

Instead of focusing on reducing the attack surface, MTD approaches seek to enlarge the exploration surface during the design phase and move the attack surface at runtime to force the attacker to re-explore the exploration surface. Intuitively, by increasing the exploration surface and moving the attack surface, an attacker will spend more effort and *time* locating and re-locating vulnerabilities. Previous research such as network [2, 8]and memory address space randomization [27, 15, 23], instruction set randomization [4], host IP mutation [1], and software diversification [7, 11] all attempt to increase the difficulty and time required to discover a target systems' configuration by enlarging the exploration surface or proactively moving the attack surface. Moreover, by taking advantage of adaptive hardening approaches, more advanced MTD systems can incorporate IDS feedback to change or move the attack surface during runtime, thus increasing penetration difficulty even further [29, 31].

Although several research efforts are underway, MTD is still in its infancy. Most of the previous work focuses on specific aspects of system configuration, such as IP addresses [10, 2, 8], memory layouts [27, 15], instruction sets [13, 4], html keywords [28, 7], SQL queries [5], database table keywords [7], etc. Recently a few comprehensive frameworks [20, 14] have been proposed, but most are still conceptual and require significant theoretical and practical effort to bring them to fruition.

2. WHY A THEORY OF MTD

In the scientific world, a theory is generally something that defines a set of concepts, their relationships and principles. Theories are used to understand and explain things we observe or predict things that have not been proven. The past few years have seen a growing need within the research community to develop a science of security [25]. The motivation is to develop a systematic body of knowledge with strong theoretical and empirical underpinnings to inform the engineering of systems that can better resist known and unanticipated attack types. A theory for moving target defense will not only create a set of common terms and define essential problems, but it will also provide a framework and systematic way to think and analyze MTD problems and solutions.

2.1 A Theoretical Framework

A theoretical framework for moving target defense systems should clearly define the concepts, relationships and principles of MTD in such a way as to provide understanding of the essence of MTD and its applicability to cyber security. The theory should be consistent with research results while being complete enough to help predict new results. In addition, the theory should provide illuminate key decision points and choices in order to focus research and support key design decisions. At this point, however, simply defining key concepts such as adaptation, diversification, randomization, attack surface, and exploration surface in a way that is both formal and appropriate to the dynamic nature of MTD systems, would be a major step forward.

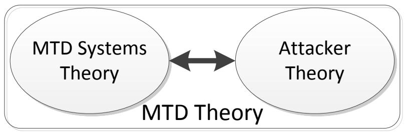

Figure 3: MTD Theory Overview

Pragmatically, the theory should be practical enough to inform design decisions during the design of MTD systems. Ultimately, the theory should be able to answer questions such as

- What features should be implemented to reduce the attacker's intrusion success likelihood?
- What are the best features to implement to defeat a given set of attack types?
- How much can diversification actually increase system security?
- How often does the system have to adapt in order to maintain security at certain level with acceptable costs?
- What is the expected benefit of implementing a specific MTD system as compared to only using traditional defenses?

Ideally, the theory should also be straightforward enough to allow MTD system designers to decide how to use existing configuration choices and diversification to increase security. It should allow them to analyze the effectiveness of adapting various combinations of configuration aspects to thwart different types of attacks. To support analysis, the framework should include an analytical model that can be used by designers to determine how different parameter settings will impact system security.

2.2 Approach

Obviously, the theoretical framework described above cannot be developed and presented in a single paper. Although this paper only presents a small first step toward development of a complete theory, we will layout our approach to developing the overall theory, followed by a description of what is actually accomplished in this paper.

Our high level approach to developing a theory for MTD is shown in Figure 3. The first step, which we start in this paper, is to develop a theory of MTD Systems. This theory focuses only on the system itself and how it adapts over time to achieve its overall goals. The second step is to develop an Attacker Theory, which will describe an attacker's goals and the actions they can take to reach their goals. The final step will be to combine the two into an overall MTD Theory. The objective of MTD Theory will be to define how elements of the MTD Systems and Attacker theories interact. This is especially important in being able to understand the true effect of an MTD system as its effectiveness only makes sense in light of actions from an attacker for a specific type of attack.

Once the base theory is in place, we plan to define an analytical model that is capable of comparing the effectiveness of various MTD systems using different design parameter settings. These design parameter settings will include the aspects of the system configuration that can be changed, the type and timing of the adaptations, the system diversification, etc. These design parameters will be able to be analyzed against a set of attack types for their comparative effectiveness and costs.

In this paper we start the process of developing the MTD Systems Theory. We start in Section 3 by defining the concepts of a configurable system, system goals, and policies as the basis of the definition of an MTD system. We then propose definitions for adaptation, diversification, and randomization based on the previous definitions. Next, in Section 4, we present specific problems of interest that are raised by the concepts defined so far. We also formally propose the Entropy Hypothesis, which we have heard discussed informally by several researchers in conferences and symposia. Finally, we wrap up with a discussion of the implications of MTD Systems Theory and the next steps towards a complete MTD Theory in Section 5.

3. CONCEPTS AND DEFINITIONS

In this section we define the key concepts required to formally talk about MTD systems and their basic properties. The focus is on formally defining an MTD system, although we do discuss adaptation, diversification, and randomization in light of that definition as well. Since the essence of an MTD system is adapting the configuration of the system over time, we start by defining what we mean by a configurable system, borrowing terms and concepts from existing configuration management theory [6].

3.1 Configurable System

When we talk about configuring a system, we generally refer to the physical devices that are part of a system, the software installed on those devices, and the settings of that software. For our purposes, we define these elements of a configuration as a configuration parameter (borrowing from [6]) that can take on various values to specify the details of the configuration.

3.1.1 Configuration Parameter

We start by defining a configuration parameter as a variable with an associated type.

DEFINITION 3.1. *A* configuration parameter*, π, is a unit of configuration information that can take on a value based on its type.*

DEFINITION 3.2. *A* configuration parameter type, Π*, is a label identifiable with the domain of possible values that the parameter can assume [6]. We denoted the associated domain of a configuration parameter π_i as Π_i.*

In essence, a configuration parameter can be viewed as a variable to which we can assign values. These values can be used to describe a piece of hardware, the software installed on that hardware, or the settings of the software itself. Examples of configuration units can be a specific physical or virtual machine host, the amount of memory installed, the speed of the processor, the operating system installed, the IP address of the host, the ports that are open, etc. While some configuration parameters are basically fixed (e.g., the size of memory in a physical host), we are obviously interested in the configuration parameters that can be modified during execution (the size of memory in a virtual host or the IP address of a host). We refer to a configuration parameter that can be modified during execution as an adaptable configuration unit.

To capture the configuration of an entire system or a complex component of that system, we introduce the notion of composite configuration parameters.

DEFINITION 3.3. *A* composite configuration parameter*, π, is a configuration parameter that is composed of a set of sub configuration parameters, $\pi = \langle \pi_1, \pi_2, \ldots, \pi_n \rangle$. The domain of a composite configuration parameter π is derived from the sub configuration parameter domains,* $\Pi = \Pi_1 \times \Pi_2 \times \ldots \times \Pi_n$.

Thus, a single virtual host may have a composite configuration parameter associated with it to describe its overall configuration. Appropriate sub configuration parameters might include the host memory size, hard disk size, CPU type, operating system, application software, IP address, open ports, etc.

3.1.2 Configuration State

The process of reconfiguration, which is at the heart of MTD systems, is the process of moving from one configuration to another. To capture the notion of a specific configuration, we introduce the notion of a configuration state. Here we typically assume we are talking about the overall configuration of a system, but we can also talk about the configuration states as part of that system as well.

DEFINITION 3.4. *A* configuration state*, s, is a unique assignment of value(s) from Π to a configuration parameter π. An assignment of some value z in Π to π is denoted as $\pi \leftarrow z$. If π is a composite configuration parameter and $\pi = \langle \pi_1, \pi_2, \ldots, \pi_n \rangle$, then s is a configuration state of π if $s = \langle s_1, s_2, \ldots, s_n \rangle$ where $\forall i \in [1, n], s_i \in \Pi_i \wedge \pi_i \leftarrow s_i$.*

Therefore, we can now discuss the current configuration of a system. Using the example of the host configuration parameters above, the configuration state would be an assignment of valid values to each of the configuration parameters. For instance, host memory size = 4gb, hard disk size = 100gb, CPU type = Intel i7, operating system = Ubuntu 14.04 LTS, IP address = 192.168.0.54, etc.

3.1.3 Action

To change one configuration into another configuration requires actions to be taken on the part of the system administrator. Traditionally, these actions are performed manually, although recently new configuration management tools have automated much of the tedious nature of these actions. However, ultimate control of what to change still resides with the administrators.

DEFINITION 3.5. *A configuration* action*, α, is an operation that can modify the value of an existing configuration parameter π, or add/delete a configuration sub parameter from a composite configuration parameter. When adding a sub parameter, we assume the parameter is initialized with a valid value. An action can also be composed of a sequence of sub actions.*

See Section 3.3 for what a valid value means.

3.1.4 Configurable System

We have now defined the configuration of a system as a set of configuration parameters that can be modified by a set of configuration actions. We combine these concepts with configuration states to define a configurable system, upon which we build our definition of an MTD system.

Definition 3.6. *A* configurable system *is a labeled transition system, $\Gamma = (S, \Lambda, \tau)$, where $S = \{s_1, s_2, \ldots\}$ is a finite or recursively enumerable set of configuration states the system can be in, $\Lambda = \{\alpha_1, \alpha_2, \ldots\}$ is a finite or recursively enumerable set of actions, and $\tau : S \times \Lambda \to S$ is the state transition function.*

We based the configurable system definition on labeled transition systems as the essence of configuration management is change. Here the set of allowable changes is captured in the state transition function, τ. Til now, we have ignored the notion of configuration consistency and validity, which we address in the next section using system goals and constraints.

3.2 System Goals

The heart of the MTD paradigm is adapting the system to keep an attacker from taking the time required to successfully attack and compromise the system. However, a potential problem associated with this constant adaptation is that we do not intentionally change the system in a way that keeps it from achieving its intended mission. It has recently been recognized by several in the adaptive systems community that the key to *effective* adaptive systems is explicitly modeling the requirements or objectives of the system [21]. Understanding the objectives of the system is critical in making the determination of what is a *valid* adaptation. Therefore, we define the notion of system goals, which we will use later to define the notion of valid adaptations.

Definition 3.7. *A* goal*, g, captures an intended function of a computer system. There are two types of goals of interest: operational goals and security goals. Each system has a set of goals, G that is captured by a tuple $\langle G_o, G_s\rangle$, where $G_o = \{g_{o1}, g_{o2}, \ldots g_{oj}\}$ represents the operational goals of the system and $G_s = \{g_{s1}, g_{s2}, \ldots\ldots g_{sk}\}$ represents the security goals of the system.*

Operational goals capture the mission the system was built to support. An operational goal is a high-level concept whose purpose is to organize the IT elements of the system around business objectives. The goals are typically espoused in the business terms of the users. The operational goals are guideposts used to ensure that any adaptations to the system configuration still support the overall mission. Security Goals, on the other hand, define the critical parts of the system that should be protected. If an eCommerce website relies on a database of consumer information, then protecting that database becomes an important security goal.

While a system can be in a variety of configuration states, only some of those states are actually capable of achieving the system's operational goals. For instance, an eCommerce system may be in a configuration that does not include a customer database server (thus ensuring the security goal not to allow the database to be compromised). However, this configuration is not really helpful since one of the key goals of the system is to allow customers to log in and buy items through the website. Therefore, we define a relation between configurations of the system and the goals that those configurations achieve. We use this relation later to ensure that as we adapt, our system is capable of achieving its overall operational goals.

Definition 3.8. *If a goal g can be realized in a system in configuration state s, we say that s* achieves *g and define a relation* achieves: *$G \times S$ to capture this relationship. If $\forall g \in G, \langle s, g\rangle \in achieves$, we say s achieves a set of goals G, denoted as $s \vdash G$.*

As discussed in Section 2.2, our eventual objective is to define an Attacker Theory as well as an MTD Systems Theory and then show they are related. We envision attacker's having their own set of goals and thus it seems obvious that when the goals of the attacker and the MTD system conflict, then the MTD system must be able to take steps to counter that attack.

3.3 System Policies

To ensure the parts of a system can function together efficiently and effectively, every system has a set of policies that define how the system can and cannot be structured. Often, these policies are implicit, which leads to significant problems when changes are made to the system without understanding all the pertinent policies. Ideally, an MTD system will require system designers to explicitly state these policies so that the MTD system can reason over them before making adaptions.

Definition 3.9. *A* policy*, p, defines a restriction on the configuration state of a system. Each system has a finite or recursively enumerable set of policies, $P = \{p_1, p_2, \ldots\ p_l\}$. The aggregated restrictions of system policies P on system configuration states S define a relation* satisfies: *$S \times P$. We say a state s* satisfies *a set of policies P, denoted as $s \succ P$, if $\forall p \in P, \langle s, p\rangle \in satisfies$.*

If a configuration policy is violated the system will not operate as intended. Thus, as the system adapts, it is critical that it does not move to an *inconsistent state* that violates policies. Next we define a consistent configuration state.

Definition 3.10. *A system is said to be in a* consistent state *s if $s \succ P$, where P represents the current system policies. The set of all consistent states of a system is denoted as $S_c = \{s | \forall s \in S \wedge s \succ P\}$. Any state that is not a consistent state is an inconsistent state.*

While being in a consistent state is necessary, we still need to ensure the system is in a state so that it can achieve its intended goals. Thus we use the definition of achieves from Definition 3.8 to define a valid state.

Definition 3.11. *A* valid state *s is a consistent state that is capable of achieving all of the existing system operational goals, G, i.e., $s \vdash G$. We define the set of* all valid states *of a given system, S_v, as $S_v = \{s | \forall s \in S, s \succ P \wedge s \vdash G\}$.*

Knowing a state is consistent and valid is very useful. However, when we are dealing with an entire system, we need to be able to reason about the configurations that are really of interest – those that define a complete system for the purposes of achieving the operational goals of the system. Thus, we define a complete configuration as one that has all the configuration parameters required to configure a system that is actually capable of achieving the goals of the system.

Definition 3.12. *A* complete configuration *is a configuration parameter whose value can be a valid state. Formally, this is stated as $\exists s \in S_v, \pi \leftarrow s$.*

Since a complete configuration might also contain unnecessary configuration parameters, we need to restrict it to the configuration information that is required to achieve the overall goals of the system. We do this by defining a minimum complete configuration. Again, it should be noted that this a minimum complete configuration whose valid states can achieve the system goals G.

DEFINITION 3.13. *A* minimum complete configuration *is a complete configuration parameter that has the minimal required parameters. Formally, this is stated as* $\nexists \pi_i \in \pi$, *such that* $\exists s \in S_v, \pi - \{\pi_i\} \leftarrow s$.

3.4 Adaptation

While we have defined actions above as operations that take a system from one configuration state to another, what we are really interested in is transforming a system from one configuration state to a *valid* configuration state. This will be the basis of defining the kind of MTD systems of interest and will be discussed further in the next section. To handle this specific type of transformation operation, we now define an adaptation as a restriction on the set of actions.

DEFINITION 3.14. *An* adaptation *is a sequence of actions* $A = \{a_1, a_2, \ldots, a_k\}$ *that transforms a system from a state* s *to a valid state* s_v.

We have now defined the necessary concepts to allow us to define an MTD system, which we do in the following section.

3.5 MTD System

An MTD system is a configurable system that can adapt its configuration during execution. As discussed in the previous section, an MTD system should also be able to configure itself so that it is always in a consistent state that can achieve the overall goals of the system.

DEFINITION 3.15. *A* moving target defense (MTD) system Σ *is a tuple* $\langle \Gamma, G, P \rangle$, *where* Γ *is a configurable system,* G *is the set of goals which includes both operational goals and security goals and* P *is the set of polices.*

Notice that although S in Γ will be restricted to S_v by G and P, an MTD system is still defined using S, the set of all configuration states of Σ, instead of S_v. This is because an MTD system is not necessarily always in a valid state. Initial system configurations or operational failures all could lead the MTD system to an invalid state. However, as defined above, when we apply an adaptation to MTD system, it should result in a valid state.

In order to reason and discuss the configuration of an MTD system, we must be able to refer to the *current* configuration of a system at any given point in time. Since we need to reason about the entire configuration, the configuration must be complete as defined in Definition 3.12. Since we do not want to include extraneous configuration information, it must be minimum as defined in Definition 3.13. As configuration actions may add or remove configuration parameters from a composite configuration parameter, the configuration state of an MTD system can change in terms of the configuration parameters as well as their values. We call the current configuration of an MTD system as its configuration state.

DEFINITION 3.16. *At any point in time, each MTD system* Σ *has a minimum complete configuration denoted* Σ_π. Σ_π *also has a unique value,* $s \in S$, *which is called the* configuration state *of* Σ.

3.6 Configuration Space

The set of all valid states S_v captures the overall *configuration space* of an MTD system's valid configuration. This can also be derived from the domain of Σ_π, which has been defined in Definition 3.3.

DEFINITION 3.17. *The* configuration space *of an MTD system* Σ *with configuration parameter* Σ_π *is the domain of* Σ_π, *which is* S_v. S_v *can be computed as* $S_v = \prod \Pi_i$, *where the cross product operation should obey constraints such that* $\forall s \in S_v, s \succ P \wedge s \vdash G$.

As defined above, S_v can be computed as the cross product from Π_i of each configuration parameter π_i, where the cross product operation should ensure that the result is a valid state. Each Π_i actually captures the configuration space of the configuration parameter π_i.

As discussed earlier, a critical part of a cyber attack is the reconnaissance or exploration of the target system's configuration. To understand the effect of an MTD system, we must be able to characterize the space – the *exploration surface* – that the attacker must explore before attacking. Clearly, the exploration surface is related to the *configuration space* of the MTD system. If we assume the attacker knows the exact domain of the configuration parameter Σ_π in an MTD system, then the exploration surface equals the configuration space. But obviously, this assumption is often not the case. For instance, if an attacker is looking for a specific target host in an IPv4 system, the attacker need check each possible IP address, or 256 different addresses (assuming a /24 address). However, if the configuration constraints of the MTD system limits the possible IP address of the host to a range of $100 \ldots 255$, the configuration space does not truly reflect the exploration surface of the attacker. As the exploration surface is associated with ongoing attacks, we leave its definition until after defining attacker theory (more is said on this in Section 5.1).

3.7 Diversification

In the MTD literature, diversification typically has two related concepts. The first refers to the number of configuration choices available in the system, while the second refers to a technique to increase the number of configurations available to the MTD system. Based on our definitions above, we can see that these both refer to the configuration space. The first definition relates to the size of the configuration space while the second concerns a variety of techniques used to increase the size of the configuration space. To eliminate this confusion of terms, we propose to use the term *artificial diversification* to refer to the second definition.

DEFINITION 3.18. *The* diversification *of an MTD system is the cardinality of its configuration space, or* $|S_v|$. *We can also refer to the diversification of any configuration parameter* π, *which is simply the cardinality of its domain* $|\Pi|$.

DEFINITION 3.19. Artificial Diversification *is a technique to increase the configuration space,* S_v, *of an MTD system.*

Kant [12] points out that attack surface modification can be aided by introducing diversity, while Christodorescu [7] cites the need to understand the impact of diversification on the attacker. As shown in our definitions, diversification provides the potential to measure security of an MTD system through configuration space. In an MTD system, a system with a higher diversification is likely to be more difficult to compromise than one of lower diversification, given that all else is equivalent. Artificial diversification seeks to enhance the security provided by an MTD system by introducing functionality equivalent alternatives for a configuration parameter. Diversification determines the size of the configuration space that provides the space for adaptation.

3.8 Randomization

Randomization is another term often used in MTD literature. Of course, there is nothing in the general understanding of MTD systems that requires random choices, although there are good arguments for its use. The overall goal of randomization is to make full use of the available configuration space introduced by diversification, where a larger configuration space provides more space for adaptation.

In MTD systems, randomization typically refers to choosing random configurations in order to make full use of configuration space while adding the notion of non-predictability. Random choices are usually understood as making selections assuming that the probability of each choice forms a uniform distribution. In previous work [31], we proposed both a purely random MTD system, which select the next configuration via a uniform distribution, as well as an *intelligent* MTD system, which also considers potential vulnerabilities and attack alerts when choosing configurations.

If we view randomization as a decision making process that chooses the next valid state based on a specific probability distribution over states in S_v, then choosing the next configuration from a uniform distribution of states as well as considering alerts become specific instances of randomization. Indeed, given S_v and a specific set of environmental information, if we want to ensure that state $s_i \in S_v$ is chosen, we simply define a probability distribution that says the probability of picking s_i is 1. This way, randomization as a decision making process generalizes so that all kinds of AI techniques, such as genetic algorithms, machine learning, game theory, etc. all are just different ways to select the appropriate probability distributions over states S_v. Thus, we define the randomization as follows.

Definition 3.20. Configuration randomization *is a decision making process of selecting the next valid system configuration value $s \in S_v$ for Σ_π. If P_j represents the probability that s_j is chosen and p_j represents a specific probability value assigned to P_j through randomization, then $\forall s_j \in S_v, 1 \leq j \leq |S_v|$, we have $P_j = p_j \wedge 0 \leq p_j \leq 1 \wedge \sum_j p_j = 1$.*

4. PROBLEMS AND HYPOTHESES

Using the terminology and concepts developed in Section 3, we now discuss the implications of these concepts in the MTD process as shown in Figure 1. First we discuss some problems that we believe are common to any MTD system that follows from MTD Systems Theory. We then discuss a common assumption made about MTD systems.

4.1 Problems

To carry out a process similar to the one described in Figure 1, there are three essential problems:

1. How to select the next configuration state of the MTD system.
2. How to select the adaptations to take to get to the next configuration state.
3. When to carry out the adaptations to actually change the state of the system.

Each of these interrelated problems are discussed below.

4.1.1 MTD Problem

The essential problem of an MTD system is to move the system from current state to a valid state in such a way as to make an attackers job of compromising the system more difficult. Thus, the key problem will be deciding what state to move to.

Definition 4.1. *Given the current state, s, of an MTD system, the* MTD problem *is how to choose the next configuration state of the system, s', subject to $s' \in S_v$, to increase the effectiveness of the MTD system.*

The definition makes it clear that we are only talking about choosing a valid configuration state. In addition, measuring effectiveness requires additional theory to define the attacker as well as the relationships as discussed in Section 2.1. Actually, there are several approaches that could be taken including random selection, intelligent selection based on intrusion detection alerts, or cost-based strategies. We believe this will be a prime area of MTD research in the future.

4.1.2 Adaptation Selection Problem

The solution to the MTD problem will lead to a valid next state s' being chosen. However, moving the system to this state requires solving the adaptation selection problem, which can be stated informally as finding an adaptation that can transition the system from s to s'.

Definition 4.2. *Given the current state of an MTD system, s and a valid next state, s', the* adaptation selection problem *is how to synthesize a sequence of actions $A = \{a_1, a_2, \ldots, a_k\}$ such that $\tau : s \times A \rightarrow s'$. This problem may also consider constraints such as time and costs.*

This problem is analogous to a planning problem and thus future research will likely lead in that direction for the general case. The complexity lies in that there could be multiple sequence of actions that could result in the same configuration state, the optimal solution would require to take constraints such as time and costs into consideration.

4.1.3 Timing Problem

The final problem we consider is the MTD Timing Problem, or "when to adapt". Timing is a critical factor in the success of an MTD system. Our previous work provides some insight into the relationship between several key MTD system factors which include the adaptation time interval T_r and attack time interval T_a [30].

Definition 4.3. *In an MTD system, the* Timing Problem *is at what point should the MTD system launch an adaptation to increase the effectiveness of the MTD system, while maintaining a reasonable cost, c.*

Here the reasonable means the decision of when to launch the adaptation should be made based on the trade off between operation and security. As discussed, after the Attacker theory is in place, we will be able to start reason over the interactions between MTD system and the attacker to make such decision.

4.2 MTD Entropy Hypothesis

As we have written papers, submitted proposals, and attended workshops and symposia on MTD, we have heard a general belief that the more random and chaotic an MTD system is, the more effective it should be. As a way to test our power of the theory presented in this paper, we formalized that assumption in what we call the *MTD Entropy Hypothesis*.

Definition 4.4. *We can state the* MTD Entropy Hypothesis *as follows: the greater the entropy of the configuration of an MTD system, the more effective the MTD system will be.*

This belief (at least among those in the MTD community) is that we can use diversification, randomization and adaptation together to increase the uncertainty of a system's configuration and make it harder for an attacker to compromise the system. Of course, as a hypothesis, this has yet to be proved. However, the proof of the hypothesis could yield some interesting theoretical results. We provide some initial insights into the effect of such a finding below. First we define the concept of entropy as related to MTD systems. If we treat Σ_π as a random variable over S_v then we can use the mathematical formulation of entropy based on Shannon's information entropy [24] and use it to introduce configuration entropy.

Definition 4.5. *Let $H_C(\Sigma)$ represent the* configuration entropy *of MTD system Σ. Since S_v includes all possible states of the Σ's configuration parameter Σ_π, then we can define the configuration entropy of Σ as*

$$H(\Sigma)_C = H(\Sigma_\pi) = -\sum_{s \in S_v} p(s)\ log(p(s))$$

Given this definition, there are a couple of obvious statements we can make about this configuration entropy. The first is that the overall configuration entropy is less than or equal to the sum of all the sub configuration parameter's entropy.

Theorem 4.1. *An MTD system's configuration entropy is less than or equal to the summation of sub configuration parameter's entropy. Given an MTD system Σ, let $\Sigma_\pi = \pi = \langle \pi_1, \pi_2, \ldots, \pi_n \rangle$, this can be denoted as:*

$$H(\Sigma_\pi) = H(\pi_1, \pi_2, \ldots, \pi_n) \leq \sum_{i=1}^{n} H(\pi_i)$$

Proof. According to the chain rule for entropy, for random variables $X_1, X_2, \ldots, X_n$, we have $H(X_1, X_2, \ldots, X_n) = \sum_{i=1}^{n} H(X_i \mid X_{i-1}, \ldots, X_1)$, thus for a given MTD system Σ and random variable $\Sigma_\pi = \pi = \langle \pi_1, \pi_2, \ldots, \pi_n \rangle$, then

$$H(\Sigma_\pi) = H(\pi_1, \pi_2, \ldots, \pi_n) = \sum_{i=1}^{n} H(\pi_i \mid \pi_{i-1}, \ldots, \pi_1) \quad (1)$$

According to the conditioning reduces entropy theorem, for random variables X, Y, we have $H(X \mid Y) \leq H(X)$ and with equality holds if and only if X and Y are independent, thus

$$H(\pi_i \mid \pi_{i-1}, \ldots, \pi_1) \leq H(\pi_i) \quad (2)$$

Combine equation 1 and 2 and the theorem is proved. □

A corollary to Theorem 4.1 is the well known condition of entropy that the entropy is maximized when sub parameters are mutually independent.

Corollary 4.1. *An MTD system's configuration entropy is* maximized *when $\pi_1, \pi_2, \ldots, \pi_n$ are mutually independent.*

Proof. The proof of this corollary can be derived from Theorem 4.1 when the equality condition is true. □

This corollary would suggest that, in practice, an MTD system should attempt to reduce the constraints and dependencies between different sub configuration states. For example, assume we have two different operating systems (OS) to choose from to implement three different implementations of a service. This corollary states that implementing all three variations on both OS will reduce the dependency between OS type and service implementation and thus increase the configuration entropy of the system.

It is also important to understand the theoretical maximum entropy that can be achieved by an MTD system, which is given below.

Theorem 4.2. *Given an MTD system Σ, let $\Sigma_\pi = \pi = \langle \pi_1, \pi_2, \ldots \pi_n \rangle$, the* maximum configuration entropy *of Σ can achieve, $H(\Sigma)_C = H(\Sigma_\pi)$, is $\sum_{i=1}^{n} log(\Pi_i)$.*

Proof. As each π_i is a random variable, $H(\pi_i) \leq log|\Pi_i|$ where the equality is satisfied if and only if π_i is distributed uniformly on Π_i, thus

$$H(\Sigma_\pi) = H(\pi_1, \pi_2, \ldots, \pi_n) \leq \sum_{i=1}^{n} H(\pi_i) \leq \sum_{i=1}^{n} log|\Pi_i|.$$

□

From Theorem 4.2, we can see that this maximum entropy can only be achieved when the sub configuration parameters π_i of Σ_π are mutually independent and the MTD process ensures that π_i is uniformly distributed on Π_i.

5. DISCUSSION

Clearly, this paper represents only an initial foray into the theory of MTD. We have yet to tackle timing elements of the system that describe how often an MTD system adapts and how long adaptation can take. As described in Section 2.2, in addition to the MTD Systems Theory that we describe here, an Attacker Theory must also be defined to help understand the characteristics and effectiveness of MTD systems. Attacker theory will capture the key elements of an attacker's goals and attacks that can be carried out to achieve those goals. Each type of attack requires knowledge of certain information and is focused on vulnerabilities related to specific aspects of a system's configuration. Once these two key theoretical elements are developed, we can actually start to tie them together in a complete MTD Theory. MTD Theory will be able to discuss the interplay between system goals and attacker goals as well as the more technical aspects of attacks and adaptations.

As stated earlier, the goal of an MTD system is to *eliminate* the attacker's asymmetric advantage of time. Curiously, MTD Systems Theory as currently presented hardly mentions it at all except in its discussion of the Timing Problem. Clearly time must be a major part of any theory of MTD and it will be in ours as well. However, the main importance of time comes when an attacker makes an intrusion attempt at an MTD system. In this case, the key time elements involved are the interval between adaptions, the time it actually takes the system to adapt, and the time it takes for an attack to occur, from the required reconnaissance phase through system compromise. These timing parameters, along with the how configurations changes made by the MTD system interacts with various types of attacks, will go a long way in defining the impact and effectiveness of an MTD system.

Quantifying the effectiveness of MTD systems is still an open issue. However, the development of an attacker theory will greatly benefit our understanding of the interaction between the attacker goals and MTD system goals. This will likely include an understanding of the cost factors related to attacker and MTD actions, which we believe will be strongly related to the time issues discussed above. Once this interaction is better understood, we should be able to quantify the effectiveness of MTD systems and create an analytical model to analyze it. This analytical model will greatly benefit the key decision makings involved in both MTD system design and MTD runtime parameter settings to help balance the operation and security goals.

As part of this theoretical development, we will constantly need to verify the theory against real systems and applications. While we are keeping the theory as broad as possible, we realize that it will be impossible for a single theory to fit all possible system types. Our hope is that there will be a core MTD Theory that will be broadly applicable, with extensions developed to meet the needs of sub-communities.

5.1 A New Definition for Attack Surface

One area that will need significant work in relation to MTD Theory is a new, or extended definition of an attack surface. Huang [11] points out that existing attack surface definitions [18] are not suitable for evaluating a moving *attack surface* due to the violation of two basic assumptions of the original definition. The first violated assumption is that the attack surface remains unchanged during an attack. This is the whole purpose of MTD systems. The second assumption, that the target attack surface is always reachable by attackers is also violated by MTD systems as the target itself may change its configuration during the attack. Therefore, a new attack surface definition is needed for an MTD's changing and unpredictable nature.

Manadhata extended his original attack surface definition in [17] to include definitions for a shifting attack surface. This extension supports modeling the interaction between the defender and the attacker as a two player game, using game theory to determine optimal defense strategies. However, as admitted by Manadhata, the potential state and action space explosion are serious problems. The paper also leaves the instantiation of the model as future work.

While we introduced the concepts of exploration and attack surfaces in Section 1.3, we cannot yet precisely define those concepts. One reason is that both exploration and attack surfaces are closely tied to the attacks being proposed. Thus, once we have a definition of attacker theory, we should be able to define the exploration surface and attack surface in terms of the potential attacks being proposed, the MTD system in place, and timing issues. We believe this approach will produce more powerful definitions to capture the dynamic and changing nature of MTD system.

6. CONCLUSIONS

This paper presented a start toward the definition of MTD Theory, which combines MTD System Theory and Attacker Theory. Specifically, it defines the basic concepts and problems associated with an MTD system. While the work is far from complete, the paper should be an interesting place to start discussions about the foundations of MTD systems and the theory required to support the research, development, and implementation of MTD systems to combat the growing threat of cyber attacks in today's world.

We founded MTD Theory on configuration management theory in order to define a configurable system, upon which we built the definition of an MTD system. The theory in this paper also builds on years of adaptive systems research and captures the goals (both operational and security) and policies (or constraints) of the system as essential elements to determine what should be considered complete and valid configurations of an MTD system. Based on that, we defined the configuration space and informally discussed its relation to a new concept introduced in this paper, called the exploration surface, to help capture the required reconnaissance effort that an MTD system will impose on an attacker. We also formally defined the concepts of diversification and randomization and showed how they relate to MTD systems and how their use can increase the effectiveness of those systems.

We also formally defined key problems and hypothesis related to MTD systems. Specifically, we defined the essential problem of an MTD system as how to select the next valid configuration state of the system. This problem drives the solution of the other problems such as the Adaption Selection Problem and the Timing Problem. Finally, we introduced the Entropy Hypothesis as a commonly held belief whose truthfulness still requires evidence.

We understand that this formalization is not unique in its ability to define MTD concepts. However, it is our hope that the MTD community will grasp the usefulness of basing work in theory as well as practice and that a significant body of theoretical MTD work will develop along with the implementation of various systems and techniques.

7. ACKNOWLEDGMENTS

This work was supported by the Air Force Office of Scientific Research award FA9550-12-1-0106 and U.S. National Science Foundation awards 0954138 and 1018703. Any opinions, findings and conclusions or recommendations expressed herein are those of the authors and do not necessarily reflect the views of the above agencies.

8. REFERENCES

[1] E. Al-Shaer. Toward network configuration randomization for moving target defense. In *Moving Target Defense*, pages 153–159. Springer, 2011.

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

[3] D. Barrett. Hackers penetrate NASDAQ computers. http://online.wsj.com/article/, Feb. 2011.

[4] S. W. Boyd, G. S. Kc, M. E. Locasto, A. D. Keromytis, and V. Prevelakis. On the general applicability of instruction-set randomization.

Dependable and Secure Computing, IEEE Transactions on, 7(3):255–270, 2010.

[5] S. W. Boyd and A. D. Keromytis. SQLrand: Preventing SQL injection attacks. In *Applied Cryptography and Network Security*, pages 292–302. Springer, 2004.

[6] M. Burgess and A. L. Couch. Modeling next generation configuration management tools. In *LISA*, pages 131–147, 2006.

[7] M. Christodorescu, M. Fredrikson, S. Jha, and J. Giffin. End-to-end software diversification of Internet services. In *Moving Target Defense*, pages 117–130. Springer, 2011.

[8] M. Dunlop, S. Groat, W. Urbanski, R. Marchany, and J. Tront. MT6D: a moving target ipv6 defense. In *Military Communications Conference, 2011-MILCOM 2011*, pages 1321–1326. IEEE, 2011.

[9] B. Foo, Y. Wu, Y. Mao, S. Bagchi, and E. Spafford. Adepts: Adaptive intrusion response using attack graphs in an e-commence environment. In *Proceedings of the International Conference on Dependable Systems and Networks*, pages 508–517, Piscataway, NJ, 2005.

[10] S. Groat, M. Dunlop, R. Marchany, and J. Tront. Using dynamic addressing for a moving target defense. In *Proceedings of the 6th International Conference on Information Warfare and Security. Academic Conferences Limited*, page 84, 2011.

[11] Y. Huang and A. K. Ghosh. Introducing diversity and uncertainty to create moving attack surfaces for web services. In *Moving Target Defense*, pages 131–151. Springer, 2011.

[12] K. Kant. Configuration management security in data center environments. In *Moving Target Defense*, pages 161–181. Springer, 2011.

[13] G. S. Kc, A. D. Keromytis, and V. Prevelakis. Countering code-injection attacks with instruction-set randomization. In *Proceedings of the 10th ACM conference on Computer and communications security*, pages 272–280. ACM, 2003.

[14] A. D. Keromytis, R. Geambasu, S. Sethumadhavan, S. J. Stolfo, J. Yang, A. Benameur, M. Dacier, M. Elder, D. Kienzle, and A. Stavrou. The meerkats cloud security architecture. In *Dist. Computing Systems Workshops, 2012 32nd International Conference on*, pages 446–450. IEEE, 2012.

[15] C. Kil, J. Jun, C. Bookholt, J. Xu, and P. Ning. Address space layout permutation (aslp): Towards fine-grained randomization of commodity software. In *Computer Security Applications Conference, 2006. ACSAC'06. 22nd Annual*, pages 339–348. IEEE, 2006.

[16] W. Lee, W. Fan, M. Miller, S. Stolfo, and E. Zadok. Towards cost-sensitive modeling for intrusion detection and response. *Journal of Computer Security*, 10(1-2):5–22, 2002.

[17] P. K. Manadhata. Game theoretic approaches to attack surface shifting. In *Moving Target Defense II*, pages 1–13. Springer, 2013.

[18] P. K. Manadhata and J. M. Wing. An attack surface metric. *Software Engineering, IEEE Transactions on*, 37(3):371–386, 2011.

[19] NITRD. National Cyber Leap Year Summit 2009 co-chairs' report, networking and information technology research and development. Technical report, National Office for the Federal Networking and Information Technology Research and Development Program, Sept. 2009.

[20] G. Portokalidis and A. D. Keromytis. Global ISR: Toward a comprehensive defense against unauthorized code execution. In *Moving Target Defense*, pages 49–76. Springer, 2011.

[21] P. Sawyer, N. Bencomo, J. Whittle, E. Letier, and A. Finkelstein. Requirements-aware systems: A research agenda for re for self-adaptive systems. In *Requirements Engineering Conference (RE), 2010 18th IEEE International*, pages 95–103. IEEE, 2010.

[22] D. Schnackenberg, H. Holiday, R. Smith, and et al. Cooperative intrusion traceback and response architecture citra. In *DARPA Information Survivability Conference and Exposition I*, 2001.

[23] H. Shacham, M. Page, B. Pfaff, E.-J. Goh, N. Modadugu, and D. Boneh. On the effectiveness of address-space randomization. In *Proceedings of the 11th ACM conference on Computer and communications security*, pages 298–307. ACM, 2004.

[24] C. E. Shannon. A note on the concept of entropy. *Bell System Tech. J*, 27:379–423, 1948.

[25] M. P. Singh. Towards a science of security. `http://www.computer.org/portal/web/computingnow/archive/january2013`, 2013. Online, accessed June 30, 2014.

[26] N. Stakhanova, S. Basu, and J. Wong. A taxonomy of intrusion response systems. *International Journal of Information and Computer Security*, 1(1/2):169–184, 2007.

[27] P. Team. PaX address space layout randomization (ASLR), 2003.

[28] S. Vikram, C. Yang, and G. Gu. Nomad: Towards non-intrusive moving-target defense against web bots. In *Communications and Network Security (CNS), 2013 IEEE Conference on*, pages 55–63. IEEE, 2013.

[29] J. Yackoski, J. Li, S. A. DeLoach, X. Ou, and A. Singhal. Mission-oriented moving target defense based on cryptographically strong network dynamics. In *CSIIRW '12: Eight Annual Workshop on Cyber Security and Information Intelligence Research*, New York, USA, 2013. ACM.

[30] R. Zhuang, S. A. DeLoach, and X. Ou. A model for analyzing the effect of moving target defenses on enterprise networks. In *Proceedings of the 9th Cyber and Information Security Research Conference*, Oak Ridge, Tennessee, April 2013. ACM.

[31] R. Zhuang, S. Zhang, A. Bardas, S. A. DeLoach, X. Ou, and A. Singhal. Investigating the application of moving target defenses to network security. In *Resilient Control Systems (ISRCS), 2013 6th Intl Symposium on*, pages 162–169. IEEE, 2013.