A Framework for Moving Target Defense Quantification

Warren Connell, Massimiliano Albanese(✉), and Sridhar Venkatesan

George Mason University, Fairfax, VA 22030, USA
{wconnel2,malbanes,svenkate}@gmu.edu

Abstract. Moving Target Defense (MTD) has emerged as a game changer in the security landscape, as it can create asymmetric uncertainty favoring the defender. Despite the significant work done in this area and the many different techniques that have been proposed, MTD has not yet gained widespread adoption due to several limitations. Specifically, interactions between multiple techniques have not been studied yet and a unified framework for quantifying and comparing very diverse techniques is still lacking. To overcome these limitations, we propose a framework to model how different MTD techniques can affect the information an attacker needs to exploit a system's vulnerabilities, so as to introduce uncertainty and reduce the likelihood of successful attacks. We illustrate how this framework can be used to compare two sets of MTDs, and to select an optimal set of MTDs that maximize security within a given budget. Experimental results show that our approach is effective.

Keywords: Moving target defense · MTD quantification · Framework

1 Introduction

Moving target defense offers a great potential in turning the typical asymmetry of the cyber security landscape in favor of the defender [9], and many techniques have been developed since the term first surfaced in the literature. However, each technique only addresses a narrow subset of potential attack vectors and different techniques tend to measure their effectiveness in different and incompatible ways. Additionally, in order to provide a comprehensive security solution, multiple techniques should be used jointly, but this requires the selection of an optimal subset of available techniques. Although several surveys note where certain MTDs might not work well together [12], or give a qualitative estimate of their effectiveness and cost [6], a quantitative framework that can accommodate any existing or future MTDs is still needed for this area of research to progress

This work was partially supported by the Army Research Office under grants W911NF-13-1-0421 and W911NF-13-1-0317, and by the Office of Naval Research under grant N00014-13-1-0703. We acknowledge Dr. George Cybenko and Dr. Sushil Jajodia for the invaluable inputs they provided in the early stages of this work.

S. De Capitani di Vimercati and F. Martinelli (Eds.): SEC 2017, IFIP AICT 502, pp. 124–138, 2017.
DOI: 10.1007/978-3-319-58469-0_9

past specialized, isolated solutions. To address this pressing need, we present a novel framework that captures the relationships between available MTDs and the information such MTDs may affect through probabilistic measures. It also captures the relationships between services, their weaknesses, and the knowledge required to exploit such weaknesses to probabilistically determine the effectiveness of any given technique or set of techniques, regardless of how they operate. Our framework presents the following desirable attributes: (i) **generality** – the relationship between MTDs and the knowledge they protect defines an interface that enables to plug any MTD into the framework; (ii) **extensibility** – the model can be extended to accommodate future MTDs by introducing new elements, such as additional knowledge blocks or classes of weaknesses; (iii) **resilience** – as the framework addresses generic classes of weaknesses rather than specific vulnerabilities, the model can address both known and unknown (zero-day) attacks; (iv) **usability** – the framework is simple and intuitive, can be used to compute utility estimates at different levels of granularity, and can incorporate cost in the estimation of utility.

The remainder of the paper is organized as follows. Section 2 discusses related work, whereas Sect. 3 covers our threat model and underlying assumptions. The framework itself is presented in detail in Sect. 4 with a simple running example, while a more complex case study is discussed in Sect. 5. Then, Sect. 6 shows two applications of the proposed model. Finally, Sect. 7 discusses potential future work and gives some concluding remarks.

2 Related Work

Many different metrics are used in the literature to measure the effectiveness of MTDs, such as attacker's success rate [3], or metrics for deception, deterrence, and detectability [8]. Still others utilize multiple metrics (productivity, success, confidentiality, and integrity) for both the attacker and the defender [16], leading to confusion over the multiple dimensions. However, all these metrics only evaluate a few select MTDs. One expert survey provides a thorough assessment of the effectiveness and cost of many techniques across the spectrum of existing MTDs [6], but the survey is qualitative in nature and potentially subject to reviewer's bias. Our work leverages existing work on attack graphs [10], particularly those approaches that evaluate security by looking at how the probability of a successful attack propagates over an attack graph [15]. The TREsPASS project[1] provides a holistic view of an organization's information security risk. It provides a visualization framework that combines the impact of vulnerability exploitation, physical security breach and social engineering on the target organization. This framework can be used to analyze several properties of multi-step attacks such as the required effort or time, and likelihood of success. However, attack graphs cannot be readily used with every MTD, as they are often tied to specific vulnerabilities. In fact, several MTDs can drastically alter a system's attack surface, requiring to generate an entirely new attack graph every time

[1] http://www.trespass-project.eu.

the MTD changes the system's configuration, which is not feasible in practice. Our work is also inspired by research on autonomous systems, particularly self-protecting systems [1], which autonomously change their settings to adapt to their environment, implicitly creating a *moving target*. In order to do so efficiently, they must quantify the effectiveness and cost of all possible changes.

3 Threat Model and Assumptions

The general nature of our model lets us make very broad, worst-case assumptions about the cyber threats we are trying to protect against. In particular, we assume that *attackers can exploit any possible attack vector*. Most techniques described in the literature only protect against a narrow subset of possible attacks and no single MTD can protect against all possible attack vectors. This is handled by our model by providing the ability to combine multiple MTDs in a defense-in-depth approach. We also make the worst-case assumption that *no static defense can prevent an attack*, as the attacker has virtually unlimited time to plan and execute an attack and zero-day exploits can evade static defenses. Only MTDs are considered to have an effect on the attacker's success rate, and even then, an MTD may not be perfect. We assume that *attackers can be stopped or at least delayed by preventing them from acquiring accurate knowledge* about the target system. Our primary focus here is on the reconnaissance phase, when that knowledge is gathered prior to planning and executing attacks. Our goal can be achieved by either preventing attackers from accessing that knowledge or delaying them until that knowledge is no longer useful.

Finally, we make several additional simplifying assumptions throughout the paper that we summarize here. Future work will allow us to revise many of our assumptions in order to further generalize our approach. We assume that services and weaknesses are time-invariant. We also assume that services and knowledge blocks are independent, but multiple services with dependencies could be modeled. We currently assume that each MTD has a predefined optimal configuration of its parameters, and that, if multiple MTDs affect a knowledge block, they do not interact and only the most effective one is considered.

4 Quantification Framework

In this section, we present the proposed quantification framework, which, as shown for the motivating example of Fig. 1, consists of four layers: (i) a time-invariant service layer representing the set $\mathcal{S}$ of services to be protected; (ii) a weakness layer representing the set $\mathcal{W}$ of general classes of weaknesses that may be exploited; (iii) a knowledge layer representing the set $\mathcal{K}$ of all possible knowledge blocks required to exploit those weaknesses; and (iv) an MTD layer representing the set $\mathcal{M}$ of available MTD techniques.

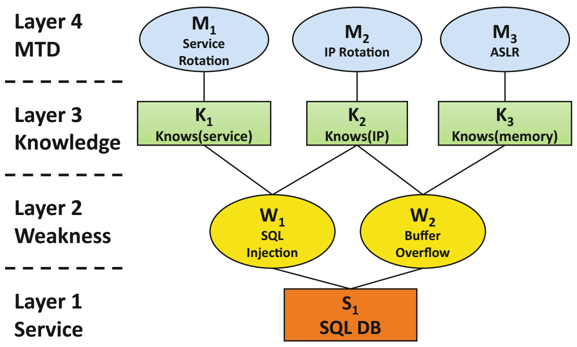

Fig. 1. Quantification framework layers

4.1 Mathematical Model

The proposed MTD quantification framework can be formally defined as a 7-tuple $(\mathcal{S}, \mathcal{R}_{SW}, \mathcal{W}, \mathcal{R}_{WK}, \mathcal{K}, \mathcal{R}_{KM}, \mathcal{M})$, where: (i) $\mathcal{S}$, $\mathcal{W}$, $\mathcal{K}$, $\mathcal{M}$ are the sets of services, weaknesses, knowledge blocks, and MTD techniques, respectively; (ii) $\mathcal{R}_{SW} \subseteq \mathcal{S} \times \mathcal{W}$ represents relationships between services and the common weaknesses they are vulnerable to; (iii) $\mathcal{R}_{WK} \subseteq \mathcal{W} \times \mathcal{K}$ represents relationships between weaknesses and the knowledge blocks required for an attacker to exploit them; and (iv) $\mathcal{R}_{KM} \subseteq \mathcal{K} \times \mathcal{M}$ represents relationships between knowledge blocks and the MTD techniques that affect them. The proposed model induces a k-partite graph (with $k = 4$) $G = (\mathcal{S} \cup \mathcal{W} \cup \mathcal{K} \cup \mathcal{M}, \mathcal{R}_{SW} \cup \mathcal{R}_{WK} \cup \mathcal{R}_{KM})$.

Layer 1: Service Layer. The first layer represents the set $\mathcal{S}$ of services we wish to protect against attacks. We assume that the services are time-invariant, i.e., the functionality of the services does not change over time, and services cannot be taken down to prevent attacks, as this action would result in a denial-of-service. We only consider one service in the case studies presented in this paper, but the model could be extended to consider multiple interdependent services, similarly to how an exploit chain might occur within attack graphs [10,15].

Layer 2: Weakness Layer. The second layer represents the set of weaknesses $\mathcal{W}$ that services are vulnerable to. We choose general classes of weaknesses rather than specific vulnerabilities because there are too many vulnerabilities to enumerate, some vulnerabilities are unknown, and, depending on the MTD used (e.g., OS rotation), specific vulnerabilities may change over time. Using general classes of weaknesses when building the model makes them time-invariant.

The examples used in this paper draw these weaknesses primarily from MITRE's Common Weakness Enumeration (CWE) project [5], particularly from those known as the *"Top 25 Most Dangerous Software Errors."* Although many of the top software errors are primarily the result of bad coding practices and better solved at development time, the top software errors enabling exploits such as *SQL Injection*, *OS Injection*, and *Classic Buffer Overflow* can be addressed at runtime by MTDs (e.g., SQLrand) and make for good general categories of weaknesses. The Microsoft STRIDE Threat Model [7] has also been used as a source of general threats in MTD research [14] and can fill in areas where CWE may be lacking. For example, *Information Disclosure* (eavesdropping) and *Denial of Service* are not specifically addressed by CWE. Our example shows two weaknesses, *SQL Injection* and *Buffer Overflow*. More weaknesses, such as *OS Injection*, might be included in a more complex example, while other weaknesses, such as *Cross-Site Scripting*, would not be applicable to this service.

Layer 3: Knowledge Layer. The third layer represents the knowledge blocks $\mathcal{K}$ required to effectively exploit weaknesses. This knowledge is required to plan an attack even when no MTD is deployed (such as a victim's IP address) or it may be an additional piece of information required due to the use of an MTD. For example, SQLrand [2] adds a keyword to SQL commands, which must be known for a malicious user to perform SQL injection. We assume that knowledge blocks are independent and must be acquired using different methods. For example, IP address and port number should not modeled as separate knowledge blocks because a method to determine one would also reveal the other.

The relationship between the knowledge and weakness layers is many-to-many. A weakness may require several pieces of knowledge to be exploited, and a knowledge block may be key to exploiting several weaknesses. This layer may also be extended as new MTDs – disrupting new and different aspects of an attacker's knowledge – are developed.

In our example, we assume that, in order to execute a SQL Injection attack, the attacker must gather information about the service (e.g., name and version of the specific DBMS) and the network configuration (e.g., IP address). In order to execute a Buffer Overflow attack, an attacker must know the IP address and some information about the vulnerable memory locations. A higher-fidelity version of this model may take a knowledge block and break it down into smaller, more specific items that are specifically targeted by available MTDs.

Layer 4: MTD Layer. The fourth layer of the model represents the set $\mathcal{M}$ of available MTDs. As MTD techniques provide probabilistic security, we model the impact of an MTD M_i on the attacker's effort to acquire knowledge K_j by associating a probability $P_{i,j}$ – representing the attacker's success rate – with the relation (K_j, M_i). As mentioned in the Sect. 3, when only static defenses (i.e., no MTD) are deployed, an attacker will acquire the necessary knowledge without significant effort, which we model by associating a probability of 1.

For example, if technique M_1 in Fig. 1 (*Service Rotation*) reduces an attacker's likelihood of acquiring knowledge block K_1 (i.e., correct version of the service) by 60%, we would label that edge with $P_{1,1} = 0.4$. If an MTD delays an attacker by some factor, we can also express that as a probability that the attacker will not gather the correct information in a timely manner. For example, an MTD that expands addressable memory by a factor of 10 might reduce the attacker's probability of success to 0.1, so $P_{i,j} = 0.1$. The exact methodology for determining the value of $P_{i,j}$ may vary from MTD to MTD, and we are investigating this problem as a separate line of research that goes beyond the scope of this paper. Specifically, we are developing a general approach to model the tradeoff between cost and effectiveness of MTD techniques, as we vary the values of a technique's tunable parameters and other aspects of the attacker/defender interaction. Ultimately, this approach will enable us to identify the optimal configuration for each technique. Therefore, in this paper, we assume that such optimal configuration has already been identified for each available MTD technique, along with the corresponding value of $P_{i,j}$ and the corresponding cost.

Expressing MTD effectiveness in terms of the probability an attacker will succeed in acquiring required knowledge enables us to analyze multiple techniques using a uniform approach that yields values in the [0,1] range, with a theoretically perfect MTD yielding $P_{i,j} = 0$, and a completely ineffective MTD yielding $P_{i,j} = 1$. In our example, we use *service rotation* to disrupt knowledge about the version of the service, and naïvely assume that rotating between 4 services reduces the attacker's probability of gathering the correct information to $P_{1,1} = 0.25$. We apply an *IP address rotation* scheme to mask the victim's IP address. It has been shown that perfect shuffling reduces the attacker's likelihood of guessing the correct IP address by 37% [3]. Using a conservative estimate, we assume $P_{2,2} = 0.75$. Finally, to protect knowledge of the memory layout, we use a dynamic ASLR scheme. Although dynamic ASLR only adds a single bit of entropy compared to typical ASLR [13], this further delays the attacker, resulting in a probability $P_{3,3} = 0.5$ of gathering the correct information.

4.2 Computing MTD Effectiveness

We compute an MTD's effectiveness starting from layer 4 of the model and working our way down to find the overall probability of attacker's success. First, we define $P(K_j)$ as the probability that the attacker has the correct information about knowledge block K_j, and compute $P(K_j)$ for each K_j in layer 3, based on the active MTDs affecting it. If there is no active MTD, we assume that the attacker is guaranteed to obtain that information, i.e., $P(K_j) = 1$.

In our example, each knowledge block is affected by only one MTD. When multiple MTDs affect the same knowledge block, we make the simplifying assumption that the resulting effect is driven by the best-performing MTD. Thus:

$$P(K_j) = \begin{cases} 1, \quad \text{if } \nexists M_i \in \mathcal{M} \text{ s.t. } (K_j, M_i) \in \mathcal{R}_{KM} \wedge active(M_i) \\ \min\limits_{M_i \in \mathcal{M} \text{ s.t. } (K_j, M_i) \in \mathcal{R}_{KM}} P_{i,j} \wedge active(M_i), \quad \text{otherwise} \end{cases} \quad (1)$$

A possible improvement to the model would be to capture the effect of multiple MTDs acting on the same knowledge block by using a function modeling either diminishing returns or some other interaction between multiple MTDs.

Next, we determine the probability $P(W_k)$ that an attacker has gained all the knowledge required to exploit a given weakness W_k. Since each knowledge block is independent, this is simply the product of the probabilities associated with all knowledge blocks leading to it, as shown by Eq. 2.

$$P(W_k) = \prod_{K_j \in \mathcal{K} \text{ s.t. } (W_k, K_j) \in \mathcal{R}_{WK}} P(K_j) \tag{2}$$

In our example, when calculating $P(W_1)$ and $P(W_2)$ for *SQL Injection* and *Buffer Overflow*, respectively, we obtain $P(W_1) = 0.25 \cdot 0.75 = 0.1875$ and $P(W_2) = 0.75 \cdot 0.50 = 0.375$.

Finally, we determine the defender's utility U gained by deploying MTD techniques based on the reduced probability of exploit for each class of weaknesses. In this work, the utility is defined as a function of the probability $P(S_l)$ that an attacker can compromise a service S_l by exploiting any of the weaknesses leading to it. $P(S_l)$ can be computed as the probability of the union of non-mutually exclusive events, using the *Inclusion-Exclusion Principle* [4]. With respect to our running example, $P(S_1)$ can be computed as follows:

$$P(S_1) = P(W_1 \cup W_2) = P(W_1) + P(W_2) - P(W_1 \cap W_2) \tag{3}$$

As W_1 and W_2 are not necessarily independent (as shown in this example), we cannot assume $P(W_1 \cap W_2) = P(W_1) \cdot P(W_2)$. Instead, we must express each $P(W)$ in terms of its corresponding independent knowledge blocks K_j, that is $P(W_1) = P(K_1) \cdot P(K_2)$, $P(W_2) = P(K_2) \cdot P(K_3)$, and $P(W_1 \cap W_2) = P(K_1) \cdot P(K_2) \cdot P(K_3)$, and then express $P(S_1)$ as a function of probabilities $P(K_j)$:

$$P(S_1) = P(K_1) \cdot P(K_2) + P(K_2) \cdot P(K_3) - P(K_1) \cdot P(K_2) \cdot P(K_3)$$

which results in

$$P(S_1) = 0.25 \cdot 0.75 + 0.75 \cdot 0.5 - 0.25 \cdot 0.75 \cdot 0.5 = 0.469$$

For graphs with 3 or more weaknesses $\mathcal{W}^* \subseteq \mathcal{W}$, we can expand Eq. 3 to the generalized form of the *Inclusion-Exclusion Principle* [4]:

$$P\left(\bigcup_{W_k \in \mathcal{W}^*} W_k\right) = \sum_{i=1}^{|\mathcal{W}^*|} \left((-1)^{i-1} \cdot \sum_{\mathcal{W}' \in 2^{\mathcal{W}} \text{ s.t. } |\mathcal{W}'|=i} P\left(\bigcap_{W_j \in \mathcal{W}'} W_j\right)\right)$$

Computing the probability of the union of multiple events is an NP-hard problem that cannot be solved in better than $O(2^n)$ time [4]. However, the general nature of the weaknesses in layer 2 of the model limits their number – as opposed to vulnerabilities which may number in the thousands – keeping the computing time manageable.

After computing $P(S_l)$, we can easily compute the defender's utility as $U = 1 - P(S_l)$. Besides this simple approach, the utility could be a sigmoid function of $P(S_l)$ with an inflection point centered around a desired effectiveness.

Such functions are commonly used in autonomic computing [1]. The complete computation for each of the values in our example is shown in Fig. 2. Note that this choice of utility function relies upon the expectation that at least some measure of protection will be guaranteed for at least one knowledge block for each weakness, otherwise the attacker will be guaranteed to exploit that weakness and reduce the utility to 0. To handle this issue, utility can be defined as a function of the probabilities to exploit each weakness.

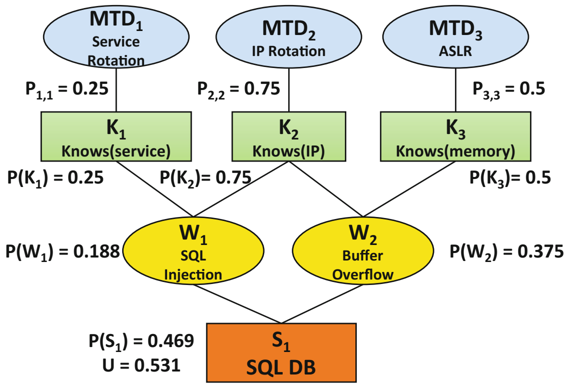

Fig. 2. Computing MTD effectiveness

5 Experimental Evaluation

We now present a more complex example to demonstrate the capabilities of our model. As seen in Fig. 3, we consider the same basic service but protect against two additional classes of weaknesses, OS Injection [5] and Eavesdropping (related to Information Disclosure from the STRIDE model [7]).

In this case study, more fine-grained knowledge blocks have been considered in order to provide more detail or to fit the specific MTDs selected for the case study. For example, knowledge block *Knows(memory)* has been broken down into separate blocks related to system call mapping, memory address, and stack direction. Similarly, SQL Injection now explicitly requires knowledge of keywords appended to SQL commands and some knowledge of the database schema, both of which are disrupted by SQLRand. Most importantly, we can now observe the many-to-many relationships between weaknesses, knowledge blocks, and MTDs, and conclude that finding the optimal solution is no longer trivial. However, using approximate yet reasonable values of $P_{i,j}$ for each MTD and cost constraints, we can determine the final utility as a function of selected MTDs using the steps previously shown and find an optimal solution using a problem solving method, such as stochastic hill climbing or evolutionary methods.

As a proof of concept, we can take the model in Fig. 3 and perform all the necessary computations programatically. As mentioned earlier, we are studying

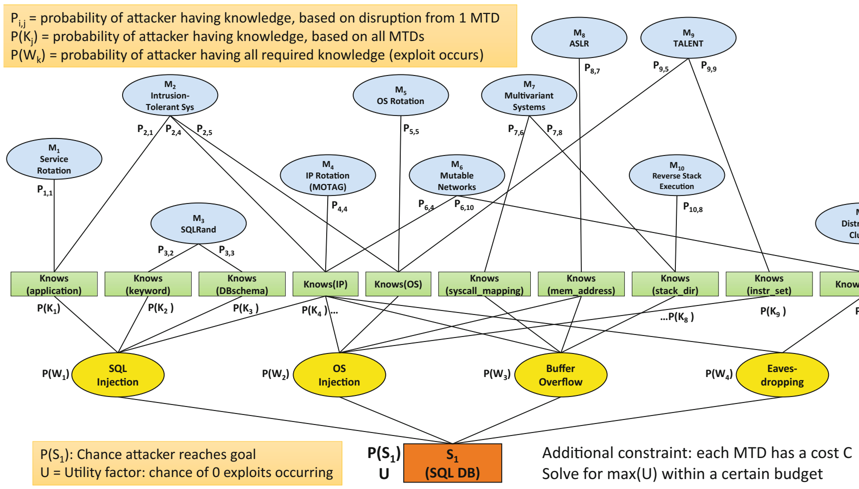

Fig. 3. Case study quantification framework

Table 1. Sample case study evaluation

MTD	$P_{i,j}$		Cost	Active?	$P_{i,j}$ (effective)	Cost (effective)
M_1 (Service Rotation)	$P_{1,1}$	0.500	15	No	1.000	0
M_2 (Intrusion Tolerant Systems)	$P_{2,1}$	0.900	25	No	1.000	0
	$P_{2,4}$	0.900			1.000	
	$P_{2,5}$	0.900			1.000	
M_3 (SQLRand)	$P_{3,2}$	0.300	20	No	1.000	0
	$P_{3,3}$	0.300			1.000	
M_4 (IP Rotation/MOTAG)	$P_{4,4}$	0.900	25	No	1.000	0
M_5 (OS Rotation)	$P_{5,5}$	0.700	15	No	1.000	0
M_6 (Mutable Networks)	$P_{6,4}$	0.500	20	Yes	0.500	20
	$P_{6,10}$	0.500			0.500	
M_7 (Multivariant Systems)	$P_{7,6}$	0.500	20	No	1.000	0
	$P_{7,8}$	0.500			1.000	
M_8 (ASLR)	$P_{8,7}$	0.500	10	Yes	0.500	10
M_9 (TALENT)	$P_{9,5}$	0.500	20	No	1.000	0
	$P_{9,9}$	0.500			1.000	
M_{10} (Reverse Stack Execution)	$P_{10,8}$	0.500	20	No	1.000	0
M_{11} (Distraction Cluster)	$P_{11,10}$	0.500	20	No	1.000	0

Knowledge:	
Knows(application)	1.000
Knows(keyword)	1.000
Knows(DBschema)	1.000
Knows(IP)	0.500
Knows(OS)	1.000
Knows(syscall_mapping)	1.000
Knows(mem_address)	0.500
Knows(stack_dir)	1.000
Knows(instr_set)	1.000
Knows(path)	0.500

Chance of attack success:	
SQL Injection	0.500
OS Injection	0.250
Buffer Overflow	0.250
Easvesdropping	0.250
Chance of attacker success:	0.500
Utility	**0.500**

Total Cost	30
Total Budget	120

Cost:	
High	25
Medium	15
Low	5

Effectiveness:	
High	0.3
Medium	0.5
Low	0.9

the relationship between cost and effectiveness of MTD techniques as part of another line of research. For the purpose of this paper and the evaluation we are presenting, we obtained qualitative values of $P_{i,j}$ and cost from an expert survey [6] which estimates the relative effectiveness and cost of several MTD techniques by grouping them into coarse-grained categories of Low, Medium, or High. Whether or not an MTD is active can be treated as a Boolean variable, with inactive MTDs implying an attacker's probability of success of 1 and a cost of 0. The values from a sample MTD setup are shown in Table 1. The interim calculations for the probabilities of each knowledge block being acquired and each weakness being able to be exploited are also shown.

6 Applications

In these section, we discuss two different applications of our framework.

6.1 Comparing MTDs

Given a set $\mathcal{M}$ of MTD techniques, we want to identify the one that provides the highest overall utility. With respect to the example of Fig. 3, we start from the baseline deployment, shown earlier in Table 1, including M_6 (Mutable Networks) and M_8 (ASLR) to ensure we have a utility value to compare with. We then measure the updated utility value after individually adding each of the other MTDs to our baseline deployment. From the results reported in Table 2, we find that M_3 (SQLRand) offers the greatest increase in utility, with M_1, M_2, and M_3 being the only ones offering any increase at all. To explain these results, we observe that there is a lower bound on $P(S_1)$ that translates into an upper bound on U, defined by $\max(P(W_1), P(W_2), P(W_3), P(W_4))$.

In other words, the overall defense can only be as strong as the protection against exploitation of its most vulnerable weakness, which in turn benefits from the deployment of multiple MTDs. Therefore, given the baseline conditions, only an MTD that affects the most vulnerable weakness will yield any improvement in our utility value. This procedure could be used iteratively in an attempt to find an optimal solution in a greedy manner, but there would have to be some way to handle cases where no MTD adds any utility (such as random selection).

Table 2. Improvement from adding MTDs

MTD	M_1 (Service Rotation)	M_2 (Intrusion Tolerant Systems)	M_3 (SQLRand)	All others
Utility	0.5625	0.513	0.614	0.5
Delta	0.0625	0.013	0.114	0.0

6.2 Selecting Optimal Defenses

Given a set $\mathcal{M}$ of MTDs and a budget, we would like to select the optimal set of MTDs that yield the highest utility with a total cost within the budget. As we now have a tool to evaluate the utility of any MTD deployment, we can also solve for the optimal selection of MTDs, given the constraints that the deployment of each MTD is a Boolean variable (either active or not) and that the sum of the costs of selected MTDs be under our budget. For the purpose of evaluating our framework and making the problem interesting, we selected a value of the budget (120) halfway between 0 and the total cost of deploying all available MTDs (i.e., 210). This choice ensured that a solution with utility greater than 0 would be found and that approximately half the MTDs would be chosen as part of the optimal solution. We solved using the *generalized reduced gradient non-linear algorithm* [11] with random restarts to eliminate finding local maxima. After solving, we obtain an optimal solution with the selected MTD highlighted with a thicker red outline in Fig. 4 and detailed results, including margins of error for our estimates of effectiveness, shown in Table 3.

We can observe that our choice of a utility function forces the selection of a variety of MTDs such that each weakness has at least one MTD affecting one of its knowledge blocks and that protection is evenly distributed over the 4 weaknesses. Visually, we can also observe that an MTD with the ability to affect multiple knowledge blocks is inherently more powerful than one that only affects one. However, if their cost is too high or effectiveness too low, it will still not be chosen as part of an optimal solution. Similarly, an MTD that only affects one knowledge block may be chosen if it is effective, low-cost, or affects a knowledge block that still receives relatively weak protection from other MTDs.

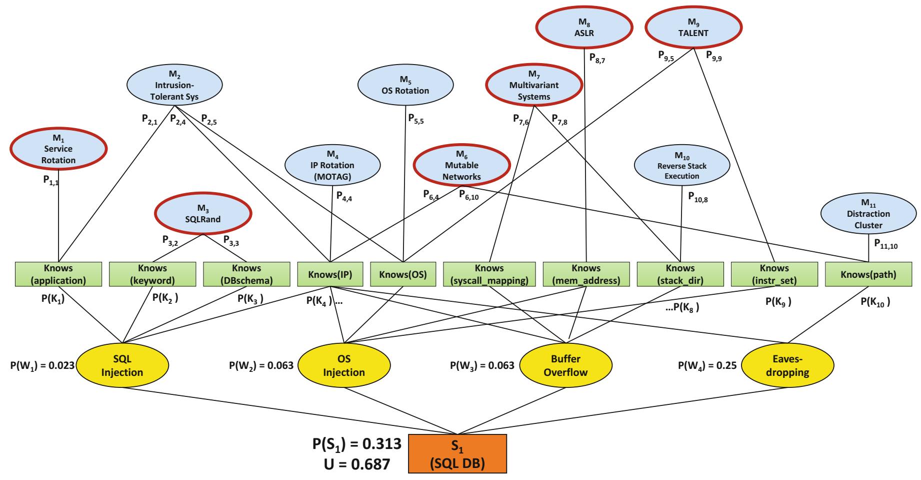

Fig. 4. Case study optimal configuration

Table 3. Case study optimal configuration

MTD	$P_{i,j}$		C	Active?	Pi,j (effective)	C (effective)
M_1 (Service Rotation)	$P_{1,1}$	0.500 ± 0.05	15	Yes	0.500 ± 0.05	15
M_2 (Intrusion Tolerant Systems)	$P_{2,1}$	0.900 ± 0.05	25	No	1.000	0
	$P_{2,4}$	0.900 ± 0.05			1.000	
	$P_{2,5}$	0.900 ± 0.05			1.000	
M_3 (SQLRand)	$P_{3,2}$	0.300 ± 0.05	20	Yes	0.300 ± 0.05	20
	$P_{3,3}$	0.300 ± 0.05			0.300 ± 0.05	
M_4 (IP Rotation/MOTAG)	$P_{4,4}$	0.900 ± 0.05	25	No	1.000	0
M_5 (OS Rotation)	$P_{5,5}$	0.700 ± 0.05	15	No	1.000	0
M_6 (Mutable Networks)	$P_{6,4}$	0.500 ± 0.05	20	Yes	0.500 ± 0.05	20
	$P_{6,10}$	0.500 ± 0.05			0.500 ± 0.05	
M_7 (Multivariant Systems)	$P_{7,6}$	0.500 ± 0.05	20	Yes	0.500 ± 0.05	20
	$P_{7,8}$	0.500 ± 0.05			0.500 ± 0.05	
M_8 (ASLR)	$P_{8,7}$	0.500 ± 0.05	10	Yes	0.500 ± 0.05	10
M_9 (TALENT)	$P_{9,5}$	0.500 ± 0.05	20	Yes	0.500 ± 0.05	20
	$P_{9,9}$	0.500 ± 0.05			0.500 ± 0.05	
M_{10} (Reverse Stack Execution)	$P_{10,8}$	0.500 ± 0.05	20	No	1.000	0
M_{11} (Distraction Cluster)	$P_{11,9}$	0.500 ± 0.05	20	No	1.000	0

Knowledge:	
Knows (1,application)	0.500 ± 0.05
Knows (1,keyword)	0.300 ± 0.05
Knows (1,DBschema)	0.300 ± 0.05
Knows (1,IP)	0.500 ± 0.05
Knows (1,OS)	0.500 ± 0.05
Knows (1, syscall_mapping)	0.500 ± 0.05
Knows (1, Mem_Address)	0.500 ± 0.05
Knows (1,stack_dir)	0.500 ± 0.05
Knows (1,instr_set)	0.500 ± 0.05
Knows (1,path)	0.500 ± 0.05
Chance of attack success:	
SQL Injection	0.023 ± 0.006
OS Injection	0.063 ± 0.013
Buffer Overflow	0.063 ± 0.013
Easvesdropping	0.250 ± 0.035
Chance of attacker success:	0.313 ± 0.043
Utility	**0.687 ± 0.043**

Total Cost	105
Total Budget	120

Cost:	
High	25
Medium	15
Low	5

Effectiveness:	
High	0.3 ± 0.05
Medium	0.5 ± 0.05
Low	0.9 ± 0.05

6.3 Extending the Framework

Our framework can accommodate any existing MTD as long as we can identify the knowledge blocks it affects, the extent to which it disrupts that knowledge, and how it relates to the weaknesses we plan to protect against. Another important feature of our framework is the ability to be extended to accommodate any future MTD that may be developed. A new MTD that affects existing knowledge blocks may be simply added to the MTD layer of the model, while an MTD that works in ways we have not yet considered might also require the addition of new knowledge blocks. Even a new class of weaknesses could be added to the model if the situation warrants it, making our model "future-proof" against new developments in cyber threats.

7 Conclusions and Future Work

In this paper, we have introduced a framework for quantifying moving target defenses. Our approach to quantifying the benefits of MTDs yields a single, probability-based utility measure that can accommodate any existing or future MTD, regardless of their nature. Our multi-layered approach captures the relationship between MTDs and the knowledge blocks they are designed to protect and the relationship between knowledge blocks and generic classes of weaknesses that can be exploited using that knowledge. We have shown through case studies that we can compute the joint effectiveness of multiple MTDs as a function of their individual effectiveness and, by doing so, we can make informed decisions about which MTD or set of MTDs provide better protection based on the security requirements or cost constraints.

Although the work presented in this paper represents a significant step towards effective MTD quantification, several limitations still exist and will be addressed as part of our planned future work. Specifically, limitations exist in the following areas: (i) **probability computation** – our methods for computing the probability $P_{i,j}$ of knowledge disruption provide rough estimates, so a procedure needs to be developed to accurately assess the effectiveness of any MTD; (ii) **cost modeling** – currently, we adopt a very simple notion of cost, and use cost just as an additional constraint, whereas a more sophisticated notion of cost could be introduced and taken into account in the computation of utility values; and (iii) **choice of utility function** – the proposed utility function is based on the assumption that all weaknesses need to be at least partially protected by MTDs to prevent the utility from dropping to 0, therefore, if the risk of leaving a specific weakness unprotected can be accepted, other classes of utility functions could be explored. To address these limitations and further refine our model, we plan to work on several aspects of the framework, as briefly described below.

Implementation and validation. To validate the model, we plan to deploy multiple MTDs on our computing infrastructure and then examine their effectiveness both in isolation – in order to determine the value of $P_{i,j}$ for each MTD – and jointly – in order to accurately study the combined cost and performance.

Application to multiple attack phases. Our model aims at disrupting an attacker's knowledge in the reconnaissance phase of the cyber kill chain. While this may be the most cost-effective way to approach cyber security, no defense is perfect, and we need to ensure multiple layers of defense. Some MTDs can disrupt an attacker's ability to maintain a foothold in the system, so we plan to extend our framework to model this additional class of MTDs.

Application to dependent services. Our framework currently models only independent services. Similar to attack graphs, an attacker may need to execute a sequence of exploits to reach a specific goal. Thus, we plan to extend our framework by introducing a meta-model that captures the relationships between services and the MTDs that can protect them from multi-step attacks.

Heuristics. Because of the $O(2^n)$ runtime to evaluate utility with the current model, it may be necessary to develop heuristics to speed up the evaluation in the case that the number of weaknesses grows to the point where using the model becomes infeasible.

Confidence intervals. Because of the level of uncertainty of our probabilistic values, we may not have a completely accurate utility value. With enough experimental samples, we could introduce confidence intervals into our assertion that a certain MTD or set of MTDs has a higher utility.

References

1. Alomari, F., Menascé, D.A.: An autonomic framework for integrating security and quality of service support in databases. In: Proceedings of the 6th International Conference on Software Security and Reliability (SERE 2012), Gaithersburg, MD, USA, pp. 51–60, June 2012
2. Boyd, S.W., Keromytis, A.D.: SQLrand: preventing SQL injection attacks. In: Jakobsson, M., Yung, M., Zhou, J. (eds.) ACNS 2004. LNCS, vol. 3089, pp. 292–302. Springer, Heidelberg (2004). doi:10.1007/978-3-540-24852-1_21
3. Carroll, T.E., Crouse, M., Fulp, E.W., Berenhaut, K.S.: Analysis of network address shuffling as a moving target defense. In: IEEE International Conference on Communications (ICC 2014), Sydney, Australia, pp. 701–706, June 2014
4. Chen, S.G.: Reduced recursive inclusion-exclusion principle for the probability of union events. In: Proceedings of the IEEE International Conference on Industrial Engineering and Engineering Management (IEEM 2014), Malaysia, pp. 11–13, December 2014
5. Christey, S.: 2011 CWE/SANS top. 25 most dangerous software errors (2011). http://cwe.mitre.org/top.25/
6. Farris, K.A., Cybenko, G.: Quantification of moving target cyber defenses. In: Proceedings of SPIE Defense + Security 2015, Baltimore, MD, USA, April 2015
7. Howard, M., LeBlanc, D.: Writing Secure Code. Microsoft Press, Redmond (2002)
8. Jafarian, J.H., Qi Duan, E.A.S.: Spatio-temporal address mutation for proactive cyber agility against sophisticated attackers. In: Proceedings of the 1st ACM Workshop on Moving Target Defense (MTD 2014), Scottsdale, AZ, USA, pp. 69–78. ACM (2014)
9. Jajodia, S., Ghosh, A.K., Swarup, V., Wang, C., Wang, X.S. (eds.): Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats, Advances in Information Security, 1st edn., vol. 54. Springer, New York (2011)
10. Jajodia, S., Noel, S., O'Berry, B.: Topological analysis of network attack vulnerability. In: Kumar, V., Srivastava, J., Lazarevic, A. (eds.) Managing Cyber Threats: Issues, Approaches, and Challenges, Massive Computing, vol. 5, pp. 247–266. Springer, USA (2005)
11. Lasdon, L.S., Fox, R.L., Ratner, M.W.: Nonlinear optimization using the generalized reduced gradient method. Revue française d'automatique, d'informatique et de recherche opérationnelle. Recherche opérationnelle **8**(3), 73–103 (1974)
12. Okhravi, H., Rabe, M.A., Mayberry, T.J., Leonard, W.G., Hobson, T.R., Bigelow, D., Streilein, W.W.: Survey of cyber moving targets. Technical report 1166, MIT Lincoln Laboratory, Lexington. MA, USA, September 2013
13. Shacham, H., Page, M., Pfaff, B., Go, E.J., Modadugu, N., Boneh, D.: On the effectiveness of address-space randomization. In: Proceedings of the 11th ACM Conference on Computer and Communications Security (CCS 2004), Washington DC, USA, pp. 298–307. ACM, October 2004
14. Soule, N., Simidchieva, B., Yaman, F., Watro, R., Loyall, J., Atighetchi, M., Carvalho, M., Last, D., Myers, D., Flatley, B.: Quantifying minimizing attack surfaces containing moving target defenses. In: Proceedings of the Resilience Week (RWS 2015), August 2015

15. Wang, L., Islam, T., Long, T., Singhal, A., Jajodia, S.: An attack graph-based probabilistic security metric. In: Atluri, V. (ed.) DBSec 2008. LNCS, vol. 5094, pp. 283–296. Springer, Heidelberg (2008). doi:10.1007/978-3-540-70567-3_22
16. Zaffarano, K., Taylor, J., Hamilton, S.: A quantitative framework for moving target defense effectiveness evaluation. In: Proceedings of the 2nd ACM Workshop on Moving Target Defense (MTD 2015), Denver, CO, USA, pp. 3–10. ACM, October 2015