Markov Modeling of Moving Target Defense Games

Hoda Maleki*†§, Mohammad H. Valizadeh*†§, William Koch‡✠, Azer Bestavros ‡✠, and Marten van Dijk †§

§Computer Science and Engineering Dep., University of Connecticut, CT, USA.
✠Computer Science Dep., Boston University, MA, USA.

July 25, 2016

Abstract

We introduce a Markov-model-based framework for Moving Target Defense (MTD) analysis. The framework allows modeling of broad range of MTD strategies, provides general theorems about how the probability of a successful adversary defeating an MTD strategy is related to the amount of time/cost spent by the adversary, and shows how a multi-level composition of MTD strategies can be analyzed by a straightforward combination of the analysis for each one of these strategies. Within the proposed framework we define the concept of security capacity which measures the strength or effectiveness of an MTD strategy: the security capacity depends on MTD specific parameters and more general system parameters. We apply our framework to two concrete MTD strategies.

Keywords— Moving Target Defense, Security Capacity, Markov Models, MTD Games, IP Hopping, Multiple-Target Hiding.

1 Introduction

For a static system in which configuration parameters and system's settings remain relatively unchanged over relatively long periods of time, attackers have enough time in order to infer system's configuration, vulnerabilities, and corresponding attack vectors which may lead to the attacker's success in compromising the system in any number of ways [2]. In order to mitigate such vulnerability, MTD has been introduced as the notion of controlling change across multiple system dimensions to intensify uncertainty and ostensible complexity for attackers, reducing their window of opportunity and increasing the costs of their probes and attack efforts [30].

In spite of the fact that today's computer systems are designed and implemented with security in mind and take advantage of secure services, security primitives and mechanisms, and crypto protocols, the history of computer systems has patently revealed that perfect security is unattainable [1], i.e., all systems have vulnerabilities that can be exploited to launch successful attacks.

*Hoda Maleki and Mohammed H. Valizadeh, in alphabetical order, share lead authorship.

†{hoda.maleki,mohammad.valizadeh,marten.van_dijk}@uconn.edu

‡{wfkoch,best}@bu.edu

Research in MTD focuses on enabling the continued safe operation of a system by dynamically hiding vulnerabilities or attack points (which we call targets in our framework, see section 3). Before a system can be successfully attacked, the adversary first needs to find attack points and this is made complex and costly by MTD.

To date many different MTD schemes have been introduced as proactive defensive solutions by making the system less stationary and less deterministic. In general terms, these techniques can be categorized into five different domains based on their placement within the execution stack [20]: (1) *Dynamic Data* which mostly involves changing data representation or its format [5, 7, 17], (2)*Dynamic Software* which includes changing the application code [24, 25], (3)*Dynamic Runtime Environment* which is about altering the execution environment [14, 22, 28], (4)*Dynamic Platform* which deals with modifying platform properties [8, 11, 26] and finally (5)*Dynamic Network* which is about changing network configurations and properties [4, 13]. A thorough study of most of these methods, including their advantages and weaknesses, can be found in [21]. Most research focuses on introducing a new MTD scheme in one of those five categories and then using a simulation based approach in order to gauge how effective the scheme can be. Even though the use of MTD seems to be a promising defense strategy and many efforts have been made to introduce different practical MTD schemes, there has been very little research that shows the effectiveness of these methods, specially when multiple MTD schemes at different layers of the software stack are combined.

In this paper we introduce a theoretical framework for modeling the effectiveness of MTD schemes:

- In section 3 we define MTD as an interaction between a defender and attacker modeled by probabilistic algorithms and we show that such an interaction corresponds to a defender-attacker game characterized by a Markov model. Our framework is general in that it formally defines (and can be used to analyze) a large class of MTD strategies as games based on Markov models.

- Section 5 proves a first theorem analyzing the number of time steps needed for an adversary to "win" an MTD game. Based on this theorem we introduce *security capacity* as a measure of the effectiveness of an MTD strategy; an MTD strategy with security capacity c has the property that, for all s and t such that $c = s + t$, the probability that the adversary wins the MTD game within 2^t time steps is at most 2^{-s}. We show that time steps can be converted into any cost measure for the adversary.

- Although a single MTD may not be very effective in that it has a small security capacity, it may be possible to obtain a large security capacity if multiple MTDs are deployed together. We call this approach a "Layered MTD" or "MTD^*". In this regard, section 6 is an expansion of our framework by introducing compositions of MTD games for which we extend our theorem.

In sections 4 and 5 we analyze two concrete examples to demonstrate the applicability of our framework beyond a mere theoretical exercise.

2 Related Work

In order to design and maintain secure systems, the system architect or operator must be aware of the systems' attack points and vulnerabilities. This can be achieved only if a higher-level view of the system and these attack points is presented. A Cyber Attack Life Cycle (CALC) also known as "Cyber Kill Chain", provides this higher-level view of an attack in multiple stages where each stage accomplishes an intermediate goal. A more holistic approach for analytics and sensing can be taken into consideration by using the knowledge of the entire CALC [38]. In general terms, CALC stages can be categorized as follows:

- *Reconnaissance*: Collecting necessary information for attack purposes including investigation and identification of the targets.
- *Weaponize*: Developing and preparing a package of attack tools and exploits.
- *Delivery*: Devising and deploying the attack weapon delivery.
- *Persistence*: Maintaining the attack environment to guarantee continued long-term access.

The research summarized in [21] considers a large class of MTD schemes establishes various CALC stages for which each of these schemes can be effective.

On the other hand, rather than using attack life cycles, a more detailed representation of threats is through attack graphs [27], where each node represents a system state with attacker capabilities and a (labeled) directed edge represents an adversarial behavior causing a change in system state. An attack graph teaches how compositions of individual (local) attacks can produce larger attack paths (reaching a global goal).

Since MTD strategies are by definition designed to dynamically change the character of attack points (e.g., by "changing the location of a vulnerability" or by "using a different system implementation thereby exchanging one set of vulnerabilities for another"), the complexity for the attacker reaching intermediate adversarial goals (e.g., as represented by an attack graph) is made more difficult if compositions of MTD strategies are employed.

In MTD research, the central theme is to first present a new feasible technique and then estimate its effectiveness. In this regard, many MTD techniques have been proposed, e.g., Address Space Layout Randomization [14], Software Diversity [19,23], Dynamic Data [3,17], etc.

Existing methods for estimating the effectiveness are mainly divided in two categories: a) simulation based [10,28,29,31,37] and b) based on a mathematical model [9,12,28,35].

Simulation based: Two good examples of simulation-based methods are [31,37]. In [31] the authors introduce a quantitative framework for evaluating the effectiveness of MTD strategies (in the network layer) using simulation experiments. Because of the limited scope of experimental conditions, it is difficult to compare the effectiveness of an MTD strategy for different system settings. Zhuang et al. [37] present a preliminary design schema of MTD in enterprise networks. The authors investigate an adversary's chance of success, by means of conducting a simulation-based study on proactively altering a network's parameter, based on the introduced design schema. One of the problems with simulation based MTD evaluation is that these techniques are case-dependent meaning that they only stand for their specific introduced example.

Based on mathematical models: The work described in [9] considers the effectiveness of "Diversity defenses" (including Address Space, Instruction Set and Data Randomization) as an MTD method. The authors present a model for analyzing the effect of MTD, focusing on scenarios where only the undefined semantics of the underlying programming language is changed. The study is worthwhile as the authors show that the effectiveness of MTD depends on multiple aspects: the type of the execution properties that are changed, address randomization entropy, etc. However, the study does not lead to a generally applicable framework.

Of specific interest to us are game theoretic models that assist in the design and analysis of MTD mechanisms [18]. They can be used to maximize the defender's "security" while minimizing/constraining the "diversity rate", defined as the time interval between each system adaptation (in our terminology the rate λ at which the defender moves in his MTD strategy), to a usable setting (i.e., to a setting giving acceptable performance overhead): Lye and Wing [15] use a two players general-sum stochastic game to model the interaction between an attacker and a defender (administrator) for analyzing the security of computer networks. Manadhata [16] uses a perfect information and complete game for attack surface shifting based on MTD. Zhu et al. [32] propose a game theoretical framework for multi-layer attack surface shifting with an adaptive defender who changes defense strategy based on real-time feedback information and risk analysis. Carter et al. [6] study dynamic platform MTD by using a game theoretical approach modeling an attacker who can monitor defender's moves.

Rather than using game theory, we propose a general framework where the moves of the attacker and defender (system administrator) are represented in a Markov model allowing an immediate and straightforward characterization of the effectiveness of the used MTD strategy by the defender (with respect to the modeled attacker).

Zhuang [33, 36] made a first attempt using Markov models to analyze MTD strategies; each state in the proposed Markov model represents the complete configuration of the whole system which means that the number of states in the Markov model grows exponentially in the number of machines in the system. This attempt being too complex, Zhuang [33,35] considers an attack graph as a Markov model where transitions are labeled with a probability density and where nodes keep track of the machine's configuration together with necessary overall system parameters of which the probability densities are a function. Nodes can represent physical machines, software stacks, or software objects such as VMs. A transition from one node to another node means that the adversary extends its footprint to the other node. MTD can now be modeled as being part of the overall system parameters. From this attack graph perspective precise formulas can be derived with respect to the time needed for an adversary reaching its objective as a function of these probability density functions, which in turn are functions of the MTD and system parameters [33, 34]. Such formulas cannot become concrete as the whole system architecture with MTD and adversarial behavior needs to be modeled – the proposed Markov model requires too much detail. As a result only simple relationships can be proven such as a larger diversity rate λ implies a larger expected time till successful attack, and faster expected transition times imply a smaller expected time till successful attack.

In this paper we realize that the first approach of a Markov model where each state represents a complete system configuration is useful if we compress the Markov model to one where states only represent the projection to necessary system and MTD parameters. This leads to precise formulas and relations which are easy to interpret by plugging in known parameters.

3 MTD Games

In this section we introduce an abstract game between a defender and an attacker which models a large class of MTD strategies. Below we first give an algorithmic definition after which we explain how it naturally translates into an equivalent definition based on a Markov modeling interpretation of MTD games.

Definition 1 *An MTD game is defined by a set of possible defender moves* $\mathcal{D} = \{\epsilon, d_1, d_2, \ldots\}$ *and attacker moves* $\mathcal{A} = \{\epsilon, a_1, a_2, \ldots\}$ *(where* ϵ *indicates "no move") together with a probabilistic algorithm* $\mathcal{M}$ *which outputs a stream of triples* $\{(D_i, A_i, w_i)\}_{i\geq 1}$ *with* $D_i \subseteq \mathcal{D}$*,* $A_i \subseteq \mathcal{A}$*, and* $w_i \in \{0,1\}$ *indicating whether the attacker has beaten the defender's moving target strategy; we say the attacker is in a winning state at time step* i *if* $w_i = 1$*.*

The above definition as an interplay of a defender and adversarial strategy is very general: Algorithm $\mathcal{M}$ is probabilistic and uses coin tosses which, together with the current state of $\mathcal{M}$, decide at each time step i which collection of defender and adversarial moves are being played and in what order. The changes result in a change in the state of algorithm $\mathcal{M}$ from time step to time step. $\mathcal{M}$'s state represents the combination of the adversary's and defender's state together with the state of the system which is attacked/defended. This means that $\mathcal{M}$ also models adaptive strategies.

Algorithm 1 shows how $\mathcal{M}$ can be simulated; *Simstate* keeps track of the system state s, the view of the defender $view^D$, the view of the attacker $view^A$, and whether the attacker is in a winning position; $NextState(.,.,.)$ simulates how the moves by the attacker and defender change the system state; the defender and attacker are represented by algorithms D and A where $D.update(.,.)$ and $A.update(.,.)$ keep track of how their views change because of the new system state; notice that even if only one party moves, both views may be affected.

Algorithm 1 MTD Game Simulator

```
function SIMULATOR($\mathcal{M}$)
    $s$ = initial state system;
    $view^A$ = adversary's initial state of the system used in his adversarial strategy;
    $view^D$ = defender's initial state of the system used in his MTD strategy;
    Initialize $simstate = (s, view^D, view^A, 0)$;
    while true do
        $(moveD, moveA, w) \leftarrow \mathcal{M}(simstate)$;
        $s = NextState(s, moveD, moveA)$;
        $view^D \leftarrow D.update(view^D, s)$;
        $(view^A, w) \leftarrow A.update(view^A, s)$;
        $simstate = (s, view^D, view^A, w)$;
        if $w = 1$ then
            Return "Attacker is in a winning position";
```

In this paper we measure the effectiveness of an MTD strategy by analyzing *when the attacker will be in a winning state for the first time.*[1] For this purpose we do not need to keep track of the exact moves being played by the defender and attacker; we only need to keep track of how the state of $\mathcal{M}$ changes from time step to time step and whether the new state corresponds to a winning

[1]Additional analysis of *how long an attacker will be in a winning state once it arrives at a winning state* and *when the attacker will again be in a winning state once he exits a winning state* is required to understand the effect of an MTD strategy in more detail. Since an adversary reaching a winning state can start launching a successful attack, the authors feel that *when the attacker will be in a winning state for the first time* is the most important question to analyze first.

state for the adversary. This defines a Markov model[2] where state transitions are a function of coin tosses. This leads to the following definition:

Definition 2 *A M-MTD game is defined by a $(k+1)\times(k+1)$ transition matrix M which describes a Markov model of state transitions that reflect both defender and attacker moves. Initially the game starts in the 0 state. At each next time slot the game transitions from its current state i to a new state j with probability $M_{i,j}$. We say the attacker wins the MTD game within T time slots if (the winning) state k has been entered during at least one of the first T time slots of the MTD game.*

Note that state k represents the winning state from the adversary's perspective. In other words, it depicts full control by the adversary; once the adversary reaches state k, it is possible for the attacker to launch an attack that will be successful against the system which the defender attempts to protect using his MTD strategy. From the above discussion we obtain the following lemma.

Lemma 1 *Each MTD game can be described as a M-MTD game for some transition matrix M.*

A subclass of MTD games with a more natural and intuitive correspondence to MTD strategies is given in the next definition:

Definition 3 *A (M^D, λ, M^A, μ)-MTD Game is defined by*

1. *parameters λ and μ satisfying $0 \leq \lambda + \mu \leq 1$ which represent the rate of play of the defender and attacker, respectively.*
2. *$(k+1)\times(k+1)$ transition matrices M^D and M^A; for $i, j \in \{0, \dots, k\}$, $M^D_{i,j}$, respectively $M^A_{i,j}$, represents the probability of transitioning from state i to state j when the defender, respectively attacker, plays a move in state i.*

Initially the joint defender-attacker state is 0. At each next time slot either with probability λ the defender makes a move according to M^D, or with probability μ the attacker makes a move according to M^A, or with probability $1-\lambda-\mu$ neither the defender nor attacker makes a move.

We say the attacker wins the MTD game within T time slots if he has entered state k during one of his moves during the first T time slots of the MTD game. In the above definition we exclude the possibility of both the defender and attacker playing at the same time: A (three valued) random coin decides with probability λ that the defender can move, with probability μ that the attacker can move, and with probability $1-\lambda-\mu$ that no one moves.

Notice that a (M^D, λ, M^A, μ)-MTD game is fully described by the transition matrix

$$M = \lambda M^D + \mu M^A + (1-\lambda-\mu)I_{k+1}, \tag{1}$$

where I_{k+1} is the $(k+1)\times(k+1)$ identity matrix. At each time slot the joint defender-attacker state transitions to a new state according to M; M represents the MTD game as a Markov model as in Definition 2.

A couple questions concerning the limitations of this abstract definitional framework immediately arise: The above definitions relate to a logical discrete time axis where at each fixed time step or " logical clock cycle" the system state changes according to transition matrix M. How can this

[2]We avoid unnecessary state explosion by representing states by system and MTD parameters needed for describing the MTD game between adversary and defender with sufficient precision.

logical time be translated to real time (which is what we are interested in) or cost of playing (in some cases we may be able to assign a cost to adversarial moves)? The above definitions are very general and may lead to very complex huge Markov models in order to model real systems. How much detail is needed in order to model a complex MTD game, is it possible to decompose games in multiple layers simplifying analysis? Finally, how can we measure the security offered by an MTD strategy? The remainder of this paper answers these questions demonstrating the richness of the language provided by our definitional framework.

4 MTD games Examples

In the following subsections we consider a system taking advantage of IP Hopping as an MTD scheme. Two different scenarios are considered in which the adversary's goal is to locate and control one, respectively k, vulnerable hosts in the network. This is referred to as single-target IP hopping and multiple-target IP hopping. Our analysis of single-target IP hopping assumes an optimal adaptive adversary while our analysis of multiple-target IP hopping assumes an almost optimal adaptive adversary (being almost optimal is shown in section 5).

4.1 Single-Target IP Hopping

IP Hopping (also known as IP address randomization) as an MTD technique makes reconnaissance difficult for an adversary when attempting to locate a host by an IP address. By frequently changing the IP addresses of the hosts on a network, the adversary is forced to continually rescan the network to locate the host, while also limiting the time to execute their attack. Implementations of this technique have been proposed for OpenFlow [13] and DHCP [4].

The adversary is forced to perform a spatial search for the target host in a pool of N IP addresses, which is performed by sending network probes to specific IP addresses. The probe may be a simple SYN packet to locate a host with a particular open port or an operating system/software fingerprint to locate a host running a particular system. If the adversary receives an acknowledgment the probe found a target, then the adversary moved successfully and is now in control of the target. The adversary continues his search until the target (or targets) are found. The stronger adversary has the ability to uniquely identify the target once found, e.g., the adversary can perform a reverse DNS lookup on the IP address to obtain a unique host name.

In general, in IP hopping techniques new IP addresses from a pool of unused addresses in the network subnet will be selected and assigned to the host at some (average) rate λ. E.g., the defender plays a memoryless MTD strategy such that every time step he randomizes the IP address with probability λ (and he will do nothing with probability $1-\lambda$). Meanwhile, the adversary sends a probe at rate μ in an attempt to identify a target. He may play an adaptive strategy: If the attacker has tried a set of say q IP addresses after the defender's last IP address randomization, then his $(q+1)$-st trial should not be one of the q IP addresses he already tried, instead he should guess a new one. Even though the attacker does not learn when the defender randomizes the IP address, if he tries each IP address in sequential round robin order, then he knows his next guess will never be of an IP address he already tried since the defender's last IP address randomization. As the defender plays memoryless, it does not matter whether the attacker tries each IP address in fixed known sequential round robin order. This optimal adaptive adversarial strategy is described by the Markov model in Figure 1.

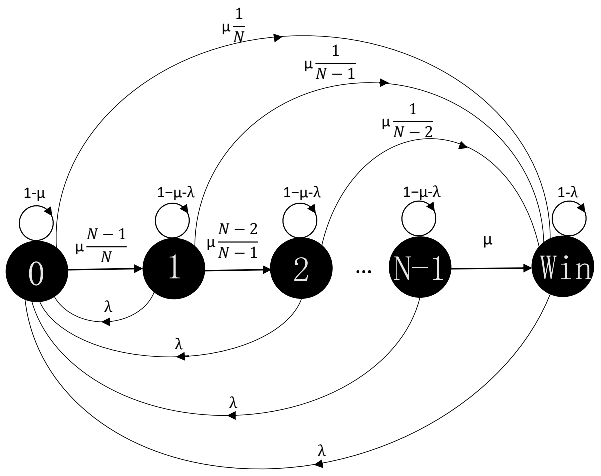

Figure 1: Single-Target IP Hopping game Markov model

The Markov model has $N + 1$ states where N equals the total number of possible IP addresses. Let state N (in the figure denoted by "Win") represent the winning state for the adversary and let states $0, 1, \ldots, N - 1$ correspond to when the defender randomized the IP address for the last time, i.e., state q means that exactly q time steps ago the IP address was randomized (by the defender). For $0 \leq q \leq N - 1$,

- $M_{q,0} = \lambda$; with probability λ the defender randomizes the IP address, as a result the Markov model transitions back to the 0 state,
- $M_{q,q+1} = \mu(N - q - 1)/(N - q)$; with probability μ the attacker tries the next IP address which with probability $1 - 1/(N - q)$ is not the correct one,
- $M_{q,N} = \mu/(N - q)$; with probability μ the attacker tries the next IP address which is with probability $1/(N-q)$ correct meaning that the Markov model transitions to the winning state for the adversary,
- $M_{q,q} = 1-\lambda-\mu$; with probability $1-\lambda-\mu$ both the defender and attacker do not do anything,
- $M_{N,0} = \lambda$ and $M_{N,N} = 1 - \lambda$; the adversary does not need to try IP addresses, he will only be kicked out of the winning state if the defender randomizes the IP address.

The IP hopping strategy above deals with the scenario where the adversary only needs to find one single target. Multiple-Target IP Hopping can be seen as an example of the more abstract MTD game described next.

4.2 Multiple-Target Hiding

Another possible scenario one can think of is the case in which the adversary needs to find/penetrate more than one host/target in a network taking advantage of IP-hopping. Let N be the total number of "locations" where K targets (i.e., attack points or vulnerabilities) may hide; only if the combination of a smaller subset of k locations of these K target attack points are known by the attacker, the attacker is capable of launching a successful attack by exploiting the combination of the k associated vulnerabilities, i.e., the attacker wins the MTD game. The defender moves by re-allocating a target causing the attacker to renew his search for its location. The attacker moves by selecting one of the N possible locations and testing whether it corresponds to one of the targets. E.g., one can think of hit list based worms targeting specific servers on the network or servers part of a redundant system an adversary is attempting to disrupt. The multiple-target hiding defense limits the success of an adversary attempting to compromise and persist on a set of servers on a network.

Both the defender and attacker interface with the defender "system", the system implements the defender's MTD strategy and can also control the rate at which each party may access the system. Each time step allows three possibilities for defender-attacker moves (giving an example of the more abstract (M^D, λ, M^A, μ)-MTD game):

[**Defender moves**] *If the defender issues a move, it will take priority over any other player's request.* We assume a system which controls its own MTD: if the system/defender thinks it is time to move (i.e., re-allocate a target), then it will stall any other requests (in particular, those made by an adversary who wishes to figure out where targets are hiding by getting in touch with locations and testing them).

In our analysis we only consider a defender who plays a memoryless strategy: at every time step, the defender plays (issues a move) with probability λ regardless any learned information about the attacker's strategy. Also, when the defender plays, he will choose one of the K targets at random and re-allocates the selected target to a random location (among one of the other $N - K$ possible locations).

[**Attacker moves**] The defender has not issued a move, but the attacker has: We consider *adaptive adversarial strategies*, i.e., the adversary can be modeled by a probabilistic algorithm $\mathcal{A}$ which uses previously learned information in order to select a new future time step at which the adversary will issue his next move.

Previously learned information are, e.g., the times at which the attacker moves and his discovery about when he gains or loses the location of a target. In our analysis we give the adversary more detailed information (for free): we assume that as soon as the defender re-allocates a target which location before re-allocation was known to the attacker, the attacker is told that this location does not any more correspond to one of the K targets.

[**No moves**] No moves are issued by either the defender or adversary. We consider an adaptive adversarial strategy which, given some number of time steps T, maximizes the probability of winning the game (by finding a subset of k locations) within T time steps:

Suppose the attacker moves and decides to wait by not issuing a move the very next time step. Then the attacker runs the extra risk to lose one of his locations during the next time step as the defender plays a memoryless strategy and will play/move with probability λ. This extra risk translates in a higher probability of not being able to learn a sufficient number of k locations within T time steps. To reduce this unnecessary probability of losing a location while waiting, the attacker should not wait in the first place.

This means an optimal adaptive adversarial strategy is to issue moves at every time step as much as the system allows access. We assume a system which limits the rate r at which a party is allowed to access the system (in more detail: whenever a party requests to access the system, the request is put in a FIFO queue and the system will serve requests according to a memoryless

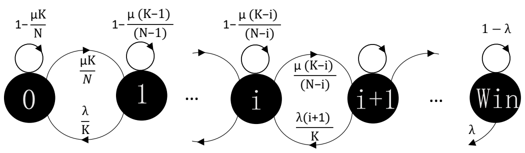

Figure 2: Introduced Multiple-Target Hiding game Markov model

Poisson process corresponding to r). Since defender moves take priority, this implies that the attacker only plays when the defender does not play. Therefore, if the defender plays memoryless with probability λ, then either the defender moves with probability λ, or the attacker moves with probability $\mu = \min\{r, 1-\lambda\}$, or with probability $1-\lambda-\mu$ no one moves.

Figure 2 depicts the Markov model which describes a close approximation of the above MTD game (the winning state "k" is denoted by state "Win"); as in the single-target IP hopping game an optimal adversary should select new locations according to a round robin strategy – we analyze a suboptimal adversary who selects random new locations: If the game is in state i and if the attacker plays, which happens with probability μ, then the attacker will hit one of the $K-i$ unknown locations with probability $(K-i)/(N-i)$ (since the defender uniformly chooses random new locations): this explains the transition probability $M_{i,i+1}$ from state i to state $i+1$. If the game is in state $i+1$ and if the defender plays, which happens with probability λ, then the defender will randomize one of the $i+1$ locations known to the adversary with probability $(i+1)/K$: this explains the transition probability $M_{i+1,i}$ from state $i+1$ to state i. Hence, transition matrix M is given by

$$\begin{aligned} M_{i,i+1} &= \mu(K-i)/(N-i) \quad \text{for } 0 \le i \le k-1, \\ M_{i+1,i} &= \lambda(i+1)/K \quad \text{for } 0 \le i \le k-1, \end{aligned} \tag{2}$$

$M_{i,i} = 1 - M_{i,i+1} - M_{i,i-1}$ for $1 \le i \le k-1$, $M_{0,0} = 1 - M_{0,1}$, and $M_{k,k} = 1 - M_{k,k-1}$.

5 Security Analysis

We prove the following theorem which provides an upper bound on the probability of an attacker to win a M-MTD game within T time steps and which interpretation leads to the definition of "security capacity" of MTD games:

Theorem 1 *For an M-MTD game with equilibrium vector π (i.e., $\pi M = \pi$) and the initial 0 state representing the worst state to start the game in from the adversary's perspective,*[3] *the probability that the attacker wins in the first T time steps is $\le T \cdot \pi(k)$, where k is the winning state.*

In order to prove this theorem, we give the adversary the advantage to initially start in a state drawn from the equilibrium vector π rather than starting in the 0 state which represents the worst starting state for the adversary. For π we prove the following lemma:

[3]This excludes a scenario where initially the adversary has a leg up – e.g., because the initial state of the system is a "default" state where the adversary can assume specific "default" settings before the defender had the opportunity to move "targets" around.

Lemma 2 *If the MTD game starts in a state drawn from the equilibrium vector π, then E, the expected number of times the attacker enters state k within the first T time steps, is equal to $E = T \cdot \pi(k)$.*

Proof. Let Z be the random variable representing the sequence of T states entered in the first T time steps, i.e., if $Z = (z_1, \ldots, z_T)$ then in the i-th time slot the game entered state z_i. Let X_i be the indicator random variable with $X_i = 1$ if $z_i = k$ and $X_i = 0$ if $z_i \neq k$. By the definition of expectation

$$\begin{aligned} E &= \sum_{(z_1,\ldots z_T)} Prob[Z = (z_1, \ldots z_T)] \cdot |\{i : z_i = k\}| \\ &= \sum_{(z_1,\ldots z_T)} Prob[Z = (z_1, \ldots z_T)] \cdot \sum_i X_i \\ &= \sum_i \sum_{(z_1,\ldots z_T)} Prob[Z = (z_1, \ldots z_T)] \cdot X_i \\ &= \sum_i Exp[X_i]. \end{aligned}$$

Notice that $Exp[X_i]$ is equal to the probability that the game enters state k during the i-th time slot. Since π is the equilibrium vector and the game is assumed to start from a state drawn from π, this probability is equal to the k-th entry of $\pi M^i = \pi$, i.e., $\pi(k)$. This proves $E = \sum_i Exp[X_i] = T \cdot \pi(k)$.

Lemma 3 *Let E be the expected number of times the attacker enters state k within the first T time steps. Then, the probability that the attacker wins in the first T time steps is $\leq E$.*

Proof. Let $p(j)$ be the probability the attacker enters state k exactly j times in the first t time steps. By the definition of expectation, $E = \sum_j p(j) \cdot j$. Now notice that the probability that the attacker wins in the first T time steps is equal to $\sum_{j=1}^{T} p(j) \leq \sum_{j=0}^{T} p(j) \cdot j = E$.

The two lemmas together prove[4] Theorem 1 which we can interpret as follows: Suppose $\pi(k) = 2^{-c}$. Then, the probability that the attacker wins in the first $T = 2^t$ time steps is $\leq 2^{-(c-t)}$. We may interpret $s = c - t$ as a security parameter and interpret c as the "security capacity" of the MTD game. The capacity $c = s + t$ is split towards "a probability 2^{-s} of successful attack" and "a considered time frame of 2^t logical time steps".

Definition 4 *An M-MTD game has at least security capacity c if the probability that the attacker wins in the first $T = 2^t$ time steps is $\leq 2^{-s}$ for all $t + s = c$. Notice that an M-MTD game with with equilibrium vector π satisfying the conditions of Theorem 1 has at least a security capacity[5] $c = -\log \pi(k)$.*

If we know the cost of the real resources (i.e. money, time, etc.) an adversary needs to spend in order to move, then the next theorem shows how to compute the adversarial cost needed for a successful attack.

[4]We notice that the upper bound is close to the actual probability for $T \cdot \pi(k) \ll 1$ since (1) the assumed equilibrium vector has its probability mass at and around the initial 0 state for well designed MTD strategies and (2) $\sum_{j=1}^{T} p(j) \approx \sum_{j=0}^{T} p(j) \cdot j$ as $p(j) \ll p(1)$ for $j > 1$.

[5]See the previous footnote, the security capacity will not be much larger than $-\log \pi(k)$.

Theorem 2 *For an M-MTD game with expected adversarial cost per logical time slot equal to α, equilibrium vector π, and the initial* 0 *state representing the worst state to start the game in from the adversary's perspective, the probability that the attacker wins the M-MTD game by paying at most C is $\leq C \cdot \pi(k)/\alpha$, where k is the winning state.*

Notice that C/α estimates the number of logical time slots T within which the adversary wins the M-MTD game. Applying Theorem 1 with $T = C/\alpha$ immediately gives the above result. A precise proof turns out to be more complex and is given in the appendix.

5.1 Single-Target IP Hopping

We compute the equilibrium distribution for IP address randomization as follows: Equation $\pi M = \pi$ gives for $1 \leq q \leq N-1$

$$\pi(0) = (1-\mu)\pi(0) + \lambda\pi(N) + \lambda \sum_{q=1}^{N-1} \pi(q), \tag{3}$$

$$\pi(q) = (1-\lambda-\mu)\pi(q) + \mu \frac{(N-q)\pi(q-1)}{(N-q+1)}, \tag{4}$$

$$\pi(N) = (1-\lambda)\pi(N) + \sum_{q=0}^{N-1} \mu\pi(q)/(N-q). \tag{5}$$

Equation (4) yields the recurrence relation

$$\pi(q) = \frac{\mu}{\mu+\lambda} \frac{N-q}{N-q+1} \pi(q-1)$$

which is solved by

$$\pi(q) = \left(\frac{\mu}{\mu+\lambda}\right)^q \frac{N-q}{N} \pi(0).$$

Equation (3) combined with $\sum_{q=1}^{N} \pi(q) = 1 - \pi(0)$ is solved by

$$\pi(0) = \frac{\lambda}{\mu+\lambda}.$$

Now we are able to derive

$$\sum_{q=0}^{N-1} \pi(q)/(N-q) = \left\{1 - \left(\frac{\mu}{\lambda+\mu}\right)^N\right\}/N.$$

Substituting this in (5) gives the equilibrium probability of the winning state for the adversary:

$$\pi(N) = \left\{1 - \left(\frac{\mu}{\lambda+\mu}\right)^N\right\} \cdot \frac{\mu}{\lambda N}. \tag{6}$$

In practice, the security capacity $-\log \pi(N) \approx \log(\lambda N/\mu)$ is rather small as N is restricted to the number of unused addresses in the network subnet.

5.2 Multiple-Target Hiding

In order to find the equilibrium distribution π of M in (2), we observe that $\sum_{j=0}^{i} \pi(j)$ and $\sum_{j=i+1}^{k} \pi(j)$ do not change after multiplying with M. This means that the probability which leaves state $i+1$ to i must equal the probability which leaves state i to $i+1$ as a result of one multiplication by M: For $0 \leq i \leq k-1$,

$$\pi(i) \cdot M_{i,i+1} = \pi(i+1) \cdot M_{i+1,i}.$$

Rewriting the recurrence relation yields

$$\begin{aligned}
\pi(k) &= \pi(0) \prod_{i=0}^{k-1} \frac{\pi(i+1)}{\pi(i)} = \pi(0) \prod_{i=0}^{k-1} \frac{M_{i,i+1}}{M_{i+1,i}} \\
&= \pi(0) \prod_{i=0}^{k-1} \frac{\mu(K-i)/(N-i)}{\lambda(i+1)/K} \\
&= \pi(0) \left(\frac{\mu K}{\lambda}\right)^{k} \prod_{i=0}^{k-1} \frac{(K-k+i+1)}{(i+1)(N-i)} \\
&\leq \left(\frac{\mu K(K-k+1)}{\lambda(N-k+1)}\right)^{k}.
\end{aligned}$$

Notice that $K = k = 1$, which describes the single-target IP hopping game, has the upper bound $\mu/(\lambda N)$ which is close to (6) implying (by extrapolation) that the suboptimal adversary considered in the multiple-target hiding game is close to optimal.

The security capacity $-\log \pi(k) \geq k(\log(\lambda(N-k+1)) - \log(\mu K(K-k+1)))$ scales linearly with k showing that multiple-target hiding is a good MTD design principle.

6 Composition of MTD games

Composition seems to be a natural defense mechanism meaning that although a single MTD may not have sufficient security capacity, a composition of multiple MTDs deployed together may have sufficient security capacity. In this section we analyze the composition of several levels/layers of MTD where the attacker can only play at a higher level if it is in the winning state of the previous level. Equivalently, the game at the higher level does not allow the attacker to move if it is not in the winning state of the previous level. The main idea is that the attacker must slowly penetrate the defense mechanism as in an Advanced Persistent Threat (APT). We call this approach a Layered MTD or MTD^*:

Definition 5 *A Layered MTD game (or MTD^*) is a composition of a sequence of MTD games M_j, $0 \leq j \leq f$, defined by the following rules: The defender plays each game each timeslot and the adversary plays MTD game M_0 each timeslot. However, for $0 \leq j \leq f-1$, the adversary only plays MTD game M_{j+1} in a timeslot if the adversary is in the winning position of MTD game M_j at the beginning of that timeslot.*[6] *Game MTD^* is won by the adversary if the winning state in M_f has been reached.*

[6]If the adversary is not in the winning position of MTD game M_j, then these rules effectively change M_j temporarily into an MTD game M'_j where only the defender plays.

The above definition assumes that each timeslot transitions each Markov model M_j exactly once, i.e., for all M_j transitions occur at the same frequency (we analyse different frequencies in section 6.1). The next theorem analyses the security capacity of a layered MTD game:

Theorem 3 *Let M_j with winning state k_j, equilibrium vector π_j and at least a security capacity $c_j = -\log \pi_j(k_j)$, $0 \leq j \leq f$, represent a sequence of MTD games. Assume that for each M_j-MTD game the initial 0 state represents the worst state to start the game in from the adversary's perspective. Furthermore, assume that in each M_j-MTD game the defender implements moves which can transition from state k_j (the losing state from the defender's perspective) to a state $\neq k_j$, hence, $\lambda_j = 1 - (M_j)_{k_j,k_j} > 0$.*

Then, the probability that the attacker wins the layered MTD game based on $M_0, \ldots, M_f$ in the first T time steps is

$$\leq T \cdot \pi_0(k_0) \prod_{j=1}^{f} \pi_j(k_j)/\lambda_j.$$

I.e., the layered MTD game has at least a security capacity

$$c = \sum_{j=0}^{f} c_j + \log \prod_{j=1}^{f} \lambda_j. \tag{7}$$

Proof. To analyze this composition game, we slightly modify the composition game by giving the adversary two extra advantages. First, the attacker is also able to play in M_j if it is currently already in a winning state in M_j (as we will explain this is not really an advantage that helps the adversary). Second, the defender can only play game M_j if the attacker is also allowed to play. This restriction of the defender implies that in the modified composition game only if the attacker is in a winning state in M_j or in M_{j+1}, then the defender and attacker are able to play in M_{j+1}; if the attacker is not in a winning state in M_j or M_{j+1}, then M_{j+1} is put "on hold" as both the defender and attacker do not issue moves in M_{j+1}. The modified composition game is formally defined as follows:

Definition 6 *Let M_j with winning (for the adversary) states k_j, $0 \leq j \leq f$, represent a sequence of MTD games. We define the $(M_0, \ldots, M_f)$-MTD game as the following composition of the individual M_j-MTD games:*

1. *The M_0-MTD game is played in every time slot.*
2. *For $0 \leq j < f$, the M_{j+1}-MTD game is only played in a time slot if*
   - *the M_j-MTD game is in its winning state k_j at the start of the time slot or*
   - *the M_{j+1}-MTD game is in its winning state k_{j+1} at the start of the time slot.*
3. *The composed game is won by the adversary as soon as he enters the winning state k_f of the M_f-MTD game.*

We say that the M_j-MTD game is played at level j in the composition of the M_j-MTD games.

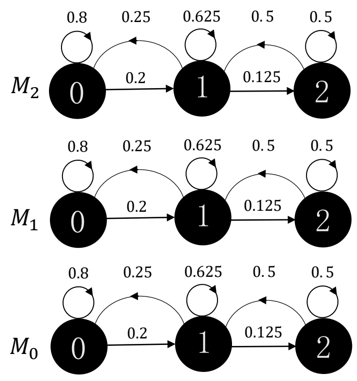

Figure 3: An example of a composed game made of 3 similar levels

In order to get a feeling for how a $(M_0,\ldots,M_f)$-MTD game behaves we consider the toy example of Figure 3 which depicts the Markov models of a sequence of (in our example equivalent) games M_0, M_1, and M_2 (with similar time scales) at *levels* 0, 1, and 2. A sample run of a simulation of the (M_0,M_1,M_2)-MTD game is illustrated in Figure 4 in which after 439 time steps the attacker wins. Notice that the defender and attacker can play moves in games at different levels at the same time. E.g., if at the start of a time slot the attacker is not in the winning position in the game at level 0 but is in the winning position in the game at level 1, then during the time slot he can play moves in both M_0 and M_2, but not in M_1.

From the attacker's point of view (in Definition 6), if it is in the winning state of the game at level $j+1$, then playing the game at level $j+1$ means not doing anything at all as it does not want to lose its winning position. So, equivalently, the adversary only plays the game at level $j+1$ if it is in the winning state of the game at level j.

From the defender's point of view, the definition unnecessarily restricts it: The defender stops playing the game at level $j+1$ if the game at level j is not in state k_j and at the same time the game at level $j+1$ is not in state k_{j+1}. This is an advantage for the adversary which does not happen in practice. In practice the defender does not know the states of the various games and it will continue playing its MTD strategy as always.

The $(M_0,\ldots,M_f)$-MTD game is only introduced for proving Theorem 3: We now show that the theorem holds for $(M_0,\ldots,M_f)$-MTD games which implies the theorem must hold for layered MTD games as these represent adversaries without additional advantages.

Let E_j be the expected number of times the attacker enters state k_j in the game at level j within the first T time steps. Let $p_j(t)$ be the probability that the attacker enters state k_j exactly t times in the game at level j in the first T time steps. Let $q_j(w)$ be the probability that the game at level j is played during exactly w time slots in the first T time steps. We define $p_j(t|w)$ as the probability that the attacker enters state k_j exactly t times in the game at level j in the first T time steps conditioned on the assumption that the game at level j is played during exactly w time slots in the first T time steps. Notice that $p_j(t)=\sum_{w=t}^{T} q_j(w)p_j(t|w)$.

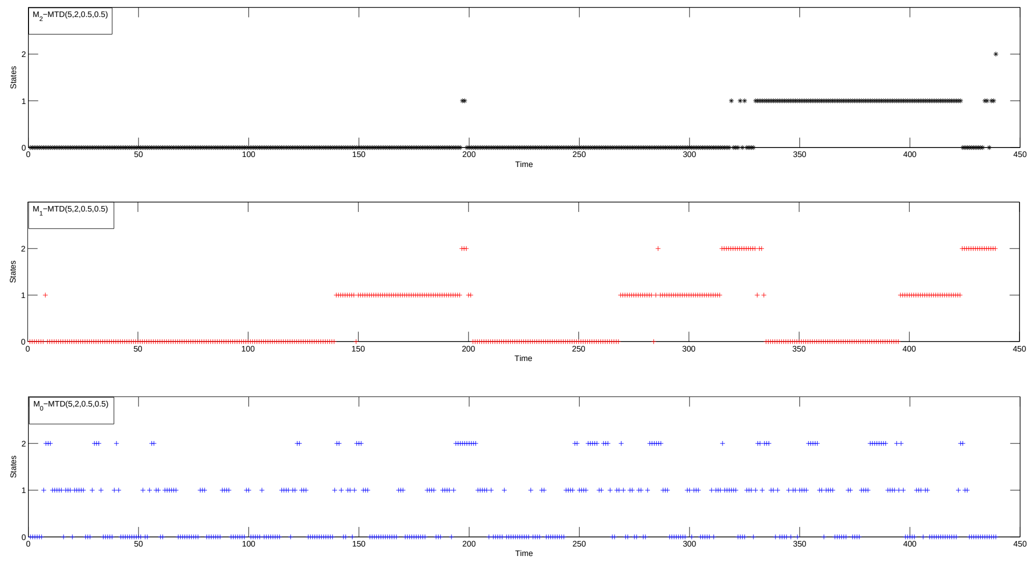

Figure 4: An illustration of a sample run of the modified composed game example Simulation in which after 439 time steps the attacker wins the modified composed game.

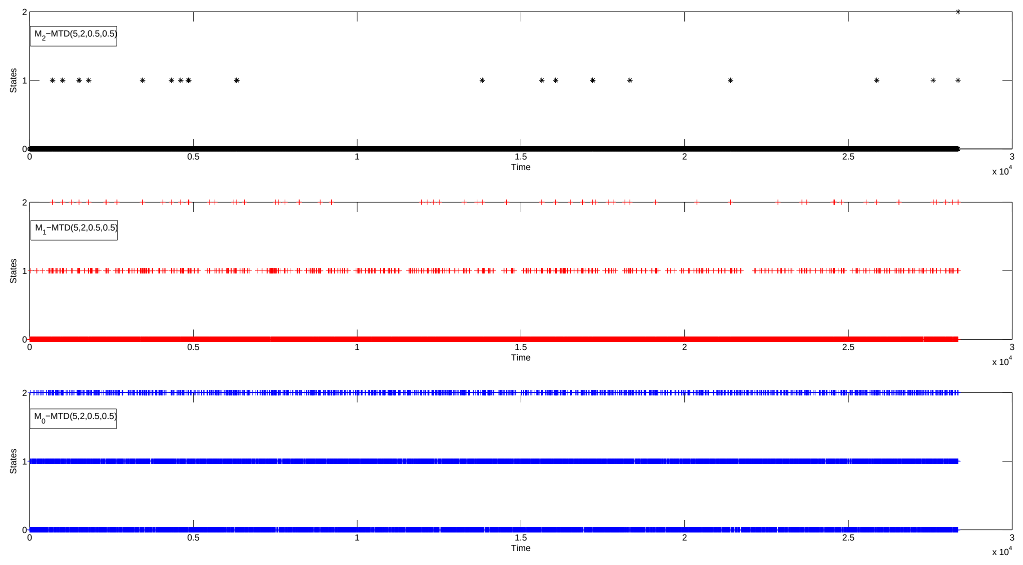

Figure 5: An illustration of a sample run of the Layered MTD game example Simulation, in which the adversary wins the game after 28349 time steps

We derive

$$\begin{aligned} E_{j+1} &= \sum_{t=0}^{T} p_{j+1}(t) \cdot t \\ &= \sum_{w=0}^{T} q_{j+1}(w) \sum_{t=0}^{w} p_{j+1}(t|w) \cdot t. \end{aligned}$$

Notice that Lemma 2 applies directly to the expectation $\sum_{t=0}^{w} p_{j+1}(t|w) \cdot t$: By Lemma 2,

$$\sum_{t=0}^{w} p_{j+1}(t|w) \cdot t = w \cdot \pi_{j+1}(k_{j+1}).$$

Hence,

$$E_{j+1} = \sum_{w=0}^{T} q_{j+1}(w) \cdot w \cdot \pi_{j+1}(k_{j+1}).$$

By the definition of composition, the game at level $j+1$ is only played at time slot t when the game at level j is in k_j at the start of time slot t or when the game at level $j+1$ is in k_{j+1} at the start of time slot t. Let t_0 be a time slot during which the game at level $j+1$ transitions from k_{j+1} to a state $\neq k_{j+1}$. Let t_1 be the most recent time slot before t_0 during which the game at level $j+1$ transitioned from a state $\neq k_{j+1}$ to state k_{j+1}. This means that (a) the game at level $j+1$ enters (or remains in) state k_{j+1} at time slots $t_1, t_1+1, \ldots, t_0-1$, and (b) the game at level $j+1$ was played in time slot t_1 when at the start of time slot t_1 the game at level $j+1$ was not in k_{j+1}.

By the definition of composition and choice of t_0, (a) implies that the game at level $j+1$ is played during time slots $t_1+1, \ldots, t_0$; at the start of each of these time slots the game at level $j+1$ is in k_{j+1}; during time slot t_0 the game at level $j+1$ exits state k_{j+1}. This process is described by a Poisson process with parameter λ_{j+1}, i.e., the probability that the game at level $j+1$ behaves this way is equal to $(1-\lambda_{j+1})^{t_0-1-t_1}\lambda_{j+1}$. Therefore the expectation of $t_0 - t_1$ is equal to $1/\lambda_{j+1}$.

By the definition of composition, (b) implies that the game at level j must have been in k_j at the start of time slot t_1. It may happen that during the Poisson process as described above the game at level j enters state k_j once more in which case the current Poisson process proceeds and a new Poisson process is not spun off. In the worst case analysis for the defender, however, each time the game at level j is in k_j at the start of a time slot a Poisson process (as described above) is spun off (which adds more time slots during which the game at level $j+1$ is being played). We conclude that the expected number of times the game at level $j+1$ is played is at most the expected number of times the game at level j is played times $1/\lambda_{j+1}$. That is,

$$\sum_{w=0}^{T} q_{j+1}(w) \cdot w \leq E_j/\lambda_{j+1}.$$

This leads to the recurrence

$$E_{j+1} = E_j \cdot \pi_{j+1}(k_{j+1})/\lambda_{j+1}$$

with $E_0 = T \cdot \pi_0(k_0)$ by Lemma 2. Its solution is

$$E_f = T \cdot \pi_0(k_0) \cdot \prod_{j=1}^{f} \pi_j(k_j)/\lambda_{j+1}.$$

Application of Lemma 3 proves the theorem.

Since an $(M_0,\ldots,M_f)$-MTD game assumes a much weaker defender than the corresponding layered MTD game, the layered MTD game may have a security capacity which is significantly larger than (7). In the example of Figure 4 and 5 the average (simulated) time to win the (M_0,M_1,M_2)-MTD game is 471.8 time steps while the average time to win the corresponding layered MTD game is 14257 timesteps.[7] Theorem 3 is a first analysis of layered MTD games and future research is needed to understand the exact effect of such composition.

Is it possible to derive a much stronger bound on the security capacity for $(M_0,\ldots,M_f)$-MTD games if the defender has stronger capabilities? E.g., if the defender is able to detect whether the adversary enters a state k_j, then we may give the defender the ability to immediately kick the adversary out of state k_j after one time slot. In the proof of Theorem 3 this means that we do not need to consider the arguments related to the Poisson process; the upper bound can be improved by discarding the factors λ_j. However, this does not change the asymptotic behavior of the upper bound and therefore extra investment in such advanced detection will likely not pay off (we expect this intuition to also hold for layered MTD games).

6.1 Time Conversion

How do we define a composition game made of levels with different time scales? Let us measure the time t_j per logical transition at level j in a universal time unit (e.g. seconds) such that $t_j \geq 1$ for all j. In reality defender and adversarial moves complete in very little time and corresponding transitions are close to immediate; the rate of play, however, measured in universal time units scales with $1/t_j$. The corresponding new transition matrix for level j is equal to

$$M_j^{new} = \frac{1}{t_j}M_j + (1-\frac{1}{t_j})I.$$

Since the time per transition in M_j^{new} is now 1 universal time unit regardless the level, we can apply the composition theorem to these new transition matrices.

Clearly the transformation to the new transition matrices does not have any effect on the equilibrium distribution π_j since for the new transition matrix we also have $\pi_j = \pi_j M_j^{new}$. This proves that $\pi_j(k_j)$ stays the same for M_j^{new}. However we notice that

$$\begin{aligned}\lambda_j^{new} &= 1-(M_j^{new})_{k_j,k_j} = 1-[\frac{1}{t_j}(M_j)_{k_j,k_j} + (1-\frac{1}{t_j})] \\ &= \frac{1}{t_j}(1-(M_j)_{k_j,k_j}) = \frac{\lambda_j}{t_j}\end{aligned}$$

Moreover, $T^{new} = T \cdot t_0$ since it is measured in time steps for the lowest level game. This yields the following corollary.

[7]The security capacity of an individual game M_i is at least $-\log \pi_i(2) = -\log 0.1 = 3.32$ and (7) gives at least a security capacity of $3 \cdot 3.32 + \log 0.25 = 7.97$ for the (M_0,M_1,M_2)-MTD game. The simulation results fit up to a constant factor the theoretical prediction as the average time to win an individual game M_i is 24.3 and is approximately twice $2^{3.32}$ and the average time to win the (M_0,M_1,M_2)-MTD game is 471.8 and is approximately twice $2^{7.97}$.

Corollary 1 *Assume that for each M_j-MTD game the transitions take on average $t_j \geq 1$ time units. Then, the probability that the attacker wins the layered MTD game based on $M_0, \ldots, M_f$ in the first T time units is*

$$\leq T \cdot t_0 \pi_0(k_0) \prod_{j=1}^{f} t_j \pi_j(k_j)/\lambda_j.$$

Hence, the security capacity of the layered MTD game measured in time units is at least (7) minus a time conversion factor $\log \prod_{j=0}^{f} t_j$.

7 Concluding Remarks and Future Directions

We have introduced a Markov model based framework for MTD analysis. The framework allows modeling of concrete MTD strategies, provides general theorems about how the probability of a successful adversary defeating an MTD strategy is related to the amount of time/cost spend by the adversary, and shows how a multi-level composition of MTD strategies can be analyzed by a straightforward combination of each individual MTD strategy's own analysis.

Central in the proposed framework is the concept of security capacity which measures the strength of an MTD strategy: the security capacity depends on MTD specific parameters and more general system parameters. As a first direction for future research the security capacity versus system performance can be analyzed as a function of system and MTD parameters. This allows the defender to understand the price in performance paid for his MTD strategy and how MTD and system parameters (given system constraints) should be set. Also several different MTD strategies can be compared and the most effective one with respect to the defender's cost in terms of performance can be selected.

MTD strategies hide targets which is a combinatorial game leading to information theoretic guarantees expressed by the security capacity. As demonstrated in the analysis of the multiple-target hiding MTD game, the security capacity is the largest if an adversary needs to find multiple targets (and not just one single target) which are each being hidden by the MTD strategy. The analysis shows that the probability of finding each target scales with the product of the probabilities of finding each specific target. This exponential dependency on the number of hidden targets is also demonstrated by the analysis of Layered MTD strategies (in our example, the number of hidden targets is equal to the number of levels). A second direction for future research is to analyse the Layered MTD composition in more detail and improve the lower bound on the security capacity given in this paper (as demonstrated for a toy example, even though the current proven bound shows an exponential dependency on the number of layers, it is much weaker when compared to simulation results). Additional future research in composition is to investigate into what extend MTD strategies can be composed according to a more effective (new) mechanism where the state in each MTD game gives feedback to each other MTD game in their composition. Feedback from one MTD game can be used to adapt the MTD parameters of another MTD game and this could lead to an overall larger security capacity of the composition.

References

[1] N. Adams and N. Heard. *Dynamic Networks and Cyber-security*, volume 1. World Scientific, 2016.

[2] E. Al-Shaer and M. A. Rahman. *Security and Resiliency Analytics for Smart Grids: Static and Dynamic Approaches*, volume 67. Springer, 2016.

[3] P. E. Ammann and J. C. Knight. Data diversity: An approach to software fault tolerance. *Computers, IEEE Transactions on*, 37(4):418–425, 1988.

[4] 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.

[5] C. Cadar, P. Akritidis, M. Costa, J.-P. Martin, and M. Castro. Data randomization. Technical report, Technical Report TR-2008-120, Microsoft Research, 2008. Cited on, 2008.

[6] K. M. Carter, J. F. Riordan, and H. Okhravi. A game theoretic approach to strategy determination for dynamic platform defenses. In *Proceedings of the First ACM Workshop on Moving Target Defense*, pages 21–30. ACM, 2014.

[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] B. Cox, D. Evans, A. Filipi, J. Rowanhill, W. Hu, J. Davidson, J. Knight, A. Nguyen-Tuong, and J. Hiser. N-variant systems: a secretless framework for security through diversity. In *Usenix Security*, volume 6, pages 105–120, 2006.

[9] D. Evans, A. Nguyen-Tuong, and J. Knight. Effectiveness of moving target defenses. In *Moving Target Defense*, pages 29–48. Springer, 2011.

[10] J. Han, D. Gao, and R. H. Deng. On the effectiveness of software diversity: A systematic study on real-world vulnerabilities. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 127–146. Springer, 2009.

[11] D. A. Holland, A. T. Lim, and M. I. Seltzer. An architecture a day keeps the hacker away. *ACM SIGARCH Computer Architecture News*, 33(1):34–41, 2005.

[12] T. Jackson, B. Salamat, G. Wagner, C. Wimmer, and M. Franz. On the effectiveness of multi-variant program execution for vulnerability detection and prevention. In *Proceedings of the 6th International Workshop on Security Measurements and Metrics*, page 7. ACM, 2010.

[13] J. H. Jafarian, E. Al-Shaer, and Q. Duan. Openflow random host mutation: transparent moving target defense using software defined networking. In *Proceedings of the first workshop on Hot topics in software defined networks*, pages 127–132. ACM, 2012.

[14] C. Kil, J. Jun, C. Bookholt, J. Xu, and P. Ning. Address space layout permutation (aslp): Towards fine-grained randomization of commodity software. In *null*, pages 339–348. IEEE, 2006.

[15] K.-w. Lye and J. M. Wing. Game strategies in network security. *International Journal of Information Security*, 4(1):71–86, 2005.

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

[17] A. Nguyen-Tuong, D. Evans, J. C. Knight, B. Cox, and J. W. Davidson. Security through redundant data diversity. In *Dependable Systems and Networks With FTCS and DCC, 2008. DSN 2008. IEEE International Conference on*, pages 187–196. IEEE, 2008.

[18] A. Nochenson and C. L. Heimann. Simulation and game-theoretic analysis of an attacker-defender game. In *Decision and Game Theory for Security*, pages 138–151. Springer, 2012.

[19] A. J. O'Donnell and H. Sethu. On achieving software diversity for improved network security using distributed coloring algorithms. In *Proceedings of the 11th ACM conference on Computer and communications security*, pages 121–131. ACM, 2004.

[20] H. Okhravi, T. Hobson, D. Bigelow, and W. Streilein. Finding focus in the blur of moving-target techniques. *Security & Privacy, IEEE*, 12(2):16–26, 2014.

[21] H. Okhravi, M. Rabe, T. Mayberry, W. Leonard, T. Hobson, D. Bigelow, and W. Streilein. Survey of cyber moving targets. *Massachusetts Inst of Technology Lexington Lincoln Lab, No. MIT/LL-TR-1166*, 2013.

[22] K. Onarlioglu, L. Bilge, A. Lanzi, D. Balzarotti, and E. Kirda. G-free: defeating return-oriented programming through gadget-less binaries. In *Proceedings of the 26th Annual Computer Security Applications Conference*, pages 49–58. ACM, 2010.

[23] M. Petkac, L. Badger, and W. Morrison. Security agility for dynamic execution environments. In *DARPA Information Survivability Conference and Exposition, 2000. DISCEX'00. Proceedings*, volume 1, pages 377–390. IEEE, 2000.

[24] T. Roeder and F. B. Schneider. Proactive obfuscation. *ACM Transactions on Computer Systems (TOCS)*, 28(2):4, 2010.

[25] B. Salamat, A. Gal, and M. Franz. Reverse stack execution in a multi-variant execution environment. In *Workshop on Compiler and Architectural Techniques for Application Reliability and Security*, pages 1–7, 2008.

[26] B. Salamat, T. Jackson, G. Wagner, C. Wimmer, and M. Franz. Runtime defense against code injection attacks using replicated execution. *Dependable and Secure Computing, IEEE Transactions on*, 8(4):588–601, 2011.

[27] B. Schneier. Attack trees. *Dr. Dobb's journal*, 24(12):21–29, 1999.

[28] 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.

[29] A. N. Sovarel, D. Evans, and N. Paul. Where's the feeb? the effectiveness of instruction set randomization. In *Usenix Security*, 2005.

[30] O. website of the Department of Homeland Security. Moving target defense. https://www.dhs.gov/science-and-technology/csd-mtd. [Online; accessed 10-May-2016].

[31] K. Zaffarano, J. Taylor, and S. Hamilton. A quantitative framework for moving target defense effectiveness evaluation. In *Proceedings of the Second ACM Workshop on Moving Target Defense*, pages 3–10. ACM, 2015.

[32] Q. Zhu and T. Başar. Game-theoretic approach to feedback-driven multi-stage moving target defense. In *Decision and Game Theory for Security*, pages 246–263. Springer, 2013.

[33] R. Zhuang. *A theory for understanding and quantifying moving target defense*. PhD thesis, Kansas State University, 2015.

[34] R. Zhuang, A. G. Bardas, S. A. DeLoach, and X. Ou. A theory of cyber attacks: A step towards analyzing mtd systems. In *Proceedings of the Second ACM Workshop on Moving Target Defense*, pages 11–20. ACM, 2015.

[35] 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 Annual Cyber and Information Security Research Conference*, pages 73–76. ACM, 2014.

[36] R. Zhuang, S. A. DeLoach, and X. Ou. Towards a theory of moving target defense. In *Proceedings of the First ACM Workshop on Moving Target Defense*, pages 31–40. ACM, 2014.

[37] R. Zhuang, S. Zhang, S. A. DeLoach, X. Ou, and A. Singhal. Simulation-based approaches to studying effectiveness of moving-target network defense. In *National symposium on moving target research*, 2012.

[38] C. Zimmerman. Ten strategies of a world-class cybersecurity operations center. *MITRE Corporation report release, USA*, 2014.

Appendix: Cost Analysis

We prove Theorem 2: Let $A_{i,j}$ represent the probability the adversary moves in state i and reaches state j as a consequence of the move. Let the real number $c_{i,j} \geq 0$ indicate the adversarial cost associated to $A_{i,j}$. Then

$$\alpha = \sum_{i,j} \pi_i A_{i,j} c_{i,j} \tag{8}$$

is a convex combination of costs $c_{i,j}$ (since $\sum_{i,j} \pi_i A_{i,j} \leq \sum_{i,j} \pi_i M_{i,j} = \sum_j \pi_j = 1$). We need to prove that the probability the attacker wins the M-MTD game by paying at most C is

$$\leq \frac{C \cdot \pi_k}{\sum_{i,j} \pi_i A_{i,j} c_{i,j}}.$$

We first introduce a mapping of matrix M to a new matrix M^* which we can analyze using our theorem but has all the necessary cost information embedded:

First, we split each transition from state i to state j with probability $M_{i,j}$ into two transitions. One from i to j with probability $A_{i,j}$ and the other from i to j with probability $D_{i,j} = M_{i,j} - A_{i,j}$: $A_{i,j}$ represents the probability that the adversary moves in state i and reaches state j as a consequence of that move, while $D_{i,j}$ represents the probability of a defender's move or no move (if $i = j$) from state i to state j.

Second, let $c_{i,j} \geq 0$ be a real number representing the cost for the adversary if he moves from i to j with probability $A_{i,j}$. Let $\epsilon \geq 0$ (later in the proof we will take the limit $\epsilon \to 0$) and define

$$C_{i,j} = \lceil \frac{c_{i,j}}{\epsilon} \rceil \in \mathbb{N}$$

We transform each edge from $i \to j$ with probability $A_{i,j}$ into a path $i \to (i,j;1) \to (i,j;2) \to \ldots \to (i,j;C_{i,j}) \to j$ with probabilities as depicted in Figure 6. The transition $i \to (i,j;1)$ has probability $A_{i,j}$. If $C_{i,j} = 0$, then i is directly connected to j without intermediary nodes and $i \to j$ is transitioned with probability $A_{i,j}$ by the adversary. We denote the transformed matrix by M^*.

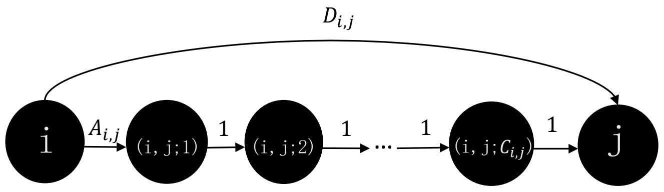

Figure 6: Each transition $M_{i,j}$ from $i \to j$ will be replaced with above transition from state i to j

Lemma 4 *If M has an equilibrium probability distribution π, then M^* has an equilibrium distribution π^* defined by:*

$$\begin{aligned} \pi_i^* &= \frac{\pi_i}{1+\alpha} \textit{ and} \\ \pi_{i,j;h}^* &= \frac{\pi_i \cdot A_{i,j}}{1+\alpha} \textit{ for } 1 \leq h \leq C_{i,j}. \end{aligned}$$

Proof. From $\pi^* = \pi^* M^*$ and the definition of M^* we infer the following equations:

$$\begin{aligned} \pi_{i,j;1}^* &= \pi_i^* \cdot A_{i,j}, && \text{if } C_{i,j} \geq 1, \\ \pi_{i,j;h+1}^* &= \pi_{i,j;h}^*, && \text{for } 1 \leq h \leq C_{i,j}, \\ \pi_j^* &= \sum_i \pi_i^* D_{i,j} + \sum_{\substack{i \\ C_{i,j} \neq 0}} \pi_{i,j;C_{i,j}}^* + \sum_{\substack{i \\ C_{i,j}=0}} \pi_i^* \cdot A_{i,j}. \end{aligned}$$

By combining the above equations we have,

$$\begin{aligned} \pi_j^* &= \sum_i \pi_i^* D_{i,j} + \sum_{\substack{i \\ C_{i,j} \neq 0}} \pi_{i,j;1}^* + \sum_{\substack{i \\ C_{i,j}=0}} \pi_i^* \cdot A_{i,j} \\ &= \sum_i \pi_i^* D_{i,j} + \sum_i \pi_i^* \cdot A_{i,j} = \sum_i \pi_i^* M_{i,j} \end{aligned}$$

We conclude that the vector $(\pi_1^*, \pi_2^*, ...)$ is proportional to vector π and this proves $\pi_i^* = \pi_i/\rho$ for some ρ. As a result

$$\pi_{i,j;h}^* = \pi_{i,j;1}^* = \pi_i^* \cdot A_{i,j} = \frac{\pi_i \cdot A_{i,j}}{\rho} \text{ for } 1 \leq h \leq C_{i,j}.$$

The proportionality factor ρ satisfies

$$\begin{aligned} 1 &= \sum_i \pi_i^* + \sum_{\substack{i,j,h \\ C_{i,j}\neq 0 \\ 1\leq h\leq C_{i,j}}} \pi_{i,j;h}^* \\ &= \sum_i \frac{\pi_i}{\rho} + \sum_{\substack{i,j,h \\ C_{i,j}\neq 0 \\ 1\leq h\leq C_{i,j}}} \frac{\pi_i \cdot A_{i,j}}{\rho} \\ &= \sum_i \frac{\pi_i}{\rho} + \sum_{i,j} \frac{\pi_i \cdot A_{i,j} \cdot C_{i,j}}{\rho} \end{aligned}$$

Together with $\sum_i \pi_i = 1$ and (8), this proves $\rho = 1 + \alpha$ and the lemma follows.

In order to prove the theorem we introduce new random variables: Let T_M be the random variable indicating the number of transitions $i \xrightarrow{D_{i,j}} j$, $i \xrightarrow{A_{i,j}} (i,j;1)$ (if $C_{i,j} \geq 1$) and $i \xrightarrow{A_{i,j}} j$ (if $C_{i;j} = 0$), made before the adversary wins the MTD-game corresponding to M^*.

Let $T_C = \sum_{i,j} N_{i,j} \cdot C_{i,j}$ be the random variable indicating the number of all other transitions where $N_{i,j}$ represents the random variable counting the number of times the path$(i,j;1) \xrightarrow{1} (i,j;2) \xrightarrow{1} \ldots \xrightarrow{1} (i,j;C_{i,j})$ is traversed. Notice that $T = T_M + \sum_{i,j} N_{i,j}$ is the random variable representing the number of transitions in the M-MTD game before the attacker wins the game.

Let the random variable C_R represent the real cost of the adversary before he wins the MTD-game corresponding to M. Notice that, for random variables $N_{i,j}$ in the MTD-game corresponding to M^*,

$$C_R = \sum_{i,j} N_{i,j} \cdot c_{i,j}.$$

This shows how the MTD games based on M and M^* are related.

We derive

$$\begin{aligned} T_C &= \sum_{i,j} N_{i,j} \cdot C_{i,j} = \sum_{i,j} N_{i,j} \cdot \lceil \frac{c_{i,j}}{\epsilon} \rceil \\ &= \frac{1}{\epsilon} \sum_{i,j} N_{i,j} \cdot (\lceil \frac{c_{i,j}}{\epsilon} \rceil \cdot \epsilon) \\ &= \frac{1}{\epsilon} \sum_{i,j} N_{i,j} \cdot c_{i,j} + \frac{1}{\epsilon} \sum_{i,j} N_{i,j} \cdot (\lceil \frac{c_{i,j}}{\epsilon} \rceil \cdot \epsilon - c_{i,j}) \\ &= \frac{C_R}{\epsilon} + E \text{ where } 0 \leq E \leq \sum_{i,j} N_{i,j} \qquad (9) \end{aligned}$$

Since $T = T_M + \sum_{i,j} N_{i,j}$ represents the number of transitions in the M-MTD game before the attacker wins the game, we can use Theorem 1 in combination with (9) and derive for all t:

$$\begin{aligned}
& \frac{C \cdot \pi_k^*}{\epsilon} && (10) \\
\geq\ & Prob(T_M + T_C \leq \frac{C}{\epsilon}) = Prob(T_M + \frac{C_R}{\epsilon} + E \leq \frac{C}{\epsilon}) \\
=\ & Prob(C_R \leq C - \epsilon(T_M + E)) \\
\geq\ & Prob(C_R \leq C - \epsilon t | T_M + E \leq t) Prob(T_M + E \leq t) \\
=\ & Prob(C_R \leq C - \epsilon t) - \\
& Prob(C_R \leq C - \epsilon t | T_M + E \geq t) Prob(T_M + E \geq t) \\
\geq\ & Prob(C_R \leq C - \epsilon t) - Prob(T_M + E \geq t) \\
\geq\ & Prob(C_R \leq C - \epsilon t) - Prob(T_M + \sum_{i,j} N_{i,j} \geq t) \\
=\ & Prob(C_R \leq C - \epsilon t) - Prob(T \geq t). && (11)
\end{aligned}$$

If we choose $\epsilon = \frac{1}{t^2}$ and let $t \to \infty$ then (11) tends to $Prob(C_R \leq C)$.

We now analyse upper bound (10) by using Lemma 4:

$$\frac{C \cdot \pi_k^*}{\epsilon} = \frac{C \cdot \pi_k}{\epsilon(1+\alpha)}, \text{ where}$$

$$\begin{aligned}
\epsilon(1+\alpha) &= \epsilon + \sum_{i,j} \pi_i A_{i,j} C_{i,j} \cdot \epsilon = \epsilon + \sum_{i,j} \pi_i A_{i,j} \lceil \frac{c_{i,j}}{\epsilon} \rceil \epsilon \\
&= \epsilon + \sum_{i,j} \pi_i A_{i,j} c_{i,j} + \sum_{i,j} \pi_i A_{i,j} (\lceil \frac{c_{i,j}}{\epsilon} \rceil \epsilon - c_{i,j})
\end{aligned}$$

Hence,

$$\begin{aligned}
& \epsilon(1+\alpha) - \sum_{i,j} \pi_i A_{i,j} c_{i,j} \leq \epsilon + \sum_{i,j} \pi_i A_{i,j} (\lceil \frac{c_{i,j}}{\epsilon} \rceil \epsilon - c_{i,j}) \\
& \leq \epsilon + \sum_{i,j} \pi_i M_{i,j} \epsilon = \epsilon + \sum_j \pi_j \epsilon = 2\epsilon.
\end{aligned}$$

This proves that, for $\epsilon = \frac{1}{t^2}$ and $t \to \infty$, $\epsilon(1+\alpha) \to \sum_{i,j} \pi_i A_{i,j} c_{i,j} = \alpha$ and the theorem follows:

$$Prob(C_R \leq C) \leq C \cdot \pi_k / \alpha.$$