A Stackelberg Game and Markov Modeling of Moving Target Defense

Xiaotao Feng[1], Zizhan Zheng[2], Prasant Mohapatra[3], and Derya Cansever[4]

[1] Department of Electrical and Computer Engineering, University of California, Davis, USA
[2] Department of Computer Science, Tulane University, New Orleans, USA
[3] Department of Computer Science, University of California, Davis, USA
[4] U.S. Army Research Laboratory, USA
Email: {xtfeng, pmohapatra}@ucdavis.edu, zzheng3@tulane.edu, derya.h.cansever.civ@mail.mil

Abstract. We propose a Stackelberg game model for Moving Target Defense (MTD) where the defender periodically switches the state of a security sensitive resource to make it difficult for the attacker to identify the real configurations of the resource. Our model can incorporate various information structures. In this work, we focus on the worst-case scenario from the defender's perspective where the attacker can observe the previous configurations used by the defender. This is a reasonable assumption especially when the attacker is sophisticated and persistent. By formulating the defender's problem as a Markov Decision Process (MDP), we prove that the optimal switching strategy has a simple structure and derive an efficient value iteration algorithm to solve the MDP. We further study the case where the set of feasible switches can be modeled as a regular graph, where we solve the optimal strategy in an explicit way and derive various insights about how the node degree, graph size, and switching cost affect the MTD strategy. These observations are further verified on random graphs empirically.

1 Introduction

In cybersecurity, it is often the case that an attacker knows more about a defender than the defender knows the attacker, which is one of the major obstacles to achieve effective defense. Such information asymmetry is a consequence of time asymmetry, as the attacker often has abundant time to observe the defender's behavior while remaining stealthy. This is especially the case for incentive-driven targeted attacks, such as Advanced Persistent Threats (APT). These attacks are highly motivated and persistent in achieving their goals. To this end, they may intentionally act in a "low-and-slow" fashion to avoid immediate detection [1].

Recognizing the shortage of traditional cyber-defense techniques in the face of advanced attacks, Moving Target Defense (MTD) has been recently proposed as a promising approach to reverse the asymmetry in information or time in cybersecurity [2]. MTD is built upon the key observation that to achieve a successful

compromise, an attacker requires knowledge about the system configuration to identify vulnerabilities that he is able to exploit. However, the system configuration is under the control of the defender, and multiple configurations may serve the system's goal, albeit with different performance security tradeoffs. Thus, the defender can periodically switch between configurations to increase the attacker's uncertainty, which in turn increases attack cost/complexity and reduces the chance of a successful exploit in a given amount of time. This high level idea has been applied to exploit the diversity and randomness in various domains, including computer networks [3], system platforms [4], runtime environment, software code, and data representation [5].

Early work on MTD mainly focus on empirical studies of domain specific dynamic configuration techniques. More recently, decision and game theoretic approaches have been proposed to reason about the incentives and strategic behavior in cybersecurity to help derive more efficient MTD strategies. In particular, a stochastic game model for MTD is proposed in [6], where in each round, each player takes an action and receives a payoff depending on their joint actions and the current system state, and the latter evolves according to the joint actions and a Markov model. Although this model is general enough to capture various types of configurations and information structures and can be used to derive adaptive MTD strategies, solutions obtained are often complicated, making it difficult to derive useful insights for practical deployment of MTD. Moreover, existing stochastic game models for MTD focus on Nash Equilibrium based solutions and do not exploit the power of commitment for the defender. To this end, Bayesian Stackelberg games (BSG) has been adapted to MTD recently [7]. In this model, before the game starts, the defender commits to a mixed strategy – a probability distribution over configurations – and declare it to the attacker, assuming the latter will adopt a best response to this randomized strategy. Note that, the defender's mixed strategy is independent of real time system states, so does the attacker's response. Thus, a BSG can be considered as a repeated game without dynamic feedback. Due to its simplicity, efficient algorithms have been developed to solve BSG in various settings, with broad applications in both physical and cyber security scenarios [8]. However, a direct application of BSG to MTD as in [8] ignores the fact that both the attacker and the defender can adapt their strategies according to the observations obtained during the game.

In this paper, we propose a non-zero-sum Stackelberg game model for MTD that incorporates real time states and observations. Specifically, we model the defender's strategy as a set of transition probabilities between configurations. Before the game starts, the defender declares its strategy to the attacker. Both players take rounds to make decisions and moves. In the beginning of each round, the defender moves from the current configuration to a new one (or stay on the current one) according to the transition probabilities. Note that this is more general than [8], where the defender picks the next configuration independently of the current one. Our approach also allows us to model the long-term switching cost in a more accurate way. Moreover, we assume that the attacker can get some feedback during the game. This is especially true for advanced attacks. In this

paper, we consider the extreme case where the attacker knows the previous configuration used by the defender in the beginning of each round (even if it fails in the previous round). This is the worst-case scenario from the defender's perspective. However, our model can be readily extended to settings where the attacker gets partial feedback or no feedback.

To derive the optimal MTD strategy for the defender, we model the defender's problem as a Markov decision process (MDP). Under the assumptions that all the configurations have the same value to the defender and require the same amount of effort to compromise for the attacker, we prove that the optimal stationary strategy has a simple structure. Based on this observation, we derive efficient value iteration algorithm to solve the MDP. We further study the case where the switching cost between any pair of configurations is either a unit or infinite. In this case, the configuration space can be modeled as a directed graph. When the graph is regular, we derive the optimal strategy in an explicit way and prove that it is always better to have a higher degree in the graph, but the marginal improvement decreases when the diversity increases. This observation is further verified on random graphs empirically.

We have made the following contributions in this paper

- We propose a Stackelberg game model for moving target defense that combines Markovian defense strategies and realtime feedback.
- We model the defender's problem as a Markov decision process and derive efficient algorithms based on some unique structural properties of the game.
- We derive various insights on efficient MTD strategies using our models. In particular, we study how the diversity of the configuration space affects the effectiveness of MTD, both analytically and empirically.

The remainder of the paper is organized as follows. We introduce the related work in Section 2 and propose the game model in Section 3. Detailed solutions for optimal strategies and a special case study are presented in Section 4. The performance of optimal strategies under different scenarios are evaluated via numerical study in Section 5. Finally, we conclude the paper in Section 6.

2 Related Work

As a promising approach to achieve proactive defense, MTD techniques have been investigated in various cybersecurity scenarios [2,3,4,5]. A fundamental challenge of large scale deployment of MTD, however, is to strike a balance between the risk of being attacked and the extra cost introduced by MTD including the extra resource added, the migration costs and the time overhead. To this end, game theory provides a proper framework to analyze and evaluate the key tradeoffs involved in MTD [9].

In this paper, we propose a non-zero-sum Stackelberg game model for MTD where the defender plays as the leader and the attacker plays as the follower and both players make their decision sequentially. Sequential decision making with limited feedback naturally models many security scenarios. Recently, inspired

by poker games, an efficient sub-optimal solution for a class of normal-form games with sequential strategies is proposed in [10]. However, the solution is only applicable to zero-sum games, while the MTD game is typically non-zero-sum as the defender usually has a non-zero migration cost.

Stackelberg game models have been extensively studied in cybersecurity as they capture the fact that a targeted attacker may observe a finite number of defender's actions and then estimate the defender's strategy [11]. This is especially true for an APT attacker. By exercising the power of commitment, the defender (leader) can take advantages of being observed to alert the attacker.

In the context of MTD, several Stackelberg game models have been proposed [7,12,8]. In particular, a Stackelberg game is proposed for dynamic platform defense against uncertain threat types [7]. However, this work does not consider the moving cost for platform transitions, which should be taken into consideration on strategy design. A Stackelberg game for MTD against stealthy attacks is proposed in [12], where it is shown that MTD can be further improved through strategic information disclosure. One limitation of this work is that the authors only consider a one-round game.

More recently, a Bayesian Stackelberg Game (BSG) model is proposed for MTD in Web applications [8], where multiple types of attackers with different expertise and preferences are considered. Both theoretical analysis and experimental studies are given in [8]. However, to adapt the classic BSG model to MTD, the defender's strategy is defined as a probability distribution over states and is *i.i.d.* over rounds, which is a strong limitation. In contrast, we defined the defender's strategy as the set of transition probabilities between states. Such a Markovian strategy is not only more natural in the context of MTD, but also allows us to incorporate real time feedback available to the players.

Our model is similar in spirit to stochastic game models [6] and recent Markov models for MTD [13,14]. However, existing stochastic game models for MTD focus on Nash Equilibria instead of Stackelberg Equilibria. Moreover, solutions to stochastic games are often complicated and hard to interpret. More recently, several Markov models for MTD have been proposed [13,14]. Due to the complexity of these models, only preliminary analytic results for some special cases are provided. In particular, these work focus on analyzing the expected time needed for the attacker to compromise the resource under some simple defense strategies.

3 Game model

In this section, we formally present our MTD game model. There are two players in the game who fight for a security sensitive resource. The one who protects the resource is called the defender while the one who tries to comprise the resource is called the attacker. Below we discuss each element of the game model in details.

Resource: We consider a single resource with N features, where for the i-th feature, there are m_i possible configurations that can be chosen by the defender, denoted by $\mathbf{c}_i$ with $|\mathbf{c}_i| = m_i$. We define the *state* of the resource at any time as

the set of configurations of all the features, $s = \{c_i \in \mathbf{c}_i, i = 1, 2, \cdots, N\}$. For example, the resource can represent a critical cyber system with features such as its processor architecture, operating system, storage system, virtual machine instances, network address space, and communication channels, etc. Each feature has several possible configurations such as Windows/Linux for operating system, a range of IP addresses for network address space and so on. Moreover, the concept of resource is not limited to the cyber world. It can also represent physical entities such as military units, vulnerable species, and antiques.

We define a state as *valid* if it is achievable by the defender and the resource can function properly under that state. Although the maximum possible states of the resource can be $\prod_{i=1}^{N} m_i$, typically only a small number of them are valid. For instance, consider a mobile app that with two features: $\big\{$program language $\in$ $\{$Objective-C, Java, JavaScript$\}$, operating system $\in \{$iOS, Android$\}\big\}$. The maximum number of states for the app is 6. However, since a Java based app is incompatible with iOS, and an Objective-C based app is incompatible with Android, there are only 4 valid states. We denote the set of valid states as $V = \{1, 2, \cdots, |V|\}$.

Defender: To protect the resource, the defender periodically switches the state to make it difficult for the attacker to identify the real state of the resource. A switch is achieved by changing the configurations of one or more features and is subject to a cost. Note that not all the switches between valid states are feasible as it can be extremely difficult or even impossible to switch between two valid states in some cases.

Attacker: We assume that the attacker can potentially attack all the valid states of the resource. Note that if the defender knows that the attacker does not have technical expertise to attack certain states, then the defender should always keep the resource in those states. We leave the case where the defender is uncertain about the attacker's capability in the future work.

Before each attack, the attacker selects an attack scheme that targets at a specific configuration combination (state) of the resource. We assume that the attacker can compromise the resource successfully if and only if the selected attack scheme matches the real state of the resource. Due to this 1-1 correspondence, we simply define the attacker's action space as the set of valid states V. We further assume that the attacker can only observe and exploit the state of the resource but cannot modify it through successful attacks. That is, the state of the resource is completely under the control of the defender.

The rules of the MTD game are introduced below.

1. The game is a turn based Stackelberg Game in which the defender plays as the leader and the attacker plays as the follower.
2. The game starts at turn $t = 0$ with the resource initially in state $s_0 \in V$ (chosen by the defender), and lasts for a possibly infinite number of turns T.
3. Each turn begins when the defender takes action. We assume that the defender moves periodically and normalize the length of each turn to a unit.

4. At the beginning of turn t, the defender switches the resource from s_t to s_{t+1} with a switching cost $c_{s_t s_{t+1}}$, and the attacker selects one state $a_t \in V$ to attack. We assume that the attacker attacks once each turn. Moreover, both switching and attacking are effective instantly.
5. If the attacker is successful at turn t (that is, if $a_t = s_{t+1}$), he obtains a reward of 1, while the defender incurs a loss of 1 (not including the switching cost). Otherwise, there is no reward obtained or loss incurred.

A Graphical View: We can model the set of states and state switches as a directed graph. For example, Fig. 1a shows a fully connected graph with the set of states as nodes and state switches as links. We then eliminate some invalid states and invalid switches to get Fig. 1b. The defender chooses one node as initial state s_0 at the beginning of the game. The attacker selects one node a_t as the target in each turn. Every valid state has a self loop meaning that that no switch is always one option for the defender. We define the *outdegree* (or *degree* for short) of a node as the number of outgoing links from the node, or equivalently, the number of states that can be switched to from the state. We define the *neighbor* of state s as a set $N(s) = \{s' \in V | c_{ss'} \neq \infty\}$, $\forall s \in V$. The degree of node s is equal to $|N(s)|$.

The graph can be uniquely determined by V and a matrix $\mathcal{C} = \{c_{ss'}\}_{|V|\times|V|}$, where $c_{ss'}$ represents the switching cost between two states s and s'. There is no link between s and s' if $c_{ss'} = \infty$, and $c_{ss'} = 0$ if $s' = s$. We expect that the switching costs can be learned from history data and domain knowledge [8].

Consider again the example given above. There are four valid states corresponding to four nodes. Let nodes $1, 2, 3$ and 4 represent $\{\text{Objective-C}, \text{iOS}\}$, $\{\text{JavaScript}, \text{iOS}\}$, $\{\text{JavaScript}, \text{Android}\}$ and $\{\text{Java}, \text{Android}\}$, respectively. An example of the cost matrix $\mathcal{C}$ and the corresponding graph are given in Fig. 2. In this example, if the current state of the resource is at node 1, the defender may keep the state at node 1 without any expense, or switch the state from node 1 to node 2 or node 3 with a switching cost 0.8 and 1.5, respectively. However, the defender cannot switch the resource from node 1 to node 4 in one step as there is no direct link between them.

$$\mathcal{C} = \begin{pmatrix} 0 & 0.8 & 1.5 & \infty \\ 0.7 & 0 & 0.6 & 1.6 \\ 1.3 & 0.5 & 0 & 0.4 \\ \infty & 1.2 & 0.4 & 0 \end{pmatrix}$$

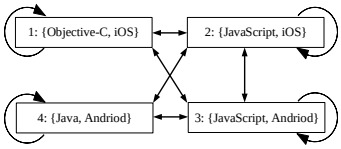
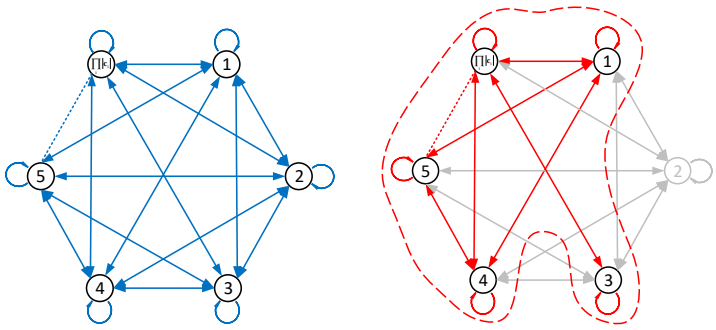


Fig. 2: A resource with 4 states and 14 switch pairs

3.1 Attacker's Strategy

We define the attacker's strategy and payoff in this subsection. In order to decide a_t, the attacker forms a prior belief $\mathbf{q}_t = \{q_s \mid s \in V\}$ regarding the probability distribution of states according to the feedback obtained during the game and the previous actions (to be discussed). For the sake of simplicity, we assume that



(a) A fully connected graph (b) A subgraph after elimination of invalid states and valid links

Fig. 1: All the possible switch pairs modeled by a graph

the attacking cost is identical for all the states and it is always beneficial to attack. Thus, the attacker always selects $a_t = \text{argmax}_{s \in V} q_s$ at turn t.

3.2 Defender's Strategy and Cost

The defender's objective is to strike a balance between the loss from attacks and the cost of switching states. To this end, the defender commits to a strategy and declares it to the attacker before the game starts. As in Bayesian Stackelberg Games, the defender should adopt a randomized strategy taking into account the possible response of the attacker. In this work, we define the defender's strategy as a set of transition probabilities $P = \{p_{ss'}\}_{|V| \times |V|}$, where $p_{ss'}$ is the probability of switching the resource to s' given that the current state is s. The defender commits to an optimal P in the beginning and then samples the state in each turn according to P. We require that $p_{ss'} = 0$ if $c_{ss'} = \infty$ and $\sum_{s' \in V} p_{ss'} = 1$, $\forall s \in V$. Given a pair of states s_t, s_{t+1}, the defender's cost at turn t can be then defined as follows:

$$c(s_t, s_{t+1}) = 1_{\{a_t = s_{t+1}\}} + c_{s_t s_{t+1}} \tag{1}$$

The first term in (1) represents the loss from being attacked where $1_{\{a_t = s_{t+1}\}} = 1$ if $a_t = s_{t+1}$ and is 0 otherwise. The second term depicts the switching cost.

3.3 Feedback During the Game

The main purpose of MTD is to reverse information asymmetry. Thus, it is critical to define the information structure of the game. We assume that both players know the defender's strategy and all the information about the resource such as V and $\mathcal{C}$ before the game starts. However, the players have different feedback during the game:

– **Defender**: As the leader of Stackelberg game, the defender declares her strategy P and initial state s_0 to the public. The defender would not change P and $\mathcal{C}$ during the game. In each turn, the defender knows if the attacker has a successful attack or not.
– **Attacker**: As the follower of Stackelberg game, the attacker knows P and s_0. After attacking at any turn t, the attacker knows if the attack is successful or not. If the attack is successful, the attacker knows s_t immediately. Otherwise, we assume that the attacker spends this turn to learn s_t and will know s_t at the end of this turn. In both cases, $\mathbf{q}_t = \mathbf{p}_{s_t}$, where $\mathbf{p}_{s_t}$ represents the s_t-th row in P. This is the worst-case scenario from the defender's perspective. We will leave the case where attacker only gets partial feedback or no feedback to the future work.

3.4 Defender's Problem as a Markov Decision Process

Given the feedback structure defined above, we have $a_t = \text{argmax}_{s \in V} p_{s_t s}$ for any t. Hence, the defender's expected loss at turn t is:

$$E\left[1_{\{a_t = s_{t+1}\}}\right] = E\left[1_{\{s_{t+1} = \text{argmax}_{s \in V} p_{s_t s}\}}\right] = \max \mathbf{p}_{s_t} \tag{2}$$

Therefore, given P and s_t, the defender's expected cost at turn t is

$$\begin{aligned} c_P(s_t) &\triangleq E_{s_{t+1}}\left[c(s_t, s_{t+1})\right] \\ &= \max \mathbf{p}_{s_t} + \sum_{s_{t+1} \in N(s_t)} p_{s_t s_{t+1}} c_{s_t s_{t+1}} \end{aligned} \tag{3}$$

In this work, we consider the defender's objective to be minimizing its long-term discounted cost defined as $\sum_{t=0}^{\infty} \alpha^t c(s_t)$ where $\alpha \in (0, 1)$ is the discounted factor. One interpretation of α is that the defender would prefer to minimize the cost at current turn rather than future turns because she is not sure if the attacker will attack at the next turn. A higher discount factor indicates that the defender is more patient.

For a given P and an initial state s_0, the state of the resource involves according to a Markov chain with V as its state space and P as the transition probabilities. Thus, the defender's problem can be considered as a discounted Markov decision problem where the defender's strategy and the transition probabilities coincide. We can rewrite the defender's long-term cost with the initial state $s_0 = s$ as follows:

$$\begin{aligned} C_P(s) =& \sum_{t=0}^{\infty} c_P(s_t) \\ =& c_P(s) + \alpha \sum_{s' \in N(s)} p_{ss'} E\left[\sum_{t=0}^{\infty} \alpha^t c(s_{t+1}, s_{t+2}) \mid s_1 = s'\right] \\ =& c_P(s) + \alpha \sum_{s' \in N(s)} p_{ss'} C_P(s') \end{aligned} \tag{4}$$

3.5 Discussion about the MTD Model

In the BSG model for MTD in [8], the defender's strategy is defined as a probability distribution $\mathbf{x} = \{x_s \mid \forall s \in V\}$ over states, and the expected switching cost is defined as $\sum_{s,s' \in V} c_{ss'} x_s x_{s'}$. This model implies that at each turn, the defender samples the next state independent of the current state of the resource. In contrast, we define the defender's strategy as a set of transition probabilities between states. Our choice is not only more natural for MTD, but also considers a richer set of defense strategies. Note that different transition probability matrices may lead to the same stationary distribution of states, but with different switching costs, which cannot be distinguished using the formulation in [8]. Our approach provides a more accurate definition of the defender's real cost. We show that by modeling the problem as a MDP, we can still find the optimal defense strategy in this more general setting. Moreover, the MDP can be solved in an explicit way under certain system settings, which provides useful insights to the design of MTD strategies, as we discuss below.

4 Defender's Optimal Strategy and Cost

In this section, we solve the defender's optimal strategy as well as the optimal cost under different scenarios. Recall that the defender's problem is to find a strategy such that the cost in (4) is minimized from any initial state. Let $C^*(s)$ denote the defender's optimal cost with an initial state s, where

$$C^*(s) = \min_P C_P(s) \tag{5}$$

According to the theory of MDP, it is possible to find an optimal strategy P^* that simultaneously optimizes the cost for any initial state $s \in V$; that is,

$$P^* = \text{argmin}_P C_P(s), \forall s \in V \tag{6}$$

4.1 Algorithms for Solving the MDP

According to (3) and (4), we expand $C_P(s)$ in (5) and rewrite $C^*(s)$ in the following form,

$$C^*(s) = \min_P \left[\max \mathbf{p}_s + \sum_{s' \in N(s)} \big(c_{ss'} + \alpha C_P(s')\big) p_{ss'} \right] \tag{7}$$

In order to solve (7), we employ the standard value iteration algorithm to find the defender's optimal cost as well as the optimal strategy. Algorithm 1 shows the value iteration algorithm, where $C^\tau(s)$ is the cost at state s in the $\tau-$th iteration. Initially, the value of $C^\tau(s)$ is set to 0 for all s. In each iteration, the algorithm updates $C^\tau(s)$ by finding the optimal strategy that solves (7) using the costs in the previous iteration (step 4), which involves solving a Min-Max problem.

Although the value iteration algorithm is standard, solving the Min-Max problem in step 4 of Algorithm 1 directly is computationally expensive. Note that the decision variables $p_{ss'}$ can take any real value in $[0,1]$. One way to solve the problem is to approximate the search space $[0,1]$ by a discrete set $\{0, \frac{1}{M}, \frac{2}{M}, ..., \frac{M-1}{M}, 1\}$ where M is a parameter. The search space over all the neighbors of s has a size of $O(M^{|V|})$. A suboptimal solution can be obtained by searching over this space, which is expensive when M and $|V|$ are large. Rather than solving it directly, we first derive some properties of the MDP, which helps reduce the computational complexity significantly.

Algorithm 1 Value Iteration Algorithm for the MTD game

Input: V, $\mathcal{C}$, α, ϵ.
Output: P^*, $C^*(s)$.
1: Set $\tau = 0$, $C^\tau(s) = 0$, $\forall s \in V$; $\{C^\tau(s)$ is the cost at state s in the $\tau-$th iteration$\}$
2: **repeat**
3: $\quad \tau = \tau + 1$;
4: $\quad \mathbf{p}_s^* = \text{argmin}_{\mathbf{p}_s} \left[\max \mathbf{p}_s + \sum_{s' \in N(s)} p_{ss'} \left(c_{ss'} + \alpha C^{\tau-1}(s')\right)\right]$, $\forall s \in V$;
5: $\quad C^\tau(s) = C_{P^*}(s)$, $\forall s \in V$;
6: **until** $\sum_{s \in V} |C^\tau(s) - C^{\tau-1}(s)| \leq \epsilon$
7: $C^*(s) = C^\tau(s)$, $\forall s \in V$

Before presenting the results, we first give some definitions. Fix a state s. For any $s' \in N(s)$, let $\theta_{s'} = c_{ss'} + \alpha C^{\tau-1}(s')$ denote the coefficient of $p_{ss'}$ in the second term of the Min-Max problem in the τ-th iteration. Let $s^1, s^2, ..., s^{N(s)}$ denote the set of neighbors of s sorted according to their θ values nondecreasingly. We abuse the notation a little bit and let $\theta_i = \theta_{s^i}$.

The following lemma shows that the Min-Max problem can be simplified as a minimization problem.

Lemma 1. *Let P be the optimal solution to the Min-Max problem in the τ-th iteration of Algorithm 1. We have $p_{ss^1} = \max \mathbf{p}_s$.*

Proof. Assume $p_{ss^1} < \mathbf{p}_s$. Let $p_{ss^i} = \max \mathbf{p}_s$ for some $s^i \in N(s)$ and $p_{ss^i} = p_{ss^1} + \epsilon_1$ for some $\epsilon_1 > 0$. By the definition of s^1, there is $\epsilon_2 \geq 0$ such that $\theta_i = \theta_1 + \epsilon_2$. From the definition of P and s^i, we have

$$\begin{aligned}
C^\tau(s) &= p_{ss^i} + \sum_{s^j \in N(s)} p_{ss^j}\theta_j \\
&= p_{ss^1}\theta_1 + p_{ss^i}(1+\theta_i) + \sum_{s^j \in N(s)\setminus\{s^1,s^i\}} p_{ss^j}\theta_j \\
&= p_{ss^1}\theta_1 + (p_{ss^1}+\epsilon_1)(1+\theta_1+\epsilon_2) + \sum_{s^j \in N(s)\setminus\{s^1,s^i\}} p_{ss^j}\theta_j \\
&> (p_{ss^1}+\epsilon_1)(1+\theta_1) + p_{ss^1}(\theta_1+\epsilon_2) + \sum_{s^j \in N(s)\setminus\{s^1,s^i\}} p_{ss^j}\theta_j \\
&= p_{ss^i}(1+\theta_1) + p_{ss^1}\theta_i + \sum_{s^j \in N(s)\setminus\{s^1,s^i\}} p_{ss^j}\theta_j \qquad (8)
\end{aligned}$$

The value in (8) can be obtained by a strategy P' that switches the values of p_{ss^1} and p_{ss^i} while keeping everything else in P unchanged. This contradicts the optimality of P.

According to Lemma 1, the Min-Max problem in the τ-th iteration can be simplified as follows:

$$\begin{aligned} C^\tau(s) &= \min_P \left[p_{ss^1} + \sum_{s^j \in N(s)} \theta_j p_{ss^j} \right] \\ &= \min_P \left[(1+\theta_1)\, p_{ss^1} + \sum_{s^j \in N(s)\setminus\{s^1\}} \theta_j p_{ss^j} \right] \end{aligned} \tag{9}$$

The following lemma gives a further relation among the elements in the optimal solution to the Min-Max problem.

Lemma 2. *Let P be the optimal solution to the Min-Max problem in the τ-th iteration of Algorithm 1. If $i < j$, then $p_{ss^i} \geq p_{ss^j}$ $\forall s^i, s^j \in N(s)$.*

Proof. Assume $p_{ss^i} < p_{ss^j}$ for some $i < j$. Then we have $p_{ss^j} = p_{ss^i} + \epsilon$ for some $\epsilon > 0$. It follows that

$$\begin{aligned} C^\tau(s) &= \max \mathbf{p}_s + \sum_{s^k \in N(s)} \theta_k p_{ss^k} \\ &= \max \mathbf{p}_s + \theta_i p_{ss^i} + \theta_j (p_{ss^i} + \epsilon) + \sum_{s^k \in N(s)\setminus\{s^i, s^j\}} \theta_k p_{ss^k} \\ &> \max \mathbf{p}_s + \theta_i (p_{ss^i} + \epsilon) + \theta_j p_{ss^i} + \sum_{s^k \in N(s)\setminus\{s^i, s^j\}} \theta_k p_{ss^k} \end{aligned} \tag{10}$$

The value in (10) can be obtained by a strategy P' that switches p_{ss^i} and p_{ss^j} while keeping everything else in P unchanged. This contradicts the optimality of P.

From Lemma 1 and Lemma 2, we can obtain a complete characterization of the optimal solution to the Min-Max problem, as stated in the following proposition.

Proposition 1. *Let P be the optimal solution to the Min-Max problem in the τ-th iteration of Algorithm 1. Let $k < |N(s)|$ be the smallest positive integer such that $\theta_{k+1} > \frac{1+\sum_{i=1}^{k+1} \theta_i}{k+1}$, then we have $p_{ss^i} = \frac{1}{k}$, $\forall i \leq k$ and $p_{ss^i} = 0, \forall i > k$. If no such k exists, $p_{ss^i} = \frac{1}{|N(s)|}$, $\forall i \in N(s)$.*

Proof. First note that since $\theta_1 < 1 + \theta_1$, we must have $k \geq 1$ (if it exists). We first show that $p_{ss^i} = 0$ $\forall i > k$. Assume $p_{ss^j} = \epsilon > 0$ for some $j > k$. From

Lemma 1, we have

$$\begin{aligned} C^{\tau}(s) &= p_{ss^1} + \sum_{s^j \in N(s)} \theta_j p_{ss^j} \\ &\geq p_{ss^1} + \sum_{i=1}^{k} \theta_i p_{ss^i} + \theta_j \epsilon \\ &> p_{ss^1} + \sum_{i=1}^{k} \theta_i p_{ss^i} + \frac{1 + \sum_{i=1}^{k} \theta_i}{k+1} \epsilon \\ &= (p_{ss^1} + \frac{\epsilon}{k+1}) + \sum_{i=1}^{k} \theta_i (p_{ss^i} + \frac{\epsilon}{k+1}) \end{aligned} \tag{11}$$

Consider another strategy P' where $p'_{ss^i} = p_{ss^1} + \frac{\epsilon}{k+1}$ for all $i \leq k$ and $p'_{ss^i} = 0$ for all $i > k$. According to (11), a smaller cost $(p_{ss^1} + \frac{\epsilon}{k+1}) + \sum_{i=1}^{k} \theta_i (p_{ss^i} + \frac{\epsilon}{k+1})$ can be obtained by adopting P'. This contradicts the optimality of $C^{\tau}(s)$.

We then show that $p_{ss^i} = \frac{1}{k}$ for all $i \leq k$. To this end, we first prove the following claim: $\theta_i \leq 1 + \theta_1$ for all $i \leq k$. We prove the claim by inducion. For $i = 1$, it is clear that $\theta_1 \leq 1 + \theta_1$. Assume the claim is true for all $i \leq m - 1 < k$. We need to show that $\theta_m \leq 1 + \theta_1$. Since $\theta_m \leq \frac{1 + \sum_{i=1}^{m} \theta_i}{m}$, we have $(m-1)\theta_m \leq 1 + \theta_1 + \sum_{i=2}^{m-1} \theta_i \leq 1 + \theta_1 + (m-2)(1 + \theta_1) = (m-1)(1 + \theta_1)$, which implies $\theta_m \leq 1 + \theta_1$.

To show that $p_{ss^i} = \frac{1}{k}$ for all $i \leq k$, it suffices to show that $p_{ss^1} = \frac{1}{k}$. Assume $C_P(s)$ obtains the minimum value at P^* where $p_{ss^1} > \frac{1}{k}$. Without loss of generality, assume $p_{ss^1} > p_{ss^2}$. Then there exists an $\epsilon > 0$ such that $p_{ss^1} - \epsilon \geq \frac{1}{k}$ and $p_{ss^1} - \epsilon \geq p_{ss^2} + \epsilon$. Consider another strategy P'' where $p''_{ss^1} = p_{ss^1} - \epsilon$, $p''_{ss^2} = p_{ss^2} + \epsilon$, $p''_{ss^i} = p_{ss^i}$ for $i \geq 3$. We have

$$\begin{aligned} C_{P''}(s) &= p_{ss^1} - \epsilon + \theta_1 (p_{ss^1} - \epsilon) + \theta_2 (p_{ss^2} + \epsilon) + \sum_{i=3}^{k} \theta_i p_{ss^i} \\ &= C_{P^*}(s) - (1 + \theta_1 - \theta_2)\epsilon \\ &< C_{P^*}(s) \end{aligned} \tag{12}$$

where the last inequality follows from the claim above. This contradicts the optimality of P. Therefore, $p_{ss^1} = \frac{1}{k}$, which implies that $p_{ss^i} = \frac{1}{k}$ for all $i \leq k$.

If $\theta_k \leq \frac{1 + \sum_{i=1}^{k} \theta_i}{k}$ for all $k \leq |N(s)|$, we can use a similar argument as above to show that $C_P(s) \geq p_{ss^1} + \theta_1$, where the equality can be achieved by setting $p_{ss^1} = \frac{1}{|N(s)|}$, which implies that $p_{ss^i} = \frac{1}{k}$ for all i.

Proposition 1 has several important implications. First, each row of the optimal P has at most two different values 0 and $\frac{1}{k}$, where k is bounded by the degree of the corresponding node. This implies that the defender may move the resource to several states with the same switching probability even if their switching costs are different. Second, depending on the structure of the state

graph, the defender may prefer switching to a state with larger cost or never switch the resource from one state to another even if there is a link between them. Third, for any state s, the value of k in the $(\tau+1)-$th iteration only depends on the s-th row of $\mathcal{C}$ and $\{C^{\tau}(s)|s \in V\}$ from the τ-th iteration. Thus, the minimization problem in (9) can be easily solved. Forth, according to the proof of Proposition 1, if $\theta_k \leq 1+\theta_1$ for $\forall k \in [1, |N(s)|]$, then $p_{ss^1} = \frac{1}{|N(s)|}$. Otherwise, $p_{ss^1} = \frac{1}{k}$.

According to the above observations, we can derive an efficient solution to the step 4 in Algorithm 1, as shown in Algorithm 2.

Algorithm 2 Solving the Min-Max problem in the τ-th iteration of Algorithm 1

Input: V, $\mathcal{C}$, $C^{\tau-1}(\cdot)$, α.
Output: P^*.
1: **for** $s \in V$ **do**
2: $\{s^1, s^2, ..., s^{|N|}\} \leftarrow$ a nondecreasing ordering of $s' \in N(S)$ in terms of $c_{ss'} + \alpha C^{\tau-1}(s')$;
3: $\theta_i \leftarrow c_{ss^i} + \alpha C^{\tau-1}(s^i), \forall s^i \in N(s)$;
4: $k \leftarrow 1$;
5: **while** $\theta_{k+1} \leq \frac{1+\sum_{i=1}^{k+1}\theta_i}{k+1}$ and $k < |N(s)|$ **do**
6: $k \leftarrow k+1$
7: **end while**
8: $p^*_{ss^i} = \frac{1}{k}$, for all $i \leq k$, $p^*_{ss^i} = 0$, for all $i \geq k+1$;
9: **end for**

The running time of Algorithm 2 is dominated by sorting the neighbors of a node according to their θ values. Thus, the complexity of the algorithm is bounded by $O(|V|^2 \log |V|)$. This is much faster than the searching approach with complexity of $O(M^{|V|})$.

4.2 Solving the MDP in Regular Graphs

In this section, we consider a special case of the MTD game where each state has $K+1$ neighbors (including itself) and the switching costs between two distinct switchable states have the same value $c > 0$ as the beginning step. In this case, the state switching graph becomes a regular graph (with self loops on all the nodes). Intuitively, the regular graphs are hard to attack since all the vertices (states) look the same. It will be beneficial for the defender to construct regular or approximately regular graphs to protect the resource if this hypothesis is true. We will show that explicit formulas can be obtained for the MDP under this scenario.

Due to the symmetric nature of the regular graph, it is easy to see that the defender has the same optimal cost at every state. Let $C^{(K)}$ denote the optimal

cost when each state has $K+1$ neighbors. We have

$$\begin{aligned} C^{(K)} &= \max \mathbf{p}_s + \sum_{s' \in N(s)} p_{ss'}(c_{ss'} + \alpha C^{(K)}) \\ &\overset{(a)}{=} p_{ss}(1 + \alpha C^{(K)}) + \sum_{s' \in N(s) \setminus s} p_{ss'}(c + \alpha C^{(K)}) \end{aligned} \tag{13}$$

where (a) is due to the fact that $c_{ss} + \alpha C^{(K)} = \alpha C^{(K)} < c_{ss'} + \alpha C^{(K)}$ for any $s' \neq s$, which implies that p_{ss} is the maximum element in $\mathbf{p}_s$ according to Lemma 1. If $c > 1$, then $\theta_2 = c + \alpha C^{(K)} > \frac{1+\alpha C^{(K)} + c + \alpha C^{(K)}}{2} = \frac{1+\theta_1+\theta_2}{2}$. We have $p_{ss} = 1$ and $p_{ss'} = 0$ for all $s' \neq s$ according to Proposition 1, and $C^{(K)} = \frac{1}{1-\alpha}$. In this case, the defender will keep the resource at the original state all the time. If $c \leq 1$, then $\theta_k \leq \frac{1+\sum_{i=1}^{k} \theta_k}{k}$ for all $k \leq K+1$. We have $p_{ss'} = \frac{1}{K+1}$ for all $s' \in N(s)$ according to Proposition 1. In this case, we can solve the value of $C^{(K)}$ as

$$\begin{aligned} C^{(K)} &= \frac{1}{K+1}(1 + \alpha C^{(K)}) + \frac{K}{1+K}(c + \alpha C^{(K)}) \\ \Rightarrow C^{(K)} &= \frac{1 + Kc}{(1-\alpha)(1+K)} \end{aligned} \tag{14}$$

Putting the two cases together, we have

$$C^{(K)} = \begin{cases} \frac{1}{1-\alpha} & \text{if } c > 1, \\ \frac{1+Kc}{(1-\alpha)(1+K)} & \text{if } c \leq 1. \end{cases}$$

Assume $c \leq 1$ in the rest of this section. It is clearly that $C^{(K)}$ is increasing with c. Taking the partial derivative of $C^{(K)}$ w.r.t. K, we have

$$\frac{\partial C^{(K)}}{\partial K} = -\frac{1-c}{(1-\alpha)(1+K)^2} < 0 \tag{15}$$

Therefore, $C^{(K)}$ is strictly decreasing with K. Further, we find that $C^{(K)}$ is a convex function of K by taking the second partial derivative of $C^{(K)}$ w.r.t. K,

$$\frac{\partial^2 C^{(K)}}{\partial K^2} = \frac{1-c}{(1-\alpha)(1+K)^3} > 0 \tag{16}$$

which implies that for larger K, the marginal decrease of $C^{(K)}$ is smaller. We further notice that $C^{(K)}$ is independent of the number of valid states $|V|$ and total links in the graph. Hence, adding more states and switching pairs is not always helpful. For example, in a 8-node regular graph with $K = 2$, the defender has an optimal cost of $\frac{1+2c}{3(1-\alpha)}$. However, given the same switching cost and discount factor, the defender has a smaller cost of $\frac{1+3c}{4(1-\alpha)}$ in a 4-node regular graph with $K = 3$.

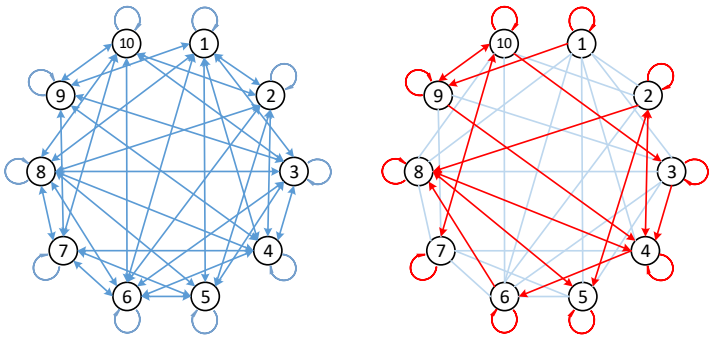
5 Numerical Results

In this section, we examine our proposed model with numerical study under different system scenarios and configurations.

5.1 Warm-up Example

We first use a simple example to illustrate the defender's optimal strategy P^* and optimal cost C^*. We consider a resource with $n = |V|$ valid states and model the valid state switches as an Erdős - Rényi $G(n, p)$ random graph [15], where every possible link between two distinct states occurs independently with a probability $p \in (0, 1)$.

Fig. 3a shows a small state switching graph sampled from $G(10, 0.6)$ (we also add self links to all the nodes). The switching costs between any two distinct connected states follow the uniform distribution $U(0, 2)$ as shown in Fig. 4, and the discount factor is set to 0.5. Fig. 5 gives the defender's optimal strategy P^* and optimal cost $C^*(s)$. The s-th row of C^* represents the optimal cost with an initial state s. Fig. 3b highlights the optimal strategy P^*, where from a current state s, the resource may switch to any of the neighboring states connected by red links with an equal probability. From the optimal P^* given in Fig. 5, we can make some interesting observations. First, the defender abandons some switching pairs and only switches the resource to the rest of states with equal probability. Second, the defender may prefer switching to a state with larger switching cost. For example, when the resource is currently at state 5, the probability of switching to state 2 is higher than the probability of switching state 7, even though $c_{52} > c_{57}$ ($c_{52} = 0.39, c_{57} = 0.30$). Third, a state s with more neighbors does not necessarily has smaller $C^*(s)$. For instance, state 2 has 7 neighbors and state 6 has 9 neighbors, but $C^*(2) = 0.8639 < C^*(6) = 1.0798$.



(a) State switching graph (b) A graphical view of P^*

Fig. 3: An example of the MTD game where the state switching graph is sampled from the Erdős - Rényi random graph $G(10, 0.6)$.

$$\mathcal{C}=\begin{bmatrix}
0.00 & 1.48 & 0.91 & 0.95 & 1.64 & 1.12 & \infty & 1.82 & 0.23 & \infty \\
0.60 & 0.00 & \infty & 0.07 & 0.28 & 0.60 & \infty & 0.16 & \infty & 1.43 \\
0.50 & \infty & 0.00 & 0.08 & 1.28 & 0.90 & \infty & 1.52 & 0.64 & 1.05 \\
1.91 & 0.61 & 0.69 & 0.00 & \infty & 0.42 & 1.58 & 0.56 & 1.59 & \infty \\
1.37 & 0.39 & 1.63 & \infty & 0.00 & 0.58 & 0.30 & 0.07 & \infty & \infty \\
1.75 & 1.56 & 0.55 & 1.34 & 1.29 & 0.00 & 1.22 & 0.15 & \infty & 0.98 \\
\infty & \infty & \infty & 1.94 & 1.47 & 1.23 & 0.00 & 1.75 & 1.82 & 0.44 \\
1.39 & 1.69 & 1.24 & 0.01 & 0.28 & 1.48 & 0.99 & 0.00 & \infty & 0.33 \\
1.31 & \infty & 1.67 & 0.58 & \infty & \infty & 1.93 & \infty & 0.00 & 0.17 \\
\infty & 1.87 & 1.06 & \infty & \infty & 1.30 & 0.88 & 1.95 & 1.02 & 0.00
\end{bmatrix}$$

Fig. 4: Switching cost matrix

$$\mathrm{P}^*=\begin{bmatrix}
0.50 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.50 & 0.00 \\
0.00 & 0.25 & 0.00 & 0.25 & 0.25 & 0.00 & 0.00 & 0.25 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.50 & 0.50 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.25 & 0.00 & 0.25 & 0.00 & 0.25 & 0.00 & 0.25 & 0.00 & 0.00 \\
0.00 & 0.33 & 0.00 & 0.00 & 0.33 & 0.00 & 0.00 & 0.33 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.50 & 0.00 & 0.50 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.50 & 0.00 & 0.00 & 0.50 \\
0.00 & 0.00 & 0.00 & 0.33 & 0.33 & 0.00 & 0.00 & 0.33 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.33 & 0.00 & 0.00 & 0.00 & 0.00 & 0.33 & 0.33 \\
0.00 & 0.00 & 0.25 & 0.00 & 0.00 & 0.00 & 0.25 & 0.00 & 0.25 & 0.25
\end{bmatrix}
\quad
\mathrm{C}^*=\begin{bmatrix}
1.2401 \\ 0.8639 \\ 1.1009 \\ 1.1511 \\ 0.9463 \\ 1.0798 \\ 1.5231 \\ 0.9358 \\ 1.2668 \\ 1.6877
\end{bmatrix}$$

Fig. 5: The defender's optimal strategy P^* and the corresponding optimal cost $C^*(s)$

5.2 Evaluation of the Optimal MTD Strategy

We then conduct large scale simulations to evaluate our MTD strategies and investigate how the structure of the state switching graph affect the defender's cost.

We first compare our strategy with two baseline strategies: (1) A simple uniform random strategy (URS) where the defender switches the resource to each neighbor of the current state with the same probability. This is the simplest MTD strategy one can come up with. (2) A simple improvement of the uniform random strategy (IRS) where the transition probabilities are inversely proportional to the switching costs. More concretely, we set $p_{ss} = \frac{1}{|N(s)|}$ and ensure that $p_{ss'}c_{ss'}$ is a constant for all $s' \in N(s)\backslash s$. The objective is to compare the average cost over all the states achieved by our algorithm and the two baselines.

The state switching graph is sampled from $G(50, 0.1)$. 100 samples are generated. We set the discount factor $\alpha = 0.5$. The switching costs between two distinct connected nodes follow an uniform distribution $U[0, 2a]$ where a varies between 0.2 and 1.

Fig. 6 shows the mean average cost over all the random graphs generated. As we expected, the optimal strategy (OS) has significant better performance than the two baselines, especially when the mean switching cost becomes larger. One thing to highlight is that, although URS is the simplest strategy that one can think of, it may actually perform better than a more complicated strategy such as IRS in certain scenarios. Hence, one has to be careful when adapting a

heuristic based strategy to MTD. This observation also indicates the importance of developing optimal strategies for MTD.

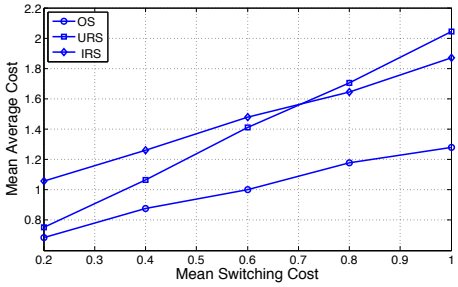


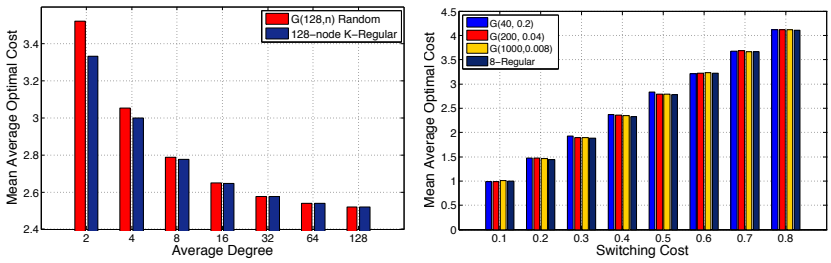
Fig. 6: Mean average cost v.s. mean switching cost

5.3 Impact of Switching Graph Structures

In Section 4.2, we have derived explicit relations between the optimal defense cost and the structure of the switching graph when the graph is regular. It is interesting to know if such relations hold in more general settings. In this section, we conduct simulations to answer this question for random graphs. To have a fair comparison between regular graphs and random graphs, we set the switching costs between distinct connected nodes to a constant c in this section. We consider two scenarios.

We first fix $|V| = 128$ and the switching cost $c = 0.5$, and vary the average degree K of the switching graph, by using different values of p in the $G(128, p)$ model. We compare this case with a regular graph with the same K. Fig. 7a gives the mean average costs for the two models. We observe that when the average degree increases, the defender's optimal cost follows a similar trend in both models. In particular, the cost reduces sharply in the small degree regime, which is consistent with our analysis in Section 4.2. In addition, the defender's performance in regular graphs is always better than that in random graphs, especially when the average degree is small. This can be explained by the convexity of $C^{(K)}$ over K shown in Section 4.2. More specifically, the degree distribution of a random graph is more diverse than that of a regular graph with the same average degree. Due to the convexity of $C^{(K)}$, we have $C^{(K+\epsilon)} + C^{(K-\epsilon)} > 2C^{(K)}$ (ϵ is a small positive integer), which implies that a graph where the degree distribution is more concentrated has better performance. In addition, the gap between $C^{(K+\epsilon)} + C^{(K-\epsilon)}$ and $2C^{(K)}$ is bigger for smaller K. Hence, regular graphs perform much better than random graphs when the average degree is small.

We then fixe the average degree $K = 8$ and vary $|V|$ and the switching cost c. From Fig. 7b, we observe that the defender's optimal costs in different $|V|$ are almost the same when both the average degree and the switching cost are fixed. Moreover, by increasing the switching cost, the defender's optimal cost in the random graph model increases linearly. Both observations are consistent with our analysis for the regular model in Section 4.2.



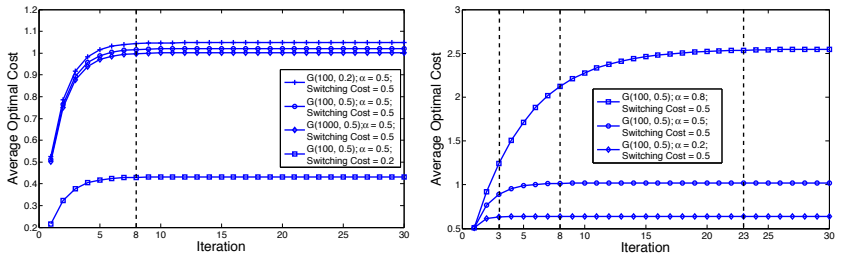
(a) Same switching cost, varying average degree

(b) Same average degree, varying switching cost

Fig. 7: Mean average optimal cost under different settings

5.4 Rate of Convergence

Previous studies have analyzed the convergence rate of discounted MDP [16]. We will examine the convergence speed of proposed Algorithm 1 using simulations with a similar setup as in Section 5.3. In Fig. 8a, we vary both $|V|$ and the mean switching cost c, while fixing the discount factor $\alpha = 0.5$. We observe that each curve converges to a relative stable value after 8 iterations. We then fix $|V|$, p, and mean switching cost c, while varying the discount factor α. From Fig. 8b, we observe that the convergence speed gets slower with larger α, which is expected. We draw the conclusion that the main factor that affects the convergence rate of Algorithm 1 is the discount factor.



(a) (b)

Fig. 8: Rate of Convergence with different parameters

5.5 Suggestions to the Defender

Based on the results and observations above, we make the following suggestions to the defender for holding a more secured resource:

– Due to the fact that the defender's cost is largely determined by the average degree of the switching graph, adding more switching pairs can help reduce the cost. In particular, for a given number of states, the average degree can

be maximized adopting a complete graph where the resource can switch between any two states.
- Since the defender's cost is approximately convex with the average degree and linear with the switching cost, the defender should pay more attention to increasing the number of states rather than reducing the switching cost if the average degree is small. While if the average degree is already large enough, reducing switching cost is more useful.
- Introducing a large number of states is not always helpful. The main reason is that the attacker could obtain full feedback about the previous configuration used by the defender in our model. Under this assumption, adding more states does not necessarily means that the defender has more choice to switch. Instead of increasing the number of states, adding more switching pairs is more beneficial to the defender.

6 Conclusion

In this paper, we propose a Stackelberg game model for Moving Target Defense (MTD) between a defender and an attacker. After fully characterizing the player's strategies, payoffs and feedback structures, we model the defender's problem on optimizing the switching strategy as a Markov Decision Process (MDP) and further derive an efficient value iteration algorithm to solve the MDP. By employing a directed graph to illustrate the pattern of switching states, we obtain the relation between defender's performance and the properties of the graph in an explicit way when the graph is regular. Similar results are further verified on random graphs empirically. Through theoretical analysis and numerical study of the proposed model, we have derived several insights and made suggestions to the defender towards more efficient MTD.

7 Acknowledgement

The effort described in this article was partially sponsored by the U.S. Army Research Laboratory Cyber Security Collaborative Research Alliance under Contract Number W911NF-13-2-0045. The views and conclusions contained in this document are those of the authors, and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes, notwithstanding any copyright notation hereon. This research was also supported in part by a grant from the Board of Regents of the State of Louisiana LEQSF(2017-19)-RD-A-15.

References

1. C. Tankard, "Advanced persistent threats and how to monitor and deter them," *Network security*, vol. 2011, no. 8, pp. 16–19, 2011.

2. S. Jajodia, A. K. Ghosh, V. Swarup, C. Wang, and X. S. Wang, *Moving target defense: creating asymmetric uncertainty for cyber threats*. Springer Science & Business Media, 2011, vol. 54.
3. 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*, 2012, pp. 127–132.
4. B. Salamat, T. Jackson, G. Wagner, C. Wimmer, and M. Franz, "Runtime defense against code injection attacks using replicated execution," *IEEE Transactions on Dependable and Secure Computing*, vol. 8, no. 4, pp. 588–601, 2011.
5. A. Nguyen-Tuong, D. Evans, J. C. Knight, B. Cox, and J. W. Davidson, "Security through redundant data diversity," in *IEEE International Conference on Dependable Systems and Networks*, 2008, pp. 187–196.
6. Q. Zhu and T. Başar, "Game-theoretic approach to feedback-driven multi-stage moving target defense," in *International Conference on Decision and Game Theory for Security (GameSec)*, 2013, pp. 246–263.
7. 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*, 2014, pp. 21–30.
8. S. Sengupta, S. G. Vadlamudi, S. Kambhampati, A. Doupé, Z. Zhao, M. Taguinod, and G.-J. Ahn, "A game theoretic approach to strategy generation for moving target defense in web applications," in *International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, 2017, pp. 178–186.
9. A. Nochenson and C. L. Heimann, "Simulation and game-theoretic analysis of an attacker-defender game," in *International Conference on Decision and Game Theory for Security (GameSec)*, 2012, pp. 138–151.
10. V. Lisỳ, T. Davis, and M. H. Bowling, "Counterfactual regret minimization in sequential security games," in *Association for the Advancement of Artificial Intelligence (AAAI)*, 2016, pp. 544–550.
11. Z. Yin, D. Korzhyk, C. Kiekintveld, V. Conitzer, and M. Tambe, "Stackelberg vs. nash in security games: Interchangeability, equivalence, and uniqueness," in *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2010, pp. 1139–1146.
12. X. Feng, Z. Zheng, P. Mohapatra, D. Cansever, and A. Swami, "A signaling game model for moving target defense," in *IEEE Conference on Computer Communications (INFOCOM)*, 2017.
13. 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*, 2014, pp. 73–76.
14. H. Maleki, S. Valizadeh, W. Koch, A. Bestavros, and M. van Dijk, "Markov modeling of moving target defense games," in *ACM Workshop on Moving Target Defense*, 2016, pp. 81–92.
15. P. Erdős and A. Rényi, "On the evolution of random graphs," *Publ. Math. Inst. Hung. Acad. Sci*, vol. 5, no. 1, pp. 17–60, 1960.
16. M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.