Using Artificial intelligence to automate the deployment of MTD operation

Project Proposal

Wai Him Ho

Word Count: 4996

School of Engineering
University of Western Australia
20th May, 2024

Contents

1 Project Summary

In the field of cybersecurity, Moving Target Defense (MTD) is recognized for its adaptive nature in addressing evolving cyber threats. Despite existing research on MTD, a gap persists in effectively integrating hybrid MTD strategies with comprehensive defense mechanisms against diverse cyber attacker profiles. This project aims to provide a detailed literature review of hybrid MTD, automation of MTD, and related security metrics that evaluate the effectiveness of MTD. This project will also develop a reinforcement learning model that utilizes the deep Q-learning algorithm to enhance the strategic deployment of hybrid MTD techniques across different adversary scenarios. The MTD simulator, MTDSimTime will be used as a platform for the model development and simulation. The methodology process involves data collection, model development, and deployment. The proposed model which uses static and dynamic metrics as features will be evaluated against different other variant AI models. The project timeline spans 32 weeks, with a primary focus on the implementation and comprehensive evaluation of the AI model.

2 Introduction

Traditional static defense mechanisms have become ineffective due to advanced attacks such as Advanced Persistent Threat(APT) [1], which can perform long-term vulnerability analysis and penetration testing on the target, making firewalls and signature-based detection obsolete. Moving Target Defence offers a solution by dynamically modifying the attack surface [2] of the system, increasing uncertainty and raising the cost of attacks while preserving its fundamental functionality.

Many existing work on MTD focuses on developing a single timeliness-based MTD technique against a specific attacker type. Research on the optimal deployment of hybrid MTD techniques is limited, and the effectiveness of a single MTD is often insufficient [3]. Although hybrid MTD approaches have been proposed [4, 5, 6, 7], they often overlook the consideration of multiple attacker types and rarely account for the availability and performance of the system [8, 9, 10] after deploying the MTD techniques. Due to the limitations of manually selecting MTD configurations based on experience [11], other research has been done in the area of automating MTD techniques, ranging from MTDeep [12] to reinforcement learning based MTD deployment [13]. However, most of these studies focus on automating the deployment of a single MTD technique against a specific attacker type. There remains a gap in using machine learning to optimally deploy one or more MTD against various attacker types.

3 Literature Review

3.1 Combinations of Moving Target Defense

MTD techniques can be classified into three categories: shuffling, diversity, and redundancy [2]. Each combination of MTD techniques(e.g. Shuffling and Diversity) should be examined and assessed individually due to their emergent properties [2].

Alavizadeh et al. proposed an approach that combines both Shuffling and Redundancy Moving Target Defense (MTD) techniques and evaluated its effectiveness using a 2-layer Hierarchical Attack Representation Model (HARM) [4, 5]. The paper used HARM and SHARPE to measure two security metrics: system risk and reliability, to evaluate the MTD techniques. Additionally, two Network Centrality Measures (NCMs), betweenness centrality and closeness centrality, were used to rank the most crucial VMs in the cloud to improve security analysis. The results showed that deploying redundancy (R) and the combination of shuffle and redundancy (S + R) increased system risk, with S + R increasing risk significantly more than S alone. However, S + R outperforms S significantly in terms of reliability. While S + R reduced more risk compared to R, R achieved slightly better reliability compared to S + R. In addition to their previous work [4], Alavizadeh et al. introduced the unattackability metric [5]. The paper also concluded that the combination of two MTD techniques, such as S + R, contributed to a higher likelihood of increasing the unattackability of the system.

3.1.1 Shuffling and Diversity

Rajakumaran et al. proposed a shuffling and diversity hybrid MTD technique against DoS mitigation in Amazon web service [6]. The experiment incorporated a 2-layer HARM security model and used the Importance Measure(IM) to assess the vulnerability. Resource constraints, delay in proxy switchover, and reduction in attack probability were used as the other important metrics. The experiment showed that the attack probability is 0.5% when using either shuffle, diversity, or redundancy MTD technique where the proposed hybrid MTD reduced the attack probability down to 0.1%.

Wenxiao Zhang evaluated various combinations of shuffling and diversity(S+D) MTD strategies across different scenarios using the proposed MTDSimTime simulator [14]. Two shuffling MTD techniques (complete topology shuffle and IP shuffle) and two diversity MTD techniques (service diversity and OS diversity) were implemented. These techniques were executed either randomly, alternately, or simultaneously to combat adversaries. The paper evaluated both single and pairwise hybrid MTD techniques and found that network-shuffle-based MTD techniques had a similar impact on the MTTC metric score. There was no significant difference between using these techniques in combination with other MTD techniques or using them individually. However, the S + D MTD technique demonstrated more favorable outcomes, yielding an average improvement of approximately 20% in the MTTC score compared to using shuffling techniques alone.

J.B. Hong et al. evaluated the combined effects of shuffle and diversity MTD techniques against two attack scenarios [7] through a simulation framework MTDSim [15]. An extra layer is added to the 2-layer HARM model to capture services provided by each host to closely model the attack behavior to improve the attack simulation. The paper used the Cyber Kill Chain [16] and MITRE ATT&CK Framework [17] to simulate realistic attack scenarios. The simulation tested against single, double, and triple MTD techniques. The experiment concluded that the best-performing results between them were similar, and using only the best single MTD is the best option in the context of the measured metrics.

3.1.2 Diversity and Redundancy

Very few papers discuss the hybrid uses of diversity and redundancy MTD techniques and the majority of the research has been done in other combinations of MTD techniques. J. H. Cho speculated that combining diversity or redundancy may not require shuffling as frequently as using shuffling alone since the diversity of the system will decrease the vulnerabilities that will be spotted by the attackers [2].

M. C. Lucas-Estañ et al. evaluated the utilization of diversity and redundancy in wireless networks to enhance the reliability and latency for mobile industrial applications [18]. They created a prototype that implements diversity and redundancy for the wireless connections between robots and conducted trials that assess the reliability and latency of wireless connections. However, the study only explored the concepts of diversity and redundancy independently and also did not involve the consideration of moving target defense.

3.1.3 Shuffling, Diversity and Redundancy

Alavizadeh et al. evaluated the effectiveness of Moving Target Defense (MTD) techniques concerning security and economic metrics in a cloud environment [19]. They employed a 2-layer Hybrid Attack-Resistance Model (HARM) to model the cloud, and Importance Metrics (IM) were utilized to measure the closeness and betweenness of virtual machines (VMs). The study employed four security metrics - system risk, attack cost, return on attack, and reliability - to assess the effectiveness of the combined MTD techniques. A combination of shuffling, diversity, and redundancy (S + D + R) was deployed with five VMs on the cloud-band for the experiment. The results showed that the values of attack cost (AC) in diversity (D)-only scenarios were lower than those of AC in S + D + R scenarios. Additionally, the deployment of S + D + R led to better reliability. Conversely, the Return on Attack (RoA) values for S + D + R were also lower than those in D-only scenarios, indicating that the attacker is less likely to attack again. The paper concluded that deploying S + D + R improved all aspects of the security and performance metrics used in the experiment.

3.1.4 Discussion

Previous works generally suggest that there is an advantage to deploying multiple MTDs over a single MTD. However, existing metrics such as attack effort [7] are not sufficient in capturing the effects of hybrid MTD deployments and there is a need for more diverse metrics. Moreover, the capability and intelligence of the attackers are underdeveloped [14, 7]. For example, the attacker only focuses on one target node even if it is too difficult to compromise [7]. There is also no substantial analysis of the effectiveness of hybrid MTD techniques against a variety of adversaries. Besides, there can be considerations of other security models other than HARM [4, 5, 19] to assess MTD deployment. Finally, event-based MTD and hybrid MTD can result in more system degradation [3, 14] due to more abrupt disconnections between a connected client and a server and trade-offs between security and performance are neglected. Therefore, extensive investigation is needed to inquire how different MTD techniques can be combined to maximize benefits while maintaining system performance.

3.2 Automation of Moving Target Defense

Zhu Y et al. conducted a holistic survey on MTD and an overview of MTD automation based on previous works. They categorized MTD automation techniques into affordable, optimized, and self-adaptive [13]. Affordable intelligent MTD focuses on reducing high overhead and minimizing additional overhead when triggering MTD. Optimized intelligent MTD involves defensive strategy solutions tailored to specific types of attacks and generalized types of attacks. Reinforcement learning is considered a suitable choice for optimized intelligent MTD. The paper gave an example of a Deep Deterministic Policy Gradient(DDPG) deep reinforcement algorithm that routes randomly against eavesdropping attacks. Self-adaptive intelligent MTD is categorized into machine learning, machine learning with legacy defense mechanisms such as dynamic honey-pot defense, and machine learning with game theories. The paper concludes that self-adaptive is the most advanced and effective type among all 3 types of intelligent MTD proposed.

J.H Cho et al. introduced an Efficient Moving Target Defense (EVADE) system that alters network topology periodically to thwart attackers [20]. They utilized the VERN algorithm to rank vulnerabilities and implemented a lightweight solution search algorithm (FSS) to expedite training. Their hybrid MTD approach combined greedy MTD with density optimization and Deep Reinforcement Learning-based MTD, triggering periodic shuffling of network topology. Comparisons were made between Deep Q-learning Networks (DQN) and Proximal Policy Optimization (PPO), alongside their hybrid versions (S-G-DQN and S-G-PPO with EVADE), and two baseline methods (GA and Random) across various network types. The study considered epidemic attacks and state manipulation attacks (SMAs). Results indicated the proposed algorithm outperformed alternatives in enhancing performance and reducing security vulnerabilities. EVADE-based models excelled in sparse and dense networks, exhibiting resilience against SMAs. However, considerations for adaptive attacks on deep reinforcement learning agents and comprehensive evaluation of attack coverage are areas for further exploration.

Tao Zhang et al. proposed an Intelligence-Driven Host Address Mutation(ID-HAM) scheme to slow down network reconnaissance using automation. The scheme utilized deep reinforcement learning with the Markov decision process(MDP) to describe time-varying network conditions in HAM [21]. Mininet was used to simulate and perform security analysis to evaluate the effectiveness of ID-HAM based on four metrics - SMT formalization, defense performance, convergence performance, and network performance. The paper examined reconnaissance attacks and considers the scanning process of the adversary. The proposed algorithm decreased a maximum of 25% times scanning hits while only influencing the quality of service(QoS) of communication slightly. The adaptivity of ID-HAM was also better than its two counterparts RHM and FRVM.

Chowdhary et. al. proposed a multi-agent reinforcement learning framework with a zero-sum game dynamic in a Software-defined Network for obtaining the optimal Moving Target Defense strategy [22]. Compared to previous work, the paper used domain-specific reward modeling which uses the CVSS score of vulnerability, the difficulty of compromising a vulnerability, the attacker and defender's effort, and the effect of MTD countermeasures as rewards. The modeling considered the performance impact induced by defensive countermeasures as a part of the reward modeling since deploying MTD actions can have some impact on the network. The experimental results suggested that the defender can mimic the attack's action using a reinforcement learning policy and obtain a higher reward using reinforcement learning. Hence, the proposed reinforcement algorithm outperforms the random MTD

deployment strategy with uniform action probabilities.

Eghtesad et. al. proposed a multi-agent partially-observable Markov decision process for Moving Target Defense [11]. The paper combined the Deep Q-learning algorithm and Double Oracle algorithm to find the optimal MTD by solving the mixed strategy Nash equilibrium(MSNE) of a game and equates finding the MSNE of the game as the method to find adaptive MTD policies. The algorithm is experimented in a two-player general-sum game between the attacker and the defender. The results showed that the double oracle algorithm converges around four pieces of training for each player, thus demonstrating that it can efficiently find the MSNE. The experiment also showed that using heuristics and NoOP as initial policy spaces resulted in very similar payoffs, which indicated that the algorithm can converge to near-optimal solutions regardless of the initial policy space.

Sengupta et al. proposed a Bayesian Strong Stackelberg Q-learning (BSS-Q) approach to advance upon previous works where incomplete information and the complexity of the adversaries were not considered [23]. The paper showed that BSS-Q converges to a game theoretic model called the Bayesian Stackelberg Markov Games (BSMGs) that models uncertainty over attacker types. The strategy of the defender starts with a uniform random combination of heuristic strategies and alternates between different strategies until it converges to Strong Stackelberg Equilibrium (SSE) for the BSMG. Two experimental scenarios were also conducted using MTD for web applications and cloud networks, demonstrating that using BSS-Q performs better than existing baseline solutions. However, the paper only considered single follower types and future research is needed to consider other possible attacker types.

Yao et al. proposed the WoLF-BSS-Q algorithm, which can adjust its strategies according to the game process while maintaining its performance compared to classic reinforcement algorithms [24]. The BSS algorithm is used to select the strategy for MTD, and the WoLF algorithm is used to dynamically adjust those strategies by the game process. The approach experiments in an OpenAI Gym-style multi-agent game simulator. The WoLF-BSS-Q algorithm is the optimum and converges to the maximal rewards compared to Nash-Q and URS-Q. The algorithm outperformed due to its ability to select the action with the greatest reward by adjusting the action selection probability. However, the attack agent was straightforward and could not identify the dynamic defense strategy of the defense agent. Furthermore, it did not consider the incomplete information aspect of the multi-agent Markov game, which can cause deviation in the results.

3.2.1 Discussion

Research towards using automation to deploy MTD against multiple adversaries remains deficient, mostly considering only two attacker types, often in the reconnaissance stage [20]. Furthermore, the complexity of the adversary is often lacking and not adaptive enough. For instance, the model described in literature [24] does not consider cheating as a method to consume defense resources and cannot identify the dynamic defense strategy of the proposed defense agent. Another study [20] acknowledges the necessity for ongoing research on characterizing different attacker types. Although research has been conducted on advanced adversaries [25], they have not been used to assess the effectiveness of intelligent MTD deployment. This literature [25] categorizes intelligent attacks targeting MTDs in an SDN environment into two main types: target attacks aiming to determine MTD properties and existing attacks improved through intelligent attack planning. Even though there have been considerations for using reinforcement learning to deploy multiple MTDs or heuristics in previous works [13, 22], there has been limited research done on it.

3.3 Security Metrics

Metrics assessing the effectiveness of Moving Target Defense (MTD) techniques can be categorized as either static or dynamic [26]. Static metrics concentrate on providing an overall assessment of security measures, while dynamic metrics evaluate the adaptability and efficacy of MTD strategies against intelligent adversaries.

1) Static metrics: Alavizadeh et al. have incorporated commonly used static metrics including the attack cost(AC), return on attack(RoA), system risk(R) and reliability into evaluating the security of the cloud environment within the context of the Hierarchical Attack Representation Model (HARM) [19, 27, 28, 29]. Wenxiao Zhang further extended the time domain aspect of MTDSim [15] and updated the implementation of attack cost and introduced the Mean Time to Compromise metric [14].

2) Dynamic metrics: Hong et al. used a Temporal Hierarchical Attack Representation Model(T-HARM) to evaluate the effectiveness of security metrics for dynamic networks [30]. The assessment included path metrics such as mean of attack path lengths(MAPL) and number of attack paths(NAP) and demonstrated that different security metrics can respond to changes in the network configuration by changing their values. Sharma et al. proposed a set of dynamic security metrics that examine the effectiveness of moving target defense techniques in software-defined network(SDN) domain [31]. Three security metrics were proposed: network and host address-based metrics, attack path-based metrics, and attack stage-based success metrics.

3.3.1 Discussion

Dynamic metrics are particularly valuable given the adaptive and dynamic aspects involved in this project. Path-based metrics, such as attack path variability (APV) [30], and attack stage-based success metrics, including exploit success probability (ESP) [30] and attack impact (AI) [30], have not received much research attention in terms of their incorporation into reinforcement learning models for deploying MTD. This project aims to leverage both static and dynamic metrics as features for the proposed model.

4 Project Objectives

This project will investigate the optimal automated deployment of combinations of MTD techniques against different types of adversaries using a reinforcement learning model. The goal of the model is to minimize network vulnerabilities while simultaneously maximizing performance. To achieve this, the MTD simulator MTDSimTime [14], an extension of MTDSim [15], will serve as a simulation and testing environment for building the machine learning model. This research aims to enhance the robustness of existing MTD studies. Previous research often overlooked thorough investigation of adversaries, resulting in a lack of validity in real-life cyber-attack scenarios. Automating the deployment of multiple MTDs is essential due to the increased complexity when considering both security and performance metrics across various adversaries.

Therefore, this project seeks to bridge different aspects of MTD research to advance the state of the art. Key contributions include:

- Review previous works on the combinations of MTD techniques, the automation of MTD, and the related security metrics and incorporate them into this research
- Propose a reinforcement learning model that uses Deep Q-network learning as the baseline model to deploy hybrid MTD techniques against various adversaries.
- Examine the effectiveness of the proposed model against other variant AI models and different metrics optimization

5 Methodology

There are three major pillars in developing this model: data collection, model development, and model deployment. The processes are further divided into two phases for convenience. Section 5.1 will illustrate the robust system required to develop the proposed model. Subsequently, Section 5.2 will give a high-level view of the reinforcement model that will be used in this project. Section 5.3 will describe the modeling process. Finally, section 5.4 will give an overview of the evaluation method of the model.

5.1 System Overview

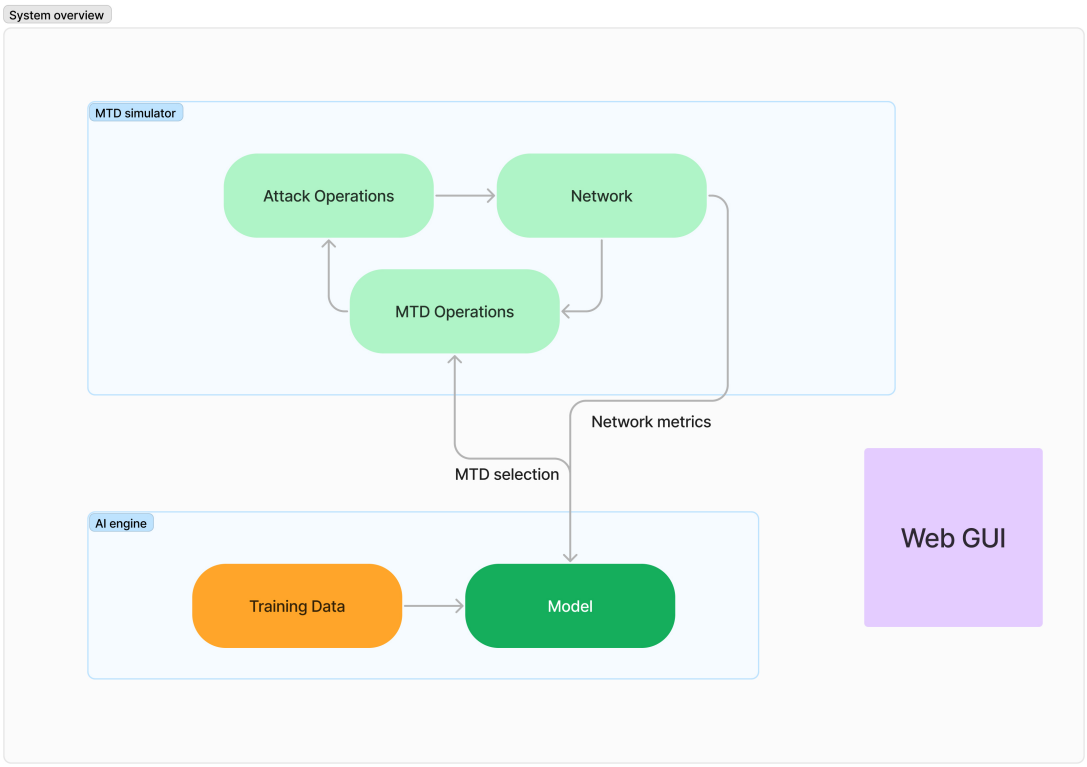

Figure 1: System Overview

As shown in Figure 1, the system consists of the AI engine, the MTD simulator, and the web GUI interface. The AI engine is the proposed model of the project. In this project, the MTD simulator MTDSimTime [14] is used as the infrastructure to create the MTD AI model. It is used as a platform for the MTD AI model to interact with the adversary. It is also used for collecting results for the evaluation of the model. The web GUI is the interface where the simulation of the MTD AI model will be displayed.

5.2 Model

Reinforcement learning uses the Markov Decision Process(MDP). Table 1 documents the key properties involved in the Markov Decision Process.

5.2.1 Deep Q-network learning

The baseline reinforcement model that this project will be using is the deep-Q-learning model. Compared to other conventional reinforcement models such as the Stackelberg Q-learning model, the deep Q-learning model is more flexible and easier to adapt to different frameworks [32]. In the original

Table 1: Markov Decision Process components

Symbol	Description
$\mathbb{S}$	Set of environment states.
$\mathbb{A}$	Set of actions available to the agent.
$\boldsymbol{E}_\pi$	Expectation operator with respect to the policy π.
γ	Discount factor.
τ	Time step relative to the current time t.
$r(t)$	Immediate reward obtained at time t.
$P(s' \mid s, a)$	Transition probability from state s to state s' under action a.
R	Accumulated reward
$\pi(a \mid s)$	Policy representing the agent's decision-making strategy, specifying the probability of taking action a in state s.
π^*	Maximum expected reward from all states
T	Total length of the experiment process

model Q-learning, the aim is to learn the state-action-value(or Q) function, which measures the overall expected reward for taking action a(t) at state s(t) with $r(t) \in \mathbb{R}$ being the intermediate reward as listed in Equation (1). However, when states and actions are broken down into smaller, specific parts, they can lead to a large number of combinations. This can create a problem called the curse of dimensionality [33].

$$Q(s(t), a(t)) = \boldsymbol{E}_{\boldsymbol{\pi}} \left(\sum_{r=0}^{\infty} \gamma^{\tau} r(t + \tau + 1) \mid s(t), a(t) \right) \tag{1}$$

Deep Q-network improves Q-learning by approximating the Q-function with a neural network. The deep-Q-network learns by minimizing the loss function presented in Equation(2) and seeks to find an optimal policy π^* that produces the maximum expected reward from all states where $\pi^* = arg\max \mathbb{E}[R|\pi]$ [34].

$$\mathbb{L}(t) = \frac{1}{2} (\underbrace{r(t) + \gamma \max_{a \in \mathbb{A}} Q(s'(t), a, \theta(t))}_{\text{target}} - \underbrace{Q(s(t), a(t), \theta(t))}_{\text{actual}})^2 \tag{2}$$

5.2.2 Features

This project will use the security and performance metrics mentioned in the literature review section as features to train the proposed model. Following are some of the major metrics that will be used:

Table 2: Metrics and Symbols

Symbol	Description
h_i	The i^{th} host
sys_{h_i}	The i^{th} host of the system
AC_{sys}	Total attack cost of the system.
ap	Single attack path in the system.
AP	List of all possible attack paths in the network.
$R(h_i)_{t_k}$	Risk of compromising host h_i at time t_k
$CSP(h_i)_{t_k}$	Compromise success probability of host h_i at time t_k.
$AI(h_i)_{t_k}$	Attack impact of host h_i at time t_k.
RoA_{h_i}	Return on Attack Cost for host h_i
SAP_{t_k}	Set of shortest attack paths at time t_k.
$\|ap\|$	Length of the attack path ap.
$SAPV_{t_k}$	Shortest Attack Path Variability at time t_k, indicating the changes in shortest attack paths over time.

1) Attack Cost (AC): The cost of exploiting the vulnerabilities of a host for an attack is defined as

the attack cost. Equation (1) shows the formula for the overall AC of the system.

$$AC_{\text{sys}} = \sum_{ap \in AP} \left(\sum_{h_i \in ap} AC_{\text{sys}_{h_i}} \right) \tag{3}$$

where ap is a single attack path in the system and AP is the list of all possible attack paths in the network.

2) Risk (R): The risk metric represents the risk of compromising a host in a network at every interval where the model takes input data from the network configuration. The change of risk per interval(before and after moving target defense deployment) is measured as an indicator of the effectiveness of the MTD deployment. A quantitative definition of the risk of compromising a host, h_i at time t_k can be defined as:

$$R(h_i)_{t_k} = CSP(h_i)_{t_k} \times AI(h_i)_{t_k} \tag{4}$$

where $CSP(h_i)_{t_k}$ and $AI(h_i)_{t_k}$ are the compromise success probability and attack impact of a host h_i at time t_k.

3) Return on Attack Cost (RoA): This metric quantifies the ratio of the benefits of an attack to the costs incurred. A higher value of Return on Attack (RoA) indicates a greater likelihood that an attacker will exploit the vulnerabilities present. The Return on Attack Cost (RoAC) for a single host h_i is calculated as the ratio of the risk associated with the host to the attack cost as shown in Equation (3). The overall RoA of a system is given in Equation (4).

$$RoA_{h_i} = \frac{R(h_i)_{t_k}}{AC(h_i)} \tag{5}$$

$$RoA_{\text{sys}} = \sum_{ap \in AP} \left(\sum_{h_i \in ap} RoA_{\text{sys}_{h_i}} \right) \tag{6}$$

4) Shortest Attack Path Variability (SAPV): The shortest path represents the length of the shortest attack paths from an attacker's initial state to the goal state. The SAPV metric estimates the changes in shortest attack paths over time. A set of the shortest attack paths at t_k estimated by:

$$SAP_{t_k} = \{ap_{t_k} \mid |ap_{t_k}| \leq |ap| \forall ap \in AP_{t_k}\} \tag{7}$$

where ap is the attack path and $\mid ap \mid$ is the length of the attack path.

The SAPV at t_k is estimated by:

$$SAPV_{t_k} = \frac{|SAP_{t_k} - SAP_{t_{k-1}}|}{|SAP_{t_k}|} \tag{8}$$

The overall SAPV for $[t_1, t_m]$ is calculated as:

$$SAPV = \frac{1}{m-1} \sum_{k=2}^{m} SAPV_{t_k} \tag{9}$$

The Common Vulnerability Scoring System (CVSS) [35] will be utilized as the initial weights for the features since it can provide a standardized measure of vulnerability severity.

5.2.3 Environment and Players

There are two players, the defender and the adversary. As shown in Figure 1, the interaction between them will occur in the MTDSimTime simulator [14] and multiple attack scenarios will be considered instead of just the current existing attack type in the simulator. The simulator uses a 3-layer HARM model which consists of the Host Attack Graph(AG), the Services on the Host Attack Graph(AG), and the Service Attack Tree(AT). Currently, the simulator can only deploy four MTD techniques either randomly, alternatively, or simultaneously against limited attacker types. In this project, this simulator will be extended to include more purposeful deployment and also consider a wide range of MTD techniques. More adversaries will also be examined in this project and they will be built using the Cyber Kill Chain [16] and MITRE ATT&CK [17] framework to try to compromise as many hosts as possible.

5.2.4 States

A state s_t at step t is defined by $s_t = (f_0^t, f_1^t, \ldots, f_N^t)$ where f_i^t is the i^{th} selected feature from the chosen features at step t.

5.2.5 Actions

The action a^t is represented by $a^t = \{MTD_1, MTD_2, \ldots, MTD_{12}, MTD_{13}, \ldots, MTD_{\sum_{i=1}^{n}}, \text{Null}\}$ where MTD_i is the deployment of the i^{th} MTD technique in the MTD set. MTD_ij represents deploying both the i^{th} and j^{th} MTD technique such as Figure(2) [13]. Null means no actions will be taken.

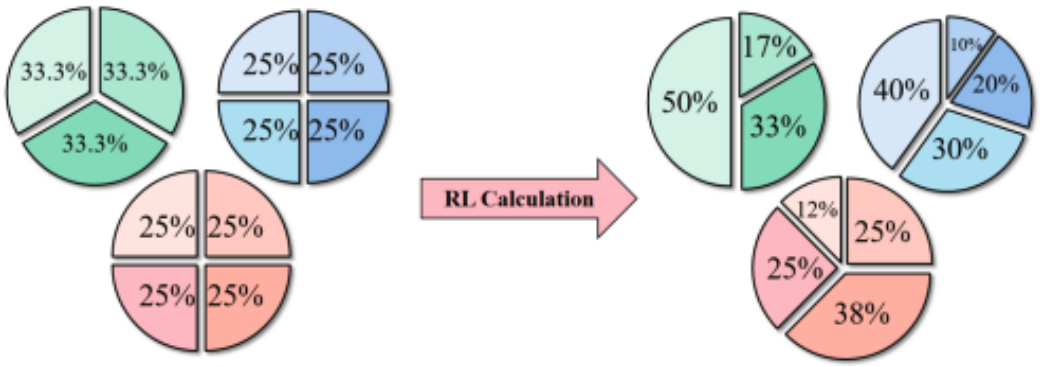

Figure 2: Diagram of optimization for multi-MTD methods

5.2.6 Rewards

The rewards for the model should be designed in a way that trains the model to maximize performance while minimizing system vulnerabilities. The reward at step t is given by $R(s_t, a_t, s_{t+1})$, where $R(s_t, a_t, s_{t+1}) = f(N_{t+1}) - f(N_t)$ where $f(x)$ is an evaluation function. The accumulative reward is categorized as $R = \sum_{t=0}^{T-1} \gamma^t r_{t+1}$.

5.2.7 Limitations

The project leverages MTDSimTime [14] for constructing the reinforcement model. Nonetheless, a disparity exists between the simulation in MTDSimTime [14] and real-world scenarios. The project lacks an assessment of the model's reliability in practical applications, indicating a need for further investigation in this area in future research.

5.3 Process

The purpose of the first phase is to lay out the data pipeline and foundational configurations in the system for a baseline model to be set up. In this phase, it is expected that by utilizing the network metrics, a simple model can deploy a specific MTD technique against a specific attacker. In the second phase, the model takes into account the attackers when deciding on MTD types and MTD triggers.

5.3.1 Data Collection

The goal of data collection is to prepare the training data and set up the data pipeline for the model. The data pipeline allows the model to collect metrics regarding the state of the network and security posture and use them as a basis to determine if an MTD should be triggered. During the first phase, the pipeline will be structured so that metrics such as MTD-triggered frequency will be logged into a separate file when the simulator is executed. The file will then be further pre-processed using Python libraries such as pandas. Metrics regarding the network configurations will be used as features and

security posture metrics will be used as ground values. The pre-processed file will be the training and testing data for the model. Additional features will be considered in the second phase such as the attack path score. The shortest path will be computed from the breach node to the sink node to determine security posture.

5.3.2 Model Development and Deployment

To evaluate the efficacy and functionality of the system, a rudimentary neural network model will be employed. This neural network will serve as a preliminary testbed to validate the overall operational flow of the system. The model should be able to deploy an MTD technique against an attacker to maintain the security posture of the network within a certain boundary. A check will be performed periodically according to certain intervals to ensure the security posture resides within the boundary at all times. The system also includes a static network degradation factor which forces the model to deploy an MTD technique if the network remains static for too long. The assumption is that leaving the network static for too long will eventually allow attacks such as APT to breach the system. At the end of this phase it is expected that by utilizing the network metrics, a simple neural model can periodically deploy a specific MTD technique against a specific attacker to maintain system security posture at a certain level. It is assumed that the intruder detection system is perfect and can detect all the attackers. The system will also apply sensitivity analysis at different thresholds. Event-based MTD deployment will be used in this phase instead of time interval-based MTD deployment so that the model can be more reactive against attackers. The development process uses the standard CI/CD (Continuous Integration/Continuous Deployment) pipeline due to its efficiency.

5.4 Evaluation

Due to the limited existing research on reinforcement learning for deploying multiple Moving Target Defense (MTD) measures, comparing the performance of the proposed model against previous works is challenging. Therefore, this project adopts a multi-variable evaluation approach, leveraging combinations of AI models outlined in Table 3 and the features described in the 5.2.2 section for a comprehensive assessment. The security robustness is measured by (1) the total actions blocked and (2) the average attempts required to compromise metrics suggested by Hong et al. [7] as they are less affected by external factors, while the system performance is assessed by (A) delay in request handling and (B) resource cost [6]. By observing the overall security changes and disruptions to normal traffic, the approach can elevate combinations(e.g. Deep Q-learning that optimizes based on attack cost) that do well in both security and performance measures.

Table 3: AI Models

AI Model
Baseline Deep Q-learning Model
Double Q-learning
Dueling DQN

6 Progress to Date

6.1 Preliminary results

In the early stages of the investigation, the Jupyter Notebook version of the simulator is used instead of the GUI version due to the complexity and difficulty in navigating the code of the GUI version. Throughout the first half of the semester, a data pipeline is established within the simulator. A basic neural network is developed to analyze various features extracted from this pipeline. Additionally, explorations are conducted within the code base to implement event-based MTD (Moving Target Defense), and initial assessments of different attackers are made. Moreover, enhancements are made to the simulator, including the incorporation of a static degradation factor and consideration of a threshold for system posture. These foundational investigations constitute the essential components of phase 1. However, to consolidate these adjustments and considerations into the initial baseline system,

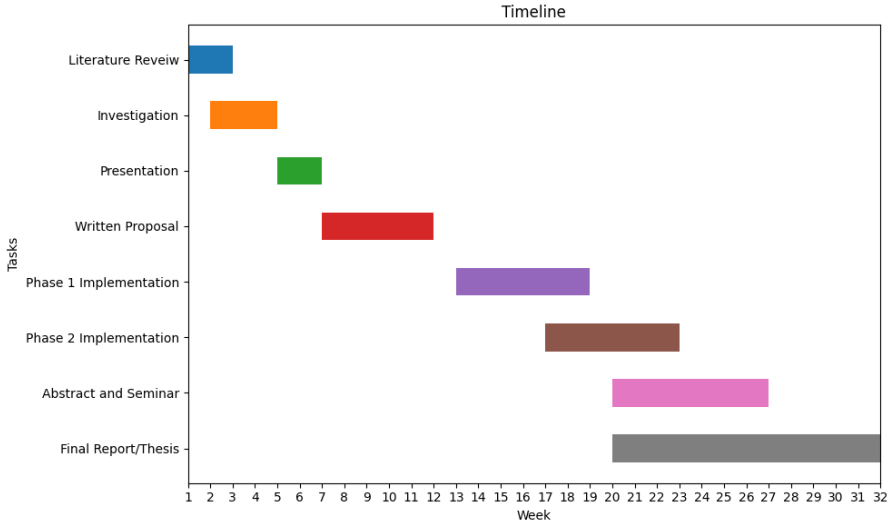

Figure 3: Timeline

further integration work is required. This integration phase is scheduled for the study break after week 12.

6.2 Progress

The overall period is 32 weeks because it includes not only the two semesters but also the exam period and winter break. Every task up to week 12 besides the written proposal has been completed. After the written proposal finishes at the start of week 12, there will be a week break since there will be a lot of other assignments due in week 12. The implementation of phase 1 is planned to be finished within the first month starting from the week 13 exam study break. Phase 2 should begin during the last quarter of the winter break and finish around week 4 into the second semester. Finally, the rest of the second semester will be used for the abstract/seminar and the final thesis report.

6.3 Risk

Although it is highly unlikely, it is possible that laptop damage can occur and lead to data and code losses. All the code and data have been pushed to a GitHub repository this risk should be mitigated. A more palatable concern is the lack of computing power to train the model. However, this can also be mitigated by using high-performance computing resources from the university CPU as mentioned in the methodology.

References

[1] Adel Alshamrani, Sowmya Myneni, Ankur Chowdhary, and Dijiang Huang. A survey on advanced persistent threats: Techniques, solutions, challenges, and research opportunities. *IEEE Communications Surveys Tutorials*, PP:1–1, 01 2019.

[2] Jin-Hee Cho, Dilli P. Sharma, Hooman Alavizadeh, Seunghyun Yoon, Noam Ben-Asher, Terrence J. Moore, Dong Seong Kim, Hyuk Lim, and Frederica F. Nelson. Toward proactive, adaptive defense: A survey on moving target defense. *IEEE Communications Surveys and Tutorials*, 22(1):709–745, 2020.

[3] Minjune Kim, Jin-Hee Cho, Hyuk Lim, Terrence Moore, Frederica Nelson, and Dan Kim. Performance and security evaluation of a moving target defense based on a software-defined networking environment. pages 119–129, 11 2022.

[4] Hooman Alavizadeh, Dong Seong Kim, Jin B. Hong, and Julian Jang-Jaccard. Effective security analysis for combinations of moving target defense techniques in cloud computing (short paper). In *Information Security Practice and Experience - 13th International Conference, ISPEC 2017, Proceedings*, Lecture Notes in Computer Science, pages 539–548, Germany, January 2017. Springer-Verlag London Ltd. 13th International Conference on Information Security Practice and Experience, ISPEC 2017; December 13-15, 2017.

[5] Hooman Alavizadeh, Jin Hong, Dong Seong Kim, and Julian Jang-Jaccard. Evaluating the effectiveness of shuffle and redundancy moving target defense techniques in the cloud. *Computers & Security*, 102, March 2021.

[6] Gayathri Rajakumaran and Neelanarayanan Venkataraman. Performance assessment of hybrid moving target defense for dos mitigation in public cloud. *International Journal of Intelligent Networks*, 2:140–147, 09 2021.

[7] Alex Brown, Tze-Wen Lee, and Jin Hong. Evaluating moving target defenses against realistic attack scenarios. pages 1–8, 05 2023.

[8] Sarah Alhozaimy and Daniel A. Menascé. A formal analysis of performance-security tradeoffs under frequent task reconfigurations. *Future Generation Computer Systems*, 127:252–262, 2022.

[9] Tuan Anh Nguyen, Minjune Kim, Jangse Lee, Dugki Min, Jae Woo Lee, and Dan Kim. Performability evaluation of switch-over moving target defense mechanisms in a software defined networking using stochastic reward nets. *Journal of Network and Computer Applications*, 199:103267, November 2021.

[10] Júlio Mendonça, Jin-Hee Cho, Terrence Moore, Frederica Nelson, Hyuk Lim, and Dan Kim. Performance impact analysis of services under a time-based moving target defense mechanism. *The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology*, 20:154851292110369, 08 2021.

[11] Taha Eghtesad, Yevgeniy Vorobeychik, and Aron Laszka. *Adversarial Deep Reinforcement Learning Based Adaptive Moving Target Defense*, pages 58–79. 12 2020.

[12] Sailik Sengupta, Tathagata Chakraborti, and Subbarao Kambhampati. *MTDeep: Boosting the Security of Deep Neural Nets Against Adversarial Attacks with Moving Target Defense*, pages 479–491. 10 2019.

[13] Rongbo Sun, Yuefei Zhu, Jinlong Fei, and Xingyu Chen. A survey on moving target defense: Intelligently affordable, optimized and self-adaptive. *Applied Sciences*, 13:5367, 04 2023.

[14] Wenxiao Zhang. Evaluating multiple moving target defense in the time domain. 2023.

[15] A. Brown. Mtdsim, 2021.

[16] Eric M. Hutchins, Michael J. Cloppert, and Rohan M. Amin. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. In *Leading Issues in Information Warfare & Security Research*, volume 1, pages 80–106. Academic Publishing Limited, Reading, UK, 2011.

[17] MITRE. Mitre ATT&CK framework: Adversary tactics and techniques, 2021.

[18] María del Carmen Lucas-Estañ, Baldomero Coll-Perales, and J. Gozalvez. Redundancy and diversity in wireless networks to support mobile industrial applications in industry 4.0. *IEEE Transactions on Industrial Informatics*, 17:311–320, 03 2020.

[19] H. Alavizadeh, S. Aref, D. Kim, and J. Jang-Jaccard. Evaluating the security and economic effects of moving target defense techniques on the cloud. *IEEE Transactions on Emerging Topics in Computing*, 10(04):1772–1788, Oct 2022.

[20] Qisheng Zhang, Jin-Hee Cho, Terrence Moore, Dan Kim, Hyuk Lim, and Frederica Nelson. *EVADE: Efficient Moving Target Defense for Autonomous Network Topology Shuffling Using Deep Reinforcement Learning*, pages 555–582. 05 2023.

[21] Tao Zhang, Changqiao Xu, Jiahao Shen, Xiaohui Kuang, and Luigi Grieco. How to disturb network reconnaissance: A moving target defense approach based on deep reinforcement learning. *IEEE Transactions on Information Forensics and Security*, PP:1–1, 01 2023.

[22] Ankur Chowdhary, Dijiang Huang, Abdulhakim Sabur, Neha Vadnere, Myong Kang, and Bruce Montrose. Sdn-based moving target defense using multi-agent reinforcement learning. In *Proceedings of the Conference Name*, 03 2021.

[23] Sailik Sengupta and Subbarao Kambhampati. Multi-agent reinforcement learning in bayesian stackelberg markov games for adaptive moving target defense, 07 2020.

[24] Qian Yao, Yongjie Wang, Xinli Xiong, Peng Wang, and Yang Li. Adversarial decision-making for moving target defense: A multi-agent markov game and reinforcement learning approach. *Entropy*, 25(4):605, 2023.

[25] Tina Moghaddam, Minjune Kim, Jin-Hee Cho, Hyuk Lim, Terrence Moore, Frederica Nelson, and Dan Kim. A practical security evaluation of a moving target defence against multi-phase cyberattacks. pages 103–110, 06 2022.

[26] Jin Hong, Simon Enoch, Dan Kim, Armstrong Nhlabatsi, Noora Fetais, and Khaled Khan. Dynamic security metrics for measuring the effectiveness of moving target defense techniques. *Computers & Security*, 79, 08 2018.

[27] Hooman Alavizadeh, Julian Jang-Jaccard, and Dong Seong Kim. Evaluation for combination of shuffle and diversity on moving target defense strategy for cloud computing. In *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pages 573–578, 2018.

[28] Hooman Alavizadeh, Dong Seong Kim, and Julian Jang-Jaccard. Model-based evaluation of combinations of shuffle and diversity MTD techniques on the cloud. *Future Generation Computer Systems*, 111:507–522, 2020.

[29] Hooman Alavizadeh, Jin B. Hong, Julian Jang-Jaccard, and Dong Seong Kim. Comprehensive security assessment of combined MTD techniques for the cloud. In *Proceedings of the 5th ACM Workshop on Moving Target Defense*, MTD '18, pages 11–20, New York, NY, USA, 2018. Association for Computing Machinery.

[30] Dilli P. Sharma, Simon Yusuf Enoch, Jin-Hee Cho, Terrence J. Moore, Frederica F. Nelson, Hyuk Lim, and Dong Seong Kim. Dynamic security metrics for software-defined network-based moving target defense. *Journal of Network and Computer Applications*, 170:102805, 2020.

[31] Simon Enoch, Mengmeng Ge, Jin Hong, Hani Alzaid, and Dan Kim. Evaluating the effectiveness of security metrics for dynamic networks. pages 277–284, 08 2017.

[32] Maxim Lapan. *Deep Reinforcement Learning Hands-On*. Packt Publishing, Birmingham, UK, 2018.

[33] Naveen Venkat. The curse of dimensionality: Inside out, September 2018.

[34] Balázs Varga, Balázs Kulcsár, and Morteza Haghir Chehreghani. Deep q-learning: A robust control approach. *International Journal of Robust and Nonlinear Control*, 33(1):526–544, 2023.

[35] Peter Mell, Karen Scarfone, and Sasha Romanosky. The common vulnerability scoring system (cvss) and its applicability to federal agency systems. *NIST Special Publication*, 800(55):1–12, 2006.