GENG5511 Engineering Research Project Part 1

Progress Report

Using Artificial Intelligence to Automate the Deployment of Moving Target Defence Operations

Joo Kai Tay
22489437
School of Engineering, University of Western Australia

Supervisor: Dr. Jin Hong
Department of Computer Science and Software Engineering,
University of Western Australia

Word count: 5194

School of Engineering
University of Western Australia

Submitted: 20 May 2024

Table of Contents

1 Introduction

Cybersecurity is a preeminent issue in today's interconnected world, with the prevalence of digital operations as the backbone of multiple sectors ranging from banking, transport and health, cybersecurity is of paramount importance [1]. While these industries reap the benefits of evolving technology, the sophistication of cyber threats is also on the rise. This makes robust cybersecurity measures essential to protect sensitive data, maintain privacy and ensure uninterrupted service delivery [2]. Effective cybersecurity practices not only shield organizations from financial and reputational damage but also bolster trust among customers and stakeholders. Furthermore, regulatory compliance with data protection laws necessitates stringent cybersecurity protocols to avoid legal repercussions [3].

Traditional methods, such as firewalls, IDS and antivirus software, focus on strengthening a static defence to prevent known threats. However, these can be less effective against new or evolving threats that exploit previously unknown vulnerabilities [4]. Compared to traditional cybersecurity techniques, which often rely on a fixed set of security measures and a well-defined perimeter, Moving Target Defence (MTD) offers a proactive and adaptive strategy. This is where MTD comes into play. Unlike static defences, which maintain a consistent configuration and attack surface, MTD continuously changes system aspects such as IP addresses, operating systems and application configurations [4]. This dynamic approach increases the complexity and cost for attackers, as the continuously evolving environment disrupts their reconnaissance efforts and disrupt chained attacks (e.g., APTs). By preventing attackers from gaining a foothold, MTD enhances overall system security and resilience. MTD's dynamic nature means that even if an attacker discovers a vulnerability, it may no longer be relevant or accessible by the time they attempt to exploit it. This fundamental difference makes MTD particularly valuable in environments where security needs to constantly evolve in response to emerging threats [5].

However, current approaches towards MTD predominantly focuses on optimizing specific MTD techniques tailored to particular types of attackers or dedicated MTD methodologies. This specialized concentration often overlooks the broader applicability and integration of MTD strategies across varied threat landscapes and defensive scenarios. As a result, the research tends to delineate narrowly defined attacker models or individual MTD tactics, such as IP shuffling or software diversification, without addressing the potential for comprehensive or unified defence frameworks that consider multiple strategies that could be taken by attackers and defenders [4]. This approach can inadvertently limit the understanding of MTD's full capabilities and its effectiveness against diverse or evolving cyber threats, thereby constraining the development of more versatile and adaptive cybersecurity solutions.

Machine learning is increasingly used in decision-making through analysing large amounts of data to identify patterns and trends. By feeding a model new data, it enables it to continuously learn, allowing decision-making processes to adapt and improve over time, allowing for reactive outcomes. This project will leverage machine learning models to identify the best combinations of MTD techniques to deploy at strategic moments, based on network security posture metrics and attacker information. To build models capable of predicting the optimal MTD deployment and type, extensive research will first focus on identifying the right metrics for assessing network security posture. Following this, model development and fine-tuning will be essential to ensure that the security framework's is responsive and adaptable but also

maximizes resource efficiency by deploying defences precisely when and where they are most needed.

The outcomes of this research can provide valuable guidance to organizations seeking to secure their networks by adding an extra layer to traditional MTD techniques, further safeguarding their systems. Specifically, the primary contributions of this work include:

- Identifies and defines network security posture metrics that will inform the machine learning models, improving decision-making for MTD deployment.
- Introduces the application of machine learning models to dynamically identify and deploy optimal MTD techniques, enhancing adaptability in cybersecurity strategies.
- Develops a unified framework that addresses a diverse range of cyber threats, and is capable of evolving to meet new challenges, expanding beyond specialized MTD approaches.
- Demonstrates how continuous system changes through MTD disrupt attackers' reconnaissance and exploitation efforts, reducing the risk of advanced persistent threats (APTs).

2 Background

2.1 MTD Design Principles

MTD techniques are about proactively preventing cyberattacks by continuously changing the attack surface of the systems involved. Therefore, when designing a MTD system, the three main design choices revolve around the questions: what to move, how to move and when to move [4].

2.1.1 What to Move

Changing the specific system configuration attributes, also known as the attack surface, is what the question “What to move” seeks to address. These attack surfaces that can be dynamically altered to thwart or confuse attackers include various system or network properties such as IP addresses, port numbers, operating systems or software versions [4]. These changes can occur in the application, OS, VM or hardware layer [6]. To consider “What to move” successful, changes in these configuration attributes must alter the attack surface significantly, causing difficulties and increased effort for attackers. Additionally, they much be sufficiently varied to prevent a brute force attack from overcoming the changes [7].

2.1.2 How to Move

The term "How to move" describes the methods used to modify the attributes targeted by MTD strategies to enhance unpredictability and uncertainty. This concept is integral to the three operational components of MTD classification: shuffling, diversity, and redundancy (SDR). Common techniques used in MTD include artificial diversity and randomizations that either rearrange or randomize various system and network attributes. Each technique can be categorized under shuffling, diversity, or redundancy [4].

2.1.3 When to Move

The concept of "When to move" in a MTD system focuses on determining the optimal timing for transitioning from the current system state to a new one, thus rendering any information or progress obtained by an attacker in the current state obsolete. There are three primary adaptation strategies for implementing these changes:

- **Reactive Adaptation**: This strategy is event-driven, activating changes in response to detected suspicious activities or alerts. Reactive MTD is designed to counteract an attacker's moves once their actions have been identified, aiming to neutralize any advantage they may have gained.
- **Proactive Adaptation**: This approach involves scheduled changes to system configurations or properties, executed at fixed or random intervals. The goal is to ensure that any intelligence gathered by an attacker becomes outdated rapidly, necessitating regular updates to maintain security. Proactive MTD operates independently of any direct threat presence, which can introduce some delay due to frequent updates.
- **Hybrid Adaptation**: Combining elements of both reactive and proactive strategies, hybrid adaptations adjust the timing of MTD operations based on specific events or security alerts, while also maintaining a maximum interval between changes to guard against unnoticed threats.

A fixed interval approach typically provides a more proactive defence mechanism, potentially leading to higher security but at the cost of increased operational frequency, which can be expensive. An optimal MTD interval, therefore, needs to be dynamically determined during runtime to balance cost-effectiveness with effective security measures. In practice, system elements might be updated either on a scheduled basis or in response to security incidents.

3 Literature Review

3.1 MTD Combination Techniques

When choosing to combine multiple MTD techniques, there are several factors that need to be taken into account. Firstly, is the type of threat that is to be mitigated, second is their combined impact on the network security metrics and lastly the impact on the system performance.

The discussion in Section 2.1.1 introduces various scenarios amenable to modification by MTD techniques, while Section 2.1.2 organizes MTD strategies into three primary categories: shuffling, diversity, and redundancy. However, the effectiveness of specific MTD approaches, such as OS Diversity or IP shuffling, hinges on the nature of the threats they are designed to counteract. It is essential to compare MTD techniques that target similar threats to ensure an accurate assessment of their effectiveness. In theory, matching MTD technique instances to specific threat instances would allow for a perfect solution but this is impractical as it assumes perfect knowledge of the attackers' methods and does not account for new attack methods being developed. Hong et al. [8] suggests that rather than focusing on countering attacks, MTD should aim to increase the attack efforts to deter attacks on the network in question.

Alavizadeh et al. [9] assessed the effectiveness of these combined MTD strategies in a cloud environment using the HARM framework. This time, they incorporated four security metrics: Redundancy (R), Return on Attack (RoA), Attack Cost (AC), and System Availability (SA). They observed improvements in all security metrics after implementing the S+D+R combination compared to using a single MTD technique. However, this study only took into account the attack related metrics as discussed in section 3.1. The deployment of multiple MTD techniques might increase resource consumption, potentially degrading the system's overall performance and adversely affecting the defence related metrics.

3.2 Network Security Metrics

Network security metrics are essential for assessing the effectiveness of security policies, detecting vulnerabilities, and ensuring compliance with regulations. These metrics provide a quantitative basis for understanding the security posture of a network, guiding security investments, and measuring the impact of security improvements. Good security metrics need to be specific, measurable, attainable, repeatable and time-dependent [10].

Behi et al. [10], details an experimental study conducted to evaluate the proposed approach for quantifying network security. The researchers collected data on various security metrics, such as web browser versions, operating system updates, vulnerabilities identified through Nessus scans, malware incidences reported by antivirus software, and the timeliness of antivirus and system scans. The analysis utilized these metrics to calculate a security score for each machine, using statistical methods like regression and correlation to determine their impact on the overall security of the network. The findings indicate the ranking of security metrics based on their impact on network security, were system updates, followed by web browser version and OS version.

Another metric used to evaluate network security is the Common Vulnerability Scoring System (CVSS). The CVSS offers a systematic method for evaluating the severity of security vulnerabilities, focusing on their characteristics and resultant effects. The CVSS framework comprises base, temporal, and environmental metrics. As delineated in the CVSS Version 2.0, the base metrics include factors such as Access Vector, Access Complexity, Authentication, Confidentiality, Integrity, and Availability [11]. These metrics provide fundamental insights into the intrinsic attributes of a vulnerability, assessing exploitability, impact, and the potential extent of damage upon exploitation. The temporal metric addresses changes in vulnerability

characteristics over time, while the environmental metrics evaluate the specific impact of vulnerabilities within a unique user environment.

Yusuf et al. [12] proposed the Temporal-Hierarchical Attack Representation Model (T-HARM) model to capture network changes and investigate how existing cyber security metrics can be adapted to dynamic networks, rather than the traditional graphical security models which assume that vulnerabilities, nodes and edges are static at all times [13]. The model leverages the concept of temporal graphs to monitor and evaluate security metrics, which expand upon a base CVSS score by assigning probabilities, impacts, and costs of attacks to create a new layer of security scores. It defines various parameters such as Risk on attack paths (R), Return on attack paths (ROA), and Cost on attack paths (AC). It also calculates the Standard deviation of attack path lengths (SDPL), Probability of attack success on paths (Pr), and a Normalised mean of attack path lengths (NMPL). Additionally, the model considers the Mean of attack path lengths (MAPL), Mode of attack path lengths (MoPL), Number of attack paths (NAP), and the shortest attack path (SAP), providing a comprehensive framework to assess and mitigate potential security threats.

Hong et al. [8] extends upon this work by differentiating security metrics into categories for attack and defence. These metrics are designed to evaluate MTD techniques by tracking changes in the security posture as the attack surface evolves. A crucial element of the attack surface is the attack paths, and monitoring variations in these paths enables an assessment of the increased effort required for attacks. Specifically, Attack Path Variation (APV) measures shifts in these paths as MTD techniques modify the network configuration. Additionally, the exposure time of attack paths is vital, as longer exposures increase the likelihood of a successful attack. The cost of attacks is also significant, estimated through the difficulty of exploiting vulnerabilities as rated by the Common Vulnerability Scoring System (CVSS). The duration it takes for an attacker to breach each point in an attack path is a critical metric; longer attack durations increase the chances of detection. The defence effort metrics primarily focus on the costs associated with deploying MTD techniques within a network. These costs are multifaceted and include the monetary expenses incurred from integrating new MTD technologies. Additionally, there are operational impacts to consider, such as the downtime necessary for replacing each node with a different variant. This process can affect network efficiency, leading to increased service overhead and potential delays in the communication medium, all of which are crucial factors in the overall assessment of defence efforts.

3.3 Machine Learning and MTD

Sections 3.1 and 3.2 lay the foundation for understanding what to deploy and when in MTDs. However, network systems often have security flaws like similarity, determinism, and static configurations, which leave them vulnerable to prolonged observation, analysis, and repeated attacks. Attackers have since identified adaptive strategies that can exploit the vulnerabilities in time-based MTD techniques [14]. Reinforcement learning is particularly effective for this task because it allows the defence agent to learn the adversarial decision-making strategy and continuously adapt its own strategy to prioritize the rewards that lead to the overall stability of the network in both the attack and defence metrics.

Yao et al. [15] proposes the adversarial decision-making strategy, WoLF-BSS-Q, which is designed for multi-agent Markov games according to the Cyber-Kill-Chain model. Multi-agent Markov games, also known as stochastic games, are a framework in game theory where multiple agents interact in a dynamic environment, characterized by states that evolve stochastically in response to the actions of all agents involved. Each state in a Markov game has a set of possible actions for each agent, and the combination of actions chosen by the agents

determines the transition to the next state according to a probabilistic rule. These transitions also produce rewards, which may differ for each agent, reflecting their individual objectives. The core of studying Markov games involves analysing the policies by which agents select actions based on their observations of the current state. The goal for each agent is typically to maximize its expected return over time, which is the cumulative sum of future rewards, discounted by a factor that accounts for the preference for immediate rewards over distant ones. This setup extends the single-agent Markov decision processes (MDPs) to scenarios with multiple decision-makers, each possibly with competing interests. The WoLF-BSS-Q algorithm combines micro-level strategy selection (using the BSS algorithm within a leader-follower dynamic) and macro-level dynamic adjustments (through the WoLF mechanism), which observes and responds to game dynamics. The WoLF mechanism adapts its learning rates based on whether the agent is currently winning or losing, thus improving the adaptability and effectiveness of the strategy. Additionally, to address the common problem of Q-value overestimation in Q-learning, the authors propose a variant called "constrained Q-learning", which builds on conservative Q-learning methods to provide a more accurate and reliable learning process in these complex game environments.

Zhang et al. [16] introduced the reinforcement learning strategy EVADE to tackle several challenges in MTD. Firstly, the cost of shuffling network topology in resource-limited environments. Secondly, it aims to overcome the scalability issues and inefficiencies in delivering multi-objective defence strategies using deep reinforcement learning (DRL). The EVADE algorithm employed a fractal-based environment (FSS) in the EVADE strategy, significantly decreasing the training duration required by DRL algorithms. The use of FSS notably accelerates the convergence towards an almost optimal solution within the constraints of network adaptation budgets. Additionally, they developed a vulnerability-aware ranking algorithm named VREN, which is integral to EVADE. This algorithm facilitates strategic edge adaptations for highly efficient and effective reconfigurations of network topology. Furthermore, EVADE incorporates a density optimization (DO)-based greedy algorithm, used in conjunction with VREN, to further compress the search space utilized by DRL algorithms. This enhancement allows their hybrid MTD approach to achieve faster convergence within a more compact solution space. However, EVADE might not be scalable as it does not take into account the overhead of running DRL in an SDN controller.

Kreischer. [17] presents a new strategy that integrates Federated Learning (FL) and Reinforcement Learning (RL) to jointly develop methods for implementing MTD to combat malware attacks. RL adapts more effectively, learning continuously to potentially address even newly emerging threats. To maintain the privacy of each device's behavioural data, the defence selection policy is trained in a federated manner. This collaborative approach could help in effectively responding to an attack globally after it has been initially detected locally. However, this work is limited in the fact that it does not consider the temporal aspects of the network and attacker.

Zhang et al. [18] identified three primary challenges in integrating MTD into Delay Tolerant Mobile Networks (DTMN). First, there was a lack of emphasis on collaborative scheduling among various MTD strategies, which could enhance the security of DTMN systems. Second, MTD schemes were noted to be resource-intensive, with limited research focusing on optimizing time allocation to minimize network resource consumption. Third, existing defence mechanisms relied solely on current data without considering future information. To address these issues, this paper proposes a collaborative mutation-based MTD (CM-MTD) approach, focusing on two specific strategies: host address mutation (HAM) and route mutation (RM). These strategies adjust network attributes to disrupt different stages of the cyber kill chain. The

approach begins with the formulation of a semi-Markov decision process (SMDP) to model time-varying security threats and dynamic MTD scheme deployment. Predictions of security events are made using long short-term memory (LSTM) networks, which serve as the network states within the SMDP. The action space of the SMDP is then refined by removing infeasible actions that do not meet network constraints. Finally, a hierarchical deep reinforcement learning algorithm is developed to facilitate effective collaborative scheduling of the MTD strategies.

Existing MTD approaches are often specialized and narrowly focused, limiting their applicability against varied and evolving cyber threats. Moreover, static and deterministic network configurations leave systems vulnerable to prolonged reconnaissance and adaptive attacks. Furthermore, the resource-intensive and inefficient MTD strategies need to be optimized for scalability, while including collaborative scheduling across different defence mechanisms to enhance the overall efficiency.

4 Methodology

The goal is to develop a machine learning model that determines the most effective timing and type of MTD based on the calculated network security score. This involves identifying the appropriate features from the existing simulator framework, calculating a network security score and using that to train a machine learning model that will be able to identify appropriate timings and MTD methods. Accordingly, section 4.1 will introduce the MTD AI module, section 4.2 will cover the model training and section 4.3 will cover the model evaluation strategy.

The existing simulator framework highlighted in green consists of three main components: the Network Module (b), MTD module (c) and Attacker module (a) [19] as shown in Figure 1. The network module is a graph comprised of nodes and edges, which symbolize hosts and the connections between. Each node has IP addresses, operating systems, services, vulnerabilities, and user access credentials. The MTD module facilitates MTD operations by deploying MTD techniques into the network on a timed basis. The attacker module conducts attack operations, executing a sequence of attack actions on hosts within the network with the aim to compromise hosts.

The original simulator framework is time-based, deploying MTD into the network on a fixed schedule with no regard for the network security posture. In order to transform this simulator from a time-based simulator to a reactive one, we introduce the MTD AI plugin highlighted in orange in Figure 1. The MTD AI will receive information about the network posture in real time from the network module and use that to determine if an MTD should be triggered. If the model determines that triggering an MTD will improve the network security posture, it will select the most appropriate MTD technique from a list and deploy that into the network via the MTD operations module.

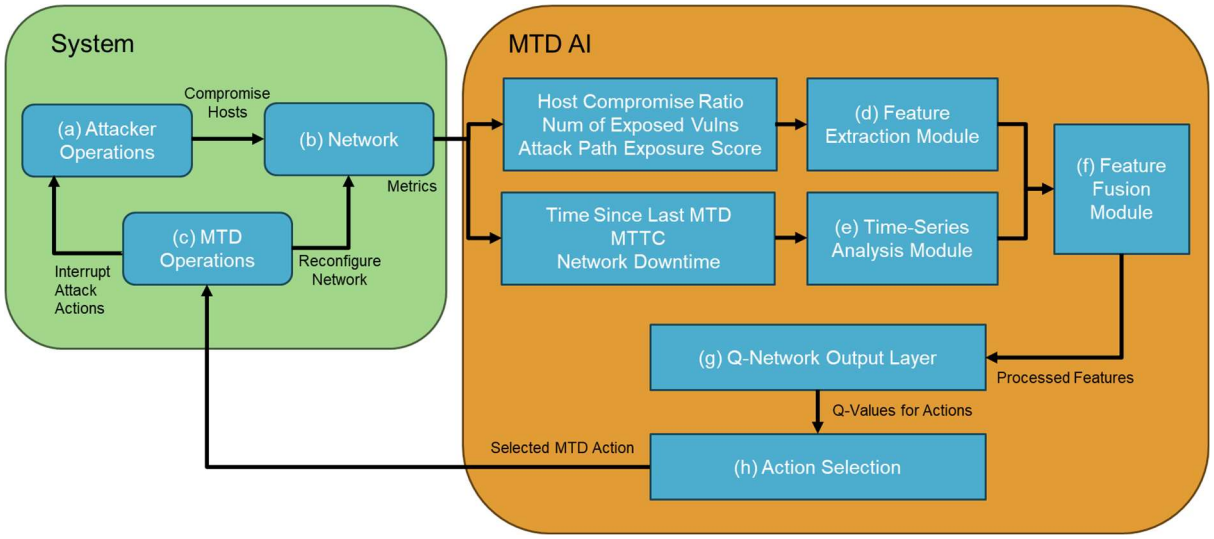

Figure 1: System Overview

4.1 MTD AI

Reinforcement Learning (RL) is a branch of machine learning where an agent learns to make decisions through trial and error, aiming to maximize cumulative rewards within an environment [20]. At its core, RL involves an agent interacting with its environment by taking actions that transition the system from one state to another, while receiving feedback in the form of rewards. Key concepts in RL include the agent (the learner and decision-maker), the environment (the context in which the agent operates), actions (possible decisions or moves),

states (current situation of the environment), and rewards (feedback signals indicating the success of actions). The policy represents the agent's strategy for selecting actions, while the value function and Q-function estimate the expected cumulative rewards. Effective learning balances exploration (trying new actions to discover their rewards) and exploitation (choosing known rewarding actions).

In this work, we propose a Deep Q-learning model to strategically address vulnerabilities within a network environment, leveraging key metrics discussed in section 3.2. The model formulates the problem as a Markov Decision Process (MDP), where the RL agent, guided by a Deep Q-Network (DQN), learns to maximize cumulative rewards through intelligent decision-making. The action space includes deploying the MTD techniques IP shuffle, OS diversity, Service Diversity and Complete Topology shuffle [19] or choosing not to deploy MTD. The state space comprises a feature vector encompassing network security metrics, including the HCR, the Number of Vulnerabilities, the Number of Exposed Vulnerabilities, the Attack Path Exposure Score, the MTTC, Downtime and Operational Impact, and Time Since Last MTD.

Obtaining the right features from the simulator is crucial in because it directly influences model performance, interpretability, and efficiency. Irrelevant or redundant features can introduce noise, leading to overfitting and reduced predictive accuracy. By carefully selecting the most informative features, we ensure that the model can optimize the deployment of MTD techniques by reducing training time, and enhance interpretability, enabling a clearer understanding of the relationship between input variables and predictions.

4.1.1 Feature Extraction Module

The feature extraction module (d) in figure 1 is used to transform the raw network security metrics into a meaningful representation that can be effectively utilized by the RL algorithm. This module consists of a series of densely connected neural network layers designed to capture the relationships between the features. Starting with the raw input data, the module first normalizes the values to ensure consistent scale and prevent biases associated with different metric magnitudes. The data then passes through multiple dense layers, each followed by a Rectified Linear Unit (ReLU) activation function that introduces non-linearity, enabling the model to learn more complex patterns in the data. This structured layering distils the raw metrics into a higher-level abstract representation, capturing underlying patterns that are not immediately obvious.

Among the metrics processed, the Host Compromise Ratio (HCR) is an indicator of network security health. It measures the proportion of hosts within a network that are compromised or at high risk due to identified vulnerabilities, providing insights into the extent of malware infections, misconfigurations, or unpatched vulnerabilities. A high HCR signals substantial exposure to potential breaches, necessitating urgent actions such as patch management, system hardening, and user education. Conversely, a low HCR indicates that the network's threat mitigation strategies are effective. By integrating the HCR, the model can better prioritize remediation efforts and bolster network defences.

The module also analyses the Number of Vulnerabilities, which totals the identified security weaknesses within the network environment. This metric is particularly important during the initialization phase of the network graph in MTDSimTime. It is further refined by the Number of Exposed Vulnerabilities, focusing on vulnerabilities that are accessible to potential attackers due to their presence in publicly accessible systems or areas lacking sufficient network

segmentation. Monitoring changes in this metric as MTD solutions are deployed allows the model to assess the ongoing effectiveness of security measures.

The Attack Path Exposure Score, quantifies the risk associated with potential attack paths in the network. This score evaluates the vulnerability of interconnected systems and resources that attackers could exploit to access critical assets. It incorporates several factors, including the number and severity of exploitable vulnerabilities (as gauged by CVSS scores), the network's configuration and topology, and the accessibility of critical nodes. Higher scores indicate more significant exposure, suggesting easier navigation for adversaries through the network to target high-value assets.

4.1.2 Time Series Analysis Module

The time series analysis module (e) in figure 1, is specifically designed to handle temporal dependencies present in network security data. This module employs Long Short-Term Memory (LSTM), to effectively process and learn from sequences of network security events, such as changes in the Mean Time to Compromise (MTTC) and the intervals between deployments of MTD techniques. By analysing how these metrics evolve over time, the LSTM module can capture long-term dependencies that traditional models might overlook. This capability allows the model to predict future network states based on past and present data. The output from this time series analysis module provides a temporal feature vector.

MTCC is a security metric that measures the average duration it takes for an attacker to successfully breach a system or network after identifying a potential vulnerability. In an MTD-enabled environment, the MTTC provides crucial insights into the effectiveness of these adaptive security measures. Ideally, by continually shifting the attack surface, MTD should significantly increase the MTTC, giving defenders more time to detect, respond to, and neutralize threats.

Downtime and Operational Impact for Node Replacement refers to the temporary loss of service and associated disruptions that occur when a network node is replaced, updated, or reconfigured during an IP Shuffle or OS Diversity. This metric is particularly relevant when deciding when is the most appropriate time to deploy an MTD technique as the availability of the network must be taken into account alongside security.

Time since last MTD is a metric used to measure the elapsed time since the last MTD technique was deployed into the network. This metric is important as the larger this gets; the more time the attack surface has remained static which increases the likelihood that reconnaissance efforts from adversaries would pay off.

4.1.3 Q-Network Output Module

The Q-network architecture combines the feature extraction module and time-series analysis module in the Feature Fusion Module (f), which concatenates their outputs and processes them through an additional dense layer ([Feature Extraction Output] + [Time-Series Analysis Output] -> Dense(64) -> ReLU). The final Q-network output layer (g) predicts Q-values for each action in the action space, enabling the agent to choose the most beneficial action based on the current network security state.

4.2 Model Training

The model is trained using a Deep Q-learning algorithm with the Bellman equation to update Q-values. Experience replay is utilized to store past experiences, breaking correlations in the training data, while a target network stabilizes Q-learning updates by providing a separate network for target computation.

A reward function ensures that the agent optimizes network security health:

- **Positive Rewards**:
  - Reduced HCR
  - Reduced Number of Vulnerabilities and Exposed Vulnerabilities
  - Increased MTCC
- **Negative Rewards**:
  - Increased Downtime and Operational Impact for Node Replacement
  - High Attack Path Exposure Score
  - High Time Since Last MTD

During training, each episode begins by resetting the environment state. While not in a terminal state, the agent selects an action using a greedy strategy, performs the action, and observes the next state and reward. The experience (state, action, reward, next state) is stored in the replay buffer, and a minibatch is sampled from the buffer to update the Q-network using the Bellman equation. Periodic updates to the target network further stabilize the learning process.

The model's real-time inference capabilities and adaptive learning approach ensure the prioritization of effective remediation measures, providing a comprehensive framework for proactive and reactive network security management. By leveraging this Deep Q-learning model, network administrators can dynamically optimize defence strategies, reducing exposure to potential breaches and improving overall network security health.

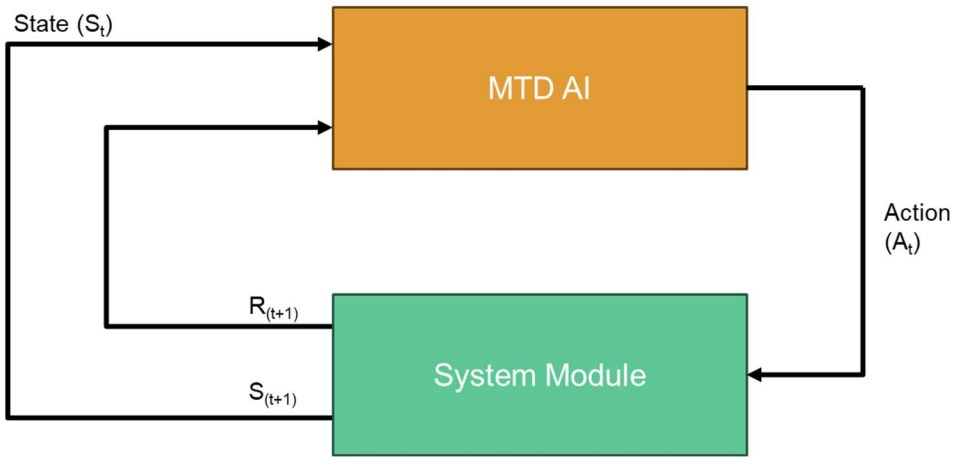

Figure 2: Training loop

4.3 Model Evaluation

The next step in the process would be testing the proposed reinforcement learning model on a real-world dataset in the simulator to ensure its applicability and robustness. We will be using a real-world intrusion dataset that has been pre-processed to align the structure of the network module described in section 3.1. The testing process begins by initializing the environment with a realistic network state and allowing the agent to interact with this environment over multiple episodes. The agent makes remediation decisions, such as deploying MTD techniques, based on current network states while adjusting its policy dynamically. The reward function provides real-time feedback, reflecting the network's security health improvements or degradations. Performance is evaluated using key security outcomes, including reductions in HCR, the Number of Vulnerabilities, and Attack Path Exposure Score, as well as increased MTTC.

Following the initial testing phase, we will conduct a comparative analysis of our reinforcement learning model against two baseline scenarios: the static defence case and the random case. In the static defence case, defensive strategies are pre-determined and do not adapt to changing network conditions or threat landscapes. In contrast, the random case involves deploying defence measures in an arbitrary manner, without any strategic consideration or pattern recognition. For each scenario, the model's effectiveness will be assessed based on the same key security outcomes used in the initial tests. We aim to demonstrate that our dynamic reinforcement learning approach outperforms these baseline methods by adapting more effectively to evolving threats, thereby optimizing security measures and reducing overall vulnerability. This comparison will provide clear insights into the benefits and limitations of implementing an adaptive security strategy within network environments.

5 Progress to Date

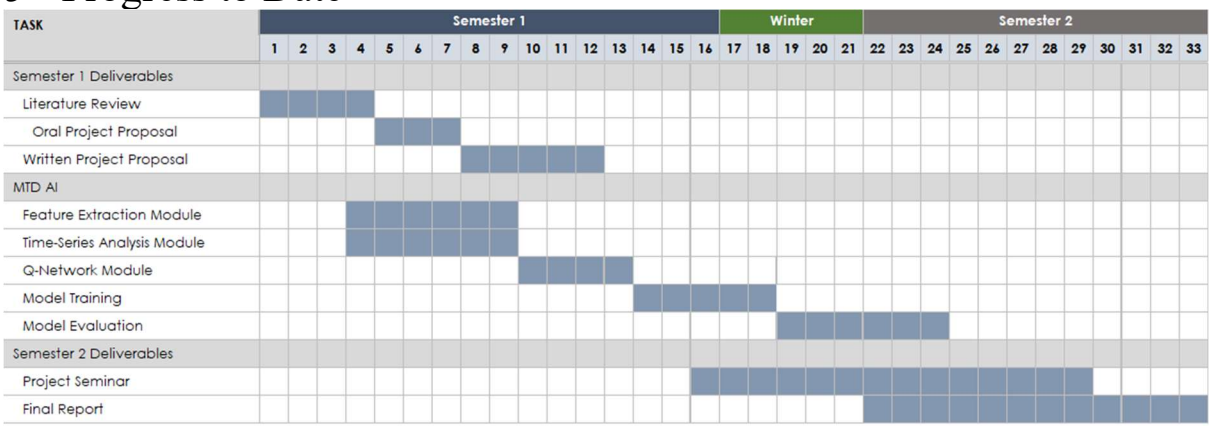

Figure 3: Timeline

The timeline for the project completion is shown in Figure 3 above. The first stage, including the literature review and oral project proposal, is complete. During this phase, we identified the existing limitations in the field and determined the necessary steps to advance research.

Phase 1: Feature Extraction and Time-Series Analysis Modules:

In the second phase of the project, we have collected the data required for the two feature modules by adapting the functions of the existing MTD simulator to return the necessary metrics. We have also commenced development of modules to integrate the features into the Q-Network.

Phase 2: Q-Network Development:

The third phase focuses on creating and fine-tuning the reinforcement learning model. This will involve selecting the most suitable model, optimizing it for accuracy and efficiency, and integrating it with the existing system.

Phase 3: Model Training:

Phase 3 involves the model being trained in the simulator using the methodology from section 4.2.

Phase 4: Model Evaluation

Next, the model will be integrated into the MTD simulator and tested using a real dataset to evaluate its performance. The model will be benchmarked against the base cases to evaluate the improvements that MTD AI brings.

Final Deliverables:

The project will conclude with a comprehensive final report and an oral presentation showcasing the progress made throughout the project.

6 References

[1] V. Sardana and S. Singhania, 'Digital Technology in the Realm of Banking: A Review of Literature', Nov. 2018.

[2] S. Al-Bassam and A. Al-Alawi, 'The Significance of Cybersecurity System in Helping Managing Risk in Banking and Financial Sector', Nov. 2019.

[3] D. Thaw, 'The Efficacy of Cybersecurity Regulation', *Ga. State Univ. Law Rev.*, vol. 30, no. 2, pp. 287–374, 2014 2013.

[4] J.-H. Cho *et al.*, 'Toward Proactive, Adaptive Defense: A Survey on Moving Target Defense', *IEEE Commun. Surv. Tutor.*, vol. 22, no. 1, pp. 709–745, 2020, doi: 10.1109/COMST.2019.2963791.

[5] S. Sengupta, A. Chowdhary, A. Sabur, A. Alshamrani, D. Huang, and S. Kambhampati, 'A Survey of Moving Target Defenses for Network Security', *IEEE Commun. Surv. Tutor.*, vol. 22, no. 3, pp. 1909–1941, 2020, doi: 10.1109/COMST.2020.2982955.

[6] H. Alavizadeh, D. S. Kim, J. B. Hong, and J. Jang-Jaccard, 'Effective Security Analysis for Combinations of MTD Techniques on Cloud Computing (Short Paper)', in *Information Security Practice and Experience*, J. K. Liu and P. Samarati, Eds., Cham: Springer International Publishing, 2017, pp. 539–548. doi: 10.1007/978-3-319-72359-4_32.

[7] G. Cai, B. Wang, W. Hu, and T. Wang, 'Moving target defense: state of the art and characteristics', *Front. Inf. Technol. Electron. Eng.*, vol. 17, no. 11, pp. 1122–1153, Nov. 2016, doi: 10.1631/FITEE.1601321.

[8] J. B. Hong, S. Y. Enoch, D. S. Kim, A. Nhlabatsi, N. Fetais, and K. M. Khan, 'Dynamic security metrics for measuring the effectiveness of moving target defense techniques', *Comput. Secur.*, vol. 79, pp. 33–52, Nov. 2018, doi: 10.1016/j.cose.2018.08.003.

[9] H. Alavizadeh, J. Jang-Jaccard, and D. S. 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)*, Aug. 2018, pp. 573–578. doi: 10.1109/TrustCom/BigDataSE.2018.00087.

[10] Mostafa Behi, Mohammad GhasemiGol, and Hamed Vahdat-Nejad, 'A New Approach to Quantify Network Security by Ranking of Security Metrics and Considering Their Relationships', *Int. J. Netw. Secur.*, vol. 20, no. 1, Jan. 2018, doi: 10.6633/IJNS.201801.20(1).15.

[11] V. R. Kebande, I. Kigwana, H. S. Venter, N. M. Karie, and R. D. Wario, 'CVSS Metric-Based Analysis, Classification and Assessment of Computer Network Threats and Vulnerabilities', in *2018 International Conference on Advances in Big Data, Computing and Data Communication Systems (icABCD)*, Durban, South Africa: IEEE, Aug. 2018, pp. 1–10. doi: 10.1109/ICABCD.2018.8465420.

[12] S. E. Yusuf, M. Ge, J. B. Hong, H. K. Kim, P. Kim, and D. S. Kim, 'Security Modelling and Analysis of Dynamic Enterprise Networks', in *2016 IEEE International Conference on Computer and Information Technology (CIT)*, Dec. 2016, pp. 249–256. doi: 10.1109/CIT.2016.88.

[13] L. Wang, M. Albanese, and S. Jajodia, ‘Network Hardening: An Automated Approach to Improving Network Security’, Mar. 2014. Accessed: May 03, 2024. [Online]. Available: https://www.semanticscholar.org/paper/Network-Hardening%3A-An-Automated-Approach-to-Network-Wang-Albanese/adbf09442409ebdb5968c9f4495471c9c7d697f6

[14] M. L. Winterrose, K. M. Carter, N. Wagner, and W. W. Streilein, ‘Adaptive Attacker Strategy Development Against Moving Target Cyber Defenses’, in *Advances in Cyber Security Analytics and Decision Systems*, S. K. Shandilya, N. Wagner, and A. K. Nagar, Eds., Cham: Springer International Publishing, 2020, pp. 1–14. doi: 10.1007/978-3-030-19353-9_1.

[15] Q. Yao, Y. Wang, X. Xiong, P. Wang, and Y. Li, ‘Adversarial Decision-Making for Moving Target Defense: A Multi-Agent Markov Game and Reinforcement Learning Approach’, *Entropy*, vol. 25, no. 4, Art. no. 4, Apr. 2023, doi: 10.3390/e25040605.

[16] Q. Zhang, J.-H. Cho, T. J. Moore, D. D. Kim, H. Lim, and F. Nelson, ‘EVADE: Efficient Moving Target Defense for Autonomous Network Topology Shuffling Using Deep Reinforcement Learning’, in *Applied Cryptography and Network Security*, M. Tibouchi and X. Wang, Eds., Cham: Springer Nature Switzerland, 2023, pp. 555–582. doi: 10.1007/978-3-031-33488-7_21.

[17] J. Kreischer, ‘Federated Reinforcement Learning for Private and Collaborative Selection of Moving Target Defense Mechanisms for IoT Device Security’, Master’s Thesis, University of Zurich, Zürich, Switzerland, 2023. doi: 10.5167/uzh-255748.

[18] T. Zhang *et al.*, ‘When Moving Target Defense Meets Attack Prediction in Digital Twins: A Convolutional and Hierarchical Reinforcement Learning Approach’, *IEEE J. Sel. Areas Commun.*, vol. 41, no. 10, pp. 3293–3305, Oct. 2023, doi: 10.1109/JSAC.2023.3310072.

[19] W. Zhang, ‘MTDSimTime’. [Online]. Available: https://github.com/MoeBuTa/MTDSimTime

[20] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, ‘Deep Reinforcement Learning: A Brief Survey’, *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 26–38, Nov. 2017, doi: 10.1109/MSP.2017.2743240.