Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.064849

ARTICLE

Multi-Agent Reinforcement Learning for Moving Target Defense Temporal Decision-Making Approach Based on Stackelberg-FlipIt Games

Rongbo Sun, Jinlong Fei*, Yuefei Zhu and Zhongyu Guo

Key Laboratory of Cyberspace Security, Ministry of Education, Zhengzhou, 450001, China

*Corresponding Author: Jinlong Fei. Email: aston_deta@163.com

Received: 25 February 2025; Accepted: 19 May 2025; Published: 03 July 2025

ABSTRACT: Moving Target Defense (MTD) necessitates scientifically effective decision-making methodologies for defensive technology implementation. While most MTD decision studies focus on accurately identifying optimal strategies, the issue of optimal defense timing remains underexplored. Current default approaches—periodic or overly frequent MTD triggers—lead to suboptimal trade-offs among system security, performance, and cost. The timing of MTD strategy activation critically impacts both defensive efficacy and operational overhead, yet existing frameworks inadequately address this temporal dimension. To bridge this gap, this paper proposes a Stackelberg-FlipIt game model that formalizes asymmetric cyber conflicts as alternating control over attack surfaces, thereby capturing the dynamic security state evolution of MTD systems. We introduce a belief factor to quantify information asymmetry during adversarial interactions, enhancing the precision of MTD trigger timing. Leveraging this game-theoretic foundation, we employ Multi-Agent Reinforcement Learning (MARL) to derive adaptive temporal strategies, optimized via a novel four-dimensional reward function that holistically balances security, performance, cost, and timing. Experimental validation using IP address mutation against scanning attacks demonstrates stable strategy convergence and accelerated defense response, significantly improving cybersecurity affordability and effectiveness.

KEYWORDS: Cyber security; moving target defense; multi-agent reinforcement learning; security metrics; game theory

1 Introduction

The pervasive digitization of societal infrastructure over two decades has rendered globally distributed entities profoundly interconnected and interdependent, with cyberspace emerging as a contested domain for adversarial engagements. Contemporary network systems increasingly exhibit automation, intelligence, and integration. Paradoxically, sophisticated attacks often exploit simple tools/vectors, whereas defenses necessitate complex coordinated mechanisms—an imbalance exacerbated by attackers' asymmetric advantages in reconnaissance and persistence.

Inspired by principles of dynamism, randomization, determinism, and diversity, Moving Target Defense (MTD) dynamically manipulates system configurations (e.g., via *shuffling*, *diversity*, and *redundancy*) to reduce attack surface predictability, forcing adversaries into perpetual reconnaissance cycles [1]. MTD research clusters around three axes: "*what to move*" (spatial decisions), "*when to move*" (temporal decisions), and "*how to move*" (technical implementation). The temporal decision-making of MTD aims to select the optimal temporal strategy for transitioning an MTD system from its current state to a new state, thereby invalidating the information or progress acquired by attackers under the current state while ensuring high

defensive efficacy and low implementation costs. Consequently, determining how to identify the optimal MTD temporal strategy through balancing system security and availability to guarantee both operational efficiency and defensive effectiveness constitutes a critical research challenge. The primary objective of MTD is to eliminate attackers' asymmetric temporal advantage, underscoring that temporal strategy represents an essential component of MTD. Key temporal parameters—including the interval between consecutive MTD activations, the duration required for MTD deployment, the time attackers allocate to executing their strategies, and the time required for successful attacks—exert significant influence on the overall effectiveness of MTD implementations. Zhuang et al. [2] provided a preliminary exploration of MTD temporal strategies by outlining fundamental issues and hypotheses. They posited that defining the probing surface and attack surface through temporal and spatial MTD strategies could better model the dynamic characteristics of MTD systems. However, their work did not establish a concrete theoretical framework for temporal decision-making in MTD systems.

However, the temporal effectiveness of MTD strategies critically influences the operational success. Overly aggressive MTD activation risks system instability or service degradation due to synchronization overhead, resource contention, or interrupted legitimate workflows. Conversely, insufficiently reactive MTD intervals grant attackers extended time windows to analyze system patterns, exploit vulnerabilities, or escalate privileges, ultimately undermining defense objectives. To balance these trade-offs, Clark et al. [3] proposed a time-based decoy deception detection technique, where a virtual network composed of decoys records temporal log information such as query and response times for attempted node connections. By analyzing the response times of nodes to probing packets, node types could be identified, and they derived a closed-form solution for the expected detection time. While stochastic approaches obscure predictable attack surfaces and complicate adversarial time-based reconnaissance, existing works struggle to rigorously harmonize defensive efficacy with operational efficiency, which necessitates advanced decision frameworks that holistically integrate temporal dynamics, adversarial behavior models, and system constraints. Therefore, this paper addresses MTD timing optimization via a Stackelberg-FlipIt game framework integrated with multi-agent Win or Learn Fast Policy Hill-Climbing (WoLF-PHC) algorithm. Key contributions include:

1. **Abstracting the Network Attack-Defense Process as a Stackelberg-FlipIt Game and Introducing Belief Factors to Control MTD Temporal Decisions.** Tan et al. [4] modeled the MTD temporal decision-making process as a FlipIt game, which effectively depicted the process of alternating control of the attack surface between the attacker and the defender, so as to integrate the temporal strategy into the game decision-making process. However, their design of the game is not accurate enough, ignoring the problem of action sequence and information asymmetry in the real game. Therefore, this paper introduces a Stackelberg game framework, constructing a Stackelberg-FlipIt game model based on the integration of Stackelberg and FlipIt games. By designing and incorporating belief factors to characterize the estimation differences between attackers and defenders regarding defense thresholds, the model more accurately reflects the asymmetric nature of real-world network attack-defense scenarios. This abstraction precisely captures the dynamics of actual network environments, facilitating the dynamic learning and updating of MTD decision-making methods and optimizing the judgment of triggering timing.
2. **Solving the Game Equilibrium by Using Multi-Agent WoLF-PHC Algorithm.** Existing works [4–6] tended to solve the equilibrium of their modeled temporal game by purely mathematical equations, such as Min-Max solving and dynamic programming. However, with the sudden increase of the dimension of the policy space, it is difficult for this kind of solutions to converge in an acceptable time and may only converge to the suboptimal strategies. Therefore, this paper tends to utilize a multi-agent reinforcement learning framework to simulate the game solving process. The WoLF-PHC

algorithm is an adaptive reinforcement learning algorithm [7] with low computational complexity, strong convergence properties, and the ability to adapt to dynamic environments. It can quickly learn and adjust strategies, making it highly suitable for MTD temporal decision-making research based on the Stackelberg-FlipIt game. Under the multi-agent framework, defenders and attackers act as independent agents, optimizing their respective strategies through interactive learning. Defenders can dynamically adjust MTD triggering times based on the attackers' strategies, thereby achieving more efficient defense.

3. **Defining a Comprehensive Reward Function.** The aforementioned methods are too simplistic to evaluate the benefits of the strategy, and most of them are evaluated based on the cost of the strategy, ignoring the benefits brought by the strategy in terms of security and other aspects. Therefore, this paper designs a reward function based on Security, Performance, Affordability, and Belief Error to guide the learning direction of the agents. The function references existing research frameworks [8–10], ensuring the accuracy and generalizability of reward quantification. It comprehensively balances the costs, defensive effects, and security of different MTD strategies while ensuring the optimal triggering timing.

The remainder of this paper is organized as follows: Section 2 reviews the relevant research progress in MTD decision-making methods; Section 3 details the Stackelberg-FlipIt model and the multi-agent WoLF-PHC algorithm; Section 4 validates the feasibility and effectiveness of the proposed method through a case study on IP address dynamic hopping against scanning attacks; Section 5 concludes the paper and discusses future directions for improvement.

2 Related Work

Decision-oriented MTD research focuses on two pivotal questions: "*What to move*" and "*When to move*". Current studies predominantly employ game theory—a mathematical framework for analyzing strategic interactions among rational agents [11]—due to two inherent characteristics of MTD scenarios:

1. **Resource Antagonism:** Attackers exploit system vulnerabilities to expand breach surfaces, while defenders constrain exposures via dynamic configuration shifts (e.g., randomization, diversification) [12].
2. **Interdependent Decision-Making:** The efficacy of adversarial strategies hinges on mutual behavioral adaptations [13].

These characteristics of MTD attack-defense interactions align with the features of game theory. Consequently, a significant number of game-theoretic approaches have been employed to develop MTD solutions: using game theory to model specific MTD attack-defense processes, proving equilibrium convergence, and ultimately deriving equilibrium-based game strategies [14]. Relevant research can be primarily categorized as follows:

1. **Game Theory-Based MTD Spatial Decision-Making Methods:** MTD spatial decision-making methods are divided into five main categories based on "*what to move*": instruction layer, data layer, network layer, platform layer, and runtime environment layer. For example, Ge et al. [15] proposed a game-theoretic MTD approach based on server migration and user service mapping to enhance system real-time performance and throughput elasticity. Carter et al. [16] utilized game theory to derive optimal migration strategies across platforms. Manadhata et al. [17] introduced a two-player stochastic game model, employing the concept of subgame perfect equilibrium to determine optimal MTD strategies based on attack surface diversification. However, due to the single-trigger mechanism of these methods, which only initiate countermeasures upon detecting an attack signal, they often leave room and time for attackers to deploy C&C infrastructure. Further temporal methodology refinements remain imperative.

2. **Game Theory-Based MTD Temporal Decision-Making Methods:** Based on "*when to move*", MTD temporal decision-making can be classified into three main categories: time-driven MTD, event-driven MTD, and hybrid time and event-driven MTD.
   - In time-driven MTD decision-making, Li et al. [5] modeled the defender's joint migration and temporal decisions as a semi-Markov decision process. However, their approach is based on a static Stackelberg game, which significantly deviates from real-world network attack-defense dynamics. Moreover, the derived temporal strategies consider only a single factor, achieving only approximate optimality.
   - In event-driven MTD decision-making, most existing MTD spatial decision methods and few temporal decision methods rely on specific attack events to trigger defenses. Zhang et al. [6] established an MTD temporal decision-making framework based on the Stochastic Markov Differential Game, which characterizes the continuous-time randomness triggered by the strategy through the Itō process, focusing on the multi-stage MTD offensive-defensive process and solved the equilibrium by dynamic programming equation. However, the game is under the condition of perfect information, which can hardly be applied to real-world attack and defense process. This kind of method suffers from low accuracy in attack event identification, potential misjudgments, and the risk of defensive responses lagging behind attack deployments. Consequently, such methods fail to ensure both low overhead and security for the maintained systems.
   - In hybrid time and event-driven MTD decision-making, Tan et al. [4] proposed an MTD temporal model based on multidimensional transitions of the attack surface. They then integrated this model with the system security state evolution of the FlipIt game [18] to establish an MTD spatiotemporal decision model. They used the decision model to derive an optimal spatiotemporal defense strategy selection algorithm. However, this algorithm relies on mathematical difference methods for solving and cannot output adaptive decisions based on environmental changes.

In summary, while game theory-based MTD methods have made significant progress in both spatial and temporal decision-making, challenges remain in terms of adaptability, accuracy, and real-time responsiveness. Further research is needed to address these limitations and enhance the practical applicability of MTD solutions. In Table 1, we compare our methods with existing MTD temporal decision-making methods in order to distinguish our game model and equilibrium algorithm.

Table 1: Comparison with existing MTD temporal decision-making methods

References	Information	Game category	Equilibrium algorithm	Payoff configuration	Temporal decision pattern
Li et al. [5]	Imperfect	Bayesian-Stackelberg game	Min-Max	Migration cost	Time-driven
Zhang et al. [6]	Perfect	Markov differential game	Dynamic programming	Attack surface resource rate and cost	Event-driven
Tan et al. [4]	Perfect	FlipIt game	Saddle point	Time and cost	Hybrid Time and Event-driven

(Continued)

Table 1 (continued)

References	Information	Game category	Equilibrium algorithm	Payoff configuration	Temporal decision pattern
Our method	Imperfect	Stackelberg-FlipIt game	Multi-agent WoLF-PHC	Security, performance, and affordability	Hybrid Time and Event-driven

3 Model and Methodology

In the real network attack and defense, there is a time series dependence and mutual stimulation of the actions of the two sides: (i) defense triggers may expose defense logic and induce attackers to adjust their strategies, and (ii) attack heuristics may reveal blind spots of the defense system and drive defense strategy iterations. However, existing MTD temporal decision methods often rely on simple random triggers or fixed intervals, neglecting the above dynamic interactions between attackers and defenders, leading to an approximately static defense logic: MTD actions are not directly coupled with the attack behavior, and cannot be optimized based on the attacker's real-time tentative feedback. Besides, information asymmetry is not exploited. The defender does not strategically hide the temporal features of MTD decisions, which reduces the learning cost of the attacker. To address this, we propose a Stackelberg-FlipIt game model, where the defender, as the "leader", plans defenses in advance, and the attacker, as the "follower", adjusts strategies dynamically. This model better reflects real-world scenarios, enhancing defense precision and effectiveness.

We also extend a multi-agent reinforcement learning framework. Both attackers and defenders use reinforcement learning to optimize their temporal strategies. Defenders minimize costs and maximize effectiveness, while attackers maximize rewards. This approach balances dynamic behaviors and fine-tunes network attack-defense processes.

The network attack-defense processes discussed in this paper are constructed based on the Cyber Kill Chain model to establish a Stackelberg-FlipIt game in cyberspace. Both attackers and defenders are intelligent agents utilizing reinforcement learning algorithms, capable of making decisions by observing the environment. The multi-agent game relies on reinforcement learning algorithms to solve for the game's equilibrium points [19]. Section 3.1 models the Stackelberg-FlipIt game, defining and explaining each component of the model. Section 3.2 introduces the MTD decision-making method based on this game.

3.1 Stackelberg-FlipIt Game Model

In MTD temporal decision methods, dynamically adjusting the triggering timing of defense strategies is a core challenge. Existing MTD approaches often assume symmetric knowledge of network states between attackers and defenders, neglecting the process by which attackers gradually learn defense thresholds through trial and error. This assumption makes it difficult to precisely control the timing of defense strategies, thereby compromising defense effectiveness and system performance. To address this issue, we introduce a belief factor B, capturing the estimation differences between attackers and defenders regarding defense thresholds. The design is based on:

1. **Asymmetric Cognitive Modeling.** Defenders know the actual thresholds but estimate attackers' perceptions, while attackers iteratively approximate thresholds through trial and error. The belief factor dynamically reflects these differences, enabling precise defense timing.

2. **Dynamic Learning and Updating.** The belief factors are updated as the attack-defense process evolves. Defenders adjust estimates by observing attackers, while attackers refine estimates through feedback. This allows defense strategies to adapt to changing conditions, enhancing system adaptability.
3. **Optimized Defense Timing.** Using the belief factor, defenders can trigger strategies before attackers fully learn thresholds, minimizing attack surface exposure. Trigger frequency can also be adjusted to balance defense effectiveness and system performance.

By introducing the belief factor, the proposed MTD temporal decision method effectively addresses the following issues:

1. **Precise Control of Defense Timing.** It quantifies cognitive differences, enabling dynamic adjustments to avoid premature or delayed triggers.
2. **Dynamic Characterization of Attack-Defense Interactions.** Its updates reflect strategic adjustments, capturing the dynamics of network attack-defense.
3. **Balancing Defense Effectiveness and Performance Overhead.** Defense strategies based on the belief factor ensure effective protection while minimizing system performance overhead, thereby improving the overall efficiency of the MTD system.

We propose a Stackelberg-FlipIt game under the leader-follower paradigm, modeled as a 9-tuple $< P, AS, S, T, A, B, R, C, \pi >$ to comprehensively account for real-world network attack-defense parameters and variables.

- $P = \left\{P^D, P^A\right\}$ denote the participants in the attack-defense game, where P^D represents the defender and P^A represents the attacker.
- AS denotes the common resource contested by both parties in the attack-defense game, i.e., the attack surface, which represents the exploitable surfaces in the system that attackers can discover and utilize. In the game, attacker aim to take control of it and then compromise it, while defenders aim to shift it and protect it from attackers' detection. For convenience, AS_t^P represents the AS controlled by participant P at time t.
- $S_t = \left\{S_t^{norm}, S_t^{fra}, S_t^{dam}, S_t^{rec}\right\}$ are the possible network states at time t. Each network state $S = (s_1, s_2, s_3)$, where s_1 denotes the attack frequency, s_2 denotes the system load, and s_3 denotes the AS exposure time. The transition of each state is shown in Fig. 1.
    - S^{norm}: The network is in a *normal* state, with no detected attack behaviors or potential threats. In this state, the attack frequency is low, the system load remains within normal ranges, and the AS is either not exposed or has minimal exposure time.
    - S^{fra}: The network is in a *fragile* state, indicating the presence of potential threats and a possible imminent attack. In this state, the attack frequency gradually increases, the system load exhibits abnormal fluctuations, and the exposure time of the AS is prolonged.
    - S^{dam}: The network is in an under-*damage* state, where some systems or services have already been compromised. In this state, the attack frequency is high, the system load has significantly increased, and the exposure time of the AS is prolonged.
    - S^{rec}: The network is in a *recovery* state, where the attack has been mitigated, and the system is gradually returning to normal operation. In this state, the attack frequency decreases, the system load is progressively restored to normal levels, and the exposure time of the AS is reduced.

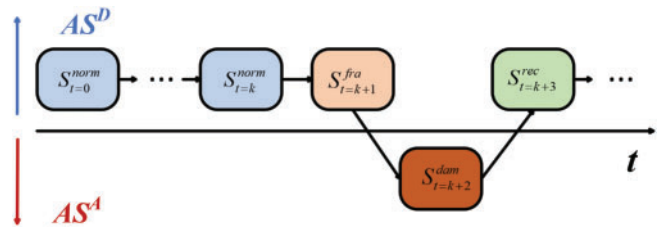

Figure 1: State transitions in the Stackelberg-FlipIt game

The conditions for classifying the four states are as follows:

$$\begin{cases} S^{norm} = \{S \mid s_1 \le \theta_1, s_2 \le \theta_2, s_3 \le \theta_3\} \\ S^{fra} = \{S \mid s_1 > \theta_1, s_2 \le \theta_2, s_3 \le \theta_3\} \\ S^{dam} = \{S \mid s_1 > \theta_1, s_2 > \theta_2, s_3 > \theta_3\} \\ S^{rec} = \{S \mid s_1 \le \theta_1, s_2 > \theta_2, s_3 \le \theta_3\} \end{cases} \tag{1}$$

where $\Theta = (\theta_1, \theta_2, \theta_3)$ is the vector of thresholds for $S = (s_1, s_2, s_3)$. These three thresholds are only transparent to the defender, while the attacker can only gradually approximate them through continuous exploration. This setup aligns more closely with real-world network attack-defense dynamics. Whether these thresholds are exceeded depends on the actions of both the attacker and the defender. For example, a state transition $S_{t-1}^{fra} \to S_t^{dam}$ hints that at time t *AS* is FLIPPED and controlled by the attacker. This aspect will be further elaborated in the subsequent description of actions and the algorithm.

- $T = T^D + T^A$, which denotes the total time required for the attack-defense game, which is the sum of the total time the attacker controls the *AS* ($T^A = T^{dam}$) and the total time the defender controls the *AS* ($T^D = T^{norm} + T^{fra} + T^{rec}$). To simplify the analysis, we assume that the attack-defense game unfolds within a finite and discrete time frame, i.e., $T = \{t_1, t_2, \ldots, t_i\}, i \in N^+$.
- $A = \{A^D, A^A\}$ denote the actions in the attack-defense game. The attacker's action set is represented as $A^A = \{a_1^A, a_2^A, \ldots, a_m^A\}, m \in N^+$ which, for simplicity, is categorized into three levels, as shown in Table 2. Similarly, the defender's action set is represented as $A^D = \{a_1^D, a_2^D, \ldots, a_n^D\}, n \in N^+$, and categorized into three levels. At any discrete time t, both the attacker and the defender may take actions to gain control of the *AS*.

Table 2: The relationship among action levels, states, and costs

Action level	For attacker	For defender
Low	High attack frequency and low costs	Low system load, high *AS* exposure time, and low costs
Medium	Medium attack frequency and medium costs	Medium system load, medium *AS* exposure time, and medium costs
High	Low attack frequency and high costs	High system load, low *AS* exposure time, and high costs

- Belief factor $B^P = \left(b_1^P, b_2^P, b_3^P\right)$ represents the participants' estimated values of the thresholds Θ perceived by the attacker, as shown in Table 3. It is a crucial factor influencing the temporal decisions of participants in the Stackelberg-FlipIt game. As the game progresses, B^A gradually approaches the actual values.

Therefore, for the attacker, the primary goal is to quickly converge B^A to the actual values, as mastering the true thresholds enables prolonged control over the *AS*. For the defender, although the thresholds are transparent, waiting until the attacker learns the actual values before taking action would prolong the loss of control over *AS*, potentially leading to defense failure. Thus, the defender also needs to estimate B^A, denoted as B^D. The defender determines the timing of strategy triggers based on $\Delta B^D = \left|\Theta - B^D\right|$, which represents the difference between the defender's estimated network state values and the attacker's estimated thresholds. This ensures targeted and timely implementation of defense measures.

Table 3: Initial and updated values of belief factor

Belief factor	Initial value	Update	Formula
B^A	Random values or estimates based on prior knowledge	by trial and observation on defender's reaction	$b_i^A \leftarrow b_i^A + \eta_i^A \cdot \left(\theta_i - b_i^A\right)$ (2)
B^D	Known threshold	by observation on attacker's reaction and network	$b_i^D \leftarrow b_i^D + \eta_i^D \cdot \left(\theta_i - b_i^D\right)$ (3)

Note: $\left(\eta_1^P, \eta_2^P, \eta_3^P\right)$ are the learning rate of participants' belief factor, and defaulted as (0.1, 0.1, 0.1).

- $R = \left\{R^D, R^A\right\}$ denote the reward functions, satisfying:

$$R^P = w_1 \cdot R^{SEC} + w_2 \cdot R^{PERF} + w_3 \cdot R^{AFF} - BE \tag{4}$$

where w_1, w_2, and w_3 are the weights that satisfy $w_1 + w_2 + w_3 = 1$. *BE* is the belief error, satisfying

$$\begin{cases} BE^A = \sum_i b_i^A - \theta_i \\ BE^D = \sum_i b_i^D - b_i^A \end{cases} \tag{5}$$

R mainly rely on three categories of reward evaluation R^{SEC}, R^{PERF}, and R^{AFF}, which evaluate the security, performance, and affordability respectively. The design of *BE* is primarily aimed at guiding strategy updates. In our presumption, the weights are configurable, so that our method can be applicable to various MTD systems, e.g., a lightweight system that aims to offer modest security and system performance with the least cost, or, a large defense system that pursues as much as possible a highly secure defense without focusing much on cost. This design is motivated by the following two reasons:

1. A multi-dimensional evaluation framework provides a more holistic assessment of strategy effectiveness, capturing various aspects of performance, security, and cost.
2. Existing MTD decision-making methods often lack clear and well-defined reward structures, which limits their practical applicability.

Clarification of reward design in our model addresses it by designing a robust and transparent reward function considering former evaluation metrics of MTD technologies over the past decade. These metrics ensure that the reward function is both comprehensive and grounded in established research, enhancing the practicality and effectiveness of the proposed method.

Assumption 1: The values of each component in the reward function are normalized to the range [0,1]. This design choice is justified because the evaluations underlying the reward function are all derived from probabilities or ratios, which naturally fall within this range.

- R^{SEC}
  - For defender: Availability Probability

To quantify the effectiveness of the defense system in terms of security, we use AP (Availability Probability) [20,21] to measure the probability that the defender controls the AS over a period T. The formula is defined as:

$$AP = \frac{T^D}{T} \tag{6}$$

  - For attacker: Mean Time to Compromise Ratio

$MTTCR$ (Mean Time to Compromise Ratio) [8,22] represents the ratio of the time the attacker controls the AS to the total time, which is essentially the probability of the attacker controlling AS. This metric indirectly reflects the attacker's ability to fully access and exploit AS.

$$MTTCR = \frac{T^A}{T} \tag{7}$$

- R^{PERF}
  - For defender: Escape Probability

EP (Escape Probability) [10,23] represents the probability that the defender transfers the AS after detecting that the attacker has gained control of it. This metric emphasizes the system's ability to promptly defend itself upon detecting an attack. The formula is expressed as:

$$EP = p\left(S^{norm}|S^{fra}\right) + p\left(S^{rec}|S^{dam}\right) \tag{8}$$

where $p\left(S_t'|S_t, a_t^D, a_t^A\right)$ represents the probability of reaching state S_t' from state S_t in which the defender takes action a^D and the attacker takes action a^A.

  - For attacker: Successful Attack Probability

SAP (Successful Attack Probability) [8,24] represents the probability that the attacker, after gaining control of the AS, proceeds with the next attack while avoiding detection and triggering an AS transfer by the defender. This metric primarily reflects the attacker's ability to conduct stealthy attacks. The formula is expressed as:

$$SAP = p\left(S^{dam}|S^{fra}\right) + p\left(S^{dam}|S^{dam}\right) \tag{9}$$

- R^{AFF}, the same as $C = \{AC, DC\}$
  - For defender: Defense Cost

DC (Defense Cost) represents the cost incurred by the defender when executing defense strategies to counter attacks. This metric measures the expenses associated with defensive actions. Due to the multi-stage nature of the attack-defense design, this indicator indirectly incorporates the time costs of both parties into consideration. Since the costs of strategies vary, no specific formula is provided; instead, the costs are comprehensively calculated after each action.

  - For attacker: Attacker Cost

Similar to DC, AC (Attack Cost) measures the overhead incurred by the attacker when executing attack strategies. Likewise, it is only necessary to comprehensively calculate the costs after each action.

- $\pi^P\left(S_t, a_t^P\right)$ denotes the policy function, representing the probability that participant P takes action a_t^P at a given time t and state S_t. Its update formula is:

$$\pi^P\left(S_t, a_t^P\right) \leftarrow \pi^P\left(S_t, a_t^P\right) + \alpha^P \cdot \Delta\pi^P\left(S_t, a_t^P\right) \tag{10}$$

where α^P is the policy learning rate for participant P. The learning rate dynamically adjusts based on policy performance to adapt to rapidly changing environments [25]. When the current policy outperforms the average policy, the learning rate decreases as α_{win}^P to reduce the magnitude of policy adjustments. Conversely, when the current policy underperforms, the learning rate increases as α_{learn}^P to quickly learn better strategies, improving the responsiveness and precision of MTD temporal decisions. This fast-slow adaptation balances the trade-off between the effectiveness and cost of temporal strategies. Its selective formula is:

$$\alpha^P = \begin{cases} \alpha_{win}^P & \text{if } \sum_{a_t^P} \pi^P\left(S_t, a_t^P\right) R^P\left(S_t, a_t^P\right) > \sum_{a_t^P} \overline{\pi}^P\left(S_t, a_t^P\right) R^P\left(S_t, a_t^P\right) \\ \alpha_{learn}^P & \text{otherwise} \end{cases} \tag{11}$$

where $\Delta\pi^P\left(S_t, a_t^P\right)$ is the policy gradient:

$$\Delta\pi^P\left(S_t, a_t^P\right) = R^P\left(S_t, a_t^P\right) - \sum_{a_{t+1}^P} \pi^P\left(S_{t+1}, a_{t+1}^P\right) R^P\left(S_{t+1}, a_{t+1}^P\right) \tag{12}$$

3.2 Multi-Agent WoLF-PHC for Stackelberg-FlipIt-MTD Scenarios

Given the setting of the game model, we model the MTD attack-defense process as a multi-stage Stackelberg-FlipIt game, incorporating the MTD scenario into the attack-defense game evolution through the asymmetric projections of both parties. Finally, the proposed multi-agent WoLF-PHC algorithm is applied to solve the MTD temporal decision-making method.

3.2.1 Stackelberg-FlipIt Game for MTD

For the attacker, the knowledge about the system network evolves as the penetration deepens. The attacker's belief factor about the network system continuously changes, and the attacker can only perceive and attack the network system through continuous probing and observation of potential defense strategies. Moreover, due to the dynamic movement of the *AS* in MTD, this local projection may also deviate from objective reality. In contrast, the system administrator (defender) has absolute physical control over the system, along with sufficient security assessment and penetration testing capabilities.

In MTD scenarios, the existence of unknown vulnerabilities does not diminish the defender's absolute cognitive advantage over the attacker. However, since MTD typically involves high defense costs, the set of movable strategies is generally limited, and defensive measures are not taken in every game. Therefore, MTD temporal decision-making methods are crucial. The defender needs to adjust the estimated belief factor of the attacker by observing the attacker's behavior, enabling precise and effective transfer of the attack surface to resist attacks while minimizing costs and avoiding aimless actions.

In summary, in MTD scenarios, the defender can proactively adjust system security configurations, while the attacker can only passively respond to these dynamic changes. Given this, we model the proactive and reactive relationship between the attacker and defender using the Stackelberg-FlipIt game model, defining the defender as the "leader" and the attacker as the "follower". Each game stage is divided into two sub-stages:

1. Defender's turn: The defender selects one of the MTD movable states or chooses not to act as a defense strategy based on B^D, aiming to hinder or delay the attacker's updates of B^A and maintain control

over *AS* for as long as possible within the time period *T*. In most cases, the defender does not need to trigger defense strategies aimlessly. Relevant experiments [26,27] have shown that even when the attacker perceives potential security configuration changes (even if none actually occur), it can increase cognitive load and delay or prevent attack activities.

2. Attacker's turn: The attacker initiates attacks based on the initial B^A and continuously adjusts it during the process to avoid detection and triggering *AS* transferring by the defender, also aiming to gain control over *AS* for as long as possible within the time period *T*.

It is evident that the differences in the estimation of the belief factor cause the attacker and defender to operate within entirely different frameworks during the game. This asymmetric interaction is a key feature of the proposed Stackelberg-FlipIt game model, enabling a more realistic and dynamic representation of MTD attack-defense processes.

3.2.2 Multi-Agent WoLF-PHC for MTD Temporal Decision-Making

In the Stackelberg-FlipIt game, the participants' actions occur in a specific sequence. Typically, the "leader" guides the game toward a globally optimal state for themselves. Leveraging cognitive advantages, the defender can conduct multi-stage risk analysis on *AS* to assess the impact of various MTD strategies on risk, which serves as the basis for the "leader's" decisions. Meanwhile, the attacker, as a rational "follower", selects the optimal attack strategy based on their current cognition. The Stackelberg game mechanism is implemented through a subsequential action signal. When the *signal* is 0, the defender observes and selects an action, and the *signal* changes to 1. Upon receiving the *signal* as 1, the attacker observes and selects an action, and the *signal* changes back to 0. This cycle repeats until the game period *T* ends.

The multi-agent WoLF-PHC algorithm continuously optimizes strategies while minimizing environmental influences to achieve faster learning speeds and better performance. Based on its characteristics, we design Algorithm 1 to solve the MTD temporal decision-making method.

Algorithm 1: Multi-agent WoLF-PHC for Stackelberg-FlipIt MTD

Input: $< P, AS, S, T, A, B >$, episode_num, batch_size
Output: π^{D*} and π^{A*}
initialize π^D and π^A, $\overline{\pi}^D$ and $\overline{\pi}^A$, α^D and α^A, $\left(\eta_1^D, \eta_2^D, \eta_3^D\right)$ and $\left(\eta_1^A, \eta_2^A, \eta_3^A\right)$
while episode_num > 0 **do**
  for k = 1 to batch_size **do**
    for t = 0 to T **do**
      $S_0 \leftarrow S^{norm}$, T^D, T^A= 0
      if *signal* = 0
        defender takes α_t^D according to $\pi^D\left(S_t, a_t^P\right)$ and ΔB^D
        $S_t \rightarrow S_{t+1}$
        if S_{t+1} belongs to S^{norm}, S^{fra}, and S^{rec}
          $T^D = T^D + 1$
        updates b_i^D, BE^D and ΔB^D
        observes $R^D\left(S_t, a_t^D\right)$
        calculates $\Delta\pi^D\left(S_t, a_t^D\right)$
        updates $\pi^D\left(S_t, a_t^D\right)$ and $\overline{\pi}^D\left(S_t, a_t^D\right) \leftarrow (1-\beta)\,\overline{\pi}^D\left(S_t, a_t^D\right) + \beta \cdot \overline{\pi}^D\left(S_t, a_t^D\right)$

(Continued)

Algorithm 1 (continued)

adjusts $\alpha^D = \begin{cases} \left(1-\Delta B^D\right)\cdot\alpha_{win} & \text{if } R^D > \overline{R}^D \\ \left(1-\Delta B^D\right)\cdot\alpha_{learn} & \text{otherwise} \end{cases}$
$t \leftarrow t+1$
$signal = 1$
else
attacker takes α_t^A according to $\pi^A\left(S_t, a_t^A\right)$
$S_t \rightarrow S_{t+1}$
if S_{t+1} belongs to S^{dam}
$T^A = T^A + 1$
updates b_i^A and BE^A
observes $R^A\left(S_t, a_t^A\right)$
calculates $\Delta\pi^A\left(S_t, a_t^A\right)$
updates $\pi^A\left(S_t, a_t^A\right)$ and $\overline{\pi}^A\left(S_t, a_t^A\right)$
adjusts $\alpha^A = \begin{cases} \alpha_{win} & \text{if } R^A > \overline{R}^A \\ \alpha_{learn} & \text{otherwise} \end{cases}$
$t \leftarrow t+1$
$signal = 0$
end for
end for
end while

4 Experiment

In this section we aim to validate the effectiveness of our proposed method through experiments on IP address dynamic hopping (short as IP hopping) against scanning attacks. The experimental design focuses on evaluating algorithm performance, optimizing strategy triggering timing, and assessing practical application value, ensuring that the proposed model and algorithm are not only theoretically innovative but also feasible in real-world scenarios.

We use simulation experiments to demonstrate the superior performance of the proposed algorithm. Section 4.1 details the design of the simulation environment, including the strategies of both attackers and defenders, as well as other critical elements. Section 4.2 compares and selects hyperparameters for the algorithm. Section 4.3 compares our algorithm with other classical algorithms and provides experimental analysis.

4.1 Experimental Setup

The IP address is a critical target for attackers launching scanning attacks, DDoS attacks, and other types of subsequent attacks. Therefore, IP hopping defense [28] is one of the key technologies in MTD. Defenders use MTD techniques to randomly change host IP addresses, achieving attack evasion. Existing research has shown that IP hopping is an effective method against scanning attacks. However, blindly triggering IP hopping without perceiving or accurately perceiving the attacker's behavior can lead to two issues:

1. It may fail to achieve the desired defensive effect and could provide attackers with more information to counteract.
2. It may result in unnecessary system resource overhead, leading to inefficiencies.

The experimental algorithm is run on OpenAI Gym, which provides a multi-agent game simulator for simulating complex network attack-defense scenarios. The experimental environment is deployed in an SDN network controlled by an OpenFlow controller [29]. We use Mininet [30] to create an OpenFlow switch network as the data plane of the SDN, while the control plane is constructed and deployed using the Ryu platform to implement the IP address dynamic defense system, as shown in Fig. 2. In this network, 400 hosts are randomly distributed across 20 subnets, with the subnets connected through OpenFlow switches.

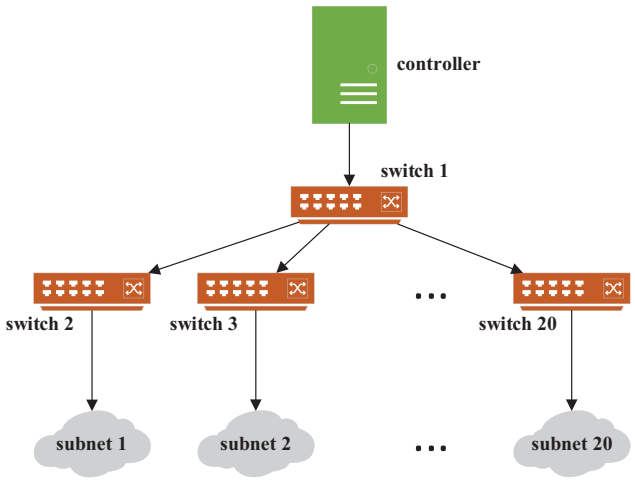

Figure 2: Virtual network topology on Mininet

The experiment simulates real-world scanning attacks and IP hopping defense techniques. The network system state $S = (s_1, s_2, s_3)$ involved information in the attack-defense process that includes attack frequency, system load, and exposure time of real IP addresses. The system load is influenced by both the attacker and the defender, as the defense system may stop services after receiving a large number of data packets, thereby reducing QoS. The individual threshold parameters of the states are normalized as a middle vector with $\Theta = (0.5, 0.5, 0.5)$ to simplify the multi-agent reinforcement learning process.

The attacker employs a coordinated scanning approach, where a certain number of hosts simultaneously initiate scanning attacks by sending probe packets to the IP addresses of the defense network. In the experiment, we assume 50 to 200 scanning hosts targeting 400 defense network hosts. The attacker is assumed to know the IP address resource pool of the defense network and can divide the entire IP address space of the defense network, evenly distributing it among the scanning hosts. These hosts perform random scans on their assigned IP address segments with a uniform distribution. The attacker's goal is to detect the IP addresses currently in use by online hosts to build an attack list, providing a foundation for subsequent attacks. Cautious attackers typically avoid rescanning already scanned IP addresses to minimize the likelihood of detection failure [31]. In the experiment, the scanning attack is designed to be propagative, meaning that detected online hosts are compromised and become new scanning hosts. The unscanned IP address ranges are redistributed among the new set of scanning hosts, with an infection time step of 1.

Each scanning host has two attribute characteristics:

1. Frequency of Scanning Packet (*FSP*): The average number of probe packets sent per second.
2. Proportion of Scanning Packet (*PSP*): The ratio of probe packets to the total packets sent.

Based on these two characteristics, the attacker's scanning behavior can be classified into three levels, corresponding to the action classification in the Stackelberg-FlipIt game model, as shown in Table 4.

Table 4: Attacker's action level

Scanning level	*FSP*	*PSP*	*Cost*
Low	0~0.5/s	0%~20%	0.1
Medium	0.5~1/s	20%~40%	0.5
High	1~2/s	40%~80%	1

The defense mechanism assigns each host within the system both a fixed real IP address and a time-varying virtual IP address. IP address alteration operates via two distinct modes: periodic time-triggered IP hopping (low-level defense), and event-driven IP hopping (mid-level and high-level defense). The periodic mode introduces fault tolerance for defensive event adjudication, providing a buffer mechanism to mitigate security risks when event determination errors occur. IP hopping behaviors are hierarchically classified into three tiers based on two metrics as systematically detailed in Table 5:

1. Hopping Frequency (*HF*): The frequency of IP address hops.
2. Hopping Range (*HR*): The span of address space covered during hopping.

Table 5: Defender's action level

IP hopping Level	*HP*	*HR*	*Cost*
Low	0~0.05/s	0%~25%	0.5
Medium	0.05~0.1/s	25%~50%	1
High	0.1~0.2/s	50%~75%	2

4.2 Hyperparameters Configuration

The experiment employs 1000 episodes with $T = 100$ steps per episode and 100 independent trials. Hyperparameter configurations are summarized in Table 6, with the following design rationales:

- Learning rate α is parameterized with two tiers:
  - α_{win} must be sufficiently large to ensure rapid convergence when policy performance is favorable but must avoid excessive magnitudes that induce destabilizing oscillations. Usually, $\alpha_{win} \in [0.1, 0.5]$.
  - α_{learn} should remain small to prioritize cautious adjustments during suboptimal policy execution, preventing abrupt deviations from near-optimal strategies, while avoiding undersized values that hinder convergence speed. Usually, $\alpha_{learn} \in [0.01, 0.1]$.
- Smoothing parameter β (average policy update) functions as a learning rate for updating the average policy. A smaller β emphasizes historical policy performance through slower updates, while a larger β prioritizes recent policy behaviors with faster adaptation. Empirically, $\beta \in [0.01, 0.1]$.
- Reward weight coefficient w directly governs optimization objectives and algorithmic performance. Weight assignments must jointly consider:
  - Security (high priority, as the core objective of MTD), $w_1 \in [0.4, 0.6]$.
  - Operational performance (moderate weight to ensure user experience and service continuity), $w_2 \in [0.2, 0.3]$.
  - Defense overhead (constrained to reasonable levels to prevent resource overconsumption), $w_3 \in [0.1, 0.2]$.

Table 6: Values configuration of learning rate, smoothing parameter, and reward weight coefficient

No.	α_{win}	α_{learn}	β	w_1	w_2	w_3
1	0.5	0.1	0.1	0.6	0.3	0.1
2	0.2	0.05	0.05	0.5	0.3	0.2
3	**0.2**	**0.1**	**0.1**	**0.6**	**0.2**	**0.2**
4	0.1	0.05	0.05	0.5	0.3	0.2

Through a series of preliminary experiments, we rigorously validated the guaranteed convergence of the algorithm within 1000 episodes. These experiments further enabled the determination of optimal hyperparameter configurations to maximize algorithmic performance, as shown in Fig. 3.

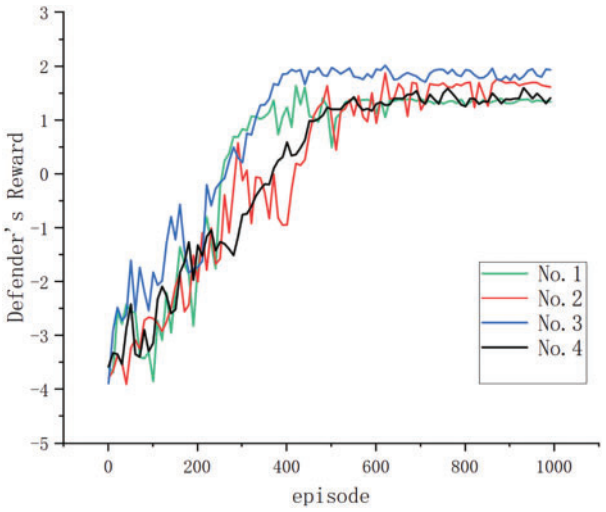

Figure 3: Defender's reward after 1000 episodes under different hyperparameters

As demonstrated in Fig. 3 and Table 7, Config. 3 (0.2, 0.1, 0.1, 0.1, 0.6, 0.2, 0.2) achieves the optimal performance, exhibiting both the fastest convergence and the highest final policy reward.

Table 7: Comparison of different configurations

No.	Convergence episode	Policy reward
1	453	1.387
2	587	1.563
3	**421**	**1.970**
4	504	1.475

Configs. 1 and 3 exhibit accelerated convergence due to their larger α_{win} and α_{learn}, enabling rapid stabilization of near-optimal policies. Conversely, Configs. 2 and 4 converge more slowly and prolong policy refinement. Despite its rapid convergence, Config. 1 yields the lowest reward because its excessive allocation to w_1, which leads to the time-occupancy weight prioritizing adversarial engagement duration at the expense

of strategy cost optimization, of which the imbalance diminishes the overall reward efficacy. Hence, Config. 3 achieves superior rewards by balancing weights to harmonize security, performance, and affordability objectives. In actual application, the configuration of reward weight coefficients can be manually changed regarding what the goal of the task is, e.g., a lightweight defense system aiming at modest security and performance with the least cost. Here, we mainly focus on the balance among security, performance, and affordability with the weight coefficient being (0.6, 0.2, 0.2).

4.3 Experiment Analysis

In the following experiments, by comparing three IP hopping methods—OF-RHM [32], SEHT [33], and DDS [34]—we comprehensively analyze the results from three perspectives: security, performance, and affordability. As for security analysis, Host Survival Ratio (*HSR*) and Host Survival Average Time (*HSAT*) are utilized to evaluate the network security under different algorithms. As for performance and affordability analysis, Actual IP Hopping Frequency (*AHF*) and Network Channel Resource Occupancy (*NCRO*) are utilized.

4.3.1 Security Analysis

The experiment assumes that the attacker performs a uniformly distributed and non-repetitive random scan on the IP address space used by the defense network. The metric *HSR* represents the proportion of remaining hosts H_{surv} that are not detected and included in the attacker's attack list, relative to the total number of hosts H in the defense network [35]. This metric is a crucial indicator of the security performance of the relevant algorithms.

Under the settings of 50, 100, 150, and 200 scanning hosts performing coordinated scans on 400 hosts in the defense network, the comprehensive results are shown in Fig. 4. In a network without MTD deployment, the *HSR* approaches 0 after approximately 100–300 s. The DDS method performs similarly to OF-RHM, while our method and SEHT significantly reduce the decline in survival rate. The final *HSR* of our method consistently remains above 0.5. It is evident that our algorithm performs slightly better than DDS, as our method can precisely determine the timing of defense strategy triggers, guiding the defense network to perform IP hopping to evade scanning attacks.

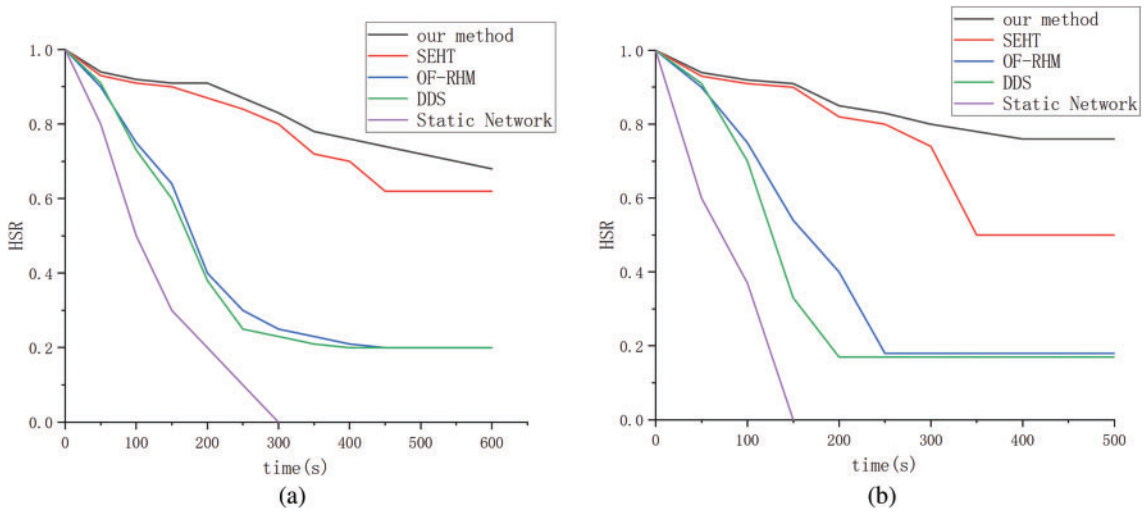

Figure 4: (Continued)

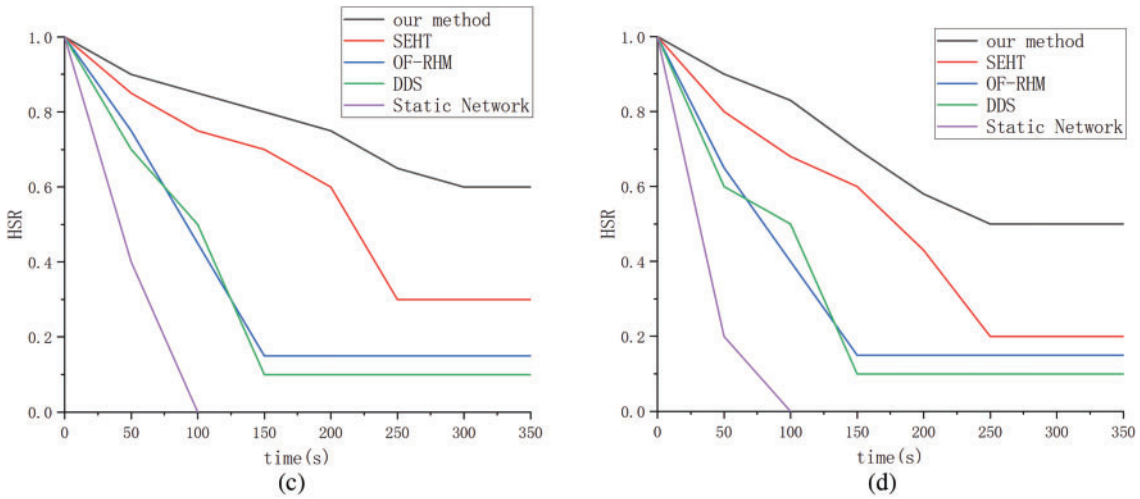

Figure 4: *HSR*s under different scanner numbers. (**a**) *HSR*s under 50 scanners; (**b**) *HSR*s under 100 scanners; (**c**) *HSR*s under 150 scanners; (**d**) *HSR*s under 200 scanners

After a successful scanning attack, the attacker may launch follow-up attacks such as worm propagation, DDoS, or other subsequent attacks based on the Hit List, aiming to gain deeper control over the *AS*. These attacks require a certain amount of time to deploy and achieve their intended effects. Therefore, the longer the *HSAT* of hosts in the attack list, the more opportunities it provides for the attacker. This metric is primarily interpreted as the duration between the scanning host receiving a response packet from a malicious probe and the IP hopping of the defense network host. The smaller this metric, the lower the success rate of the attacker's subsequent attacks [35].

Under the settings of 50, 100, 150, and 200 scanning hosts performing coordinated scans on 400 hosts in the defense network, the comprehensive results are shown in Fig. 5. Compared to other methods, as the number of scanning hosts increases, our method significantly reduces *HSAT*, effectively suppressing the development of subsequent attacks.

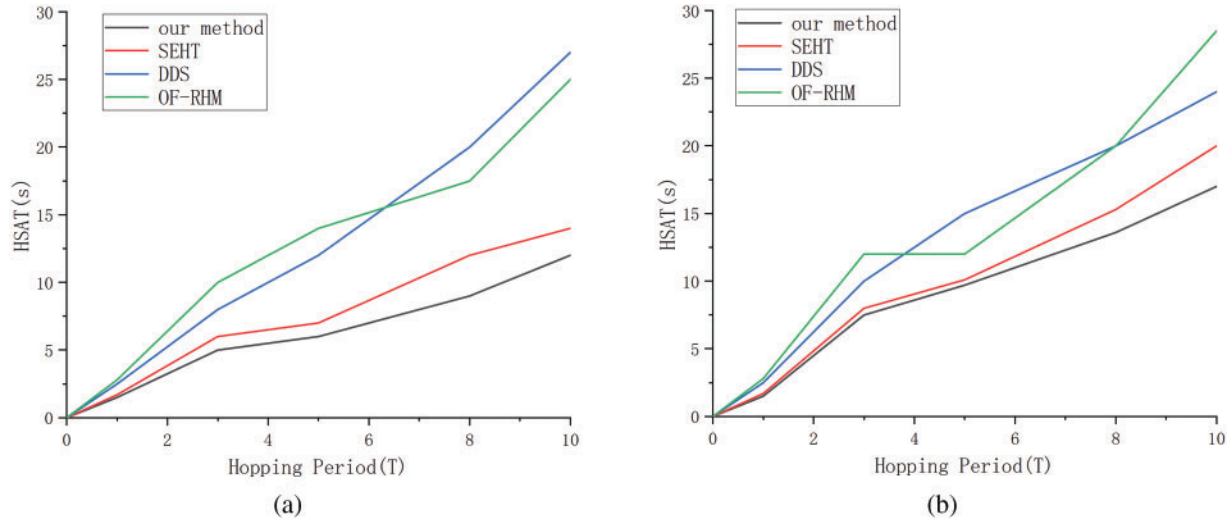

Figure 5: (Continued)

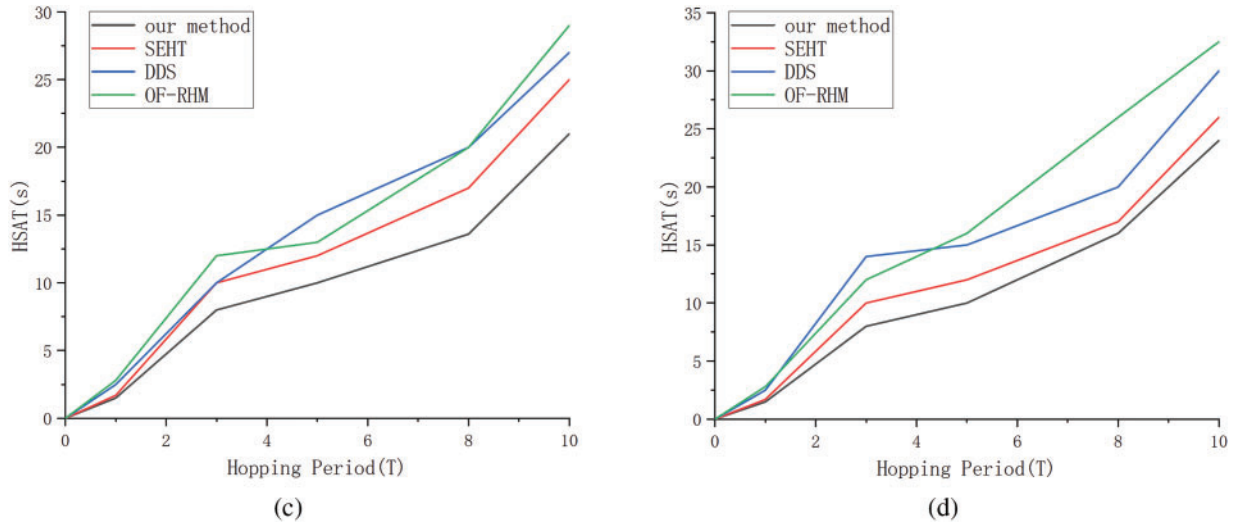

Figure 5: *HSAT*s (s) under different scanner numbers. (**a**) *HSR*s under 50 scanners; (**b**) *HSR*s under 100 scanners; (**c**) *HSR*s under 150 scanners; (**d**) *HSR*s under 200 scanners

4.3.2 Performance and Affordability Analysis

When hosts in the defense network undergo IP hopping, new forwarding flow tables must be promptly configured to maintain normal network services. This process can cause transient high latency, reducing the network QoS of the defense system and impacting normal network operations. Therefore, the flow table configuration rate and IP hopping frequency are critical factors affecting network performance. Since the former is beyond the scope of this paper, we focus on analyzing the *AHF* and *NCRO* metrics. As for *AHF*, A low *AHF* alone does not indicate the quality of the method; it must be analyzed in conjunction with the *HSR*. Specifically, under the same *HSR* condition, a lower *AHF* results in less impact on network QoS and better network performance. Additionally, the *AHF* reflects defense costs due to less triggering.

As shown in Table 8, our method achieves the lowest *AHF* across different *HSR* levels, indicating the lowest defense cost. This is because our method dynamically adjusts the IP hopping strategy trigger frequency based on the belief factor, ensuring that each hopping is effective.

Table 8: *AHF*s (time/s) of different algorithms under different *HSR*s

HSR	OF-RHM	SEHT	DDS	Our method
0.2	0.032	0.027	0.031	0.030
0.4	0.066	0.051	0.061	0.042
0.6	0.123	0.114	0.125	0.098

Real-time sensitive strategies may require the reservation of dedicated channel resources to avoid congestion, which can limit the flexibility of network resource reuse and increase the marginal cost per unit of traffic. Therefore, *NCRO* can intuitively reflect the performance and affordability of the output strategies. The lower NCRO, the temporal strategy can perform better under the same *HSR* condition.

As shown in Table 9, our method outperforms *NCRO* under different security conditions, close to SEHT. Although these temporal strategies may enhance system robustness, their frequent triggering may lead to

a decrease in effective data throughput, essentially converting overhead into bandwidth procurement costs. Therefore, providing modest security with *NCRO* under 150 kb/s [35] is the ideal solution for related methods. Under this circumstance, both SEHT and our method can be applicable.

Table 9: *NCRO*s (kb/s) of different algorithms

HSR	OF-RHM	SEHT	DDS	Our method
0.2	137	101	125	97
0.4	199	135	161	122
0.6	273	196	225	191

In summary, our method outperforms existing approaches in the comparative experiments. We attribute this to: (i) A game model that more closely aligns with real-world network attack-defense processes; (ii) The positive influence of the belief factor on strategy trigger timing; (iii) The decision-making adaptability of the multi-agent WoLF-PHC algorithm; (iv) A comprehensive reward function that guides holistic strategy optimization. These factors collectively contribute to the superior performance of our proposed method.

5 Conclusion

We proposed an innovative MTD temporal decision-making method in the field of MTD. The core components of this method include the novel Stackelberg-FlipIt game and the Multi-agent WoLF-PHC algorithm, which can be mainly concluded as follows:

- Firstly, by introducing the hierarchical decision-making structure of Stackelberg game and combining it with the dynamic process modeling of FlipIt game, we constructed the Stackelberg-FlipIt game model. The concept of belief factor was incorporated to influence the timing and planning of decisions for both attackers and defenders.
- Secondly, the multi-agent framework was integrated into the WoLF-PHC algorithm, enabling both attackers and defenders to autonomously learn and adapt their strategies in dynamic environments, which allowed them to respond to evolving network states and adversarial behaviors in real time.
- Thirdly, for the first time, four key metrics—security, system performance, affordability, and belief error—were integrated into the reward function of multi-agent reinforcement learning, which provided a new perspective for the multi-dimensional quantification of network defense strategies, helping our method output strategies with the strongest overall capabilities.
- Lastly, the effectiveness of the proposed model and algorithm was validated through experiments on IP hopping against scanning attacks. The results demonstrated the algorithm's performance, showing that, with appropriate parameters, the proposed model and algorithm significantly enhanced the adaptability and efficiency of MTD temporal decision-making methods.

In conclusion, the proposed method offers a robust and practical solution for MTD temporal decisions, balancing security, performance, and affordability while adapting to dynamic network environments. Although the model and algorithm proposed in this paper have achieved good results in simulation experiments, there are still some limitations and directions for future work:

- A more adversarial game model: The premise given is that both the attacker and the defense are rational, so the attacker in this paper will not consume defense resources through deception, that is, the "intelligence level" is not high enough. Future research can lead to the construction of more sophisticated attacker models to simulate more realistic cyberattack behaviors.

- Dynamically adjust hyperparameters: The selection of hyperparameters is mainly based on experiments and experience, and in the future, hyperparameters can be dynamically adjusted through automated methods to adapt to different network environments and attack and defense scenarios.
- Actual deployment and testing: Our model and algorithm have not yet been deployed and tested in a real-world network environment, and these methods can be applied to real-world network systems in the future to evaluate their effectiveness and robustness in the real world.

Acknowledgement: Due to the nature of this research, participants of this study agree for their data to be shared publicly as demand.

Funding Statement: This work is funded by National Natural Science Foundation of China No. 62302520.

Author Contributions: The authors confirm contribution to the paper as follows: Study conception and design: Rongbo Sun, Yuefei Zhu, Jinlong Fei; data collection: Rongbo Sun, Zhongyu Guo; analysis and interpretation of results: Rongbo Sun; draft manuscript preparation: Rongbo Sun, Jinlong Fei, Yuefei Zhu. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The data and materials used to support the findings of this study are available from the corresponding author upon request after acceptance.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Cho JH, Sharma DP, Alavizadeh H, Yoon S, Ben-Asher N, Moore TJ, et al. Toward proactive, adaptive defense: a survey on moving target defense. IEEE Commun Surv Tutor. 2020;22(1):709–45. doi:10.1109/comst.2019.2963791.
2. Zhuang R, DeLoach SA, Ou X. Towards a theory of moving target defense. In: Proceedings of the First ACM Workshop on Moving Target Defense; 2014 Nov 7; Scottsdale, AZ, USA.
3. Clark A, Sun K, Poovendran R. Effectiveness of IP address randomization in decoy-based moving target defense. In: 52nd IEEE Conference on Decision and Control; 2013 Dec 10–13; Florence, Italy.
4. Tan J, Zhang H, Zhang H, Hu H, Lei C, Qin Z. Optimal temporospatial strategy selection approach to moving target defense: a FlipIt differential game model. Comput Secur. 2021;108(3):102342. doi:10.1016/j.cose.2021.102342.
5. Li H, Zheng Z. Optimal timing of moving target defense: a Stackelberg game model. arXiv:1905.13293. 2019.
6. Zhang H, Tan J, Liu X, Wang J. Moving target defense decision-making method. In: Proceedings of the 7th ACM Workshop on Moving Target Defense; 2020 Nov 9; Virtual.
7. Cook PR. Limitations and extensions of the WoLF-PHC algorithm [master's thesis]. Provo, UT, USA: Brigham Young University; 2007.
8. Connell W, Albanese M, Venkatesan S. A framework for moving target defense quantification. IFIP Adv Inf Commun Technol. 2017;502:124–38. doi:10.1007/978-3-319-58469-0_9.
9. Xiong X, Zhao G, Wang X. A system attack surface based MTD effectiveness and cost quantification framework. In: Proceedings of the 2nd International Conference on Cryptography, Security and Privacy; 2018 Mar 16–18; Guiyang, China. p. 175–9.
10. Zaffarano K, Taylor J, Hamilton S. A quantitative framework for moving target defense effectiveness evaluation. In: MTD 2015—Proceedings of the 2nd ACM Workshop on Moving Target Defense, Co-Located with: CCS; 2015 Oct 12; Denver, CO, USA. p. 3–10.
11. Barron EN. Game theory: an introduction. Hoboken, NJ, USA: John Wiley & Sons; 2024.
12. Zhu M, Hu Z, Liu P. Reinforcement learning algorithms for adaptive cyber defense against heartbleed. In: Proceedings of the ACM Conference on Computer and Communications Security; 2014 Nov 3–7; Scottsdale, AZ, USA. p. 51–8.

13. Wright M, Venkatesan S, Albanese M, Wellman MP. Moving target defense against DDoS attacks: an empirical game-theoretic analysis. In: MTD 2016—Proceedings of the 2016 ACM Workshop on Moving Target Defense, Co-Located with CCS; 2016 Oct 24–28; Vienna, Austria. p. 93–104.
14. Tan J, Jin H, Zhang H, Zhang Y, Chang D, Liu X, et al. A survey: when moving target defense meets game theory. In: Computer science review. Vol. 48. Amsterdam, The Netherlands: Elsevier Ireland Ltd. 2023. doi:10.1016/j.cosrev.2023.100544.
15. Ge L, Yu W, Shen D. Toward effectiveness and agility of network security situational awareness using moving target defense (MTD). Sens Syst Space Appl VII. 2014;9085:185–93.
16. Carter KM, Riordan JF, Okhravi H. A game theoretic approach to strategy determination for dynamic platform defenses. In: Proceedings of the First ACM Workshop on Moving Target Defense; 2014 Nov 7; Scottsdale, AZ, USA. p. 1–30.
17. Manadhata PK. Game theoretic approaches to attack surface shifting. In: Moving target defense II. New York, NY, USA: Springer; 2012. p. 1–13 doi:10.1007/978-1-4614-5416-8_1.
18. van Dijk M, Juels A, Oprea A, Rivest RL. FlipIt: the game of "stealthy takeover". J Cryptol. 2013;26(4):655–713. doi:10.1007/s00145-012-9134-5.
19. Gao C, Wang Y. Reinforcement learning based self-adaptive moving target defense against DDoS attacks. J Phys Conf Ser. 2021;1812(1):012039. doi:10.1088/1742-6596/1812/1/012039.
20. Cremer F, Sheehan B, Fortmann M, Kia AN, Mullins M, Murphy F, et al. Cyber risk and cybersecurity: a systematic review of data availability. Geneva Pap Risk Insur-Issues Pract. 2022;47(3):698–736. doi:10.1057/s41288-022-00266-6.
21. Prakash A, Wellman MP. Empirical game-theoretic analysis for moving target defense. In: Proceedings of the Second ACM Workshop on Moving Target Defense—MTD '15; 2015 Oct 12; Denver, CO, USA.
22. Wan Z, Cho J-H, Zhu M, Anwar AH, Kamhoua CA, Singh MP. Foureye: defensive deception against advanced persistent threats via hypergame theory. IEEE Trans Netw Serv Manag. 2021;19(1):112–29. doi:10.1109/tnsm.2021.3117698.
23. Ping P, Wang K, Kong D, Chen G. Estimating probability of success of escape, evacuation, and rescue (EER) on the offshore platform by integrating Bayesian network and fuzzy AHP. J Loss Prev Process Ind. 2018;54(4):57–68. doi:10.1016/j.jlp.2018.02.007.
24. Halgamuge MN. Estimation of the success probability of a malicious attacker on blockchain-based edge network. Comput Netw. 2022;219(2):109402. doi:10.1016/j.comnet.2022.109402.
25. Mounjid O, Lehalle C. Improving reinforcement learning algorithms: towards optimal learning rate policies. Math Financ. 2024;34(2):588–621. doi:10.1111/mafi.12378.
26. Ferguson-Walter KJ, LaFon DS, Shade TB. Friend or faux: deception for cyber defense. J Inf Warf. 2017;16(2):28–42.
27. Johnson CK. Decision-making biases in cybersecurity: measuring the impact of the sunk cost fallacy to delay attacker behavior. Tempe, AZ, USA: Arizona State University; 2022.
28. Chang S-Y, Park Y, Babu BA. Fast IP hopping randomization to secure hop-by-hop access in SDN. IEEE Trans Netw Serv Manag. 2019;16(1):308–20. doi:10.1109/tnsm.2018.2889842.
29. McKeown N, Anderson T, Balakrishnan H, Parulkar G, Peterson L, Rexford J, et al. OpenFlow: enabling innovation in campus networks. ACM Spec Interest Group Data Commun. 2008;38(2):69–74.
30. Rodríguez P, Andrea Y. Using mininet for emulation and prototyping software-defined networks. In: 2014 IEEE Colombian Conference on Communications and Computing (COLCOM); 2014 Jun 4–6; Bogotá, DC, Colombia.
31. Aibekova A, Selvarajah V. Offensive security: study on penetration testing attacks, methods, and their types. In: 2022 IEEE International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE); 2022 Apr 23–24; Ballari, India. p. 1–9.
32. Jafar H, Ehab A, Qi D. 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; 2013 Aug 16; Hong Kong, China. New York, NY, USA: ACM Digital Library; 2013.
33. Lei C, Zhang HQ, Ma DH, Yang YJ. Network moving target defense technique based on self-adaptive end-point hopping. Arab J Sci Eng. 2017;42(8):3249–62. doi:10.1007/s13369-017-2430-5.

34. Miao L, Hu H, Cheng G. The design and implementation of a dynamic IP defense system accelerated by vector packet processing. In: Proceedings of the International Conference on Industrial Control Network and System Engineering Research (ICNSER2019); 2019 Mar 15–16; Shenyang, China. New York, NY, USA: Association for Computing Machinery. p. 64–9.
35. Xu X, Hu H, Liu Y, Zhang H, Chang D. An adaptive IP hopping approach for moving target defense using a light-weight CNN detector. Secur Commun Netw. 2021;2021(1):8848473. doi:10.1155/2021/8848473.