Computers & Security 142 (2024) 103871

Contents lists available at ScienceDirect

Computers & Security

journal homepage: www.elsevier.com/locate/cose

A method of network attack-defense game and collaborative defense decision-making based on hierarchical multi-agent reinforcement learning

Yunlong Tang, Jing Sun, Huan Wang *, Junyi Deng, Liang Tong, Wenhong Xu

School of Computer Science and Technology, Guangxi University of Science and Technology, Liuzhou, Guangxi, 545006, China
Liuzhou Key Laboratory of Big Data Intelligent Processing and Security, Liuzhou, Guangxi, 545006, China
Cybersecurity Monitoring Center for Guangxi Education System, Liuzhou, Guangxi, 545006, China

ARTICLE INFO

Keywords:
Cyber autonomous defense
Attack-defense game
Hierarchical multi-agent reinforcement learning
Game learning

ABSTRACT

Faced with the challenges of security strategy design brought about by the complexity of attack behavior and the dynamism of network structure, dynamic hierarchical intelligent defense methods have shown their effectiveness. However, in complex network environments, their application requires a higher level of coordination mechanisms. Therefore, this paper proposes a hierarchical multi-agent reinforcement learning network attack and defense game and cooperative defense decision-making method, which autonomously and efficiently completes the formulation of defense strategies and defense behavior responses. Firstly, we construct a Stackelberg hypergame model of cyberspace conflicts, and under the condition of information loss, characterize the multi-layer dynamic defense coordination response mechanism. Secondly, By utilizing a hierarchical multi-agent reinforcement learning method as the driving force for game evolution, we sequentially solve the Nash equilibrium of the game, and form a dynamic autonomous defense strategy. Finally, we construct a hierarchical multi-agent reinforcement learning framework, which decouples the defense decision problem, reduces the dimension of the defense action space and the exploration difficulty of the strategy space, and learns coordinated defense strategies more efficiently. We used the CybORG (Cyber Operations Research Gym) environment for simulation. We compared and analyzed the autonomously generated cyber defense strategies with other related works, verifying the superior coordination performance of our method in defense strategy generation and control.

1. Introduction

The global digital transformation has tightly integrated networks with various industries, leading to the proliferation of cybersecurity threats. Network security strategy involves the allocation of security facilities and defense resources, typically designed, planned, and implemented by security experts. The network security strategy profile represents an organization's overall defense capability and high-level security design, making it the most critical top-level design in the defense system (Waniek et al., 2019). However, the design of security strategies by human experts faces significant challenges due to the evolution of attack behaviors and the complexity of network structures. On one hand, attack behaviors have become multi-process and highly complex, with the added speed of automation and machine-level reactions facilitated by artificial intelligence technology (Apruzzese et al., 2020). It is impossible for human security strategists to enumerate all possible scenarios in advance for strategy programming. On the other hand, the hierarchical organization and dynamic topology characteristics of modern networks make it challenging to apply intelligent security decision-making methods, requiring the ability to handle and respond to increasingly complex network security situations. Consequently, designing automated cooperative defense strategies that coordinate multiple security facilities has become a matter of urgency. This method requires a flexible and resilient defense mechanism, generates continuous and dynamic cooperative defense strategies, and adaptively executes real-time defense responses (Applebaum et al., 2022).

Therefore, the integration of autonomous control theory and network security management has led to a surge in research on defense strategy games and Autonomous Cyber Operations (ACO). Network attack and defense behaviors are often modeled as game models, which qualitatively and quantitatively describe the impact of rational individual behaviors in the network space at the system level (Liang and Xiao, 2012). Previous studies have analyzed cyberspace conflicts from

* Corresponding author at: School of Computer Science and Technology, Guangxi University of Science and Technology, Liuzhou, Guangxi, 545006, China.
E-mail addresses: 221077020@stdmail.gxust.edu.cn (Y. Tang), 594588829@qq.com (J. Sun), wanghuan@gxust.edu.cn (H. Wang), dengjunyi@gxust.edu.cn (J. Deng), 20230702022@stdmail.gxust.edu.cn (L. Tong), xwhat@qq.com (W. Xu).

https://doi.org/10.1016/j.cose.2024.103871
Received 15 February 2024; Received in revised form 14 April 2024; Accepted 22 April 2024
Available online 25 April 2024

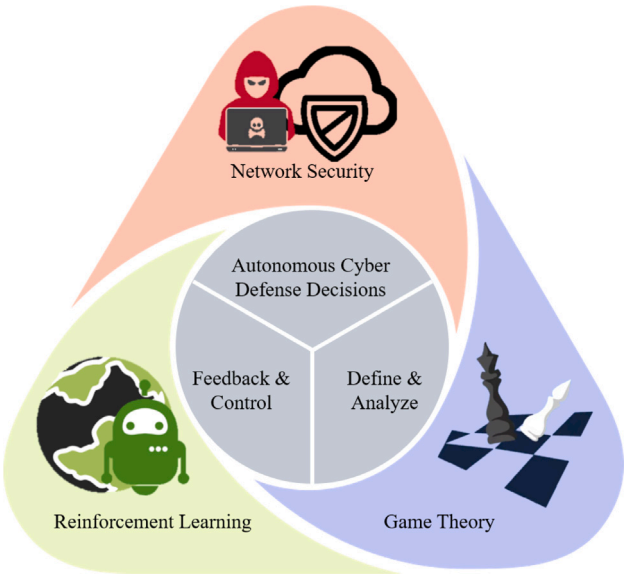

Fig. 1. Convergence of cybersecurity, game theory and reinforcement learning.

different perspectives, including static and dynamic models, complete and incomplete information models, perfect and imperfect information models, as well as cooperative and non-cooperative relationship models. Recent work has introduced information interference or deception as hyperparameters in game models to recreate more realistic network attack and defense scenarios (Cheng et al., 2022). Furthermore, the network security space is a dynamic and adversarial system, and ACO research based on learning theory is crucial for perceiving the environment, allocating network resources, and controlling network situations in complex networks. The feedback learning mechanism of reinforcement learning enables the exploration and utilization of behavioral patterns, demonstrating the capability of autonomous network operations. Single-agent and multi-agent reinforcement learning (MARL), as artificial intelligence decision-making theories, are introduced into the fields of network security penetration, defense, and planning (Adawadkar and Kulkarni, 2022). In summary, previous research provides a foundation for in-depth discussions on the following issues.

On one hand, previous research lacks discussions on the interaction of multiple players in the network security space, particularly the issue of heterogeneous defenders' collaborative responses. This constitutes a complex game model in a dynamic evolving game space. The evolution of utility functions in this game model requires rational exploration in the game space. Introducing a learning-based optimal response solving method is worth considering.

On the other hand, reinforcement learning has demonstrated excellent performance in various control and decision-making problems due to its unique learning mechanism. However, in scenarios involving interactions among multiple rational individuals, modeling different individual behaviors is crucial for analyzing strategy selection and game equilibrium. By combining game models, one can thoroughly analyze the equilibrium states of strategies and efficiently carry out strategy selection.

Based on the above, as shown in Fig. 1, this paper aims to combine game theory with learning methods to solve the problem of autonomous collaborative defense strategy decision-making. It proposes a hierarchical multi-agent reinforcement learning (HMARL) approach for network attack-defense games and collaborative defense decision-making. This approach utilizes game theory tools to model cyberspace conflicts and the collaborative actions of different defense facilities. It introduces a multi-agent mechanism for rational evolutionary game modeling, calculates the coordinated optimal response of defense facility sets, constructs elastic defense strategies, and dynamically controls defense actions in real time. The contributions of this paper can be summarized in three aspects:

- This paper models the network attack-defense mechanisms and cooperative defense response behavior using the Stackelberg hypergame model (SHM). This model fully captures the characteristics of misinformation and errors in information transmission in the network security space. It designs the collaborative response behavior of different types of defenders (followers) to attackers (leaders), establishes the relationship between the Nash equilibrium state and the attack-defense strategy representation, and constructs a stereoscopic collaborative defense strategy.
- Reinforcement learning methods are used to construct the game evolution process. This paper introduces the feedback learning mechanism of agents to dynamically evolve the game model, optimize the utility function of the game model, enable the defense side to rationally explore the attack-defense space, learn the optimal response strategies to attackers' behavior, and adjust network defense actions in real time, actively participating in network conflicts.
- A HMARL method is designed to address the curse of dimensionality in the action space during collaborative defense processes. This paper employs the concept of hierarchical reinforcement learning to decouple the defense decision problem, separating the selection of defense strategies from the formulation of specific defense actions. By specializing the reward functions for different types of agents, it reduces the dimensionality of the overall defense action space and alleviates the difficulty of policy space exploration, thereby enabling more efficient learning of coordinated defense strategies.

The structure of this paper includes the following sections: Section 2 introduces the contributions of previous research from a theoretical background perspective; Section 3 characterizes cyberspace conflicts using game theory; Section 4 designs a HMARL model for the dynamic evolution of the game model; Section 5 presents simulation cases, experimental simulations, and their results; and finally; Section 6 summarizes this paper, highlighting the contributions; Section 7 sorts out the limitations of this paper and points out the future research directions. For the reader's convenience, we summarize the notations that are frequently used in Table 1.

Table 1
Table of notations.

Symbol	Meaning
$\mathcal{H}$	Game model
l	Leader represented by attackers
f	A coalition of defenders constitutes a follower
S_l	Attacker strategy
S_f	Defender coalition Strategy
P_l	Attacker payoff
P_f	Defender payoff
I_j	Network infrastructure$j$
x	Attacker actions
y	Defender actions
Θ	Information lossy or spoofed delivery parameters

2. Related work

To integrate game theory models and learning theory effectively, it is crucial to thoroughly investigate the applications of various types of game models in the context of network attack and defense. Additionally, it is important to explore extensively how different reinforcement learning methods can address the challenges posed by the network security space. To achieve this, we will first discuss the fundamental network attack-defense game models, followed by an introduction to relevant research on enhancing network defense capabilities using reinforcement learning methods. Finally, we will present our own explorations building upon the previous efforts.

2.1. Network attack and defense game model

The non-cooperative network attack-defense models transition from static to dynamic. In static games, literature (Chen and Leneutre, 2009) abstracts network attack and defense as a two-player static game, where the actions of attack and defense are represented as probabilities over network facilities. While this approach allows for the analysis of network attack-defense strategies, it lacks scalability and fails to consider information transmission within the game. Furthermore, the players in the game are treated as a unified entity, overlooking individual decision-making. In contrast, Zhang et al. (2015) consider the imperfect and incomplete information scenarios and propose a method for selecting static active defense strategies based on Bayesian games. This represents an important extension in modeling network attack-defense.

Regarding dynamic games, An integrated approach has been proposed, which combines different types of games to create a robust, secure, and resilient framework (Huang et al., 2020). The "Bayesian" feature, as an effective means of capturing uncertainty, is widely discussed in the field of dynamic security games. The literature (Huang and Zhu, 2020) discussed Advanced Persistent Threats (APT), capturing their stealth, dynamics, and adaptability through multi-stage games with incomplete information, and used numerical experiments to iteratively calculate the perfect Bayesian Nash equilibrium, with the game participants taking strategic actions based on beliefs formed through multi-stage observation and learning. Hu et al. (2015) considering the joint threat of APT attackers and insiders, described the above interaction as a two-layer game model, analyzed the dynamic Nash equilibrium of the overall game model, and gave a dynamic defense strategy combining human factors. The literature (Liu et al., 2021) built single-stage and multi-stage Bayesian dynamic game models, calculated algorithms for perfect Bayesian equilibrium and revised prior beliefs, and summarized the general rules for network defense using mobile target defense strategies. The above work has limitations in the efficiency of deriving multi-stage Bayesian equilibrium from single-stage. The following developments have broken this constraint. Zhang and Malacaria (2021) optimized the Bayesian Stackelberg games model online through a learning mechanism, selecting the best combination of defensive actions to counter ongoing attacks, to build a decision support system for network security. This online optimization method has been proven to be more efficient than traditional equilibrium solutions. Khouzani et al. (2019) proposed a min–max multi-objective optimization framework to effectively solve multi-objective optimization problems in network security defense. In multi-stage attack problems modeled by attack graphs, the scale of security control strategies has been greatly expanded within an acceptable time range.

In dynamic games, the partial observation attribute is another means of characterizing incomplete information, and on this basis, the difficulty of handling time series problems in dynamic Bayesian security games is solved. Elderman et al. Elderman et al. (2017) explored reinforcement learning methods of neural networks, Monte Carlo learning, and Q-learning, applied in the partially observed Markov security game model, to solve the adversarial sequential decision-making problem of the attacker and the defender. Huang et al. (2018) address the issue of real-time applicability in game theory by combining differential game models with Markov decision processes. They used simulation experiments to solve the equilibrium solution of a multi-stage successive attack and defense game. This method serves as a foundation for applying learning theory in practice. The outstanding work above represents the exploration of applying learning theory to equilibrium problems in security game models, and the development of security strategies from classic design methods to intelligent design methods.

In discussing incomplete information games in the security field, its feature of incomplete information can be concretized as situations of unknown attacks, human social engineering attacks, hidden vulnerabilities, etc. The core issues it reflects are, on the one hand, the information asymmetry between the attacker and the defender, and on the other hand, the deception and misleading behavior in cyberspace conflicts. Most game models require complete information about the environment, making it difficult to depict situations with incomplete information in the conflict space (Zhu and Rass, 2018). Hypergame theory extends game theory by introducing hyperparameters to characterize the transmission of misleading information, providing a novel approach for modeling conflicts in the network space (Kovach et al., 2015). Compared to the discussion of deceptive behavior in Bayesian games, hypergame theory provides a more comprehensive and macro perspective to analyze the impact of misunderstandings in complex conflicts, rather than just focusing on the deceptive behavior itself. The outstanding work of Zhu et al. (2021) comprehensively summarized the progress in game modeling and equilibrium iteration in terms of deceptive defense.

Hypergame network attack-defense models often employ the Stackelberg model to capture deceptive network defense strategies (Cheng et al., 2022). The following works on the modeling of deceptive defense in the Stackelberg game model should not be overlooked, including discussions on deceptive defense resource allocation and deployment (Milani et al., 2020), network asset obfuscation (Bilinski et al., 2019), and deceptive network traffic (Anjum et al., 2020). Milani et al. (2020) allocated defense resources in the Stackelberg security game based on attack graphs to protect targets, and their proposed method based on layered directed acyclic graphs brought significant advantages to the defender. The literature related to this work (Bilinski et al., 2019) treated the attacker as the leader and the defender as the follower, and simulated the Markov Decision Process (MDP), studying the potential behavior of the attacker at non-critical points. The literature (Anjum et al., 2020) designed a deceptive network flow system called "Snaz" in the two-person non-zero-sum Stackelberg attack-defense game to mislead the attacker's reconnaissance. In the hypergame Stackelberg security game framework, resilient network defense is constituted by manipulating trust tags, hiding or disguising core network assets (Bakker et al., 2020; Zhan et al., 2013). In a research study (Schlenker et al., 2018), the TCP/IP protocol stack of a networked physical system was modified, and service ports were obfuscated to construct deceptive operations as hyperparameters for a deceitful Stackelberg security game model (Nguyen and Xu, 2019). The equilibrium points of this model were derived, enabling effective defense against hackers' reconnaissance of network assets.

2.2. Multi-intelligence reinforcement learning in cyber defense

Many researchers have introduced reinforcement learning algorithms into defense methods to tackle attacks at various stages. Reinforcement learning can be applied to enhance the defense capabilities of the network itself or to increase the robustness of network defense infrastructure.

In terms of managing the network security space, Hammar and Stadler (2021) abstract the intrusion detection problem as an optimal stopping problem and use deep reinforcement learning (DRL) to compute the optimal stopping time. This method characterizes network defense measures using a simple optimal stopping problem, highlighting the effectiveness of reinforcement learning in complex conceptual problems. Xu et al. (2022) focus on eavesdropping attacks, a typical form of network attack. They discuss the continuous control capabilities of DRL in a network environment, using routing as the control object to defend against eavesdropping attacks through route randomization. Optimizing communication strategies to enhance system security is another important application of reinforcement learning Li et al. (2023) utilize DRL algorithms to predict network attack paths in industrial networked physical systems, providing a new approach for attack traceability and path analysis.

For enhancing network security infrastructure, researchers have applied reinforcement learning's dynamic response and autonomous feedback mechanisms to energy network intrusion detection systems (Zolotukhin et al., 2020), edge network intrusion defense systems (Liu

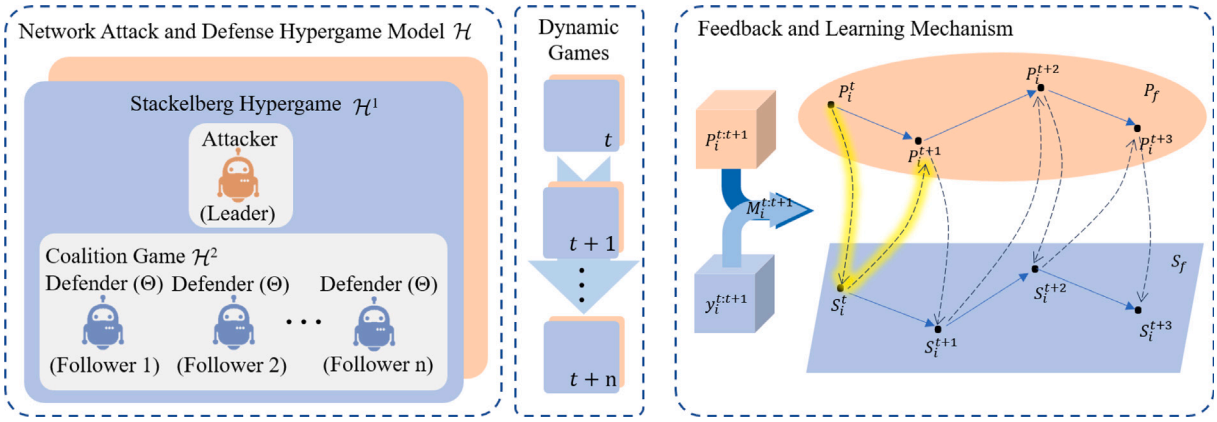

Fig. 2. Network attack and defense hypergame model construction and iterative processes.

et al., 2020), and software-defined wireless network intrusion response systems (Cardellini et al., 2022), contributing to network security warnings and blocking attack behaviors at the intrusion stage (Adawadkar and Kulkarni, 2022). Wang et al. (2020) dynamically deploy deceptive resources using reinforcement agents' policy generation, efficiently protecting internal network resources and trapping intruding attackers. Zhong et al. (2022) focus on the robustness of reinforcement learning models for detection and utilize reinforcement learning methods to counteract the generation of evasive detection samples. By expanding the machine learning training set and increasing the barriers for attackers to bypass detection, they protect the security of critical system assets.

MARL extends the discussion of the correlation between agents based on single-agent reinforcement learning, providing a new direction for research in the field of autonomous network defense (Nguyen and Reddi, 2023; Li et al., 2022). The field of intrusion detection is an early adopter of MARL technology, with many classic works summarized in the literature (Saeed et al., 2020), many of which have reached the state-of-the-art level. Adaptive defense or moving target defense is a widely discussed field at present. The literature closely related to this paper (Sengupta and Kambhampati, 2020) uses Bayesian Stackelberg Markov Games to study moving target defense problems, and iteratively solves the equilibrium state through an independent Q-learning MARL algorithm to achieve moving target defense. In the field of dynamic access control, the literature (Jin and Wang, 2020) uses multi-agent deep deterministic policy gradient (MADDPG) to optimize traffic allocation strategies, showing that reinforcement learning has application prospects in collaborative threat mitigation based on zero trust. Some recent works (Alshamrani and Alshahrani, 2023) have also made attempts to assist network defense with MARL using the multi-armed bandit algorithm, and discussed the impact of various reward situations on the strategy. Deception is a behavior with covert rationality, and it is also a key point of discussion in MARL methods. The literature (Du et al., 2022) discusses the dynamic nature of network deception defense in the attack graph scenario, uses reinforcement learning algorithms to self-play the two-person Markov game attack and defense model, and achieves adaptive network deception defense resource deployment by the defender. Similarly, the literature (Kong et al., 2023) also discusses the deployment problem of deception defense resources. The difference is that the author models the network attack and defense problem as a maze pathfinding model, and obtains the real-time deployment strategy of deception assets by solving the Nash equilibrium of multi-agent stochastic games. Cheah et al. (2023) is closely related to our paper. They propose a network cooperative defense autonomous decision-making framework in the scenario of autonomous vehicle queue driving, decompose deep rational cooperative defense actions, such as network deception behavior, to sub-agents, similar to dynamic programming optimization methods, to achieve multi-agent cooperative autonomous network defense.

Building upon previous efforts, firstly, the Stackelberg attack and defense hypergame model we discuss can effectively overcome the limitations of previous models, especially the Bayesian game model, in describing the impact of deceptive incomplete information characteristics on the overall network situation. The introduction of the hypergame model makes deceptive defense behavior more rationally profound, and by including the defense follower in the cooperative constraint to participate in the game, it is an exploration of cooperative defense games. Secondly, we use MARL methods as the driving force for the evolution of dynamic games, overcoming the limitations of numerical calculation methods that are difficult to explore complex game spaces and conventional heuristic methods that lack rational exploration. The method in this paper reflects the equilibrium state through the action strategy of the reinforcement learning agent, which is a new attempt to generate real-time dynamic defense responses. Thirdly, the problem of the evolution of game models with high dimensions or complexity in the action space of multiple agents has always been a shackle for the application of reinforcement learning. More direct iterative methods may not form deep decisions, and more complex iterative algorithms may not converge. This paper decomposes the decision-making process, introduces hierarchical thinking, reduces the dimension and decouples the action space, effectively improving the application efficiency of reinforcement learning.

3. Cyber attack and defense hypergame model

3.1. Construction of network attack and defense hypergame model

The network attack and defense hypergame model accurately depicts the scenario of multi-agent autonomous network defense, reflecting the interaction of incomplete information, imperfect information observation, and cognitive bias decision-making in cyberspace conflicts. At the same time, it analyzes the Nash equilibrium of the network confrontation strategies of the attack and defense sides and explores the coalition core form of cooperative defense. The network attack and defense hypergame model is shown in Fig. 2.

The network attack and defense hypergame model consists of two parts. The first part is described by the SHM $\mathcal{H}^1$ of the attack and defense sides, and the second part is composed of the cooperative game model $\mathcal{H}^2$ formed by multiple agents. The defensive agent participates in $\mathcal{H}^1$ as a follower, and its own strategy is constrained by $\mathcal{H}^2$. (1) gives the relationship between $\mathcal{H}^1$ and $\mathcal{H}^2$. In the composite game model $\mathcal{H}$, all defenders form a cooperative follower alliance f to participate in the game $\mathcal{H}^1$.

$$\mathcal{H} := \{\mathcal{H}^1, \mathcal{H}^2\} \tag{1}$$

The definitions of $\mathcal{H}^1$ and $\mathcal{H}^2$ will be detailed separately below.

3.1.1. Network attack and defense SHM

In the scenario of multi-agent cooperative defense, k defenders form an alliance f, which responds to system attacks as followers in the SHM, protecting resources in n network systems. The attacker is defined as the leader l, considering the attacker's first-mover advantage in general network attack and defense and its continuous influence. (2) gives the definition of the SHM $\mathcal{H}_{lf}^{1}$

$$\mathcal{H}_{lf}^{1} = \left\{\{l\} \cup f, S_l \times S_f, \{P_l\} \cup P_f, \Theta\right\} \tag{2}$$

We consider the strategy space $\mathbb{R}$, where S_l is the strategy of the leader, $S_l \in \mathbb{R}^K$, and S_f is the strategy of the follower, $S_f = S_1 \times \cdots \times S_k$, where S_i is the strategy of participant i. Θ is used to describe the impact of information transmission, $\Theta = \{\theta_l, \theta_f\}$, considering the hyperparameter θ_l as the influence on the leader, and $\theta_f = \{\theta_1, \ldots, \theta_k\}$ as the influence on the followers. This inaccurate observation will affect the payoff function, the leader's payoff function $P_l : S_l \times S_f \times \Theta \to \mathbb{R}$, and the follower's payoff function $P_f = \{P_1, \ldots, P_k\}$, where the payoff of the ith follower comes from $P_i : S_l \times S_i \times \Theta \to \mathbb{R}$.

After giving the definition of the hypergame model, we add the description of network facilities to express various game elements. The set of network facilities is defined as $\mathcal{I} = \{I_1, \ldots, I_n\}$, and the leader, i.e., the attacker, has resources R_l to execute actions x. The attack behavior on network facilities can be described as $\sum_{j=1}^{n} x^{(I_j)} = R_l$, and the attacker's strategy is represented as (3):

$$S_l = \{x | \sum_{j=1}^{n} x^{(I_j)} = R_l, x^{(I_j)} \geq 0\} \tag{3}$$

Similarly, the follower, i.e., the defender i, has resources $R_i$to execute actions y. The defense behavior of defender i on network facilities can be defined as $\sum_{j=1}^{n} y^{(I_j)} = R_i$, and the defender's strategy can be represented as (4):

$$S_i = \{y_i | \sum_{j=1}^{n} y_i^{(I_j)} = R_i, y_i^{(I_j)} \geq 0\}, \forall i \in f \tag{4}$$

After giving a detailed expression of the strategy, we also need to discuss the definition of payoff. For network facility I_j, the attacker gets a payoff of $P_l^{\alpha}(I_j)$ when executing action $x^{(I_j)}$, and a payoff of $P_l^{\beta}(I_j)$ when not executing action $x^{(k_j)}$. The defender gets a payoff of $P_i^{\alpha}(I_j)$ when executing action $y^{(I_j)}$, and a payoff of $P_i^{\beta}(I_j)$ when not executing action $y^{(I_j)}$. The payoff functions of the attacker and the defender under the strategy profile $(x, y, \{\theta_l, \theta_f\})$ are as shown in (5) for the attacker and (6) for the defender.

$$\begin{aligned} &P_l(x, y, \theta_l) \\ &= \sum_{j=1}^{n} \left(\sum_{i=1}^{n} y_i^{(I_j)}\right) \left(x^{(I_j)} P_l^{\alpha}(I_j, \theta_l) + \left(R_l - x^{(I_j)}\right) P_l^{\beta}(I_j, \theta_l)\right) \end{aligned} \tag{5}$$

$$\begin{aligned} &P_i(x, y_i, \theta_i) \\ &= \sum_{i=1}^{n} y_i^{(I_j)} (x^{(I_j)} P_i^{\alpha}(I_j, \theta_i) + (R_l - x^{(I_j)}) P_i^{\beta}(I_j, \theta_i)) \end{aligned} \tag{6}$$

Based on the above information, we can present the Hyper Nash Equilibrium(HNE) of the SHM, which is the optimal action and response given in the case of information deficiency. (7) expresses the attacker's action strategy (x^*, y^*), and (8) and (9) express the defender's optimal response $BR(x, \theta_f)$.

$$(x^*, y^*) \in \underset{x \in S_l, y \in BR(x, \theta_f)}{\operatorname{argmax}} P_l(x, y, \theta_l) \tag{7}$$

$$BR_i(x, \theta_i) = \underset{y_i \in S_i}{\operatorname{argmax}} P_i(x, y_i, \theta_i), \forall i \in f \tag{8}$$

$$BR(x, \theta_f) = BR_1(x, \theta_1) \times \cdots \times BR_k(x, \theta_k) \tag{9}$$

With the above SHM constructed, we introduce assumptions widely used in the analysis of network security games to illustrate the HNE of this model:

Assumption 1. Θ is compact and convex, the impact of θ_l may be nothing.

Assumption 2. For $f = 1, \ldots, k$, $P_i^{\alpha}(I_j, \theta_i)$ and $P_i^{\beta}(I_j, \theta_i)$ are differentiable in $\theta_i \in \Theta$.

Based on Lemma 2 and Theorem 1 of the Cheng et al. (2022), under Assumptions 1 and 2, for $\forall \theta_l, \theta_i \in \Theta$, SHM has an HNE. The reason is that for $\forall (x, y), x \in S_l, y \in S_f$, under the game sequence m, the leader's strategy converges to the optimal action sequence.

$$P_l(x_m^*, y_m^*, \theta_l) = \lim_{m \to \infty} \max_{x_m \in S_l, y \in BR(x, \theta_f)} P_l(x_m, y_m, \theta_l) \tag{10}$$

That is, $x^* \in \arg\max_{x \in S_l} P_l(x, y^*, \theta_l)$ forms the optimal response sequence x_m^*. For the follower, there also exists an optimal response sequence y_m^* composed of optimal response actions $y^* \in BR(x^*, \theta_f)$.

However, the stability of this equilibrium is insufficient. The main reason for the instability lies in Θ. The loss of information and the incorrect transmission of $\hat{\theta}$ can lead to the game participants obtaining $P_l(x_m^*, y_m^*, \hat{\theta}_l) < P_l(x_m^*, y_m^*, \theta_l)$, $P_f(x_m^*, y_m^*, \hat{\theta}_f) < P_f(x_m^*, y_m^*, \theta_f)$, thus the strategy convergence will be affected.

3.1.2. Defender coalition game model

The above provides the main game forms of network attack and defense. Another constraint to consider in this study is the cooperative relationship constraint among the defense game players. This constraint is reflected in the defender's transferable payoff function. The following defines this constraint through the definition of the cooperative game model. (11) gives the coalition game $\mathcal{H}_f^2$.

$$\mathcal{H}_f^2 = \{f, \mathcal{P}_f\} \tag{11}$$

A coalition f is formed by k defenders, satisfying conditions (a) $\bigcup_{i=1}^{k} i = f$, (b) $i \cap j = \emptyset, \forall i \neq j$, which determines the structure of the cooperative coalition. The coalition combined payoff P_f satisfies conditions (a) $P_f \geq P_i$ (b) $\sum_{i \in f} P_i = P_f$, which determines the way of profit distribution. In addition, defenders in the coalition will share the perception information of the environment, the impact of lost and incorrectly transmitted information will be weakened in the sharing link, that is, $\theta_f \leq \sum_{i=1}^{n} \theta_i$.

The cooperation mode between defenders is based on the formation of the core, and due to the existence of shared factors, the form of the coalition will not affect the convexity of SHM.

$$core = \{y_i \in S_i | P_f(x, y_f, \theta_f) \geq \sum_{i \in f} P_i(x, y_i, \theta_i)\} \tag{12}$$

3.1.3. Discussion on attack and defense SHM and HNE

To clarify the game model proposed in this paper and the dynamization process of this model in the following text, we further clarify and summarize the mathematical definition of this model, deepening the discussion on the existence of HNE.

The game model proposed in this paper is composed of the SHM $\mathcal{H}_{lf}^{1}$ describing cyberspace conflicts and the coalition game $\mathcal{H}_f^2$ describing multi-defender cooperative defense, specifically in the form of (13).

$$\mathcal{H} := \left\{ \begin{array}{l} \mathcal{H}_{lf}^{1} = \left\{l \cup f, S_l \times S_f, P_l \cup P_f, \Theta\right\}, \\ \mathcal{H}_f^2 = \left\{f, P_f\right\} \end{array} \right\} \tag{13}$$

The optimal solution problem of network attack and defense strategy corresponding to the HNE of this hypergame model is described as (14).

$$\begin{aligned} max \quad & BR(x, \theta_f) = BR_1(x, \theta_1) \times \cdots \times BR_k(x, \theta_k) \\ s.t. \quad & BR_i(x, \theta_i) = \underset{y_i \in S_i}{\operatorname{argmax}} P_i(x, y_i, \theta_i) \\ & P_f(x, y_f, \theta_f) \geq \sum_{i \in f} P_i(x, y_i, \theta_i) \\ & x \in S_l, y \in S_f. \end{aligned} \tag{14}$$

Regarding the existence of HNE in this game model, it is detailed in *Theorem 1* of Cheng et al. (2022). The difference between this paper and the problem discussed in it is that the determined SHM is a single defender as the leader and multiple attackers as followers. The core issue it discusses is the HNE stability conditions under the two situations of misunderstanding and deception. Of course, to perfect the work of this paper, a rigorous approach to proving the existence of HNE is also provided here.

Proof. Under the premise of satisfying Assumptions 1 and 2, when $\theta_i' \in \Theta$, there is a defense gain $P_i(x, y_i, \theta_i')$, we define $F(x, y) = \{(\hat{x}, \hat{y}) | \hat{x} \in \arg\max_{x \in S_l, y \in BR(x,\theta_f)} P_l(x, y, \theta_l), \hat{y} \in BR(x, \theta_f)\}$ to obtain the optimal action in the current strategy set. Because, as known from definitions (3) and (4), $S_l \times S_f$ is compact, so there exists a convergent subsequence $\{(\hat{x}_{jm}, (\hat{y})_{jm})\}_{m=1}^{\infty}$, converging to $\lim_{m\to\infty}(\hat{x}_{jm}, (\hat{y})_{jm}) = (\hat{x}^*, \hat{y}^*)$. Therefore, it only needs to prove that the result of the limit convergence is in the range of the function F, i.e., $(\hat{x}^*, \hat{y}^*) \in F(x, y)$, to show that $\mathcal{H}$ has an HNE.

In the leader's perspective, $P_l(\hat{x}^*, y^*) = \lim_{m\to\infty} P_l(\hat{x}_{jm}, y_{jm}) = \lim_{m\to\infty} \max_{x' \in S_l} P_l(x', y_{jm})$. From Lemma 17.30 in the literature (Guide, 2006), we can get $\lim_{m\to\infty} \max_{x' \in S_l} P_l(x', y_{jm}) = \max_{x' \in S_l} P_l(x', y^*)$, therefore $\hat{x}^* = \arg\max_{x' \in S_l} P_l(x', y^*)$. Similarly, from the follower's perspective, $\hat{y}^* \in BR(x, \theta')$, i.e., $(\hat{x}^*, \hat{y}^*) \in F(x, y)$. Through Theorem A.14 in the literature (Carmona, 2012), $(\hat{x}^*, \hat{y}^*)$ can be generalized to the general case, i.e., there exists (x', y') such that for the leader and each follower, $x^* = \arg\max_{x \in S_l} P_l(x, y^*)$ and $y' \in BR_i(x', \theta')$, therefore, there exists an HNE (x', y') of $\mathcal{H}$.

3.2. Hypergame model dynamic analysis

Network attack and defense is a dynamic and ongoing process, and network defense work needs to extend the normal operation time of the network core functions as much as possible during the ongoing network attack process. Defenders and attackers will reach the game equilibrium point multiple times in the game space, which is reflected in (9) where the defender's optimal response will change with the game environment. It is almost impossible to complete the numerical analysis of dynamic evolutionary equilibrium under the super game model, and its complex game process is the main obstacle to model evolution and control analysis. For this reason, this paper continues to use the symbols from the previous text, describes the evolution process of the game model with the Markov decision process, and embeds the feedback learning mechanism to guide the evolution path.

Definition 1. Multi-player Network Defense Markov Game Process from an Infinite Discrete Time Perspective

(1) The set of game players is $F = \{1, 2, \ldots, k\}$, where each player represents one or a type of defender. Define the discrete time set $\mathcal{T}_+ = \{1, 2, \ldots\}$, at each moment $t \in \mathcal{T}_+$, the game players execute actions and observe the network game space.
(2) Define the abstract action space $\mathcal{A}_i, \forall i \in f$. In network attack and defense, different defenders have different action spaces. In discrete time, $y_i^t \in \mathcal{A}_i$, and an action tuple $(y_1^t, y_2^t, \ldots, y_k^t)$ constitutes the strategy S_F.
(3) Define the observation space $\mathcal{O}$: there is a state observation $o^t \in \mathcal{O}$ in the network attack and defense game space at each moment.
(4) Define the state transition process $T : \mathcal{O} \times \prod_{i \in f} \mathcal{A}_i \to \Delta(\mathcal{O})$. In discrete time, the next moment state $o^{t+1} \sim T(o^t, y^t)$, where y^t is the set of actions made by the set of defenders.
(5) Define the transition utility $P_i : \mathcal{O} \times \prod_i \mathcal{A}_i$. At moment t, the utility $P_i^t(s^t, y^t)$ can be calculated.
(6) Define the discount factor γ, which adjusts the expected reward $\sum_{t=1}^{\infty} \gamma^t P_i(s^t, y^t)$.

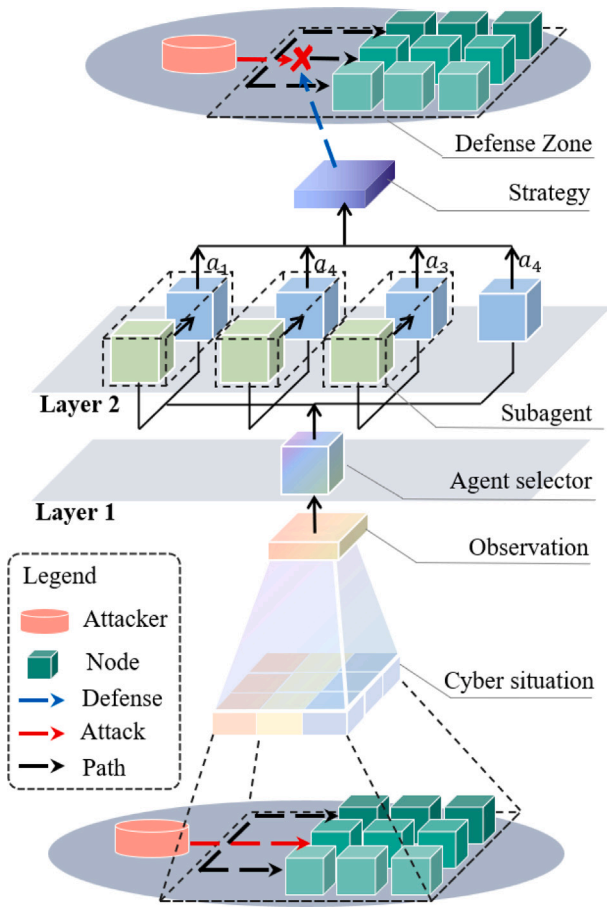

Fig. 3. Multi-intelligent hierarchical reinforcement learning of autonomous network defense processes.

Based on the above definitions, from moment t to $t+1$, we have constructed a feedback mechanism (15) through utility transition and action execution.

$$M_i^{t:t+1} = \{\{P_i^{t:t+1}, y_i^{t:t+1}\}\} \tag{15}$$

By guiding feedback through learning utility transitions, the model learns a collaborative defense strategy. In (16), we define the utility iteration over time, and in (17), we define the utility iteration guiding strategy S_f.

$$P_i^{t+1} = (1 - \mu_i^t) P_i^t + \mu_i^t G_i^t(S_i^t, P_i^t, y_i^t) \tag{16}$$

$$S_i^{t+1} = (1 - \lambda_i^t) \pi_i^t + \lambda_i^t H_i^t(S_i^t, P_i^{t+1}, y_i^t) \tag{17}$$

Where μ and λ represent the learning rates, G_i^t and H_i^t are the strategies of utility evolution and strategy evolution constructed by the feedback mechanism, and the evolution of strategies is implemented by the reinforcement learning agent.

4. HMARL network defense framework

In this section, we present a dynamic evolutionary game model and propose a HMARL approach for autonomous network defense decision-making. Firstly, we decompose the problem to establish a hierarchical reinforcement learning framework. Then, we introduce individual agents that handle different subproblems.

4.1. Problem decomposition and hierarchical framework

In the evolutionary process of the game model, the strategy profile determined encounters several challenges: the transition probabilities

of strategy states are unknown, the network attack-defense situation is highly dynamic, information transmission among players is uncertain, and the cascading effect of multi-agent collaboration all contribute to an extremely large state space, which affects action decision-making. To address these challenges, this paper introduces the idea of hierarchical reinforcement learning and decomposes defense decision-making into two subproblems: defense action type selection and defense action execution. Different-level defense agents are established based on the decomposition of action space for each subproblem.

In complex scenarios with intricate state and action spaces, introducing hierarchical reinforcement learning to decompose the state and action spaces proves to be an effective method for reducing the difficulty of policy search. This approach is widely applied in the field of complex control and optimization. For example, in Shi et al. (2020), the behavior of unmanned aerial vehicles (UAVs) for cellular services is decoupled into separate path planning and resource allocation subproblems, enabling step-by-step decision-making at different granularities and significantly improving network throughput. Tran et al. (2021), from the perspective of security penetration testing, address the exponential expansion of action space with increasing network complexity by using a hierarchical reinforcement learning approach to decouple the action space and decompose the penetration process. This allows for faster and more stable identification of optimal attack strategies. Previous efforts provide us with valuable insights. Generating collaborative defense strategies involves searching for optimal solutions in a complex and vast game space, where the dimension and scale of the action space are immense. Therefore, in this paper, the task of autonomous network defense is decomposed into two subproblems: the selection of different types of agents and the defense decision-making of each agent.

We establish the first-level defense agent selector to address the subproblem of selecting different types of agents. In this paper, we categorize defense behaviors into three types: perceiving and analyzing network situations, removing threats and restoring infrastructure functionality, and deploying deceptive defense nodes. Then, we construct an agent selector to handle different network states. The selector activates the appropriate type of agent to perform the corresponding defense actions.

Next, we establish the second-level defense action execution agents to address the subproblem of defense decision-making by the agents. The defense functionality of different types of defense agents is determined by their action spaces. When activated, defense agents explore the action space to determine optimal responses and formulate defense strategies.

Overall, in Fig. 3, we describe the process of autonomous network defense using HMARL. Firstly, the defender perceives the attack behavior, which is transformed into the observation space. Then, based on the current network situation, the first-level selector chooses the optimal defense action type and activates the corresponding type of agent. Subsequently, the respective agent receives state observations and makes defense decisions based on them, determining action parameters including defense action subtypes and execution target parameters. Finally, the defender executes the defense strategy in response to the attack behavior.

Furthermore, this paper proposes an independent two-layer agent architecture, which separates the training and operational phases. The agents are only allowed to engage in real-world operations within a network simulation environment once their policies have converged.

4.2. Layer 1 agent selector design

Given the clear task characteristics of this level, the Q-Tabular algorithm is used to quickly converge and solve the subproblem. Let us start by introducing the design of the action space, observation space, and reward function

4.2.1. Action space design

In Table 2, we provide the categorization of the action space. We classify defense behaviors into four categories, corresponding to the Layer 2 defense agents. The selector is responsible for choosing one category of agent based on the current network situation. The definitions of these defense actions are based on the Open Command and Control (OpenC2) specifications.

The action space for the selector is provided in the first column of Table 2, which includes perception action type, deception action type, recovery action type, and sleep type.

The perception action type is responsible for detecting the overall network situation. It abstracts the functionalities of network situational awareness devices and cyberspace probing devices.

The deception action type aims to implement deceptive defense strategies by transmitting misleading information to attackers in a way that simulates genuine business activities. It abstracts the functionalities of sandbox and honeypot devices.

The recovery action type involves defensive actions against network attacks. This includes clearing malicious session connections, ensuring that user privileges do not compromise system security, and resetting compromised devices. It abstracts the actions performed by security experts in network attack and defense scenarios.

The sleep action allows the selector to skip the current game round and not allocate defense resources.

4.2.2. Observation space design

The observation space is defined as a collection of asset fingerprints in the network space, which includes the following six aspects: host information, interface information, process information, session information, system information, and user information. These six types of information are used to distinguish hosts, establish network connections, provide various types of services for each user, and abstractly present the overall cyberspace profile. In addition, the formal definitions of each field also support reinforcement learning training.

4.2.3. Reward function design

The selector's task is clear and the action space is discrete, but guiding the selector to make the correct choice requires two parts of rewards. On one hand, there is the reward from the environment for the effectiveness of the defensive behavior. On the other hand, there is the internal reward of the selector. The former part of the reward is given by the environment after each sub-agent has completed training and cascades to perform defensive actions under the controller. The latter part is the reward given by the controller for correctly selecting the sub-agent. (18) provides the definition of the controller's reward.

$$R^t_{L1} = -[C/(reward(o^t, a^t_{L2}))] \tag{18}$$

At time t, the controller's reward is R^t_{L1}, and $reward(o^t, a^t_{L2})$ is the reward for the effectiveness of the defensive action given by the environment, the construction of which will be detailed in the reward functions of various defensive agents. C is the reward for agent selection judgment, which is based on the experience of human security experts when obvious attack behavior is detected. The reward value is −1 or 1. If the selector's judgment is the same as the expert's knowledge, C is 1; if it is different, C is -1. In this way, the selector can balance between quickly converging using expert prior knowledge and exploring different choices.

4.3. Design of various agents

Various types of defensive agents are parallel to the agent selector cascade. When activated by the selector, defensive agents inherit the observation space and make decisions based on their own action space, generating defensive strategies. In this paper, the observation space received by each agent is the same, as introduced in the second part of the previous section. The reward function is a common reward from the environment. Neural networks are used to explore strategies from complex action spaces, so we use the Proximal Policy Optimization (PPO) algorithm as the training algorithm in the design of sub-agents.

Table 2
Action space classification.

Type & OpenC2 Action ID	Action	Purpose
Perceive (1 & 30)	Monitor	Gathering network situational information
	Analyse	Identify aggressive behaviors
Decoy (7 & 18)	Decoy Apache	Pretending to be an Apache service
	Decoy Femitter	Pretending to be an Femitter service
	Decoy HarakaSMPT	Pretending to be an HarakaSMPT service
	Decoy SSHD	Pretending to be an SSHD service
	Decoy Svchost	Pretending to be an Svchost process
	Decoy Tomcat	Pretending to be an Tomcat service
Recover (10 & 23)	Remove	Kill malicious processes
	Restore	Restoring a system to good state
Sleep (–)	Sleep	Do Nothing

4.3.1. Action space design

The action type of each type of defensive agent is a distinguishing point of the agent, as listed in the second column of Table 2. Different action types require different action parameters. It can be intuitively understood that the complexity of action parameters is an important reason for the large scale of the action space. The parameters that need to be determined in the action space mainly include three parts: parameters for constructing network communication, including IP addresses and subnet divisions; parameters for operating various services, including target information, processes, and port numbers; and parameters for determining operation targets, including target sessions, host names, user names, and privilege passwords. After the agent takes action, it will correspondingly change the network space situation, realize game evolution, and coordinate defense.

4.3.2. Reward function design

The sub-agent reward function (15) is based on network situation and action cost. Here, $risk(o^t)$ is a network situation risk mapping, which maps different types of hosts receiving attacks to different penalty values, enabling the agent to learn strategies to protect key assets. $cost(a^t_{L2})$ maps different defensive actions to different action costs, enabling the agent to learn how to protect the system at the smallest cost. Both types of mapping values are hyperparameters, which we provide in Table 4.

$$reward(o^t, a^t_{L2}) = risk(o^t) + cost(a^t_{L2}) \tag{19}$$

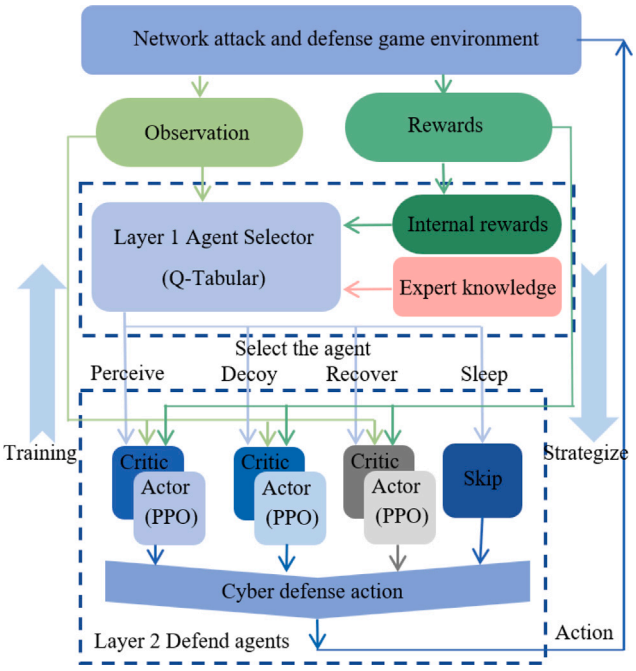

Fig. 4. HMARL framework.

4.4. Multi-agents hierarchical reinforcement learning model training

The components described above construct a HMARL model, the overall framework of which is shown in Fig. 4. The figure shows the connection relationships between the components and reflects the implementation of different algorithms by each layer of agents within the framework.

In this hierarchical reinforcement learning model, the internal reward of the first layer agent selector is influenced by the global reward, so the second layer sub-agents are trained first in the algorithm implementation. During the bottom-up training process, each type of sub-agent inputs the global observation and global reward of the network attack and defense environment. Here, the global observation is equivalent to each sub-agent being able to perform perception operations, so the recovery agent and decoy agent need to be trained separately during the training process. The converged sub-agents are cascaded in parallel under the first layer agent selector. The first layer agent selector transforms the global reward into an internal reward and guides the Q-table algorithm together with expert knowledge, enabling it to quickly converge on the strategy of selecting agents. After the overall training is completed, the hierarchical reinforcement learning model can formulate coordinated defense strategies from top to bottom.

The selection of the training algorithm for the agents depends on the tasks faced by agents. The choice of training algorithms for the two-layer agent architecture is the result of carefully balancing the task requirements and the algorithmic properties to ensure efficient and effective learning for the agents in the given cybersecurity context.

The task for the second-layer sub-agents is to select the sub-action type under the current security action space and generate the corresponding sub-action parameters, which is a policy generation type of task. It is not suitable to use the value iteration reinforcement learning algorithm, so the policy iteration method is utilized for training. The PPO is a relatively advanced policy iteration algorithm. Compared to its predecessor, the Trust Region Policy Optimization (TRPO) algorithm, PPO has a smaller computational cost and faster iteration, making it more competitive. The Deep Deterministic Policy Gradient (DDPG) algorithm is not an ideal choice here, as it trains multiple neural networks (typically 4) which makes the method highly sensitive to the environment and hyperparameters, resulting in difficulty in convergence. Therefore, the sub-agents are trained using the "Clip" optimization method of the PPO algorithm.

The task for the first-layer selector is to choose one or more sub-agents given different security situations, such that the current defense strategy yields the highest reward. This is a task with a relatively small action space and a clear correspondence between the actions and rewards. Hence, the Q-Tabular algorithm is selected for training the selector. This algorithm is simple and efficient, and can also be accelerated by incorporating expert knowledge to speed up convergence.

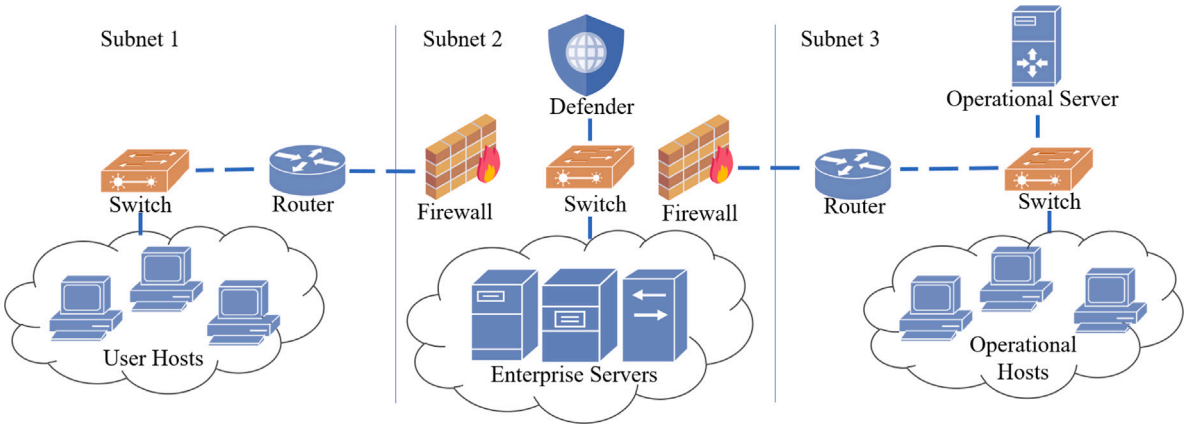

Fig. 5. Network scenario.

5. Autonomous cyber defense use cases and simulation

5.1. Autonomous cyber defense use cases

This paper validates a HMARL autonomous network defense method using a simulated environment. The Cyber Operations Research Gym (CybORG) (Cyber operations research gym, 2022a), developed by Maxwell Standen, provides a network security research environment for training and developing autonomous agents. The CybORG tool generates a scenario based on pre-generated descriptions, initializes a set of agents to perform roles in that scenario, and then implements their actions and evaluates their effectiveness. We consider the following advantages of this tool in establishing network attack and defense scenarios. Firstly, in terms of the fidelity of cyberspace scenarios, this toolkit uses cloud-based virtual machines for simulation and a general interface for simulating network environments. Through a highly modular design, scenarios modeled in finite state machines are highly realistic. Secondly, in terms of the fidelity of network operations, this toolkit defines simulated operation behaviors as state transitions, simulating the execution effects of commands with appropriate parameters in real scenarios.

5.1.1. Network attack and defense scenarios and attacker behavior patterns

This paper considers a general network service scenario as an autonomous network defense use case, and the specific settings of the use case are given in the related introduction of Cage Challenge 2 (Cyber autonomy gym for experimentation challenge 2, 2022b). The network framework is given in Fig. 5, which divides the network into three subnets. Hosts within a subnet can only communicate with hosts within the same or adjacent subnets. Subnet 1 is composed of ordinary users, which are non-critical user hosts. Subnet 2 is composed of enterprise servers, which are the objects accessed by users in Subnet 1. Subnet 3 is composed of enterprise server management hosts, including key operation servers and three user hosts. The network communication process is mapped to discrete time states. In a discrete state, ordinary users initiate normal business requests to enterprise servers, while attackers and defenders select actions from their respective action sets to operate the network. The execution order is: defender action, ordinary user action, attacker action.

The attacker's high-level attack action set is given in Table 3, with attack actions defined in accordance with the Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK) Technique definitions. This high-level action is converted into specific parameters executed in the network environment through the CybORG tool, such as low-level operation instructions for hosts and ports, thereby realizing simulated attack behavior. The attacker's modus operandi begins with using a user host in subnet 1 as a foothold. The attacker establishes communication with subnet 2 and uses the open port list obtained through the "Discover Network Services" operation to determine which vulnerabilities are available. In 75% of cases, the attacker will choose the highest-ranked available vulnerability. In 25% of cases, one of the other available vulnerabilities is chosen at random. After invading subnet 2, the attacker will escalate privileges and maintains the session. Once they have exploited a server in the enterprise network with the IP address of the operating server, they can invade Subnet 3 and ultimately take over the operating server through invasion means. The attacker's goal is to disrupt the service of the operating server for as long as possible through the "impact" action, for which the attacker will attack the operating server as directly as possible. The defender needs to prevent the attacker's behavior at each stage of the attack, discern whether the communication is from the attacker or an ordinary user, and maintain network functionality. Normal users only perform discovery operations and do not perform destructive behavior on hosts. They prevent the defender from thinking that all network activities are the work of the attacker.

The following provides an ideal instance to illustrate the operation process of the simulation environment. The defender can install monitoring tools on the host, collect and analyze event data, and determine whether it is the malicious activity of the attacker or the normal activity of a legitimate user. It can remove the attacker's access permissions, but this is only effective when the attacker's permissions have not been escalated, because once the attacker obtains system administrator permissions, the defender will receive a negative reward. As long as the attacker maintains administrator permissions, the defender will continue to receive negative rewards. The defender can also restore the system to a standard state to remove the attacker's activities, but this will destroy user data. Even after restoration, the attacker's initial foothold cannot be completely cleared, which is to ensure the sustainability of the game and simulate the difficulty of clearing an attacker who has obtained credentials in actual situations.

5.2. Simulation

Based on the use cases of network attack and defense scenarios in the previous section, the following experiments are designed to verify the performance of the HMARL autonomous network defense method. Firstly, experiments were conducted on the bottom-up training process and effect. Secondly, the decision-making performance of the top-down defense strategy was evaluated. Finally, the agent's defense strategy is analyzed through the action distribution.

5.2.1. Experiment preparation

In Table 4, the experimental setup parameters and hyperparameters for each layer of the reinforcement learning model are provided. With these basic conditions, training is conducted with each episode consisting of 100 steps, and a total of 1000 episodes are trained,

Table 3
Action space classification.

Type and ATT&CK technique	Purpose
Discover Remote Systems (ATT&CK Technique T1018)	Remote System Discovery. Discovers new hosts/IP addresses in the network
Discover Network Services (ATT&CK Technique T1046)	Network Service Scanning. Discovers responsive services on a selected host by initiating a connection with that host.
Exploit Network Services (ATT&CK Technique T1210)	Exploitation of Remote Services. This action attempts to exploit a specified service on a remote system.
Escalate (ATT&CK Tactic TA0004)	Privilege Escalation. This action escalates the privilege of attacker on the host.
Impact (ATT&CK Technique T1489)	Service Stop. This action disrupts the performance of the network

Table 4
Experimental hyperparameter settings.

Entries	Parameters
Hardware	CPU AMD EPYC 7T83 64-Core 2.45 GHz GPU NVIDIA GeForce RTX 4090 24 GB RAM 512 GB Disk 1 TB
Software	OS Ubuntu 22.04.4 jammy LTS x86_64 Python 3.9.17 Pytorch 2.1.2
PPO Algorithm	Number of layers for actor network: 3 Number of layers for critic network: 3 Number of nodes for actor layers: (62, 64, 32) Number of nodes for critic layers: (62, 64, 1) Actor network active function: (ReLU, ReLU, Softmax) Critic network active function: (ReLU, ReLU) Learning rates: 0.02 Replay buffer size, batch size: 100000, 512 Maximal episode, steps per episode: 1000, 100 Episode clip: 0.2
Reward	Attacker access User Hosts: −0.1 Attacker access Enterprise Servers: −1 Attacker access Operational Server: −1 Attacker access Operational Hosts: −0.1 The cost of decoy behavior, Maximum number of uses: −0.05, 15%Maxstep The cost of restore action: −1 The cost of remove action: −0.01 The cost of sleep action: 0 The cost of perceive action: −0.001

taking approximately 15 h. Among the hyperparameters, the design of rewards is worth discussing. On one hand, the value of server resources is much higher than that of hosts. On the other hand, the restore action consumes the most defense resources, the decoy action has usage limitations, and the perceive action is a cost-effective and efficient defense method. Based on these considerations, the following reward hyperparameters are designed.

5.2.2. Training of different agents and overall training status

The bottom-up training process is the fundamental training approach in hierarchical reinforcement learning. In this paper, the decoy agent and recover agent are first trained individually, followed by training the overall model using hierarchical reinforcement learning.

To incorporate the functionality of perceiving the safety situation, the perceive action is trained in conjunction with both types of agents during the training process. The most valuable part of the defense strategy lies in the game played against the attack behavior. The sleep action, which simply skips a game round, is a straightforward operation and is not discussed further in this context.

In Figs. 6, we present important data regarding the training of the hierarchical reinforcement learning model. In (a), we provide the reward statistics during the training process. Overall, it can be observed that the defense capability of the individual agents is inferior to that of the collaborative defense by multiple agents. When comparing the sub-agents, the recover agent has a more complex action space compared to the decoy agent, resulting in faster convergence for the decoy agent.

In Figs. 6(b), the Policy Loss of the two types of agents trained using the PPO algorithm is shown. The slightly slower convergence of the decoy agent's policy loss compared to the recover agent's supports the observations made in (a).

In Figs. 6(c), the number of steps per episode is described. In (d) and (e), the accumulation of rewards per episode is illustrated for the case of 100 steps per episode. It can be seen that the majority of rewards are concentrated, with only some dispersion in the initial training phase. This indicates that the formation process of defense strategies is relatively stable, and the agents explore various situations during the initial evolution of the game.

5.2.3. Agent defense capability assessment

After the hierarchical reinforcement learning agents form collaborative defense strategies, the paper conducts strategy evaluation and analysis. The capabilities of the defense agents trained through reinforcement learning are demonstrated based on two aspects: the distribution of autonomously formed defense strategies and the stability of the strategies.

The stability of the strategy can to some extent reflect the effectiveness of the method proposed in this paper when applied to different attack scenarios. In more complex attack scenarios, such as zero-day vulnerabilities, advanced persistent threats (APTs), and supply chain attacks, although the attack methods and techniques may vary, from a defensive perspective, the attack process generally involves reconnaissance, vulnerability exploitation, lateral movement, impact, and exfiltration. In other words, the increasing complexity of attack methods better conceals the attacker's intentions, but does not change the essential process of the attack. The core destructive effect of the attack occurs during the impact phase. By coordinating multiple defense measures to formulate defense strategies from the perspective of the "process", it is possible to detect malicious behaviors before the attack occurs and then implement security policies to prevent the attack. Even when an attack has occurred and caused damage, appropriate security policies can be implemented to achieve resilient defense.

To evaluate the strategies, the trained agents, initially trained with 100 steps per episode, are placed in scenarios with 200 steps and 300 steps per episode. The evaluation results of 1000 episodes are presented in Fig. 7. Additionally, Table 5 provides data statistics analysis for the three scenarios.

In Fig. 7, it can be observed that as the number of steps increases by 100, the reward fluctuation also increases. However, the incremental consumption of defense resources remains relatively constant. Table 5 provides statistics showing that the increase in defense resource consumption is approximately 15 units of reward, with a standard deviation increase of around 3 units.

These data reflect two key points. First, as the temporal dimension of the game process increases, the complexity of the game space also increases, leading to increased instability in the strategies. These three factors are positively correlated, highlighting the importance

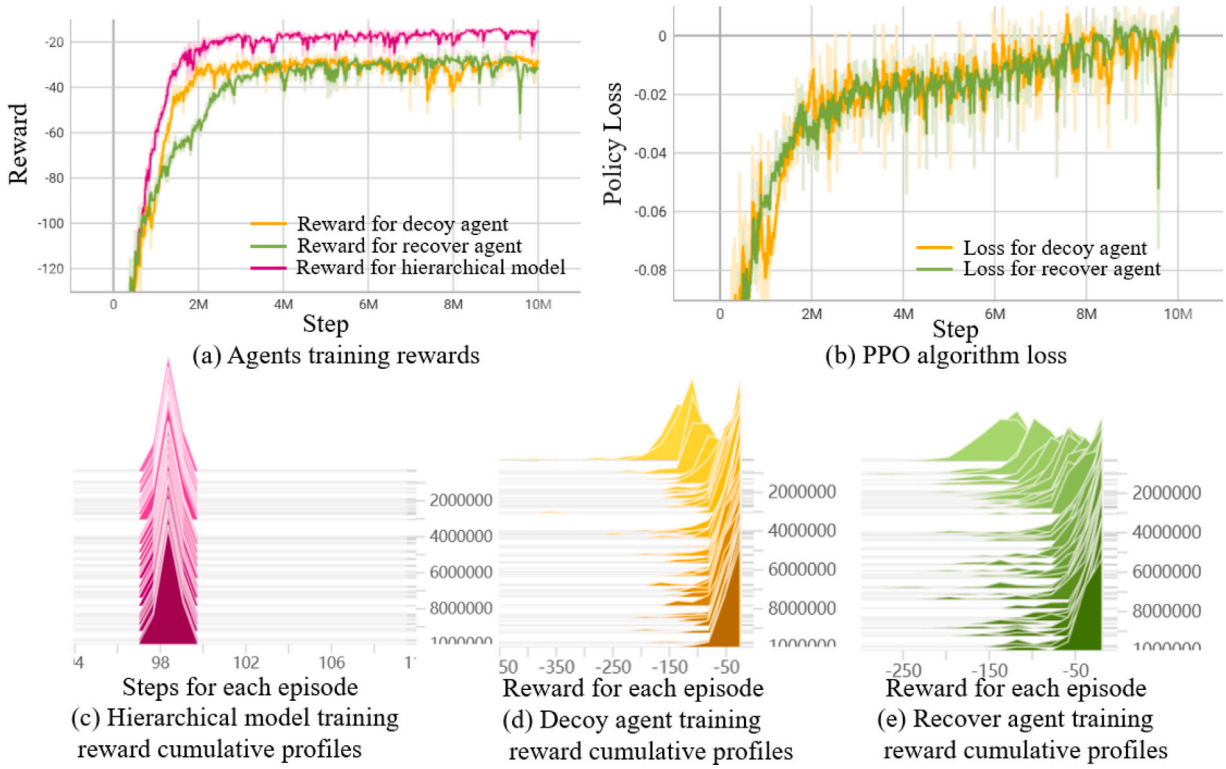

Fig. 6. Defense resource consumption in attack-defense games of different lengths.

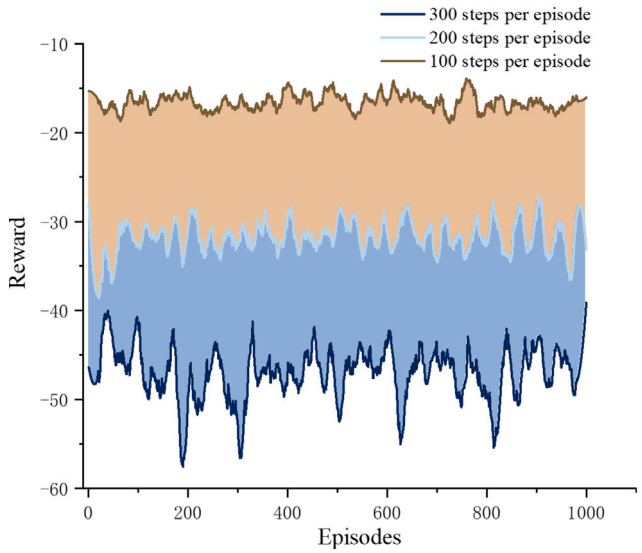

Fig. 7. Defense resource consumption for offensive and defensive games of different lengths.

of constructing autonomous defense decision-making capabilities in cyberspace.

Second, the consistent incremental consumption of defense resources indicates the effectiveness of the hierarchical reinforcement learning method in constructing autonomous network defense strategies. The defense strategies formed have the ability to maintain a linear time span of elastic defense with linearly growing resource consumption, which is challenging for human experts to achieve. The almost unchanged standard deviation increment demonstrates the relative stability of autonomous network defense strategies even in the face of increased complexity in the game space.

Some interesting conclusions are shown in the relevant results of competitions with some literature and related toolkits related to this article. We have selected some control data given in Table 6.

To align the test environment of this article and ensure variable control, the results given in the table are the same as the simulation environment and evaluation methods of this article. The data of CCS and CardiffUni are from the official data of the TTCP CAGE Challenge 2 competition, and Wiebe's results are from the literature (Wiebe et al., 2023).

The data of CCS is a baseline test for this simulation environment, and four types of tests have been done on the environment. Firstly, it is the impact on the network environment by the attacker in the absence of a defender. Secondly, the defense effect of executing defense measures with a random defense strategy. Thirdly, the result of using a simple rule defense agent, specifically, the result of executing restore immediately upon detecting a change in host permissions. Fourthly, the evaluation using the standard single-agent PPO algorithm. For the simulation environment, the attacker's attack strategy can cause serious damage to network facilities. For the defender, the intelligence level of the defense strategy is absolutely key to the defense effect. CardiffUni is the champion team in the TTCP CAGE Challenge 2 competition. The important reason for their solution to achieve such excellent results is the manual analysis of the decoy placement action strategy. This effort shows that for reinforcement learning methods, the overly large strategy search space is still a huge challenge for the learning ability of the agent. Pruning the action space manually and giving some deterministic experiences in the initialization stage can help form intelligent strategies. Their specific exploration steps are open on github.[1] Wiebe and others have given a solution of multi-agent reinforcement learning. The authors respectively apply the independent Q-learning (Tan, 1993) algorithm and QMIX algorithm (Rashid et al., 2020) to discuss the problem of network defense strategy generation. From the experimental

[1] https://github.com/john-cardiff/-cyborg-cage-2

Table 5
Results statistics of attack-defense games of different lengths.

Game length (steps)	Mean	Standard deviation	Mean value change	Standard deviation change
100	−16.5752	4.027	−0	+0
200	−31.7533	7.014	−15.1781	+2.987
300	−47.0622	9.785	−15.3089	+2.771

Table 6
Comparison of related work.

Name	Method	Grades	Clarification
CCS (Cyber autonomy gym for experimentation challenge 2, 2022b)	Sleeper	−1134.03	Do nothing
	Random	−726.92	Random defender agent
	Heuristic	−184.34	Defender react restore
	PPO	−30.19	RL Benchmark
CardiffUni (Cyber autonomy gym for experimentation challenge 2, 2022b)	PPO + Greedy Decoys	−13.76	Intervention for Deploying Decoys
Wiebe (Wiebe et al., 2023)	IQL	−18.17 ± 3.86	No communication
	QMIX	−16.75 ± 3.48	Implicit communication
	Heuristic	−31.9 ± 20.73	Rule based
Our paper	Our method	−16.6 ± 4.027	Task breakdown

results they gave, the multi-agent cooperation method is better able to solve complex decision problems, although QMIX is an implicit multi-agent cooperation method.

Compared with the above work, the method in this article is almost the same as the best performance in defense performance, but the method in this article has two advantages. Firstly, all the above work is discussing the problem of strategy generation in the field of reinforcement learning, and the interpretability of the strategy generation process is not high. This article uses the game framework to qualitatively and quantitatively model network attack and defense conflicts, providing an interpretive framework for reinforcement learning methods to iterate defense strategies. Secondly, this article uses hierarchical reinforcement learning methods to decouple the defense decision problem, reduce the dimension of the complex defense action space, agree on explicit cooperation constraints, and fully demonstrate the cooperative defense behavior of multiple agents.

5.2.4. Analysis of agent defense strategies

Based on the evaluation data of the agents, the paper conducts the following analysis of the distribution of agent strategies. The actions of both attack and defense strategies are categorized and statistically analyzed according to their types. The statistical results are presented in Figs. 8, where (a) and (b) represent the distribution of actions for the defender and attacker, respectively, under 100 steps, while (c) and (d) represent the distribution of actions for the defender and attacker, respectively, under 200 steps.

By comparing Figs. 8(a) and (b) with (c) and (d) vertically, we can confirm that the increase in the temporal dimension of the game space does not affect the stability of autonomous defense strategies. The distribution patterns of (a) and (c) are similar. Through the horizontal comparison of (a) and (c) with (b) and (d), it confirms the stability of the Stackelberg equilibrium in the game model framework. This reflects that if there is no change in the attack strategy, the optimal solution for defense response actions remains stable.

Through detailed analysis of Figs. 8(a) and (c), we can infer that defense strategies are complex mixed strategies. The distribution of Scan, Analyse, and Remove actions is relatively evenly spread, indicating that the defense agent effectively utilizes these defense behaviors. The concentrated distribution and smaller statistical values of Decoy and Restore actions suggest that our adjustment of reward hyperparameters has been incorporated into the defense agent's strategic considerations.

From the relevant results shown in Figs. 8, we can conclude that by increasing the length of the attack-defense interaction, our model's defense strategy demonstrates excellent statistical stability in its strategy profile. Therefore, we select one evaluation episode to analyze the consumption of defense resources. We extract the results of one episode of a 100-step scenario and analyze the resource consumption in the face of different types of attack techniques. The statistical results are given in Fig. 9.

Fig. 9 shows the actions taken by the attacker and defender respectively in the 100 steps, which is a statistical cross-section of the attack-defense resource consumption. Combined with specific information, the attacker enters the attack-defense simulation environment using "Discover Remote System", uses 7 steps of action resources to complete the detection of the network, and makes 77 malicious connections to the hosts and server resources in the network. Out of the 77 malicious connections, the "Exploit Remote Services" behavior succeeded in breaking through 15 times, and executed the "Escalate" action 15 times. In response to the above attack strategy, the defense agent triggered 11 analysis actions, which are mainly used to determine the source of the attack, and executed 13 decoy deployments, protecting important service resources in the system. The defense agent also learned to deploy decoy resources in advance when the attacker just entered the network environment and did not carry out effective attacks. It is worth noting that out of the 77 malicious connections, 62 were stopped by the "Remove" operation in the attempt stage, and the 15 successful breakthroughs were responded to by 14 "Restore" actions, that is, the 77 attack attempts were perceived and responded to by the defender 76 times.

6. Conclusion

In response to the inefficiency of human expert network security strategy organization and the difficulty of intelligent defense decision-making in complex environments, we propose an innovative HMARL framework for autonomously and efficiently formulating network security strategies and defense behavior decisions. This framework integrates game theory and hierarchical reinforcement learning theory. Firstly, it establishes a SHM of cyberspace conflict, describing the multi-level dynamic defense collaborative response mechanism under incomplete information. Secondly, through multi-agent reinforcement learning to simulate the game evolution dynamics process, it sequentially solves the Nash equilibrium to obtain dynamic autonomous defense strategies. Finally, using the hierarchical reinforcement learning paradigm, the defense decision problem is decoupled, significantly reducing the difficulty of exploration and efficiently learning collaborative defense strategies. This method not only enhances the automation and efficiency of defense strategy formulation, meeting the needs of real-time decision-making, but also enhances the adaptability to complex and changing network environments. We lay the theoretical and technical foundation for building an autonomous intelligent network

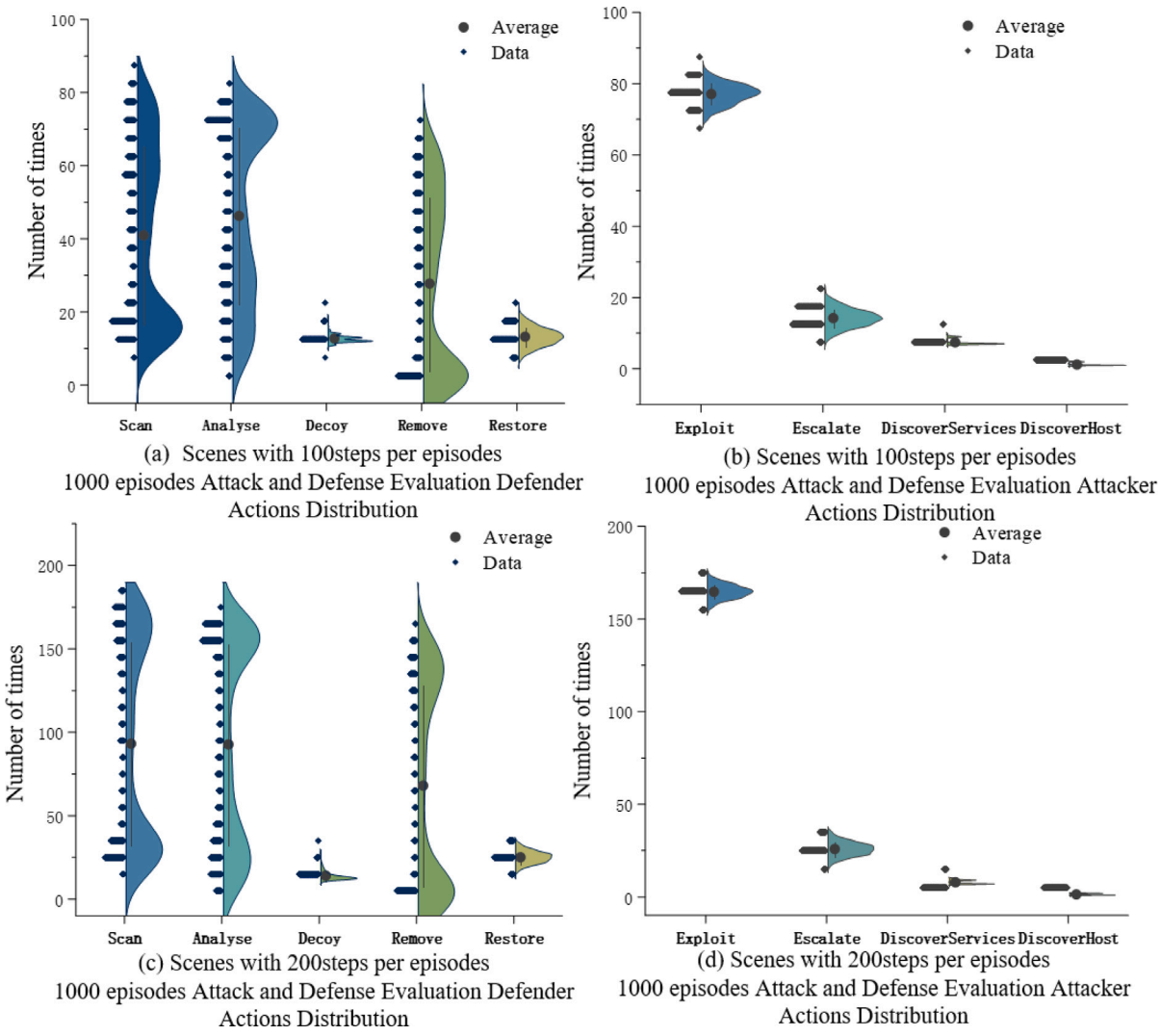

Fig. 8. Action distribution statistics in attack and defense strategies.

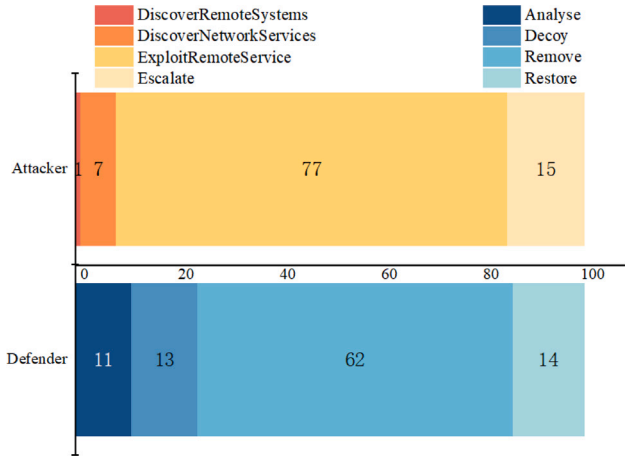

Fig. 9. Resource consumption in attack-defense interaction.

security defense system, which will promote the development of network security towards autonomous intelligence and strengthen the active defense capability of cyberspace.

The HMARL framework proposed in this paper is an efficient autonomous control and optimization method, particularly suitable for scenarios with adversarial interactions. Many cybersecurity scenarios can be abstracted into such characteristics, and it is foreseeable that applying this framework to the software vulnerability management domain can achieve autonomous vulnerability discovery, analysis, assessment, and tracking during security penetration testing. More importantly, its continuous feedback control feature can well adapt to the vulnerability management throughout the software lifecycle. Furthermore, extending this framework to the data privacy protection domain can enable a collaborative data protection mechanism among multiple users. Users can collaboratively formulate data sharing protocols, control data aggregation and linking processes, and strike a balance between preventing data leakage and maximizing the richness of the data set.

7. Limitations and future work

Based on the above theoretical modeling and model evaluation work, although we have applied new methods to discuss the problem of autonomous network defense strategy generation, it is inevitable that there are still the following limitations in this article: In terms of attack-defense game modeling, firstly, the model construction in this article is based on a series of assumptions, but the real cyberspace conflict does not conform to such perfect mathematical characteristics in the

quantification process, such as it is difficult to characterize man-made accidental factors; secondly, the game iteration process in this article is based on a rational exploration process, but real network attack methods are not all explicitly rational and interpretable, for example, the action sequence relationship of advanced persistent threat is not easy to be captured, and for another example, denial of service (DoS) attack is a violent destruction of the system.

In terms of applying multi-agent reinforcement learning methods in the security field, firstly, this article discusses the feasibility and effectiveness of exploring the generation of artificial intelligence autonomous network defense strategies in an ideal simulation environment, and expands the application range and potential application scenarios of artificial intelligence network autonomous operation capabilities. However, it needs to be calmly faced that applying such methods in real network environments still needs to overcome many problems, such as: the construction of the world model, the reasonable design of the general-purpose reward function, and how to efficiently converge in the complex action space; secondly, the method in this article is limited to generating security strategies and real-time protection for general attack behaviors, and lacks discussion on specific attack types and the security of the model itself; finally, the scale of the model in this article is not large, but the relationship between the consumption of computing resources and the improvement of the model's general-purpose ability makes the model less practical.

Based on the above various limitations and the coherence of our research direction, In terms of cross-application of cutting-edge technologies, on the one hand, we will explore the application of interpretable multi-agent reinforcement learning in the field of network security, making AI-based autonomous network defense methods more reliable and easier to be understood by human experts, to achieve intelligent and artificial collaboration, and intelligent assistance in generating and implementing security policies. On the other hand, we will combine the cutting-edge advanced reinforcement learning methods with other directions in the field of network security, especially the security issues of distributed machine learning model and training, specifically the security discussion in the field of federated learning. In terms of exploring the limitations of the underlying theory, game theory is an important mathematical tool for analyzing conflicts in cyberspace security. However, its application to analyzing network security issues involves many assumed premises. Exploring new methods to break through these assumptions, and enable deeper discussion and representation of conflicts in cyberspace, is one of the future research work.

CRediT authorship contribution statement

Yunlong Tang: Writing – original draft, Methodology, Investigation. **Jing Sun:** Supervision, Investigation. **Huan Wang:** Writing – review & editing, Resources, Project administration. **Junyi Deng:** Data curation, Conceptualization. **Liang Tong:** Investigation, Formal analysis. **Wenhong Xu:** Software, Data curation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 62106053, in part by the Natural Science Foundation of Guangxi Province of China under Grant 2024GXNSFAA010242, in part by the Innovation Project of Guangxi Graduate Education under Grant YCSW2023478, in part by the Doctoral Fund of Guangxi University of Science and Technology under Grant XiaoKe Bo19Z33.

References

Adawadkar, A.M.K., Kulkarni, N., 2022. Cyber-security and reinforcement learning—A brief survey. Eng. Appl. Artif. Intell. 114, 105116.

Alshamrani, A., Alshahrani, A., 2023. Adaptive cyber defense technique based on multiagent reinforcement learning strategies. Intell. Autom. Soft Comput. 36 (3).

Anjum, I., Miah, M.S., Zhu, M., Sharmin, N., Kiekintveld, C., Enck, W., Singh, M.P., 2020. Optimizing vulnerability-driven honey traffic using game theory. arXiv preprint arXiv:2002.09069.

Applebaum, A., Dennler, C., Dwyer, P., Moskowitz, M., Nguyen, H., Nichols, N., Park, N., Rachwalski, P., Rau, F., Webster, A., et al., 2022. Bridging automated to autonomous cyber defense: Foundational analysis of tabular q-learning. In: Proceedings of the 15th ACM Workshop on Artificial Intelligence and Security. pp. 149–159.

Apruzzese, G., Andreolini, M., Marchetti, M., Venturi, A., Colajanni, M., 2020. Deep reinforcement adversarial learning against botnet evasion attacks. IEEE Trans. Netw. Serv. Manag. 17 (4), 1975–1987.

Bakker, C., Bhattacharya, A., Chatterjee, S., Vrabie, D.L., 2020. Hypergames and cyber-physical security for control systems. ACM Trans. Cyber-Phys. Syst. 4 (4), 1–41.

Bilinski, M., Ferguson-Walter, K., Fugate, S., Gabrys, R., Mauger, J., Souza, B., 2019. You only lie twice: A multi-round cyber deception game of questionable veracity. In: International Conference on Decision and Game Theory for Security. Springer, pp. 65–84.

Cardellini, V., Casalicchio, E., Iannucci, S., Lucantonio, M., Mittal, S., Panigrahi, D., Silvi, A., 2022. Irs-partition: An intrusion response system utilizing deep Q-networks and system partitions. SoftwareX 19, 101120.

Carmona, G., 2012. Existence and stability of Nash equilibrium. World scientific.

Cheah, M., Stone, J., Haubrick, P., Bailey, S., Rimmer, D., Till, D., Lacey, M., Kruczynska, J., Dorn, M., 2023. CO-DECYBER: Co-operative decision making for cybersecurity using deep multi-agent reinforcement learning. In: European Symposium on Research in Computer Security. Springer, pp. 628–643.

Chen, L., Leneutre, J., 2009. A game theoretical framework on intrusion detection in heterogeneous networks. IEEE Trans. Inf. Forensics Secur. 4 (2), 165–178.

Cheng, Z., Chen, G., Hong, Y., 2022. Single-leader-multiple-followers stackelberg security game with hypergame framework. IEEE Trans. Inf. Forensics Secur. 17, 954–969.

Cyber autonomy gym for experimentation challenge 2, 2022b. https://github.com/cage-challenge/cage-challenge-2, 2022. Created by Maxwell Standen, David Bowman, Son Hoang, Toby Richer, Martin Lucas, Richard Van Tassel, Phillip Vu, Mitchell Kiely.

Cyber operations research gym, 2022a. https://github.com/cage-challenge/CybORG, 2022. Created by Maxwell Standen, David Bowman, Son Hoang, Toby Richer, Martin, 2022. Lucas, Richard Van Tassel, Phillip Vu, Mitchell Kiely, KC C., Natalie Konschnik, Joshua Collyer.

Du, Y., Song, Z., Milani, S., Gonzales, C., Fang, F., 2022. Learning to play an adaptive cyber deception game. In: Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems, vol. 6, Auckland, New Zealand.

Elderman, R., Pater, L.J., Thie, A.S., Drugan, M.M., Wiering, M.A., 2017. Adversarial reinforcement learning in a cyber security simulation. In: 9th International Conference on Agents and Artificial Intelligence. ICAART 2017, SciTePress Digital Library, pp. 559–566.

Guide, A.H., 2006. Infinite dimensional analysis. Springer.

Hammar, K., Stadler, R., 2021. Learning intrusion prevention policies through optimal stopping. In: 2021 17th International Conference on Network and Service Management. CNSM, IEEE, pp. 509–517.

Hu, P., Li, H., Fu, H., Cansever, D., Mohapatra, P., 2015. Dynamic defense strategy against advanced persistent threat with insiders. In: 2015 IEEE Conference on Computer Communications. INFOCOM, IEEE, Kowloon, Hong Kong, pp. 747–755.

Huang, Y., Chen, J., Huang, L., Zhu, Q., 2020. Dynamic games for secure and resilient control system design. Natl. Sci. Rev. 7 (7), 1125–1141.

Huang, S., Zhang, H., Wang, J., Huang, J., 2018. Markov differential game for network defense decision-making method. IEEE Access 6, 39621–39634.

Huang, L., Zhu, Q., 2020. A dynamic games approach to proactive defense strategies against advanced persistent threats in cyber-physical systems. Comput. Secur. 89, 101660.

Jin, Q., Wang, L., 2020. Zero-trust based distributed collaborative dynamic access control scheme with deep multi-agent reinforcement learning. EAI Endorsed Trans. Secur. Saf. 8 (27).

Khouzani, M., Liu, Z., Malacaria, P., 2019. Scalable min-max multi-objective cybersecurity optimisation over probabilistic attack graphs. European J. Oper. Res. 278 (3), 894–903.

Kong, G., Chen, F., Yang, X., Cheng, G., Zhang, S., He, W., 2023. Optimal deception asset deployment in cybersecurity: A Nash Q-learning approach in multi-agent stochastic games. Appl. Sci. 14 (1), 357.

Kovach, N.S., Gibson, A.S., Lamont, G.B., 2015. Hypergame theory: a model for conflict, misperception, and deception. Game Theory 2015.

Li, X., Hu, X., Jiang, T., 2023. Dual reinforcement learning based attack path prediction for 5g industrial cyber-physical systems. IEEE Internet Things J..

Li, T., Zhu, K., Luong, N.C., Niyato, D., Wu, Q., Zhang, Y., Chen, B., 2022. Applications of multi-agent reinforcement learning in future internet: A comprehensive survey. IEEE Commun. Surv. Tutor. 24 (2), 1240–1279.

Liang, X., Xiao, Y., 2012. Game theory for network security. IEEE Commun. Surv. Tutor. 15 (1), 472–486.

Liu, Z., Yin, X., Hu, Y., 2020. CPSS LR-ddos detection and defense in edge computing utilizing dcnn Q-learning. IEEE Access 8, 42120–42130.

Liu, L., Zhang, L., Liao, S., Liu, J., Wang, Z., 2021. A generalized approach to solve perfect Bayesian Nash equilibrium for practical network attack and defense. Inform. Sci. 577, 245–264.

Milani, S., Shen, W., Chan, K.S., Venkatesan, S., Leslie, N.O., Kamhoua, C., Fang, F., 2020. Harnessing the power of deception in attack graph-based security games. In: Decision and Game Theory for Security: 11th International Conference, GameSec 2020, College Park, MD, USA, October 28–30, 2020, Proceedings 11. Springer, pp. 147–167.

Nguyen, T.T., Reddi, V.J., 2023. Deep reinforcement learning for cyber security. IEEE Trans. Neural Netw. Learn. Syst. 34 (8), 3779–3795.

Nguyen, T., Xu, H., 2019. Imitative attacker deception in stackelberg security games. In: IJCAI. pp. 528–534.

Rashid, T., Samvelyan, M., De Witt, C.S., Farquhar, G., Foerster, J., Whiteson, S., 2020. Monotonic value function factorisation for deep multi-agent reinforcement learning. J. Mach. Learn. Res. 21 (178), 1–51.

Saeed, I.A., Selamat, A., Rohani, M.F., Krejcar, O., Chaudhry, J.A., 2020. A systematic state-of-the-art analysis of multi-agent intrusion detection. IEEE Access 8, 180184–180209.

Schlenker, A., Thakoor, O., Xu, H., Fang, F., Tambe, M., Tran-Thanh, L., Vayanos, P., Vorobeychik, Y., 2018. Deceiving cyber adversaries: A game theoretic approach. In: AAMAS'18: Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems. IFAAMAS, pp. 892–900.

Sengupta, S., Kambhampati, S., 2020. Multi-agent reinforcement learning in bayesian stackelberg markov games for adaptive moving target defense. arXiv preprint arXiv:2007.10457.

Shi, W., Li, J., Wu, H., Zhou, C., Cheng, N., Shen, X., 2020. Drone-cell trajectory planning and resource allocation for highly mobile networks: A hierarchical DRL approach. IEEE Internet Things J. 8 (12), 9800–9813.

Tan, M., 1993. Multi-agent reinforcement learning: Independent vs. cooperative agents. In: Proceedings of the Tenth International Conference on Machine Learning. pp. 330–337.

Tran, K., Akella, A., Standen, M., Kim, J., Bowman, D., Richer, T., Lin, C.-T., 2021. Deep hierarchical reinforcement agents for automated penetration testing. arXiv preprint arXiv:2109.06449.

Wang, S., Pei, Q., Wang, J., Tang, G., Zhang, Y., Liu, X., 2020. An intelligent deployment policy for deception resources based on reinforcement learning. IEEE Access 8, 35792–35804.

Waniek, M., Michalak, T.P., Alshamsi, A., 2019. Strategic attack & defense in security diffusion games. ACM Trans. Intell. Syst. Technol. 11 (1), 1–35.

Wiebe, J., Mallah, R.A., Li, L., 2023. Learning cyber defence tactics from scratch with multi-agent reinforcement learning. arXiv preprint arXiv:2310.05939.

Xu, X., Hu, H., Liu, Y., Tan, J., Zhang, H., Song, H., 2022. Moving target defense of routing randomization with deep reinforcement learning against eavesdropping attack. Digit. Commun. Netw. 8 (3), 373–387.

Zhan, Z., Xu, M., Xu, S., 2013. Characterizing honeypot-captured cyber attacks: Statistical framework and case study. IEEE Trans. Inf. Forensics Secur. 8 (11), 1775–1789.

Zhang, Y., Malacaria, P., 2021. Bayesian Stackelberg games for cyber-security decision support. Decis. Support Syst. 148, 113599.

Zhang, H., Wang, J., Yu, D., Han, J., Li, T., 2015. Active defense strategy selection based on static Bayesian game. In: Third International Conference on Cyberspace Technology. CCT 2015, IET, pp. 1–7.

Zhong, F., Hu, P., Zhang, G., Li, H., Cheng, X., 2022. Reinforcement learning based adversarial malware example generation against black-box detectors. Comput. Secur. 121, 102869.

Zhu, M., Anwar, A.H., Wan, Z., Cho, J.-H., Kamhoua, C.A., Singh, M.P., 2021. A survey of defensive deception: Approaches using game theory and machine learning. IEEE Commun. Surv. Tutor. 23 (4), 2460–2493.

Zhu, Q., Rass, S., 2018. Game theory meets network security: A tutorial. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. pp. 2163–2165.

Zolotukhin, M., Kumar, S., Hämäläinen, T., 2020. Reinforcement learning for attack mitigation in SDN-enabled networks. In: 2020 6th IEEE Conference on Network Softwarization. NetSoft, IEEE, pp. 282–286.