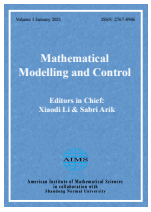
Mathematical Modelling and Control

https://www.aimspress.com/journal/mmc



MMC, 4(4): 424–438.
DOI: 10.3934/mmc.2024034
Received: 17 April 2024
Revised: 17 June 2024
Accepted: 06 August 2024
Published: 17 December 2024

Research article

Learning-based DoS attack game strategy over multi-process systems

Zhiqiang Hang[1], **Xiaolin Wang**[1,2,*], **Fangfei Li**[1,*], **Yi-ang Ren**[1] **and Haitao Li**[3]

[1] School of Mathematics, East China University of Science and Technology, Shanghai 200237, China

[2] Key Laboratory of System Control and Information Processing, Ministry of Education, Shanghai 200240, China

[3] School of Mathematics and Statistics, Shandong Normal University, Jinan 250014, China

* **Correspondence:** Email: xiaolinwang@ecust.edu.cn, lifangfei@ecust.edu.cn.

Abstract: In cyber-physical systems, the state information from multiple processes is sent simultaneously to remote estimators through wireless channels. However, with the introduction of open media such as wireless networks, cyber-physical systems may become vulnerable to denial-of-service attacks, which can pose significant security risks and challenges to the systems. To better understand the impact of denial-of-service attacks on cyber-physical systems and develop corresponding defense strategies, several research papers have explored this issue from various perspectives. However, most current works still face three limitations. First, they only study the optimal strategy from the perspective of one side (either the attacker or defender). Second, these works assume that the attacker possesses complete knowledge of the system's dynamic information. Finally, the power exerted by both the attacker and defender is assumed to be small and discrete. All these limitations are relatively strict and not suitable for practical applications. In this paper, we addressed these limitations by establishing a continuous power game problem of a denial-of-service attack in a multi-process cyber-physical system with asymmetric information. We also introduced the concept of the age of information to comprehensively characterize data freshness. To solve this problem, we employed the multi-agent deep deterministic policy gradient algorithm. Numerical experiments demonstrate that the algorithm is effective for solving the game problem and exhibits convergence in multi-agent environments, outperforming other algorithms.

Keywords: cyber-physical systems; game theory; DoS attack; AoI; multi-agent deep deterministic policy gradient

1. Introduction

Cyber-physical systems (CPSs) are multidimensional complex systems which integrate computing, network, and physical environments [1]. Through the organic integration and deep collaboration of computing, communication, and control (3C) technology, they achieve real-time perception, dynamic control, and information services for large-scale engineering systems [2]. Nowadays, they are widely applied in many fields, such as intelligent transportations [3] and health monitoring [4]. However, in recent years, a series of cyber attacks targeting CPSs have occurred, which have caused significant resource and economic losses. Thus, it is necessary and imperative to put emphasis on CPS security. A mainstream viewpoint is to study attack methods targeting CPSs. By studying attack methods, one can observe the worst performance of the CPSs, and design corresponding defense strategies to improve system security.

There are many different forms of cyber attacks, such as DoS attacks [2], false data injection attacks [5], and man-in-the-middle attacks [6]. Based on the mode of interference with the system, cyber attacks can be mainly categorized into three types [7]: confidentiality attacks, integrity attacks, and availability attacks. Confidentiality attacks involve eavesdropping on the system without the capability to interfere. Integrity attacks involve intercepting and tampering with transmitted data, resulting in significant damage and imposing strict requirements on the attackers. Availability attacks aim to disrupt the transmission of the CPS. Among availability attacks, denial-of-service (DoS) attacks are some of the most common and feasible, and they have been widely studied in recent literature [8–10].

In earlier literature [11, 12], the probability of successful channel transmission under DoS attacks was simplified as a binary variable, with attackers having only two options: 1 (attacked) or 0 (not attacked). However, in practice scenarios, this probability is influenced by many factors, such as the channel model, transmission power,

and additional noise. Therefore, recent studies on DoS attacks have focused not only on identifying which channel to attack, but also on determining how much power to allocate to jam the transmission channel and reduce the probability of a successful transmission. For instance, the authors in [13] solved the optimal DoS attack allocation for multi-channel CPS systems with additive Gaussian noise by solving the Bellman equation under the Markov decision process (MDP). Meanwhile, [14] considered the DoS attack allocation strategy from two different perspectives. However, these approaches are based on a strong assumption that DoS attackers possess a comprehensive knowledge of the system, including its model and transmission power, which may not be realistic in real-world scenarios. In typical CPS attack scenarios, defenders, acting as system regulators, should ideally understand the complete dynamics or at least estimate the state of the system. In contrast, attackers typically gather information through eavesdropping and may lack specific knowledge about system dynamics and transmission power.

Consequently, many studies assume attackers have no prior knowledge of system states or transmission power and propose using reinforcement learning (RL) algorithms to manage uncertainty [15, 16]. For example, [17] introduced the Q-learning algorithm [18] to work out the optimal DoS attack strategy on a small scale, while [1] introduced the double deep Q network (DDQN) algorithm to calculate optimal DoS attack strategies. Building on this, the deep deterministic policy gradient (DDPG) algorithm was introduced to solve the optimal DoS attack allocation strategy for a multi-process CPS where action domains are continuous. It should be noted that the abovementioned papers predominantly focus on attack strategies from the attacker's perspective. The strategy of the defender is either fixed or considered only after calculating the optimal strategy of the attacker. There is still a research gap in addressing the game problem of asymmetric information [19], where both sides make simultaneous decisions.

In this paper, we consider a multi-channel CPS with time delays [20] in wireless transmission that involves a DoS attacker and a defender (referred to as the smart sensor). At each step, both the attacker and the defender need to determine how much energy to allocate to attack each channel within their limited energy budget. However, neither side possesses knowledge about the strategy employed by their opponent. Consequently, the problem at hand can be framed as an asymmetrical information DoS attack game. We aim to find the Bayesian Nash equilibrium strategy between the defender and the attacker. Compared with other existing DoS attack game studies [21], we not only consider the Bayesian Nash equilibrium strategy in game problems with asymmetric information, but also extend the feasible regions of both sides' actions to continuous situations, which is more adoptable in actual systems with different discrete accuracy.

To address the issue of asymmetric information between the defender and attacker, we employ RL methods. However, the previously mentioned algorithms such as Q-learning, DDQN, and DDPG can have significant instability in the face of complex agents, as any agent in the environment will be affected by other agents. In this paper, we study a deep reinforcement learning algorithm called the multi-agent deep deterministic policy gradient (MADDPG) to deal with game problems in multi-agent scenarios. By introducing feasible layers, the algorithm can be applied to multi-process CPSs.

In addition, we utilize a more comprehensive metric called age of information (AoI) [22] to characterize the freshness of information and update the estimated state (see [23–25]), which is widely studied in the field of communication recently. However, there is still a lack of relevant content in the game problem of DoS attacks. Compared with the definition of the holding time in previous works [1, 2], the introduction of AoI is effective to deal with the problem of time delays in CPSs. To demonstrate the effectiveness of the MADDPG algorithm in DoS attack game problems within multi-process CPS systems with time delays, a series of extensive experiments have been conducted in this paper.

The main contributions of this work are summarized as follows:

1) In the context of a CPS facing DoS attacks, considering that DoS attacks are often confused with transmission delays, we introduces the concept of AoI to more effectively measure the freshness of the received data. Compared to prior literature that used the holding time to describe transmission delays, this paper offers enhanced estimation accuracy. Consequently, the findings of our study carry broader applicability and practical significance in mitigating DoS attacks coupled with time delays.
2) We establish an asymmetric sensor scheduling game problem for CPS systems under DoS attacks. In contrast to the majority of the prior studies, which focused on optimizing strategies separately for

attackers and defenders, we address the optimal problem considering both perspectives simultaneously. While some studies have explored real-time asymmetric information DoS attack game issues, their action spaces were limited to discrete cases due to methodological constraints. In comparison, the MADDPG algorithm introduced in our work can handle not only discrete but also continuous action settings. It is worth mentioning that we have incorporated two feasibility layers in the neural network to restrict network output actions, adapting to multi-channel CPS systems affected by time delays.

3) We demonstrate the effectiveness of the MADDPG-based algorithm for this problem through several illustrative examples, calculating the Bayesian Nash equilibrium points for different weights and the overall cumulative discount rewards for the attacker and defender at each time step.

The remainder of this paper is organized as follows: In Section 2, the system model and the DoS attack game-theoretic model are presented. The MADDPG-based algorithm to the game problem with feasibility layers is proposed in Section 3. In Sections 4 and 5, simulation results and the conclusion are provided, respectively.

Notations: $\mathbb{N}$ and $\mathbb{R}$ denote the sets of natural and real numbers, respectively. $\mathbb{N}^+$ and $\mathbb{R}^+$ represent the set of positive natural numbers and the set of non-negative real numbers, respectively. $\mathbb{S}_+^n$ is the set of $n \times n$ positive semi-definite matrices. $\mathbb{R}^n$ is the $n-$dimensional Euclidean space. $\mathbb{R}_+^n$ is the $n-$dimensional Euclidean space of non-negative real numbers. For a matrix X, X^T, $Tr(X)$, and $|X|$ stand for the transpose, trace, and determinant of X, respectively. $\|\cdot\|_1$ is the l_1 norm. The notation $\mathbb{E}[\cdot]$ stands for the expectation of a random variable. $\mathcal{N}$ denotes the standard Gaussian distribution. The spectral radius of a matrix is $\rho(\cdot)$. $\Pr(\cdot|\cdot)$ is the conditional probability. ∇ is the vector gradient operator. $\mathrm{Dim}(\cdot)$ is an abbreviation of dimension. $\mathrm{ReLU}(x) = \max(0, x)$. $\mathrm{Sigmoid}(x) = \frac{e^x}{e^x+1}$.

2. Problem setup

2.1. System model

As shown in Figure 1, the CPS is composed of discrete linear time-invariant processes, smart sensors, wireless channels, a remote estimator, and a DoS attacker. The dynamics of the i-th process ($1 \le i \le N, i \in \mathbb{R}^+$) is given by:

$$x_{k+1,i} = A_i x_{k,i} + \omega_{k,i}, \tag{2.1}$$

$$y_{k,i} = C_i x_{k,i} + \upsilon_{k,i}, \tag{2.2}$$

where $x_{k,i} \in \mathbb{R}^n$ and $y_{k,i} \in \mathbb{R}^m$ are the state and the measurement of the i-th channel, respectively. The matrices A_i and C_i are system parameters. The variables $\omega_{k,i} \in \mathbb{R}^n$ and $\upsilon_{k,i} \in \mathbb{R}^m$ indicate independent and identically distributed Gaussian noises, with their distributions given as $\omega_{k,i} \sim \mathcal{N}(0, \Sigma_{\omega_i})$ and $\upsilon_{k,i} \sim \mathcal{N}(0, \Sigma_{\upsilon_i})$, resepectively, where $\Sigma_{\omega_i} \ge 0$ and $\Sigma_{\upsilon_i} > 0$. The initial state of the i-th process $x_{0,i} \in \mathbb{R}^n$ satisfies the Gaussian distribution as $x_{0,i} \sim \mathcal{N}(0, \Sigma_{x_{0,i}})$ with $\Sigma_{x_{0,i}} \ge 0$. Suppose that the variables $x_{0,i}$, $\omega_{k,i}$, and $\upsilon_{k,i}$ are mutually uncorrelated for $k \ge 0$. Moreover, we assume that (A_i, C_i) is observable and $(A_i, \sqrt{\Sigma}_{\omega_i})$ is controllable [26].

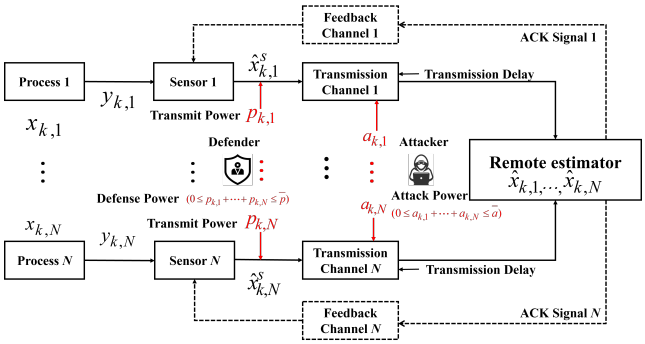


Figure 1. Multi-process system with transmission delay layout.

Taking the i-th process as an example, at every time step k, the measurement $y_{k,i}$ is obtained by the smart sensor i, which then employs a local Kalman filter [27] to estimate the actual state of the process, represented by $\hat{x}_{k,i}^s$. The corresponding estimation error covariance of the sensor is denoted by $P_{k,i}^s$, which is calculated by the following equation:

$$P_{k,i}^s = \mathbb{E}\left[(x_{k,i} - \hat{x}_{k,i}^s)(x_{k,i} - \hat{x}_{k,i}^s)^\mathsf{T} | y_{1,i}, \ldots, y_{k,i}\right], \tag{2.3}$$

where

$$\hat{x}_{0,i}^s = 0 \quad \text{and} \quad P_{0,i}^s = \Sigma_{x_{0,i}}.$$

Since $P_{k,i}^s$ converges exponentially to a certain steady state $\overline{P}_i$, we assume that

$$P_{k,i}^s = \overline{P}_i$$

when $k \ge l, l \in \mathbb{N}^+$.

After the i-th sensor calculates the estimation of $\hat{x}_{k,i}^s$, it will select transmit power $p_{k,i} \in \mathbb{R}^+$ to send $\hat{x}_{k,i}^s$ to the remote estimator through a wireless transmission channel.

However, due to the inherent properties and the transmission medium of the wireless channel, random time delays or packet loss often occur during the transmission. The time delay of the i-th channel at every step is denoted by $t_{d,i}$. A binary variable $\gamma_{k,i}$ is defined to represent whether the estimated state $\hat{x}^s_{k,i}$ is received successfully or not, as given by the following equation:

$$\gamma_{k,i} = \begin{cases} 1, & \hat{x}^s_{k,i} \text{ is received successfully,} \\ 0, & \text{otherwise.} \end{cases} \tag{2.4}$$

The probability of a successful transmission is denoted by:

$$\Pr(\gamma_{k,i} = 1 | p_{k,i}, \sigma_{k,i}) = f(p_{k,i}, \sigma_{k,i}), \tag{2.5}$$

where the scalar $\sigma_{k,i} \in \mathbb{R}^+$ is the noise power of the i-th channel, which follows a Gaussian distribution [13]. The function

$$f : \mathbb{R}^+ \times \mathbb{R}^+ \to [0, 1]$$

is determined by the specific wireless channel model and the selected modulation approach. Naturally, we suppose that f increases with $p_{k,i}$ and decreases with $\sigma_{k,i}$ according to [1,9]. That is, $\hat{x}^s_{k,i}$ is more likely to be transmitted successfully to the remote estimator with higher transmit power and less channel noise. The above assumption is intuitive and in accord with many practical communication channel models.

If $\hat{x}^s_{k,i}$ is successfully transmitted, the remote estimator will update its estimated state of the process $\hat{x}_{k,i}$ with $\hat{x}^s_{k,i}$. Conversely, if $\hat{x}^s_{k,i}$ is not obtained by the remote estimator, it will predict the current estimated state $\hat{x}_{k,i}$ according to the previous estimated state $\hat{x}_{k-1,i}$. The remote estimator utilizes a feedback channel to send an acknowledgment (ACK) signal back to the sensor, indicating whether $\hat{x}^s_{k,i}$ has been successfully transmitted ($\gamma_{k,i} = 1$ or 0).

2.2. Definition and impact of AoI

In order to better capture the impact of data timeliness, we introduce a definition, i.e., AoI, which is widely employed in the field of communication. It characterizes the information age (denoted by Δ_n) of the most recent packet received by the remote estimator. It is calculated by

$$\Delta_n = n - T_n,$$

where T_n is the most recent generation time of the latest received packet and n is the current time [22]. For the i-th process, denote $\Delta_{k,i}$ as its AoI at time step k. Therefore, the updates of $\Delta_{k,i}$ and $\hat{x}_{k,i}$, as well as the estimation error covariance $P_{k,i}$ of its remote estimator can be summarized as the following equations:

$$\Delta_{k,i} = \begin{cases} 0, & \gamma_{k,i} = 1, \\ \Delta_{k-1,i} + 1, & \text{otherwise,} \end{cases} \tag{2.6}$$

$$\hat{x}_{k,i} = \begin{cases} \hat{x}^s_{k,i}, & \gamma_{k,i} = 1, \\ A_i \hat{x}_{k-1,i} = A_i^{\Delta_{k,i}} \hat{x}^s_{k-\Delta_{k,i}}, & \text{otherwise,} \end{cases} \tag{2.7}$$

$$P_{k,i} = \begin{cases} \overline{P}_i, & \gamma_{k,i} = 1, \\ h_i(P_{k-1,i}) = h_i^{\Delta_{k,i}}(\overline{P}_i), & \text{otherwise,} \end{cases} \tag{2.8}$$

where function $h_i : \mathbb{S}^n_+ \to \mathbb{S}^n_+$ is defined as follows:

$$h_i(X) = A_i X A_i^{\mathsf{T}} + \Sigma_{w_i}. \tag{2.9}$$

According to the proof in [28], we can get

$$Tr(h_i(\overline{P}_i)) \geq Tr(\overline{P}_i).$$

In other words, a larger $\Delta_{k,i}$ results in a larger trace of the estimation error covariance $Tr(P_{k,i})$.

By introducing AoI, the systems are able to determine, to some extent, whether the time delay in the transmission channel is caused by a DoS attacker or by the transmission channel itself. We will discuss it in detail in the subsequent chapters.

2.3. DoS attack optimization problem

The objective of the attacker is to attack the transmission channels. At each time step k, the attacker exerts attack power $a_{k_i} \in \mathbb{R}^+$ as additional noise into the i-th channel, resulting in the following probability equation:

$$\Pr(\gamma_{k,i} = 1 | p_{k,i}, \sigma_{k,i}, a_{k,i}) = f(p_{k,i}, \sigma_{k,i} + a_{k,i}). \tag{2.10}$$

When $a_{k,i}$ is large, there is a higher probability of transmission failure for the estimated state, resulting in a relatively larger estimation error covariance $P_{k,i}$ and poor system performance. However, injecting attack power is costly and restricted, which forces the attacker to make a trade-off between attack performance and cost. In other words, the attacker should increase the estimation error covariance with potentially low cost. Assume that the attacker can eavesdrop the ACK signal from the feedback channels and the estimation error covariance from the transmission channels. Without loss of the generality, suppose the attack begins at time step $k = 1$. Then, from

the attacker's perspective, the attacker needs to consider the following optimization problem:

$$\begin{aligned} &\max_{\mathrm{a}_1,\mathrm{a}_2,\ldots} \ \mathbb{E}(\sum_{k=1}^{+\infty}\beta^{k-1}[\omega_A Tr(P_k)-\beta_A(1-\omega_A)\|\mathrm{a}_k\|_1]) \\ &s.t.\ \ \mathrm{a}_k\in\mathbb{R}_+^N,\ \ \|\mathrm{a}_k\|_1\le\overline{a},\ \ k\in\mathbb{N}^+,\ \ k\to\infty, \end{aligned} \tag{2.11}$$

where P_k represents a diagonal matrix composed of the covariance of the estimation error for each process, given by:

$$P_k = diag\{P_{k,1}, P_{k,2}, \cdots, P_{k,N}\}. \tag{2.12}$$

In addition,

$$\mathrm{a}_k = [a_{k,1}, \ldots, a_{k,i}, \ldots, a_{k,N}]^\mathsf{T}$$

represents the attack power at time step k for N channels in which $a_{k,i}$ is the attack power of the i-th channel, and $\omega_A \in [0,1]$ represents a weight that measures the trade-off between the attack performance and the attack cost. The positive constants $\overline{a}$ and β_A represent the upper bound for total attack power in a time step and the cost of the allocation per unit of attack power, respectively.

The variable $\beta \in [0,1)$ is the discount rate. To prevent the objective function from tending to infinity regardless of the selected attack policy, similar to [29], we assume that $\mathbb{E}(Tr(P_k))$ is bounded. Additionally, for each i, we assume that

$$\rho^2(A_i) > 1/\beta.$$

2.4. Game-theoretic model

In the presence of a DoS attack, each sensor needs to take defensive measures. Assuming that each sensor has the capability to inject defense power $p_{k,i}$ to make the transmission more likely to succeed, such as increasing the transmission power at an additional cost to reduce noise, the problem becomes a game between both sides (the defender and the attacker). With this in mind, from the perspective of the defender, we focus on optimizing the defender's power, i.e.,

$$\begin{aligned} &\max_{p_1,p_2,\ldots} \ \mathbb{E}(\sum_{k=1}^{+\infty}-\beta^{k-1}[\omega_D Tr(P_k)-\beta_D(1-\omega_D)\|p_k\|_1]) \\ &s.t.\ \ p_k\in\mathbb{R}_+^N,\ \ \|p_k\|_1\le\overline{p},\ \ k\in\mathbb{N}^+, \end{aligned} \tag{2.13}$$

where

$$p_k = [p_{k,1}, \ldots, p_{k,i}, \ldots, p_{k,N}]^\mathsf{T}$$

is the defense power at time step k for N channels in which $p_{k,i}$ is the defense power of the i-th channel, ω_D represents a weight measuring defense performance and defense cost. The positive constants $\overline{p}$ and β_D are the upper bound for the total defense power in a time step and the cost of the allocation per unit of defense power, respectively. Therefore, we can describe our game theoretic model by considering the following optimization problem:

Problem 1. Game-theoretic framework between a defender and an attacker under DoS attack.

For the attacker,

$$\begin{aligned} &\max_{\mathrm{a}_1,\mathrm{a}_2,\ldots} \ \mathbb{E}(\sum_{k=1}^{+\infty}\beta^{k-1}[\omega_A Tr(P_k)-\beta_A(1-\omega_A)\|\mathrm{a}_k\|_1]) \\ &s.t.\ \ \mathrm{a}_k\in\mathbb{R}_+^N,\ \ \|\mathrm{a}_k\|_1\le\overline{a},\ \ k\in\mathbb{N}^+,\ \ k\to\infty. \end{aligned} \tag{2.14}$$

For the defender,

$$\begin{aligned} &\max_{p_1,p_2,\ldots} \ \mathbb{E}(\sum_{k=1}^{+\infty}-\beta^{k-1}[\omega_D Tr(P_k)-\beta_D(1-\omega_D)\|p_k\|_1]) \\ &s.t.\ \ p_k\in\mathbb{R}_+^N,\ \ \|p_k\|_1\le\overline{p},\ \ k\in\mathbb{N}^+,\ \ k\to\infty. \end{aligned} \tag{2.15}$$

Remark 1. At each time step k, both the defender and the attacker select their power levels p_k and a_k, respectively, to impact the transmission channel based on their previous knowledge of the ACK signal. Unlike the research in [29], which demands full knowledge of the channel model and transmission power of the sensor for the attacker, both sides only require the ACK signal and the estimation error covariance received in the previous time step.

There are two challenges in solving this game-theoretic optimization problem. First, both sides lack information about the power allocation chosen by their opponent in the current time step, making it become an incomplete information dynamic game problem. Second, the power allocation for both sides is a vector of N dimensions in a continuous domain, leading to an infinite number of feasible choices.

3. MADDPG-based DoS attack and defend power design in a multi-channel process system

We intend to utilize deep reinforcement learning to solve Problem 1. Before introducing our method, we will first establish an MDP framework for Problem 1.

3.1. MDP formulation

The MDP is defined by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \beta)$, where $\mathcal{S}$ represents the state space, $\mathcal{A}$ denotes the action space,

$$\mathcal{P} : \mathcal{S}\times\mathcal{A}\times\mathcal{S} \to [0,1]$$

is the transition probability function,

$$r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$$

is the reward function, and $\beta \in [0, 1)$ is the discount rate. For Problem 1, the elements of the MDP model are as follows:

- State space $\mathcal{S}$: Since the defender and the attacker only have the previous state information, the state at time step k is defined as

  $$\Delta_{k-1} = [\Delta_{k-1,1}, \Delta_{k-1,2}, \cdots, \Delta_{k-1,N}]^{\mathsf{T}} \in \mathbb{R}_{+}^{N}.$$

  The state space includes all possible vectors, where all components $\Delta_{k-1,1}, \Delta_{k-1,2}, \cdots, \Delta_{k-1,N}$ are non-negative real numbers.

Remark 2. The state here represents the information for the defender and attacker to design the power allocation, which is different from the state of the dynamic systems (2.1) and (2.2).

- Action space $\mathcal{A}$: The global action of the allocation of attack and defense power at time step k is a set of all possible choices of $2N$-dimension vectors which concatenates defense power p_k and attack power a_k denoted by

  $$O_k = [(p_k)^{\mathsf{T}}, (\mathrm{a}_k)^{\mathsf{T}}]$$

  with

  $$0 \le \|p_k\|_1 \le \overline{p}$$

  and

  $$0 \le \|\mathrm{a}_k\|_1 \le \overline{a}.$$

- Transition function $\mathcal{P}$: The transition function is denoted by:

  $$\mathcal{P} = \Pr(\Delta_k | \Delta_{k-1}, p_k, \mathrm{a}_k) = \prod_{i=1}^{N} \Pr_i, \tag{3.1}$$

  where

  $$\Pr_i = \begin{cases} f(p_{k,i}, \sigma_{k,i} + \mathrm{a}_{k,i}), & \Delta_{k,i} = 0, \\ 1 - f(p_{k,i}, \sigma_{k,i} + \mathrm{a}_{k,i}), & \Delta_{k,i} = \Delta_{k-1,i} + 1, \\ 0, & \text{otherwise.} \end{cases} \tag{3.2}$$

  For simplicity, we assume that these N transmission channels are independent in the simulation. However, if this assumption does not hold and there is mutual interference between transmission channels, the corresponding function f will be replaced by:

  $$f(p_{k,i}, \sigma_{k,i} + \mathrm{a}_{k,i} + \sum\nolimits_{j=1, j\neq i}^{N} E_{ij}^{D} p_{k,j} + \sum\nolimits_{j=1, j\neq i}^{N} E_{ij}^{A} \mathrm{a}_{k,j}), \tag{3.3}$$

  where $E_{ij}^{D} p_{k,j}$ and $E_{ij}^{A} \mathrm{a}_{k,j}$ are the correlation coefficients of the defender and the attacker, respectively. Nevertheless, our learning-based method is suitable for both situations because it does not require either the defender or the attacker to possess any knowledge of the transition function.

- Reward: The one-step reward is defined as:
  (1) For the defender:

  $$r_D(\Delta_{k-1}, p_k, \Delta_k) = -\omega_D Tr(P_k) - \beta_D(1 - \omega_D)\|p_k\|_1. \tag{3.4}$$

  (2) For the attacker:

  $$r_A(\Delta_{k-1}, \mathrm{a}_k, \Delta_k) = \omega_A Tr(P_k) - \beta_A(1 - \omega_A)\|\mathrm{a}_k\|_1. \tag{3.5}$$

We denote the stationary strategies of the attacker and the defender as two functions which map the state δ_{k-1} to the power a_k and p_k as

$$\pi_A : \ \mathcal{S} \to \mathcal{A}, \pi_A \in \Pi_A$$

and

$$\pi_D : \ \mathcal{S} \to \mathcal{A}, \pi_D \in \Pi_D,$$

respectively, where Π_A and Π_D are the policy spaces that include all feasible stationary strategies of the attacker and the defender. Problem 1 can be converted into the consideration of the stationary strategies for the MDP model [30] as follows:

Problem 2. For the attacker:

$$\max_{\pi_A \in \Pi_A} \mathbb{E}\Big(\sum_{k=1}^{\infty} \beta^{k-1} r_A(\Delta_{k-1}, \pi_A(\Delta_{k-1}), \Delta_k)\Big). \tag{3.6}$$

For the defender:

$$\max_{\pi_D \in \Pi_D} \mathbb{E}\Big(\sum_{k=1}^{\infty} -\beta^{k-1} r_D(\Delta_{k-1}, \pi_D(\Delta_{k-1}), \Delta_k)\Big). \tag{3.7}$$

Remark 3. Denote $\pi_D^* \in \Pi_D$ and $\pi_A^* \in \Pi_A$ as the optimal strategies for the defender and the attacker, respectively. Based on [31], the existence of the stationary and deterministic optimal strategy can be proved. Define the Q-value function $Q^{\pi_D}(\Delta, p)$ and $Q^{\pi_A}(\Delta, \mathrm{a})$ of the MDP under the strategies π_D and π_A, respectively, as:

$$\begin{aligned} Q^{\pi_A}(\Delta, \mathrm{a}) &= \mathbb{E}(r_A(\Delta_0, \mathrm{a}_1, \Delta_1) \\ &+ \sum_{k=2}^{\infty} \beta^{k-1} r_A(\Delta_{k-1}, \pi_A, \Delta_k) | \Delta_0 = \Delta, \mathrm{a}_1 = \mathrm{a}), \end{aligned} \tag{3.8}$$

$$Q^{\pi_D}(\Delta, p) = \mathbb{E}(r_D(\Delta_0, p_1, \Delta_1) + \sum_{k=2}^{\infty} \beta^{k-1} r_D(\Delta_{k-1}, \pi_D, \Delta_k) | \Delta_0 = \Delta, p_1 = p), \tag{3.9}$$

where Δ_0 is the initial state. The functions $Q^{\pi_A}(\Delta, \mathrm{a})$ and $Q^{\pi_D}(\Delta, p)$ represent the expected cumulative reward of Δ when the attacker and defender take actions a and p, respectively, and follow their strategies π_A and π_D in the subsequent steps. Similarly, the optimal Q-value functions for the attacker and defender under their respective optimal strategies are denoted as $Q^{\pi_A^*}(\Delta, \mathrm{a})$ and $Q^{\pi_D^*}(\Delta, p)$. According to [31], if the state space and the action space are sufficiently small and discrete, $Q^{\pi_A^*}(\Delta, \mathrm{a})$ and $Q^{\pi_D^*}(\Delta, p)$ can be obtained easily by value iteration or policy iteration algorithms [32] when the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \beta)$ is accessible.

However, in Problem 2, there still exists three problems. One is that the action space is continuous, which leads to infinite strategies. Another one is that neither side has any knowledge of their opponent's actions. The last one is that the transition function $\mathcal{P}$ is determined by actions a_k and p_k of both sides, which means each side must consider their opponent's action when selecting their own action. In the following subsection, we present a MADDPG-based power allocation design algorithm that efficiently solves these challenges.

3.2. MADDPG algorithm

In order to deal with the large state space and the continuous action space, a deep learning-based algorithm is introduced, which utilizes the deep neural networks (DNNs) to parameterize the Q-value function. We divide the introduction of the MADDPG algorithm into several parts, i.e., the actor-critic algorithm, the replay buffer, double DNNs, and the updating rules.

3.2.1. Actor-critic algorithm

The actor-critic algorithm [33] is composed of two DNNs, i.e., actor and critic. It is the foundation of the MADDPG-based power allocation design algorithm. The actor contains a policy network whose structure is given by Figure 2. The input of the network is the state Δ (Δ is a time-varying variable, denoted as

$$\Delta(k) = \Delta_k,$$

where k represents the time step). Through several hidden layers such as activation function layers and fully connected layers, it outputs the corresponding action (p or a). The critic network includes an evaluation network which is shown in Figure 3. It is responsible for estimating the Q-value of the tuple (Δ, O) (O is also a time-varying variable similar to Δ) under the current strategy $\pi(\pi_D$ and $\pi_A)$. The inputs of the critic are composed of three components, the state Δ, the action generated from the actor, and the estimation of other agents' actions.

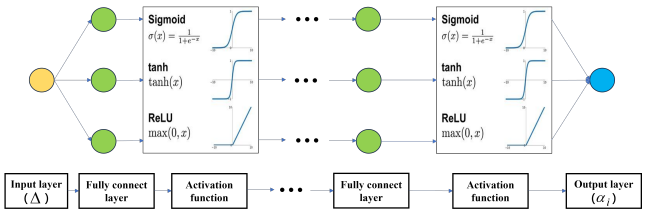


Figure 2. Policy network architecture of the actor.

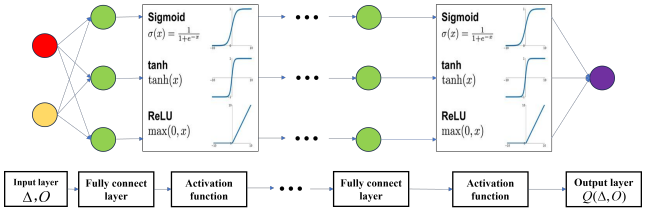


Figure 3. Evaluation network architecture of the critic.

Since the MADDPG algorithm takes into account the actions of other agents when considering the action of one agent, it needs to integrate the Q-value function and the actions of the multiple agents. Denote the centralized action-value function for the agent i as

$$Q_i^{\pi} = (\Delta, O) = (\Delta, (p, \mathrm{a})),$$

where the inputs are the actions of all agents and the state information.

Remark 4. Notice that there only exists two agents (the defender and the attacker) in Problem 1, and we define the defender and the attacker as agent 1 and agent 2, respectively (naturally, we set the number of agents as n=2). Correspondingly, denote

$$r_1 = r_D, \quad r_2 = r_A, \quad r_k = [r_1, \quad r_2], \quad \alpha_1 = p, \quad \alpha_2 = \mathrm{a}$$

for notational simplicity.

We denote the parameters of the policy network in the actor and the evaluation network in the critic of the i-th agent

as θ_i^{μ} and θ_i^{Q}, respectively. Thus the output of the policy network is given by:

$$\alpha_i = \mu_{\theta_i^{\mu}}(\Delta) = F_{a_L} \circ F_{c_L} \circ \cdots \circ F_{a_1} \circ F_{c_1}(\Delta), \tag{3.10}$$

where $\circ$ is the compound operation of functions,

$$F_{c_i}(x) = \theta_{\omega_i} x + \theta_{b_i}, \quad i = 1, 2.$$

The parameters ω_i and θ_{b_i} are the weights and biases. The form of F_{a_i} is determined by the selection of each activation function. The variable L represents the number of hidden layers. Similarly, denote the output of the evaluation network as:

$$Q_{\theta_i^Q}(\Delta, O) = F'_{a_L} \circ F'_{c_L} \circ \cdots \circ F'_{a_1} \circ F'_{c_1}(\Delta, O), \tag{3.11}$$

where the inputs are the state and the actions of all agents.

3.2.2. The replay buffer and double DNNs

Before giving the updating rules, we introduce two techniques that the MADDPG algorithm utilizes to increase the stability of the algorithms, namely, the replay buffer and double DNNs:

- Replay buffer: In MADDPG, the tuple $(\Delta_{k-1}, O_k, \Delta_k, r_k)$ obtained at time step k is stored in a buffer with a fixed size B. If the number of stored tuples exceeds B, the oldest one will be replaced by the newest one. In the updating progress, a mini batch of tuples from the replay buffer is sampled randomly to calculate the corresponding gradient and error. The introduction of the replay buffer solves the problem of the sequential correlation of data as long as the replay buffer B is large enough.
- Double DNNs: Based on DDPG, MADDPG also employs two DNNs, i.e., an online network and a target network in both the actor and the critic networks. The structures of these two DNNs are the same, but their updating rules of the parameters are quite different. For the two online networks, the parameters θ_i^{μ} and θ_i^{Q} are updated at every time step k. For the two target networks, the parameters $\theta_i^{\mu'}$ and $\theta_i^{Q'}$ are softly updated by the following rules:

$$\begin{aligned} \theta_i^{\mu'} &\leftarrow \tau\theta_i^{\mu} + (1-\tau)\theta_i^{\mu'}, \\ \theta_i^{Q'} &\leftarrow \tau\theta_i^{Q} + (1-\tau)\theta_i^{Q'}, \end{aligned} \tag{3.12}$$

where $0 < \tau \ll 1$ is a small update rate, also known as the filter parameter. The soft updating rules are employed to enhance the stability of the learning performance as the target network gradually synchronizes with the online network. The relationship between the online network and the target network is shown in Figure 4.

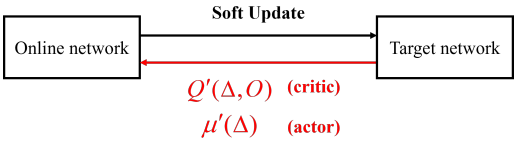


Figure 4. DNNs in the actor and the critic.

3.2.3. The updating rules

Different from the DDPG algorithm [34], the MADDPG algorithm incorporates the estimation information from other agents into the critic network of each agent. This is done to eliminate the assumption of knowing the accurate information of other agents and address the issue that the Q-function is dependent on the actions of all agents. Each agent i needs to maintain an approximation whose parameter is denoted by $\hat{\mu}_{\theta_i}^j$ (where θ represents the parameter of the approximation. For simplicity, the above variable is denoted by $\hat{\mu}_i^j$) to the actual policy by μ_j for the j-th agent. This approximation is learned by maximizing the natural logarithm of the probability of agent j's action, with an entropy regularizer, and the loss function L of the critic is as follows:

$$L(\theta_i^j) = -\mathbb{E}_{\Delta_j, \alpha_j}[\log \hat{\mu}_i^j(\alpha_j|\Delta_j) + \lambda H_e(\hat{\mu}_i^j)], \tag{3.13}$$

where H_e represents the entropy of the policy distribution, the symbol λ denotes the coefficient for entropy regularization, while $\lambda H_e(\hat{\mu}_i^j)$ refers to the aforementioned term for entropy regularization. Thus the estimated Q-value of the target network of the critic $\hat{y}$ can be calculated as follows:

$$\hat{y} = r_i + \gamma Q_i^{\mu'}(\Delta', \hat{\mu}_i'^1(\Delta_1), \ldots, \mu_i'^i(\Delta_i), \ldots, \hat{\mu}_i'^n(\Delta_n)), \tag{3.14}$$

where $\hat{\mu}_i'^j$ is the target network for the approximate action $\hat{\mu}_i^j$, $j \neq i$. Therefore, the whole updating rules are given by:
(1) Critic update (evaluation network update): for agent j, $j \in \{1, 2\}$,

$$\begin{aligned} &L(\phi_i^j) = -\mathbb{E}_{\Delta_j, a_j}[\log \hat{\mu}_i^j(a_j|\Delta_j) + \lambda H_e(\hat{\mu}_i^j)], \\ &\hat{y} = r_i + \gamma Q_i^{\mu'}(\Delta', \hat{\mu}_i'^1(\Delta_1), \ldots, \mu_i'^i(\Delta_i), \ldots, \hat{\mu}_i'^n(\Delta_n)). \end{aligned} \tag{3.15}$$

(2) Actor update (policy network update):

$$\begin{aligned}&\theta_i^{\mu} \leftarrow \theta_i^{\mu} + \nabla_{\theta_i^{\mu}} J,\\ &\nabla_{\theta_i^{\mu}} J \approx \frac{1}{S}\sum_j \nabla_{\theta_i^{\mu}} \mu_{\theta_i^{\mu}}(\Delta_i^j)\\ &\times \nabla_{\alpha_i} Q_{\theta_i^{Q}}(\Delta^j, \alpha_1^j, \cdots, \alpha_i, \cdots, \alpha_n^j)|_{\alpha_i=\mu_{\theta_i^{\mu}}(\Delta_i^j)},\end{aligned} \tag{3.16}$$

where S is the size of mini-batch samples, and $\nabla_{\theta_i^{\mu}} J$ denotes the update estimated through policy gradients.

The module of the MADDPG algorithm based on the actor-critic network is shown in Figure 5. The MADDPG algorithm for solving Problem 2 is given in Algorithm 1.

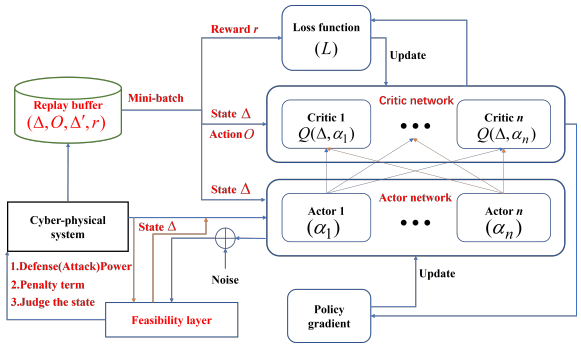


Figure 5. Framework of the MADDPG-based DoS attack game problem.

However, there still exists some other problems. For example, consider the scenario where the remote estimator receives a data packet with an AoI that is older than the one received in the previous time step due to transmission channel time delays. In other words, when

$$\Delta_k > \Delta_{k-1} + 1, \quad \Delta_{k-1} \neq 0,$$

which means the data packet received at time step $k-1$ was more recently generated. In this case, Algorithm 1 may directly overlook the problem and utilize the older data, resulting in a degradation of performance.

Remark 5. For Algorithm 1, it would be very easy to replace the AoI with the definition of holding time [1] without affecting the algorithm's results, as the one-step reward and generated actions at every time step are the same. However, this still does not solve the abovementioned problem of the freshness of data. In other words, for Algorithm 1, using the definition of holding time and the definition of AoI to derive the algorithm will lead to the same results. In subsequent experimental comparisons, we will consider Algorithm 1 to be representative of the algorithm derived from the holding time.

Algorithm 1: MADDPG-based DoS attack game-theoretic power design.

For the given system, set the number of agents n, the number of episodes M, and the maximum iteration number $Iter_{max}$, respectively.
Set the maximum power $\overline{p}$ and $\overline{a}$. Set the size of replay buffer B and the size of mini-batch S, respectively. Set a discount rate γ as well as the weight of the reward function ω_D and ω_A.
For each agent i:
Initialization: The filter parameter τ, the noise decay rate α_η, the actor network $\mu_{\theta_i^{\mu}}$, and the critic network $Q_{\theta_i^{Q}}$ with weights θ_i^{μ} and θ_i^{Q}, respectively. The target actor network $\mu_{\theta_i^{\mu'}}$ and target critic network $Q_{\theta_i^{Q'}}$ with weights $\theta_i^{\mu'} = \theta_i^{\mu}$ and $\theta_i^{Q'} = \theta_i^{Q}$, respectively.
for *episode* $= 1$ **to** M **do**
  Initialization: State $\Delta_0 = [\Delta_{0,1}, \Delta_{0,2}, \ldots, \Delta_{0,N}]$.
  for $Iter = 1$ **to** $Iter_{max}$ **do**
    Generate the exploration noise $e_i \sim \mathcal{N}_N(0, \eta)$.
    Select the defense power and attack power $\alpha_1 = p = \mu_{\theta_1^{\mu}}(\Delta_p) + e_1$, $\alpha_2 = \mathrm{a} = \mu_{\theta_2^{\mu}}(\Delta_a) + e_2$.
    $p \leftarrow \max(0, \min(p, \overline{p}))$,
    $\mathrm{a} \leftarrow \max(0, \min(\mathrm{a}, \overline{a}))$.
    Attack the system with power a and defend the system with power p, and then collect the next state Δ' and the one-step reward r.
    Store the tuple (Δ, O, r, Δ') in the replay buffer B.
    $\Delta \leftarrow \Delta'$.
    for *agent* $i = 1$ **to** n **do**
      Sample a random mini-batch of S samples $(\Delta^j, O^j, r^j, \Delta'^j)$ from B. Then obtain the local state estimation Δ_l^j for every agent l.
      Set $y^j = r_i^j + \gamma Q_{\theta_i^{Q'}}(\Delta'^j, \mu'_1, \cdots, \mu'_n)|_{\mu'_l=\mu_{\theta_i^{\mu'}}(\Delta_l^j)}$.
      Update the critic network by minimizing the loss: $L(\theta_i^{\mu}) = \frac{1}{n}\sum_j (y^j - Q_{\theta_i^{Q}}(\Delta^j, \mu_1^j, \cdots, \mu_n^j))^2$.
      Update the actor network by the sampled policy gradient: $\nabla_{\theta_i^{\mu}} J \approx \frac{1}{S}\sum_j \nabla_{\theta_i^{\mu}} \mu_{\theta_i^{\mu}}(\Delta_i^j) \times \nabla_{\alpha_i} Q_{\theta_i^{Q}}(\Delta^j, \alpha_1^j, \cdots, \alpha_i, \cdots, \alpha_n^j)|_{\alpha_i=\mu_{\theta_i^{\mu}}(\Delta_i^j)}$.
    end
    Update target network parameters for each agent i: $\theta_i^{\mu'} \leftarrow \tau\theta_i^{\mu} + (1-\tau)\theta_i^{\mu'}$,
    $\theta_i^{Q'} \leftarrow \tau\theta_i^{Q} + (1-\tau)\theta_i^{Q'}$.
    $\eta \leftarrow \alpha_\eta \eta$.
  end
end

3.3. Feasibility layers for the received AoI and power constraint

As mentioned in the previous section, measures should be taken to deal with the problem caused by time delays. This is solved by modifying the received state Δ with the following rules:

$$\Delta_{k,i} = \begin{cases} \Delta_{k,i}, & \Delta_{k,i} \leq \Delta_{k-1,i} + 1, \\ \Delta_{k-1,i} + 1, & \text{otherwise.} \end{cases} \tag{3.17}$$

In addition, notice that the power chosen during the process is calculated by the policy gradient with an exploration noise, which may sometimes be beyond the feasible domain. Thus, two feasibility layers are added to our MADDPG algorithm. The layers have three functions:

(1) Decide the freshness of the current data by AoI.

(2) Map the infeasible actions to the feasible ones.

(3) Make the action provided by the actor network be, as much as possible within the feasible domain by adding a penalty term to the reward function.

In our new algorithm, the feasibility layers are added after the network receiving the state as well as the actor network outputting the actions, resepectively. We choose the scale mapping as the method to generate the feasible action, which is as follows:

$$\alpha_{1,i} = \begin{cases} \alpha_{1,i}, & \|p_k\|_1 \leq \overline{p}, \\ \overline{p} \times \frac{p_{k,i}}{\|p_k\|_1}, i = 1, \ldots, N, & \|p_k\|_1 > \overline{p}, \end{cases} \tag{3.18}$$

and

$$\alpha_{2,i} = \begin{cases} \alpha_{2,i}, & \|\mathrm{a}_k\|_1 \leq \overline{a}, \\ \overline{a} \times \frac{\mathrm{a}_{k,i}}{\|\mathrm{a}_k\|_1}, i = 1, \ldots, N, & \|\mathrm{a}_k\|_1 > \overline{a}, \end{cases} \tag{3.19}$$

where $\alpha_{1,i}$ and $\alpha_{2,i}$ represent the i-th component of the vectors α_1 and α_2, respectively.

Furthermore, a penalty term $\lambda_D I_D$ or $\lambda_A I_A$ is added to the reward function to motivate the actor network to generate feasible actions as: for the defender,

$$r_1(\Delta_{k-1}, p_k, \Delta_k) = -\omega_D Tr(P_k) - \beta_D(1 - \omega_D)\|p_k\|_1 - \lambda_D I_D. \tag{3.20}$$

For the attacker,

$$r_2(\Delta_{k-1}, \mathrm{a}_k, \Delta_k) = \omega_A Tr(P_k) - \beta_A(1 - \omega_A)\|\mathrm{a}_k\|_1 - \lambda_A I_A, \tag{3.21}$$

where

$$I_D = \max(0, \|p_k\|_1 - \overline{p})$$

and

$$I_A = \max(0, \|a_k\|_1 - \overline{a})$$

are the infeasible parts of the given actions of the agents. The variables $\lambda_D \in \mathbb{R}^+$ and $\lambda_A \in \mathbb{R}^+$ represent the penalty weights of I_D and I_a, respectively. The scale mapping method described in (3.18) and (3.19) is reasonable and has little influence on Problem 2. First, the feasible action generated by the networks does not change. If the generated action exceeds the feasible domain, the reward will obviously decrease with a great penalty term, which causes the agent to reduce the attempts to take action that exceeds the feasible domain. Therefore, our MADDPG-based algorithm becomes Algorithm 2.

4. Simulation results

4.1. Environment setting

We consider a cyber-physical system that involves two processes with transmission delay. Here the parameters of the system are given by Table 1, where "No." is an abbreviation for numbers. The transmission delay is set as

$$t_{d,i} \in \{0, 1, 2\}, \quad i = 1, 2.$$

Table 1. Multi-process system parameters.

Processes	A_i	C_i	Σ_{ω_i}	Σ_{v_i}	σ_i
$i = 1$	1.1	1.2	0.8	0.6	0.8
$i = 2$	1.1	1.0	0.9	0.9	1.0

Assume the model of the wireless transmission channel is an AWGN channel [29]. Thus, the probability function f in (2.5) is denoted by:

$$f(p^s_{i,k}, p^a_{i,k}) = 1 - 2g\left(\sqrt{\frac{\delta_s p^s_{i,k}}{\delta_a p^a_{i,k} + \sigma_i^2}}\right), i = 1, \ldots, N, \tag{4.1}$$

where

$$g(x) = \left(\frac{1}{\sqrt{2\pi}}\right) \int_x^\infty \exp(-v^2/2) dv.$$

The variable σ_i^2 represents the white noise power of the i-th channel, and the positive parameters δ_s and δ_a represent the performance characterization of specific transmission channels under DoS attack and defense. We assume that the upper bound of the state of each process is 9 to ensure at least one successful transmission within a limited number of time steps, in other words, when AoI exceeds 9, the transmission will definitely succeed at the next time step. This assumption has a negligible influence on the problem, as proved in [21]. The maximum power of the defender and attacker is set as

Algorithm 2: Feasible MADDPG-based DoS attack game-theoretic power design.

Set n, M, $Iter_{max}$, $\overline{p}$, $\overline{a}$, γ, ω_D, ω_A, S, and B.
For each agent i:
Initialization: τ, α_η, $\mu_{\theta_i^\mu}$, $Q_{\theta_i^Q}$ with weights θ_i^μ and θ_i^Q, respectively. $\mu_{\theta_i^{\mu'}}$ and $Q_{\theta_i^{Q'}}$ with weights $\theta_i^{\mu'} = \theta_i^\mu$ and $\theta_i^{Q'} = \theta_i^Q$, respectively. The penalty terms λ_A and λ_D.
for *episode* = 1 **to** M **do**
  Initialization: state $\Delta_0 = [\Delta_{0,1}, \Delta_{0,2}, \ldots, \Delta_{0,N}]$.
  for *Iter* = 1 **to** $Iter_{max}$ **do**
    for $j = 1$ **to** N **do**
      if $\Delta_j^{'} > \Delta_j + 1$ **then**
        $\Delta_j^{'} = \Delta_j + 1$.
      end
    end
    Generate the exploration noise $e_i \sim \mathcal{N}_N(0, \eta)$.
    Select the defense power and attack power $p = \mu_{\theta_i^\mu}(\Delta_p) + e_1$, $\mathrm{a} = \mu_{\theta_i^\mu}(\Delta_a) + e_2$,
    $p \leftarrow \max(0, \min(p, \overline{p}))$, $\mathrm{a} \leftarrow \max(0, \min(\mathrm{a}, \overline{a}))$.
    if $\|a\|_1 > \overline{a}$ **then**
      Calculate α_2 by (3.19), $I_A = \max(0, \|\mathrm{a}\|_1 - \overline{a})$, $\mathrm{a} = \alpha_2$.
    end
    if $\|p\|_1 > \overline{p}$ **then**
      Calculate α_1 by (3.18), $I_D = \max(0, \|p\|_1 - \overline{p})$, $p = \alpha_1$.
    end
    Attack the system with power a and defend the system with power p, and then collect the next state Δ' and the one-step reward $r = [r_D, r_A]$.
    $r_D \leftarrow r_D - \lambda_D I_D$, $r_A \leftarrow r_A - \lambda_A I_A$.
    Store (Δ, O, r, Δ') in the replay buffer B.
    $\Delta \leftarrow \Delta'$.
    for *agent* $i = 1$ **to** n **do**
      Sample a random mini-batch of S samples $(\Delta^j, O^j, r^j, \Delta'^j)$ from B. Then obtain the local state estimation Δ_l^j for every agent l.
      Set
      $$y^j = r_i^j + \gamma Q_{\theta_i^{Q'}}(\Delta'^j, \mu'_1, \cdots, \mu'_n)|_{\mu'_l = \mu_{\theta_i^{\mu'}}(\Delta_l^j)}.$$
      Update the critic network by minimizing the loss: $L(\theta_i^\mu) = \frac{1}{S}\sum_j \left(y^j - Q_{\theta_i^Q}(\Delta^j, \mu_1^j, \cdots, \mu_n^j)\right)^2$.
      Update the actor network by the sampled policy gradient:
      $$\nabla_{\theta_i^\mu} J \approx \frac{1}{S}\sum_j \nabla_{\theta_i^\mu}\mu_{\theta_i^\mu}(\Delta_i^j) \times \nabla_{\alpha_i} Q_{\theta_i^Q}(\Delta^j, \alpha_1^j, \cdots, \alpha_i, \cdots, \alpha_n^j)|_{\alpha_i = \mu_{\theta_i^\mu}(\Delta_i^j)}.$$
    end
    Update target network parameters for each agent i:
    $\theta_i^{\mu'} \leftarrow \tau\theta_i^\mu + (1-\tau)\theta_i^{\mu'}$, $\theta_i^{Q'} \leftarrow \tau\theta_i^Q + (1-\tau)\theta_i^{Q'}$.
    $\eta \leftarrow \alpha_\eta \eta$.
  end
end

$\overline{p} = 10$ and $\overline{a} = 10$, respectively. The discount rate is set as $\beta = 0.9$. The network parameters of the MADDPG-based game algorithm are summarized in Table 2, and the learning parameters are displayed in Table 3. All simulations were run on a computer with Intel i7-10875H and 16 GB RAM.

Table 2. Network parameters.

Parameters	Policy Network	Evaluation Network
No. of inputs	Dim(Δ)	Dim(Δ)+Dim(O)
No. of hidden layers	3	2
No. of nodes in hidden layer 1	64	64
No. of nodes in hidden layer 2	64	64
No. of nodes in hidden layer 3	Dim(α)	×
No. of outputs	Dim(α)	1
Activation function 1	ReLU	ReLU
Activation function 2	ReLU	ReLU
Activation function 3	Sigmoid	×

Table 3. Learning parameters.

Parameters	n	M	$Iter_{max}$	$\overline{p}$	$\overline{a}$	γ	λ_a	λ_D	τ	α_η	B	S
Value	2	10^3	200	10	10	0.95	10	10	0.01	0.999	10^4	10

4.2. Result presentation

The performance of the attackers and defenders is measured by the cumulative discounted reward and the mutual Bayesian Nash equilibrium strategy under different states. We set the initial weights

$$\omega_A = \omega_D = 0.8$$

with

$$\beta_A = 0.48, \ \beta_D = 0.67,$$

and the experimental results are shown in Figure 6. After 100 episodes, it can be observed that the cumulative discounted rewards for both the attacker and defender reached a relatively stable stage.

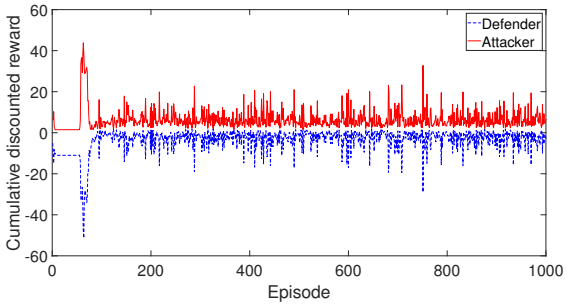


Figure 6. Attacker and defender performance with $\beta_A = 0.48$ and $\beta_D = 0.67$ in two different processes in Table 1.

The attacker's cumulative discounted reward fluctuated around 0 to 20, while the defender's cumulative discounted reward fluctuated around -20 to 0.

Based on the neural networks that have been recently updated, their Bayesian Nash equilibrium strategies can be calculated as shown in Figures 7 and 8, where each scatter point in the figure represents the optimal power allocation under its corresponding state (Δ_1, Δ_2) (the x-axis and y-axis correspond to the states of process 1 and process 2, respectively, while the legend provides precise values). The reason that both the attacker and defender chose to focus on process 1 instead of process 2 is determined by the settings of the system parameters.

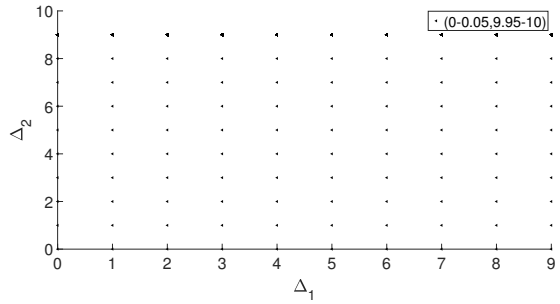


Figure 7. Attacker's optimal strategy with $\beta_A = 0.48$ and $\beta_D = 0.67$ in two different processes in Table 1.

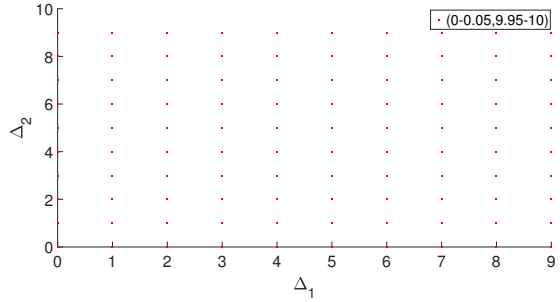


Figure 8. Defender's optimal strategy with $\beta_A = 0.48$ and $\beta_D = 0.67$ in two different processes in Table 1.

After conducting the simulation with

$$\beta_A = 0.48 \text{ and } \beta_D = 0.67,$$

we change the power discount rate with

$$\beta_A = 0.55 \text{ and } \beta_D = 0.68$$

in two identical processes as in Process 2 in Table 1. The corresponding results are shown in Figures 9–11. From the attacker's prospective (Figure 10), it tends to spare little power to attack the channel when the state is $(0, 0)$. However, if the state is not $(0, 0)$, the attacker will put most of their emphasis on attacking channel 1. The optimal strategy of the defender is much more complex than the attacker's. We present it in Figure 11 and show a part of the optimal strategies in Table 4. Affected by the attacker's attack strategy, the defender also pays higher attention to channel 1.

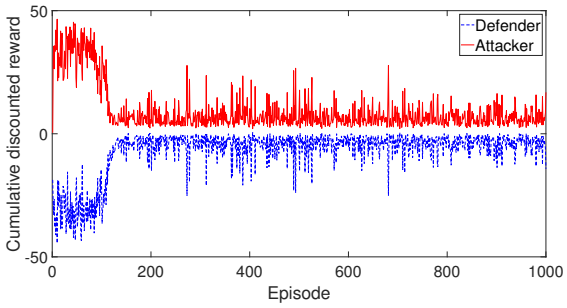


Figure 9. Attacker and defender performance with $\beta_A = 0.55$ and $\beta_D = 0.68$ in the two same processes as in Process 2 in Table 1.

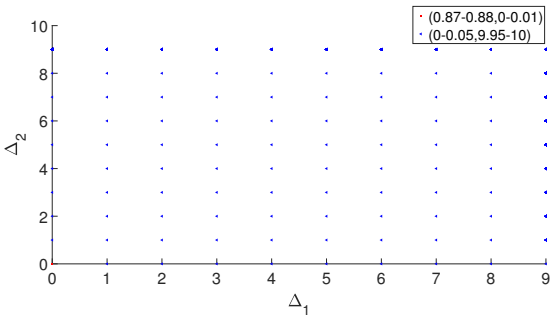


Figure 10. Attacker's optimal strategy with $\beta_A = 0.55$ and $\beta_D = 0.68$ in the two same processes as in Process 2 in Table 1.

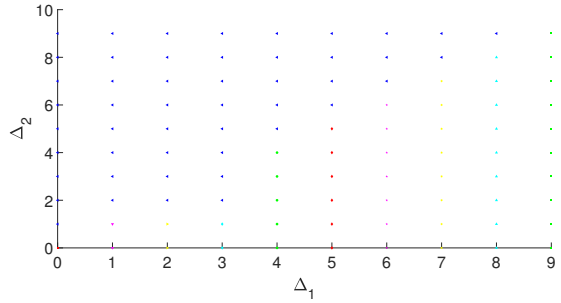


Figure 11. Defender's optimal strategy with $\beta_A = 0.55$ and $\beta_D = 0.68$ in the two same processes as in Process 2 in Table 1.

Table 4. Part of the optimal strategies of the defender in Figure 11.

$\Delta_2 \geq \Delta_1$ *or* $(\Delta_1, \Delta_2) = (3, 2)$	$(\Delta_1, \Delta_2) = (6, 0)$	$(\Delta_1, \Delta_2) = (6, 1)$	$(\Delta_1, \Delta_2) = (6, 2)$
$(6.18 - 6.26, 2.65 - 2.75)$	$(0.19, 9.52)$	$(0.41, 8.71)$	$(0.96, 7.81)$
$(\Delta_1, \Delta_2) = (6, 3)$	$(\Delta_1, \Delta_2) = (6, 4)$	$(\Delta_1, \Delta_2) = (6, 5)$	$(\Delta_1, \Delta_2) = (6, 6)$
$(2.47, 6.33)$	$(4.04, 4.96)$	$(4.86, 3.98)$	$(5.36, 3.18)$

When the AoI of channel 2 is bigger than the one of channel 1 (except for the state $(0, 0)$), the optimal strategy is around (6.18–6.26, 2.65–2.75). In addition, if the AoI of channel 1 is fixed, the defender's power allocation to channel 1 will increase as the AoI of channel 2 increases. To show that the strategies obtained by Algorithm 2 are optimal, we compare the optimal strategy of the defender with other strategies when the attacker's strategy is fixed as the one derived from Algorithm 2 in Figure 12.

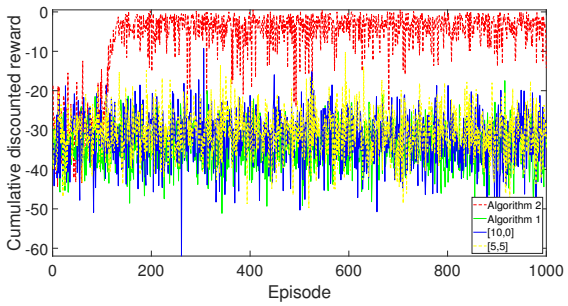


Figure 12. Defender's cumulative discounted reward with different strategies.

From Figure 12, it can be seen that the defender's cumulative discounted reward under the optimal strategy is obviously more than the ones under other fixed strategies. In addition, we compare the optimal strategy to the one derived by Algorithm 1 to show the effectiveness of the feasibility layers in CPSs with time delays. Although Algorithm 1 employs AoI as a state definition, it does not take into account that AoI is a variable used to measure the freshness of data. In essence, there is not much difference between AoI and other definitions (for example, holding time). Therefore, the discounted cumulative rewards for the defender are not as good as those of Algorithm 2. Therefore, the optimality of the strategy derived by Algorithm 2 is obvious.

It is worth noting that, in practical scenarios, such results are relatively rare. Due to the characteristics of different channels, both attackers and defenders tend to have a focus on allocating energy in one channel, which leads to the optimal action near $(\overline{p}\ or\ \overline{a}, 0)$ or $(0, \overline{p}\ or\ \overline{a})$. What is more, when ω_A and ω_D reduce, the weight of cost in the reward formulas (3.20) and (3.21) of both sides will correspondingly increase, leading them to reduce the injecting energy to the channel rather than focus on increasing (or reducing) the estimated error covariance. Therefore, in practical applications, the setting of the weights and power discount rates may be extremely important.

In addition, compared with other learning-based methods in existing literature (such as DQN, DDQN, DDPG, etc.), our method can be applied to multi-agent environments with asymmetric information. However, other mentioned algorithms may easily fail to converge due to the complexity and variability of the multi-agent environments.

5. Conclusions

We considered a learning-based game-theoretic DoS attack problem with continuous power in a multi-process CPS with random time delays of transmission channels. In order to deal with the challenges posed by a multi-agent environment with asymmetric information and continuous attack and defense power, we provided a MADDPG-based algorithm with two feasible layers, which is suitable for the above situation, and is able to reach an optimal Bayesian Nash equilibrium. In future work, we will introduce various types of attacks into our research on game theory problems. We will consider not only cyber attacks targeting transmission channels but also attacks aimed at other parts of the networks.

Use of Generative-AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This work is supported in part by the National Natural Science Foundation of China (Grants 62173142, 62303185, and 62073202), in part by the Programme of Introducing Talents of Discipline to Universities (the 111 Project) under Grant B17017, in part by the Shanghai Sailing Program under Grant 23YF1409500, and in part by the Fundamental Research Funds for the Central Universities under Grant JKM01231838.

Conflict of interest

The authors declare that there are no conflicts of interest in this paper.

References

1. M. Huang, K. Ding, S. Dey, Y. Li, L. Shi, Learning-based DoS attack power allocation in multiprocess systems, *IEEE Trans. Neural Networks Learn. Syst.*, **34** (2023), 8017–8030. https://doi.org/10.1109/TNNLS.2022.3148924
2. M. Huang, K. Tsang, Y. Li, L. Li, L. Shi, Strategic DoS attack in continuous space for cyber-physical systems over wireless networks, *IEEE Trans. Signal Inf. Process. Networks*, **8** (2022), 421–432. https://doi.org/10.1109/TSIPN.2022.3174969
3. D. Möller, H. Vakilzadian, Cyber-physical systems in smart transportation, *2016 IEEE International Conference on Electro Information Technology (EIT)*, 2016. https://doi.org/10.1109/EIT.2016.7535338
4. A. Kasun, W. Chathurika, M. Daniel, R. Craig, M. Milos, Framework for data driven health monitoring of cyber-physical systems, *2018 Resil Week (RWS)*, 2018, 25–30. https://doi.org/10.1109/rweek.2018.8473535
5. G. Liang, R. W. Steven, J. Zhao, F. Luo, Z. Dong, The 2015 Ukraine blackout: implications for false data injection attacks, of cyber-physical systems, *IEEE Trans. Power Syst.*, **32** (2017), 3317–3318. https://doi.org/10.1109/TPWRS.2016.2631891
6. I. Stojmenovic, S. Wen, X. Huang, H. Luan, An overview of Fog computing and its security issues, *Concurrency Comput. Practice Exper.*, **28** (2016), 2991–3005. https://doi.org/10.1002/cpe.3485
7. P. Griffioen, S. Weerakkody, O. Ozel, Y. Mo, B. Sinopoli, A tutorial on detecting security attacks on cyber-physical systems, *2019 18th European Control Conference (ECC)*, 2019, 979–984. https://doi.org/10.23919/ecc.2019.8796117
8. J. Qin, M. Li, L. Shi, X. Yu, Optimal denial-of-service attack scheduling with energy constraint over packet-dropping networks, *IEEE Trans. Autom. Control*, **63** (2018), 1648–1663. https://doi.org/10.1109/TAC.2017.2756259
9. H. Zhang, P. Cheng, L. Shi, J. Chen, Optimal denial-of-service attack scheduling with energy constraint, *IEEE Trans. Autom. Control*, **60** (2015), 3023–3028. https://doi.org/10.1109/TAC.2015.2409905
10. S. Li, C. K. Ahn, Z. Xiang, Decentralized sampled-data control for cyber-physical systems subject to DoS attacks, *IEEE Syst. J.*, **15** (2021), 5126–5134. https://doi.org/10.1109/JSYST.2020.3019939
11. Y. Mo, B. Sinopoli, L. Shi, E. Garone, Infinite-horizon sensor scheduling for estimation over lossy networks, *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, 2012, 3317–3322. https://doi.org/10.1109/cdc.2012.6425859
12. R. Gan, Y. Xiao, J. Shao, J. Qin, An analysis on optimal attack schedule based on channel hopping scheme in cyber-physical systems, *IEEE Trans. Cybern.*, **51** (2021), 994–1003. https://doi.org/10.1109/TCYB.2019.2914144
13. X. Ren, J. Wu, S. Dey, L. Shi, Attack allocation on remote state estimation in multi-systems: structural results and asymptotic solution, *Automatica*, **87** (2018), 184–194. https://doi.org/10.1016/j.automatica.2017.09.021
14. Z. Guo, J. Wang, L. Shi, Optimal denial-of-service attack on feedback channel against acknowledgment-based sensor power schedule for remote estimation, *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017. https://doi.org/10.1109/cdc.2017.8264567
15. X. Wang, J. He, S. Zhu, C. Chen, X. Guan, Learning-based attack schedule against remote state estimation in cyber-physical systems, *2019 American Control Conference (ACC)*, 2019, 4503–4508. https://doi.org/10.23919/acc.2019.8814961

16. X. Wang, C. Chen, J. He, S. Zhu, X. Guan, Learning-based online transmission path selection for secure estimation in edge computing systems, *IEEE Trans. Ind. Inf.*, **17** (2021), 3577–3587. https://doi.org/10.1109/TII.2020.3012090

17. P. Dai, W. Yu, H. Wang, G. Wen, Y. Lv, Distributed reinforcement learning for cyber-physical system With multiple remote state estimation under DoS attacker, *IEEE Trans. Networks Sci. Eng.*, **7** (2020), 3212–3222. https://doi.org/10.1109/TNSE.2020.3018871

18. J. Clifton, E. Laber, *Q*-Learning: theory and applications, *Ann. Rev. Stat. Appl.*, **7** (2020), 279–301. https://doi.org/10.1146/annurev-statistics-031219-041220

19. L. Wang, X. Liu, T. Li, J. Zhu, Skew-symmetric games and symmetric-based decomposition of finite games, *Math. Modell. Control*, **2** (2022), 257–267. https://doi.org/10.3934/mmc.2022024

20. Z. Yue, L. Dong, S. Wang, Stochastic persistence and global attractivity of a two-predator one-prey system with *S*-type distributed time delays, *Math, Modell. Control*, **2** (2022), 272–281. https://doi.org/10.3934/mmc.2022026

21. K. Ding, X. Ren, D. Quevedo, S. Dey, L. Shi, DoS attacks on remote state estimation with asymmetric information, *IEEE Trans. Control Networks*, **6** (2019), 653–666. https://doi.org/10.1109/TCNS.2018.2867157

22. J. Champati, M. Mamduhi, K. Johansson, J. Gross, Performance characterization using AoI in a single-loop networked control system, *IEEE INFOCOM 2019-IEEE Conference on Computer Communications Workshops*, 2019, 197–203. https://doi.org/10.1109/infcomw.2019.8845114

23. Q. He, D. Yuan, A. Ephremides, Optimal link scheduling for age minimization in wireless systems, *IEEE Trans. Inf. Theory*, **64** (2018), 5381–5394. https://doi.org/10.1109/TIT.2017.2746751

24. Y. Sun, E. Uysal-Biyikoglu, R. Yates, C. Koksal, N. Shroff, Update or wait: how to keep your data fresh, *IEEE Trans. Inf. Theory*, **63** (2017), 7492–7508. https://doi.org/10.1109/TIT.2017.2735804

25. X. Wang, C. Chen, J. He, S. Zhu, X. Guan, AoI-aware control and communication co-design for industrial IoT systems, *IEEE Int. Things J.*, **8** (2021), 8464–8473. https://doi.org/10.1109/JIOT.2020.3046742

26. L. Shi, P. Cheng, J. Chen, Sensor data scheduling for optimal state estimation with communication energy constraint, *Automatica*, **47** (2011), 1693–1698. https://doi.org/10.1016/j.automatica.2011.02.037

27. L. Shi, L. Xie, Optimal sensor power scheduling for state estimation of Gauss-Markov systems over a packet-dropping network, *IEEE Trans. Signal Process.*, **60** (2012), 2701–2705. https://doi.org/10.1109/TSP.2012.2184536

28. L. Shi, L. Xie, R. Murray, Kalman filtering over a packet-delaying network: a probabilistic approach, *Automatica*, **45** (2009), 2134–2140. https://doi.org/10.1016/j.automatica.2009.05.018

29. L. Peng, L. Shi, X. Cao, C. Sun, Optimal attack energy allocation against remote state estimation, *IEEE Trans. Autom. Control*, **63** (2018), 2199–2205. https://doi.org/10.1109/TAC.2017.2775344

30. R. Sutton, G. Andrew, *Reinforcement learning: an introduction*, MIT Press, 1998.

31. K. Hazeghi, M. Puterman, Markov decision processes: discrete stochastic dynamic programming, *J. Amer. Stat. Assoc.*, **90** (1995), 392. https://doi.org/10.2307/2291177

32. H. Tan, Reinforcement learning with deep deterministic policy gradient, *2021 International Conference on Artificial Intelligence, Big Data and Algorithms (CAIBDA)*, 2021, 82–85. https://doi.org/10.1109/CAIBDA53561.2021.00025

33. B. Shalabh, R. Sutton, M. Ghavamzadeh, M. Lee, Natural actor-critic algorithms, *Automatica*, **45** (2009), 2471–2482. https://doi.org/10.1016/j.automatica.2009.07.008

34. A. Rodriguez-Ramos, C. Sampedro, H. Bavle, P. de la Puente, P. Campoy, A deep reinforcement learning strategy for UAV autonomous landing on a moving platform, *J. Intell. Robot. Syst.*, **93** (2019), 351–366. https://doi.org/10.1007/s10846-018-0891-8

AIMS Press