drones

Article

A Q-Learning-Based Two-Layer Cooperative Intrusion Detection for Internet of Drones System

Moran Wu [1], Zhiliang Zhu [1,*], Yunzhi Xia [2], Zhengbing Yan [1], Xiangou Zhu [1] and Nan Ye [3]

1 College of Electrical and Electronic Engineering, Wenzhou University, Wenzhou 325035, China; 21451841029@stu.wzu.edu.cn (M.W.); zbyan@wzu.edu.cn (Z.Y.); zhuxo@wzu.zju.cn (X.Z.)
2 Hubei Key Laboratory of Distributed System Security, Hubei Engineering Research Center on Big Data Security, School of Cyber Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China; yzxia@hust.edu.cn
3 Yalong Intelligent Equipment Group Co., Ltd., Wenzhou 325035, China; nannan203@126.com
* Correspondence: zlzhu@wzu.edu.cn

Abstract: The integration of unmanned aerial vehicles (UAVs) and the Internet of Things (IoT) has opened up new possibilities in various industries. However, with the increasing number of Internet of Drones (IoD) networks, the risk of network attacks is also rising, making it increasingly difficult to identify malicious attacks on IoD systems. To improve the accuracy of intrusion detection for IoD and reduce the probability of false positives and false negatives, this paper proposes a Q-learning-based two-layer cooperative intrusion detection algorithm (Q-TCID). Specifically, Q-TCID employs an intelligent dynamic voting algorithm that optimizes multi-node collaborative intrusion detection strategies at the host level, effectively reducing the probability of false positives and false negatives in intrusion detection. Additionally, to further reduce energy consumption, an intelligent auditing algorithm is proposed to carry out system-level auditing of the host-level detections. Both algorithms employ Q-learning optimization strategies and interact with the external environment in their respective Markov decision processes, leading to close-to-optimal intrusion detection strategies. Simulation results demonstrate that the proposed Q-TCID algorithm optimizes the defense strategies of the IoD system, effectively prolongs the mean time to failure (MTTF) of the system, and significantly reduces the energy consumption of intrusion detection.

Keywords: Internet of Things; Internet of Drones; unmanned aerial vehicles; intrusion detection; Q-learning

Citation: Wu, M.; Zhu, Z.; Xia, Y.; Yan, Z.; Zhu, X.; Ye, N. A Q-Learning-Based Two-Layer Cooperative Intrusion Detection for Internet of Drones System. *Drones* **2023**, *7*, 502. https://doi.org/10.3390/drones7080502

Academic Editors: Wenzheng Xu, Tang Liu and Weifa Liang

Received: 19 July 2023
Accepted: 31 July 2023
Published: 1 August 2023

1. Introduction

The Internet of Things has emerged as a disruptive and transformative technology that enables ubiquitous connectivity and seamless integration of physical devices, data networks, and intelligent algorithms. Its pervasiveness and versatility have led to a plethora of diverse applications in various domains, such as smart cities [1], smart homes [2], smart car networking systems [3], medical care [4] and other industries. The Internet of Drones is an extension of the concept of the Internet of Things, where 'Things' are replaced by 'Drones'. As a layered network control architecture, the IoD plays a crucial role in the advancement of unmanned aerial vehicles (UAVs) or drones [5,6].

The IoD is a vast network of numerous information-sensing devices engaged in communication and collaboration to facilitate reliable decision-making and problem-solving. However, the use of off-the-shelf components and software in IoD systems makes them vulnerable to sophisticated attacks. This poses a significant threat to their safe operation and reliability, which is crucial for ensuring the safe and normal functioning of UAVs. For an IoD system to be reliable, the information it communicates must be credible. The presence of malicious devices within the IoD system that supply false information and collude with others to mislead the system can undermine its reliability and cause considerable loss.

Network intrusion detection systems (NIDS) [7] serve as proactive security defense technology, providing real-time monitoring of attacks from the intranet and extranet. When an attack is detected, timely alarms can be triggered, and appropriate solutions can be provided [8]. Researchers have successfully applied machine and deep learning to intrusion detection, achieving significant progress [9,10]. NIDS has evolved over the years in terms of efficiency. However, attackers have also developed advanced techniques to evade detection, particularly in the complex network layers of unmanned aerial vehicles (UAVs). Although lightweight intrusion detection algorithms have been proposed [11], many of them sacrifice recognition accuracy. Therefore, there is an urgent need to develop lightweight intrusion detection algorithms that are applicable to individual nodes.

Reinforcement learning (RL) has witnessed significant growth in research and applications, offering solutions for complex decision-making tasks in recent years [12]. In this paper, to improve the accuracy of intrusion detection decision-making, we propose a Q-learning-based two-layer cooperative intrusion detection algorithm to improve the effectiveness of intrusion detection in IoD systems. Specifically, at the host level, nodes employ Q-learning to gradually learn and optimize their individual voting strategies, enhancing collaborative decision-making among multiple nodes. At the system level, intelligent auditing is performed on the host-level voting results, followed by appropriate rewards or penalties based on the audit findings. The audit strategy is optimized using Q-learning, leading to a significant improvement in the accuracy of intrusion detection. The system level is a supplementary defense mechanism against the omission or miscalculation of the voting process at the host level.

The main contributions of this paper can be summarized as follows:

- A new intelligent voting algorithm for IoD intrusion detection is proposed, which applies Q-learning to the node voting intrusion detection. The proposed algorithm is equipped with continuous automatic learning capabilities for IoD nodes. It can interact with the network environment and cooperate with other nodes to optimize group interests and enhance their intrusion detection capabilities.
- A Q-learning-based two-layer cooperative intrusion detection algorithm (Q-TCID) is proposed for the host level and the system level, respectively. As a supplementary defense mechanism to the host-level intelligent voting algorithm, the system-level intelligent audit algorithm cooperates with the host level to effectively reduce the probability of false positives and false negatives while also reducing the energy consumption.
- The simulation results show that the proposed Q-TCID algorithm optimizes the defense strategy of the IoD system, which not only saves more energy and improves the accuracy of intrusion detection but also effectively improves the MTTF of the system.

The paper is organized as follows: Section 2 presents the related work. Section 3 explains the network architecture and problem formulation. The proposed approach is discussed in Section 4. Simulation results are presented in Section 5, and Section 6 concludes the paper.

2. Related Work

In recent years, the increasing utilization of unmanned aerial vehicles integrated with IoT technologies has led to rising concerns about the security of drone networks. Intrusion detection plays a crucial role in protecting these networks from malicious activity and safeguarding sensitive data.

Numerous research efforts have focused on developing efficient intrusion detection systems specifically designed for IoD. These efforts can be broadly classified into two primary approaches: signature-based and anomaly-based detection techniques.

Signature-based detection looks for similarities between a collection of network data and a database containing features, relying on predefined patterns or signatures of known attacks to identify and block malicious activity. Early works on intrusion detection systems (IDS) for IoD-involved signature-based intrusion detection methods [5]. Kacem et al. [13] proposed an IDS to detect B messages, incorporating knowledge from cyber defense mechanisms and aircraft motion to identify potential attacks and preserve digital evidence for forensic investigation. Condomines et al. [14] proposed a hybrid IDS for UAV spectral traffic analysis with a robust controller/observer to accurately detect anomalies and mitigate Distributed Denial of Service (DDoS) attacks and demonstrate its effectiveness through real traffic traces and practical applications. However, signature-based IDSs are complex to manage and require manual intervention for rule configuration and signature updates, which are not capable of detecting unknown attacks.

Anomaly-based intrusion detection techniques collect and analyze data on legitimate user behavior to determine if the currently observed behavior is malicious or legitimate. These technologies are effective in detecting unknown attacks [15,16]. The second research direction focuses on developing lightweight classifiers utilizing AI techniques [17,18], such as Machine Learning (ML) and Deep Learning (DL) [19–21]. J. Tao et al. [22] proposed a UAV IDS using deep reinforcement learning (DRL) for airborne communication systems. They also discussed the fundamentals of UAVs, conducted a case study on the effectiveness of their DRL-based IDS, and verified its effectiveness through simulations. A. Heidari et al. [23] proposed a blockchain-based radial basis function neural network (RBFNN) model to enhance the performance of the IoD network, improving data integrity, intelligent decision-making, and decentralized predictive analytics while outperforming state-of-the-art methods in network intrusion detection. Rui Fu et al. [24] integrated Convolutional Neural Networks(CNN) and long short-term memory (LSTM) into the CNN-LSTM algorithm to build an agricultural Internet of Things IDS. Abu Al-Haija et al. [21] proposed an autonomous IDS using a deep convolution neural network to effectively detect malicious threats from invading UAVs. Complex classification algorithms based on deep learning have also been promoted to effectively classify malicious and benign devices in IoD scenarios. Most of these attack detection methods are complex and consume high energy. The main challenge is reducing the computational cost and energy consumption of training the classifier so that it can run on IoD with limited resources. Wang et al. [11] presented a novel IDS attack–defense game that incorporates occasional system audits while relying on sensor nodes for intrusion detection through a distributed approach. However, there is a high probability of misjudgments and missed judgments in the design of their work. Therefore, an appropriate strategy is needed to realize cooperation in the system and reduce the probability of false positives and negatives in intrusion detection. The purpose of this paper is to optimize the performance of IDS by training an intrusion detection strategy.

However, most of the current research has not been discussed from the point of view of a single node, and it is based only on the system level. This consideration cannot be directly applied to the nodes of the IoD, so it is necessary to consider each node individually. To address this limitation, this paper proposes a Monte Carlo simulation method in which the host-level and the system-level agents cooperate to perform distributed intrusion detection.

3. Network Architecture and Problem Formulation

In this section, the network architecture of Q-TCID is presented, and the mathematical model of the problem is formulated.

3.1. Network Architecture

The architecture of Q-TCID is shown in Figure 1, consists of four entities: Base Station (BS), Multi-UAV, Good Nodes, and Bad Nodes. The Multi-UAVs are responsible for collecting sensor information, ensuring that each sensor is connected by a single UAV. The BS receives the collected information from all UAVs and performs processing and analysis.

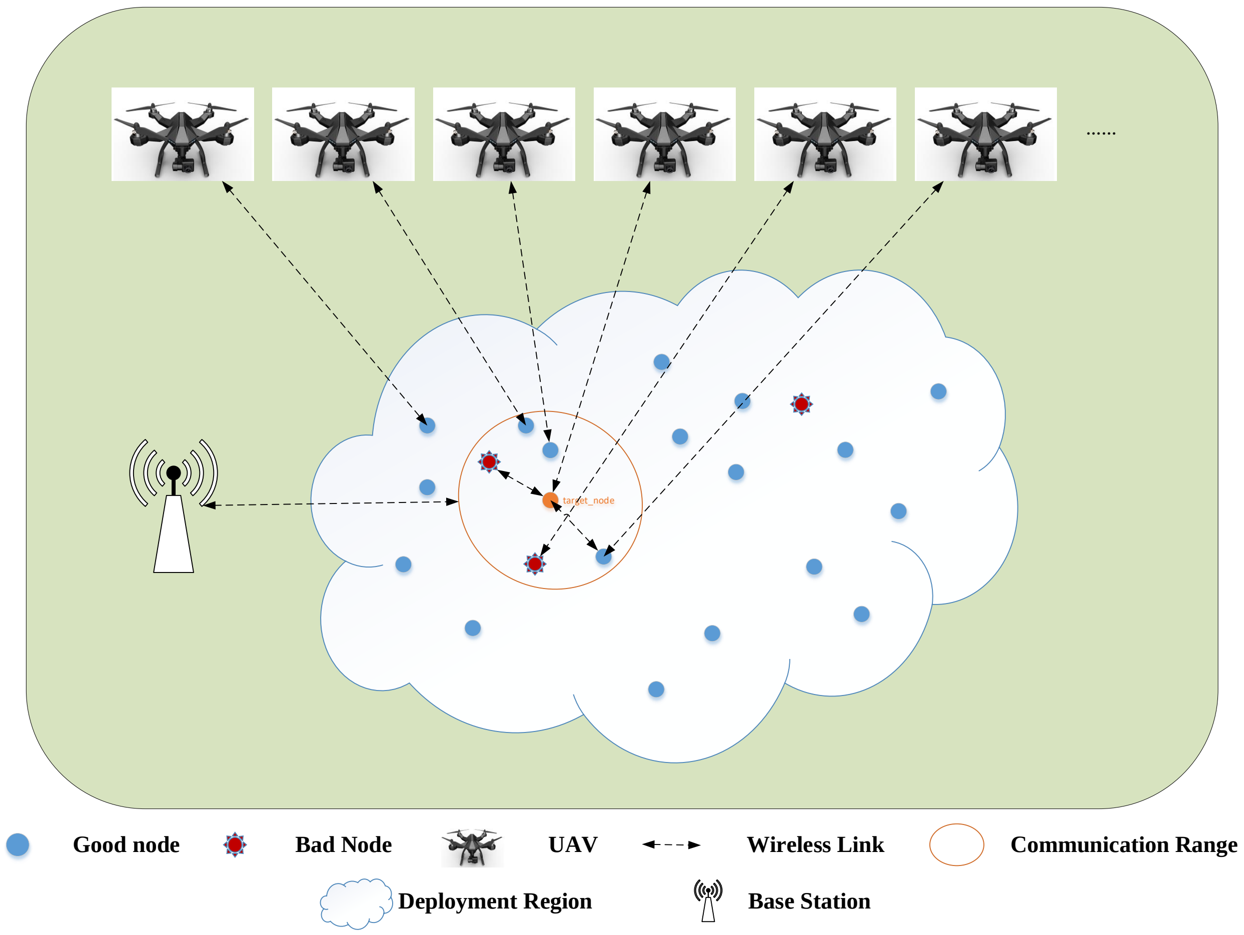


Figure 1. The architecture of Q-TCID.

The attacker exhibits two forms of behavior: device capture and malicious voting. In the first form, the attacker targets vulnerable IoD devices that lack physical protection, converting them into bad nodes. The probability of node capture is denoted as P_{cap}. The second form involves an attacker attempting to label good nodes as bad by influencing the IDS voting process. The probability of such attacks is represented as P_a.

Defensive measures include host-level and system-level actions. At the host level, neighboring nodes can assess the historical voting patterns of each other to determine malicious intent based on discrepancies between their own votes and those of the target node. System-level defense involves controlling the detection intensity through the number of voters (m) and the frequency of intrusion detection (in each T_{IDS} interval). During IDS voting sessions, m adjacent nodes participate in voting to either support or oppose a target node. If most nodes classify it as malicious, the target node is evicted. Otherwise, it is retained. The defense system audits the voting results to prevent collusion attacks and penalizes nodes whose voting behavior is inconsistent with the audit, saving energy and preventing collusion among malicious nodes.

3.2. Problem Formulation

This paper takes the maximum MTTF of the IoD system as the optimization objective and considers the constraints such as network energy consumption, percentage of malicious nodes, and communication distance. An improved intrusion detection algorithm is implemented in the system to reduce the probability of false positives and false negatives and to extend the MTTF of the system as much as possible while reducing the energy consumption.

The IoD is comprised of N sensor nodes, denoted as $SN = sn_1, sn_2, \ldots, sn_N$, which are deployed along with the UAV's position. The sensor nodes are distributed over a two-dimensional area, and their states are statistically independent. Each sensor node can be classified as either "good" or "bad" based on its vulnerability to attacks.

All nodes in the IoD system may be in one of three states: GOOD, BAD, and EVICTED; state "GOOD" means that the node has not been captured at this time and is still a good node; state "BAD" means that the node has been maliciously captured and is a malicious node; state "EVICTED" means that the node has been evicted from the system during the intrusion detection process and can no longer participate in the intrusion detection voting of the system. For a reliable link to be established between nodes i and j, both nodes must not be in the EVICTED state. Each sensor node has an initial battery capacity of E_{in} and is fully charged. The minimum energy required for a sensor node to remain operational is denoted as E_{min}. The energy consumption is expressed as E [25,26].

$$E = \rho \sum_{j \in SN}^{j \neq i} g_{ij} + \sum_{j \in SN}^{j \neq i} C_{ij} g_{ij}, \quad i \in SN \tag{1}$$

In Equation (1), SN represents the set of sensor nodes; the energy consumption rate for receiving data from node i to node j is denoted as ρ, the flow rate for communication from node i to node j is denoted as g_{ij}. In contrast, the energy consumption rate for transmitting data from node i to node j is represented as C_{ij}. The energy consumption for receiving data from other sensor nodes can be expressed as $\rho \sum_{j \in SN}^{j \neq i} g_{ij}$ and $\sum_{j \in SN}^{j \neq i} C_{ij} g_{ij}$ represents the energy consumption transmitting to sensor nodes.

$$C_{ij} = \beta_1 + \beta_2 D_{ij}^{\theta} \tag{2}$$

In Equation (2), β_1 ($\beta_1 \geq 0$) represents a distance-independent constant, β_2 ($0 \leq \beta_2 \leq 1$) is a distance-dependent variable associated with weight distance and D_{ij} denotes the communication distance between node i and node j. D_{ij} is influenced by the path loss coefficient θ.

This paper uses MTTF as a standard to measure the intrusion detection capabilities of IoD systems. A higher MTTF indicates better intrusion detection and defense effectiveness. The investigated problem can be formulated mathematically as:

$$\max_{\{E, N_{bad}, D_{ij}\}} \quad MTTF \tag{3}$$

Subject to:

The energy in the IoD system needs to ensure the minimum energy required for its normal operation:

$$E \leq E_{in} - E_{min} \tag{4}$$

If a system contains at least one-third of compromised nodes, the nodes in the system will not be able to reach a consensus, leading to a system failure [27]. Therefore, the number of malicious nodes needs to be less than one-third of the total number of "good" and "bad" status nodes in the IoD system.

$$N_{bad} < 1/3(N_{bad} + N_{good}) \tag{5}$$

where N_{bad} and N_{good} represent the number of nodes in the system that are in "good" and "bad" states, respectively.

The communication distance constraint should enable the connectivity between sensor nodes:

$$D_{ij} \leq com_{\mathrm{SN}}, \quad \forall i,j \in SN, i \neq j \tag{6}$$

where com_{SN} is the communication distance per sensor node.

In Equations (3)–(6), E, N_{bad}, D_{ij}, N_g and MTTF are variables, E_{in}, E_{min} and com_{SN} are constants.

4. Proposed Methodology

This section describes the overall framework of the proposed method and introduces the IDS cooperation scheme based on Q-learning optimization. This paper proposes an advanced stochastic Petri Net (SPN) model [11,28,29] as shown in Figure 2 to analyze the dynamics of the IDS game.

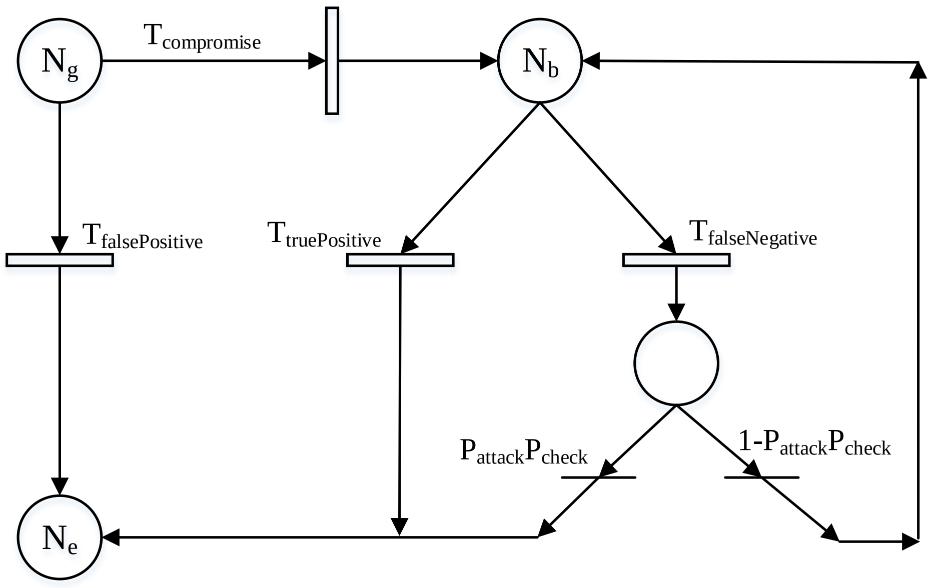


Figure 2. The SPN model.

The SPN model is initially set up with all N nodes being good nodes, and tokens are placed in N_g. Due to the invasion of attackers, every node has a probability P_{cap} of being captured and transformed into a bad node. This is modeled by relating the transition $T_{compromise}$. Firing $T_{compromise}$ will move tokens from place N_g to place N_b one at a time. The tokens in place N_b represent compromised nodes that are not detected.

During IDS voting, especially when no audit is performed, good nodes can be misidentified as bad nodes. If a good node is misjudged as a bad node, it will be removed from the IoD directly. $T_{falsePositive}$ is triggered in this situation, and the good node needs to be moved from place N_g to place N_e, where tokens in the set N_e represent the nodes that were evicted from the system (in the EVICTED state).

Bad nodes in the system will encounter two situations. The first situation is when bad nodes are correctly identified in the IDS and removed from the system. At this time, $T_{truePositive}$ is triggered, and the token of the bad node is transferred from N_b to N_e. In another case, the bad nodes missed judgment during IDS voting, and $T_{falseNegative}$ was triggered. These "false negative" bad nodes were temporarily stored in a temporary placeholder (where there is no label in Figure 2), waiting for the selection of a system-level attack–defense strategy.

In the IDS voting process, a bad node has a probability of P_a to launch an attack, and the system can decide whether to execute the audit in consideration of the overall benefits of the system. If a bad node attack and the system audit occur simultaneously, the system detects inconsistencies between the voting result of the bad node and the audit outcome. As a penalty, the system expels the bad node from the system, causing the token of the bad node to move from the temporary placeholder to place N_e. In addition to this situation, in other cases, the token of the bad node returns to N_b from the temporary placeholder, waiting for the next IDS vote.

In order to reduce the probability of false positives and false negatives, this paper proposes a two-layer cooperative intrusion detection algorithm based on Q-learning (Q-TCID). Q-learning is a reinforcement learning mechanism that is based on a finite Markov decision process (FMDP). It aims to find the optimal solution in the current state by maximizing the utility function of subsequent strategies. Q-learning can be used in both single-agent and multi-agent systems. In a single-agent system, the agent evaluates its utility function based on its own expected return and immediate return, then chooses the response action strategy according to the maximum Q-value. In a multi-agent system, agents have to consider the joint optimal returns of other related agents. The agent should not only choose its own optimal return strategy but also consider other agents' selection strategies. The optimal strategy criterion depends on the joint actions of all agents.

In this paper, at the host level, a Q-learning multi-agent algorithm is used. Each voting node plays a game with the malicious node as an agent and learns a cooperation strategy to vote for the target node in the interaction, which reduces the false positive probability in the voting process. At the system level, a Q-learning single-agent algorithm is used. The system can learn the audit strategy of voting results in the interaction with malicious nodes, thus reducing the false negative probability.

Algorithm 1 describes the Q-TCID algorithm, which is used to calculate the Mean Time To Failure (MTTF) in a monitoring system. The algorithm takes several input parameters, including *NODE* (the set of states and locations of all nodes), L and W (the length and width of the simulation region), T_{IDS} (the IDS interval), E_{in} (the initial energy of each node), C_a (the cost of performing an audit), E_{min} (the minimum energy required for a sensor node to remain operational), and com_{SN} (the communication distance per sensor node).

The algorithm begins by randomizing the positions of the nodes within the specified simulation region of length L and width W. It then proceeds with a loop that runs 1000 simulations. Within each simulation, the algorithm initializes each node's attribute as GOOD (Line 1–5).

A perpetual loop is initiated, where the algorithm iterates over each sensor node and checks if it is in the GOOD state. If the node is captured, its attribute is updated to BAD (Line 7–14). At intervals defined by T_{IDS}, the algorithm invokes Algorithm 2, obtaining the values of the action-value functions Q_{V} and Q_{C} for the host-level intelligent dynamic voting algorithm and the system-level intelligent audit algorithm, respectively. It then iterates over each sensor node that is not in the EVICTED state, determining the current state and performing the corresponding action based on the obtained Q_{V} and Q_{C} values. Following this step, the algorithm updates the residual energy of each node (Line 16–23).

If a Byzantine failure occurs or the system power becomes too low, the algorithm records the current time as the MTTF value and breaks out of the loop. The algorithm increments the time variable for each iteration of the perpetual loop (Line 24–28).

At the completion of the 1000 simulations, the algorithm returns the calculated MTTF value.

Algorithm 1 Q-TCID

```
Input: $NODE, L, W, T_{IDS}, E_{in}, C_a, E_{min}, com_{\mathrm{SN}}$
    $NODE$: Sets of states and locations of all nodes.
    $L$: Simulation region length.
    $W$: Simulation region width.
    $T_{IDS}$: IDS interval.
    $E_{in}$: Initial energy of each node.
    $C_a$: Cost of performing an audit.
    $E_{min}$:Minimum energy required for a sensor node to remain operational.
    $com_{\mathrm{SN}}$: Communication distance per sensor node.
Output: MTTF (Mean Time To Failure)
1:  Randomly deploy node positions (L, W) in the simulation area
2:  for each simulation q = 1, 2, 3, ..., 1000 do
3:      for each node $n_i$ of $NODE$ do
4:          Initialize node $n_i$ with attribute = GOOD
5:      end for
6:      while true do
7:          for each sensor node $n_i$ do
8:              if $n_i$ is in GOOD state then
9:                  Inspect whether the node is captured and transformed
10:                 if node is captured then
11:                     Update node attribute to BAD
12:                 end if
13:             end if
14:         end for
15:         if mod (time,$T_{IDS}$)==0 then
16:             Run Algorithm 2 and obtain $Q_{\mathrm{V}}$ and $Q_{\mathrm{C}}$
17:             for each sensor node $n_i$ do
18:                 if $n_i$ not in EVICTED state then
19:                     Determining the historical voting strategy and status of the node and the target node
20:                     Perform the corresponding action according to $Q_{\mathrm{V}}$ and $Q_{\mathrm{C}}$
21:                     Update and review the remaining energy of each node
22:                 end if
23:             end for
24:             if Byzantine failure occurs or system power is too low then
25:                 Set MTTF to current time and break the loop
26:             end if
27:         end if
28:         Increment time
29:     end while
30: end for
```

4.1. Host Level: Intelligent Dynamic Voting of Nodes

The host level of intrusion detection is concerned with the interaction between each voting node and its corresponding attacker and affects the dynamic change of each node's state. Intelligent voting means that good nodes keep learning voting strategies in the game with malicious nodes and formulate adaptive joint voting strategies to prevent attacks from malicious nodes. The objective of each good node is to maximize its cumulative reward.

In the process of IDS voting, the voting results of the voted nodes are related to the number of voting nodes m (when m = 5, there may be six kinds of voting results: five members all vote for "good"; four members vote for "good", and one member votes for "bad"; three members vote for "good", and two members vote for "bad"; two members vote for "good", and three members vote for "bad"; one member votes for "good" and four members vote for "bad"; five members all vote for "bad"). The action space of voting nodes is defined as $A_{\mathrm{V}} = (a_{\mathrm{V1}}, a_{\mathrm{V2}}, \ldots, a_{\mathrm{Vm}}, a_{\mathrm{Vm+1}})$, where action a_{Vt} indicates that the corresponding strategy is allocated according to the proportion of voting results of voting nodes.

The malicious nodes in the system may choose to attack or "silence" during IDS voting, just behave like a good node. Similarly, the actions of malicious nodes are defined as $A_{\mathrm{H}} = (a_{\mathrm{H1}}, a_{\mathrm{H2}})$.

The state space of IDS intelligent voting game is defined by $S_{\mathrm{V}} = (s_{\mathrm{V1}}, s_{\mathrm{V2}})$, where s_{Vt} represents the state of the target node ("good" or "bad"). The payoff of the voting result is that good nodes hope to minimize the proportion of malicious nodes in the system. Therefore, in all strategies, if all voting nodes can correctly identify the target node, the strategy reward is set to the maximum. If all voting nodes misjudge the target node, the strategy reward is set to the minimum.

In order to solve the optimal strategy, after observing the actions $(s_{\mathrm{V}}, a_{\mathrm{Vt}})$, the reward $R_{\mathrm{V}}(s_{\mathrm{V}}, a_{\mathrm{Vt}})$, and the state transition from s_{V} to s'_{V}, the value functions Q_{V} can be updated according to Equation (7),

$$Q_{\mathrm{V}}(s_{\mathrm{V}}, a_{\mathrm{Vt}}) \leftarrow Q_{\mathrm{V}}(s_{\mathrm{V}}, a_{\mathrm{Vt}}) + \alpha_{\mathrm{V}}[R_{\mathrm{V}}(s_{\mathrm{V}}, a_{\mathrm{Vt}}) + \gamma_{\mathrm{V}} \pi_{\mathrm{V}}(s'_{\mathrm{V}}) - Q_{\mathrm{V}}(s_{\mathrm{V}}, a_{\mathrm{Vt}})], \tag{7}$$

where $\alpha_{\mathrm{V}} \in [0,1]$ and $\gamma_{\mathrm{V}} \in (0,1)$ denotes the learning rate and discount factor, respectively. s'_{V} is the next state.

To prevent the system from falling into a local optimum, there should be an appropriate trade-off between exploitation and exploration in the Q-learning procedure. In this paper, the ϵ-greedy algorithm is employed for exploration.

During the exploration phase, the next action is randomly selected with a probability of ϵ $(0 \leq \epsilon \leq 1)$, ensuring a focus on long-term gains. The selection strategy of the greedy algorithm is represented by Equation (8):

$$\pi(s) = \begin{cases} random \quad action \quad A(s) & \zeta < \epsilon \\ arg \quad maxQ(s,a) & otherwise \end{cases} \tag{8}$$

In this equation, a random number ζ is generated from a uniform distribution between 0 and 1. This random number is assigned before taking any action. If $\zeta < \epsilon$, the agent randomly selects a behavior from the set of behaviors $A(s)$. On the other hand, if $\zeta \geq \epsilon$, the agent explores all the actions in the behavior set $A(s)$ and ultimately executes the action that yields the maximum Q value.

4.2. System Level: Intelligent Audit of System

The system level is concerned with the interaction between the system and malicious nodes. This layer is a defense mechanism added at the system level when the host-level intelligent dynamic voting link is missed, that is, auditing the voting results at the host level. If the voting results of a node are found not to match the audit results, the node will be punished and expelled. To preserve the energy of IoD nodes, the system cannot audit the voting results too frequently. Hence, the system uses Q-learning to optimize its own audit strategy, which is similar to the host-level intelligent dynamic voting method.

In the intelligent dynamic voting process, malicious nodes have a probability of P_a for selection attack and a probability of $1- P_a$ for not attacking, which corresponds to the state space $S_{\mathrm{C}} = (s_{\mathrm{C1}}, s_{\mathrm{C2}}, \ldots, s_{\mathrm{Ck}})$ in the intelligent audit of the system. The system can decide whether to perform the audit in consideration of reducing energy consumption.The action space of voting nodes is defined as $A_{\mathrm{C}} = (a_{\mathrm{C1}}, a_{\mathrm{C2}})$. When the malicious node is audited by the system, it means that the malicious node fails to act. At this time, the system gets corresponding rewards. Otherwise, the system gets corresponding punishments. The possible situations can be classified into four types, as shown in Table 1:

- Best Choice for System: the system chooses to check; the malicious node chooses to attack;

- False Negative: the system chooses not to check; the malicious node attacks;
- False Positive: the system chooses to check; the malicious node chooses to be silent;
- Least Damage: the system chooses to trust; the malicious node chooses to be silent.

The reward value for each type of interaction is evaluated using Equation (9).

Table 1. The game play between system and malicious node.

		System	
		Check	**Trust**
Malicious Node	Attack	Best Choice for System	False Negative
	Silent	False Positive	Least Damage

This paper introduces a constant reward value, denoted as B, which represents the gain of the system when it correctly finds the malicious node of the attack. Conversely, if the system fails to protect the IoD while a malicious node is attacking, the payoff of the system is represented as $-B$. In addition, E_a denotes the cost of performing an audit. This paper define the reward function as R_C, which is given by Equation (9):

$$R_C = \begin{cases} B - E_a & if \quad check, attack \\ -B & if \quad trust, attack \\ -E_a & if \quad check, silent \\ 0 & if \quad trust, silent \end{cases} \tag{9}$$

To acquire knowledge about the correct audit strategy, the system engages in interactions with its surroundings. The objective is to determine the cumulative reward value over time. This is achieved through the use of a learning function, denoted as Q_C, which is defined as Equation (10):

$$Q_C(s_C, a_{Ct}) \leftarrow Q_C(s_C, a_{Ct}) + \alpha_C[R_C(s_C, a_{Ct}) + \gamma_C \pi_C(s'_C) - Q_C(s_C, a_{Ct})], \tag{10}$$

where $\alpha_C \in [0,1]$ and $\gamma_C \in (0,1)$ denotes the learning rate and discount factor, respectively. s'_C is the next state.

The action selection strategy adopted in the system-level intelligent audit algorithm is consistent with the host-level intelligent dynamic voting algorithm. The execution action selected by the system for the current state during the exploration process is determined by Equation (8).

The two method flows proposed above are described in Algorithm 2.

Algorithm 2 Collaborative Q-learning for IDS

```
Input: NODE, $S_\sigma$, $A_\sigma$, $R_\sigma$, $\alpha_\sigma$, $\gamma_\sigma$, Maximum number of iterations
       NODE: Sets of states and locations of all nodes.
       $S_\sigma$:State.
       $A_\sigma$:Action.
       $R_\sigma$:Reward table.
       $\alpha_\sigma$:Learning rate.
       $\gamma_\sigma$:Discount factor.
Output: Final $Q_\sigma$ value
1: Initialization: $Q_\sigma(s_\sigma, a_\sigma) = 0$;
2: Determine the current state $s_\sigma \in S_\sigma$;
3: while ( do$Q_\sigma$is not convergence);
4:     use $\epsilon$-greedy to choose an action "$a_\sigma \in A_\sigma$" based on current $Q_\sigma$ and state;
5:     run action "$a_\sigma \in A_\sigma$" to get the reward value "$r_\sigma \in R_\sigma$", and reach to new state $s'_\sigma$;
6:     update $Q_V$ and $Q_C$ using (2) and (4), respectively;
7:     update the current state to $s'_\sigma$;
8: end while
```

5. Simulation and Analysis

The aim of this section is to comprehensively assess the effectiveness and performance of the proposed method and to explore the effect of different parameters on the system behavior. This section first describes the configuration of the simulation platform and the setting of the experimental conditions. Then, the superiority of the proposed method is analyzed by comparing its performance with existing methods. Finally, an in-depth study of the variation of key parameters in the system is carried out to explore their impact on the system's performance. Through these comprehensive analyses, this paper aims to reveal the advantages of the proposed method in various aspects as well as its adaptability.

5.1. Simulation Setting

The platform environment used for all experiments in this paper was a personal computer with an Intel (R) Core (TM) i7 processor and 16 GB of RAM. The operating system used was Windows 10 with MATLAB R2019a version installed. In this paper, the required algorithms and models were implemented using MATLAB programming, which could be used for data processing, simulation, algorithm development, etc.

In the simulation experiment, the IoD comprised a group of $N = 128$ sensor nodes randomly deployed. Once deployed, the sensors were fixed in their positions. All nodes had a probability P_{cap} of being maliciously captured by external attacks, thus being transformed into malicious nodes, which would attack in the IoD. In this experiment, the proposed algorithm's effectiveness was evaluated by simulating different attack probabilities of malicious nodes and comparing it with the basic voting strategy (BVS) [11] and the single-layer optimized intrusion detection algorithm (SOID). Table 2 lists the design parameters of attack and defense strategy for intrusion detection in this IoD system.

Table 2. System intrusion detection attack and defense strategy design parameters.

Parameter	Meaning	Default Value/Range
N	Number of nodes	128, 228, 328
P_{cap}	Per-node capture rate	1/3600, 1/1800, 1/900, 1/500
m	Number of voters	5, 7, 9, 11
T_{IDS}	IDS interval	50–1600 time units
P_a	Attack probability	0.0–1.0
α_n	Node learning rate	0.6
α_s	System learning rate	0.4
ϵ_n	Node exploration rate	0.5
ϵ_s	System exploration rate	0.5
γ_n	Node discount factor	0.9
γ_s	System discount factor	0.9
E_{in}	Initial energy	2 J
z	Energy consumption rate of a single intrusion detection	1/40, 1/20, 1/10

The system's intrusion detection voting occurs periodically at every T_{IDS} moment. In the process of intrusion detection voting, each node in the system takes turns to vote for it by m neighboring nodes, and the voting result is judged according to the rule that the minority is subordinate to the majority.

Malicious nodes have a probability P_a to attack in the voting process of intrusion detection; that is, when the target node is a malicious node, vote "good" for it, with the intention of leaving more malicious nodes in the system to attack and increasing the proportion of malicious nodes, and when the target node is a normal node, vote "bad" for it, with the intention of expelling good nodes from the system. Accordingly, there is a $1-P_a$ probability that a malicious node will not launch an attack, and at this time, its behavior is consistent with that of a normal node.

In the process of learning its own voting strategy before voting and learning its own auditing strategy in intrusion detection, the learning rate, exploration rate, and discount factor of Q-learning are all set according to its learning and training degree, which are somewhat different.

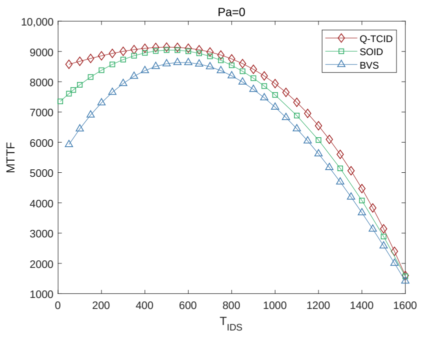
Once a Byzantine fault occurs in the system or the system energy is exhausted, the IoD system will fail.

5.2. Performance Comparision

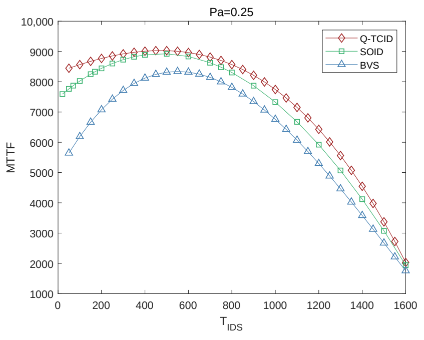
To verify the performance, this paper compares the proposed algorithm Q-TCID with the basic voting strategy (BVS) [11] and the single-layer optimized intrusion detection algorithm (SOID)in terms of attack probability, accuracy, and energy consumption. Both the proposed algorithm and the comparison algorithms utilize identical values for their basic parameters.

5.2.1. Attack Probability

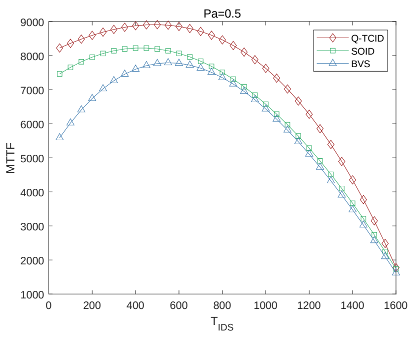
The results depicted in Figure 3 illustrate the superior performance of the Q-TCID algorithm over the basic voting strategy (BVS) proposed in [11] and the single-layer optimized intrusion detection algorithm (SOID). The MTTF values obtained using Q-TCID are consistently higher under different attack probabilities (P_a), indicating that Q-TCID is more efficient in detecting and defending against malicious nodes in IoD systems. This algorithm is especially effective when the T_{IDS} values are small (e.g., T_{IDS} from 50 to 250), as it leverages Q-learning to optimize the intrusion detection strategy, thereby reducing the probability of false positive and false negative. In addition, when the frequency of intrusion detection is higher, the advantage of Q-TCID is more obvious.



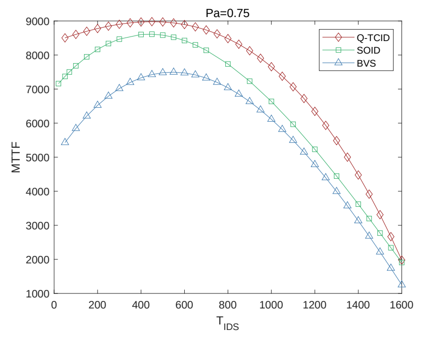
(**a**) P_a = 0



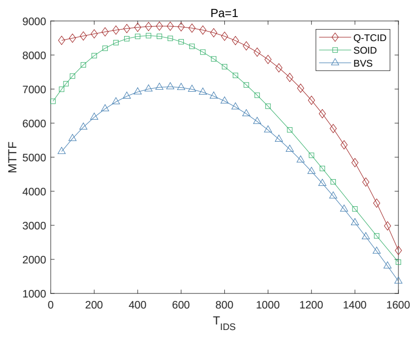
(**b**) P_a = 0.25



(**c**) P_a = 0.5



(**d**) P_a = 0.75



(**e**) P_a = 1

Figure 3. Comparison of the influence of T_{IDS} on MTTF when attack probability P_a = 0–1 under different algorithms.

5.2.2. Accuracy

Accuracy is a widely employed metric for evaluating the performance of intrusion detection models, representing the ratio of correct decisions made by the model to the total number of decisions. In the context of intrusion detection, accuracy is determined by correctly identifying nodes (true negative, TN), accurately detecting malicious nodes (true positive, TP), misclassifying benign nodes (false positive, FP), and identifying malicious nodes (false negative, FN). Equation (11) offers a formal definition of accuracy.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{11}$$

Figure 4 shows the comparison of Q-TCID with the basic voting strategy (BVS) [11] and the single-layer optimized intrusion detection algorithm (SOID). It can be seen that the identification accuracy of BVS and SOID decreases with the increase of the attack probability of malicious nodes. In contrast, Q-TCID demonstrates strong adaptability to the attacks of malicious nodes, with little difference in recognition accuracy under different attack probabilities of malicious nodes. Overall, the improved intrusion detection algorithm achieves higher recognition accuracy and better intrusion detection performance when compared to BVS and SOID.

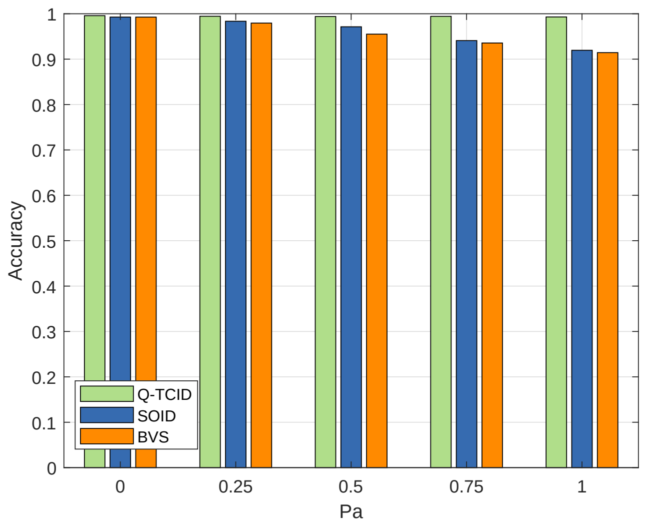


Figure 4. Comparison of average recognition accuracy of intrusion detection under different attack probabilities.

5.2.3. Energy Consumption

To evaluate the feasibility of the suggested algorithm, considering energy consumption related to intrusion detection is crucial. In this regard, energy consumption primarily arises from the system-level audit process. Figure 5 shows the comparison of energy consumption of Q-TCID, BVS, and SOID under the same T_{IDS} and different attack probabilities of malicious nodes. As the figure illustrates, Q-TCID exhibits significantly lower energy consumption compared to BVS and SOID. This is mainly because the proposed algorithm reduces the number of nodes misjudged or missed during the host-level voting process, and the optimized system-level audit strategy reduces the number of audits required, thereby significantly reducing the energy consumption of intrusion detection.

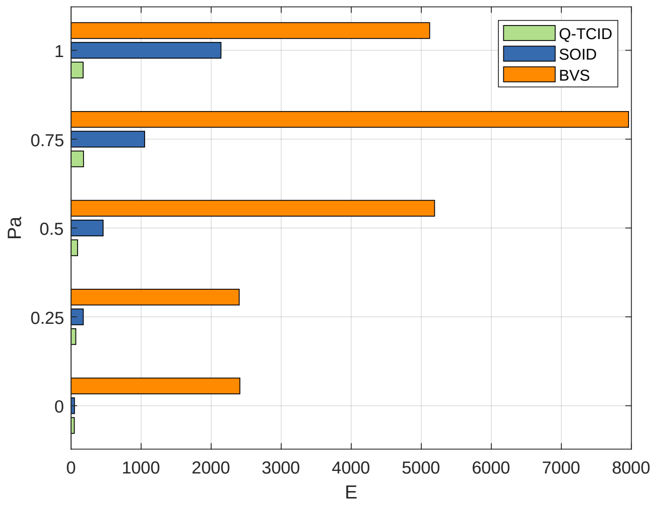


Figure 5. Comparison of energy consumption under different attack probabilities when T_{IDS} is the same.

5.3. Impact of Different Parameters

This section presents the performance of the Q-TCID algorithm through a series of experimental simulations, which evaluate its effectiveness in terms of the attack probability of malicious nodes, the number of voting nodes, the proportion of energy consumed, and the per-node capture rate.

5.3.1. Attack Probability

Figure 6 shows the impact of various attack probabilities P_a on MTTF. It is evident from the plot that there is an optimal value for T_{IDS}, which maximizes the MTTF. This optimal value achieves a balance between the energy consumption during audits and the effectiveness of defense measures in prolonging the system's lifetime. When T_{IDS} is too small, intrusion detection occurs too frequently, resulting in excessive energy consumption and a short running time of the IoD system. With the increase of T_{IDS}, the system can save more energy and extend the network running time. However, when T_{IDS} is too large, although it can save more energy, the system cannot catch malicious nodes in time, which leads to an excessive proportion of bad nodes, leading to Byzantine failure.

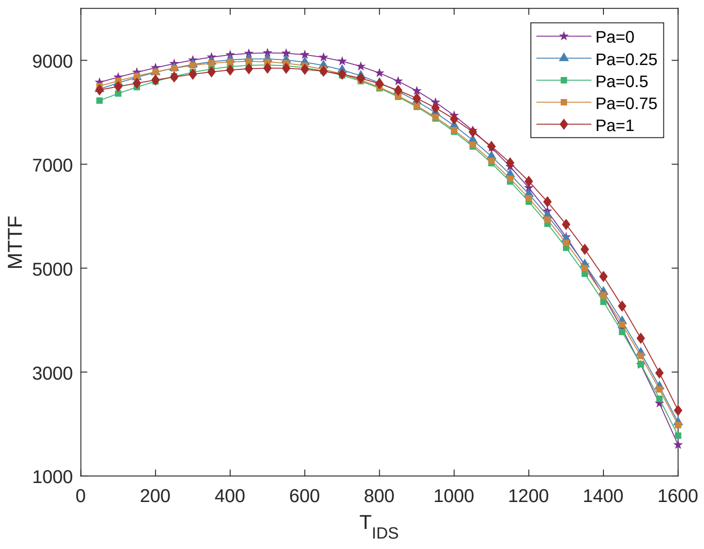


Figure 6. Impact of P_a on MTTF.

Moreover, by evaluating the impact of different attack probabilities of malicious nodes P_a on the MTTF of the IoD system, it is found that the MTTF of the system remains stable within a certain range, regardless of the value of P_a. This indicates that the intrusion detection method proposed in this paper exhibits strong adaptability to attacks by malicious nodes.

5.3.2. Number of Voting Nodes

Figure 7 illustrates the impact of the number of voters (m) on the MTTF of the IoD system under the attack probability of malicious nodes P_a = 0.5. As the number of voters increases (m rises from 5 to 7, 9, and 11), the number of nodes participating in a single intrusion detection vote also increases, and the energy consumed by each intrusion detection also increases, resulting in a decrease in MTTF. However, when T_{IDS} is too small, the frequency of intrusion detection increases, leading to higher energy consumption and potentially shorter MTTF. In this case, the number of voters (m) does not have a significant impact on the MTTF.

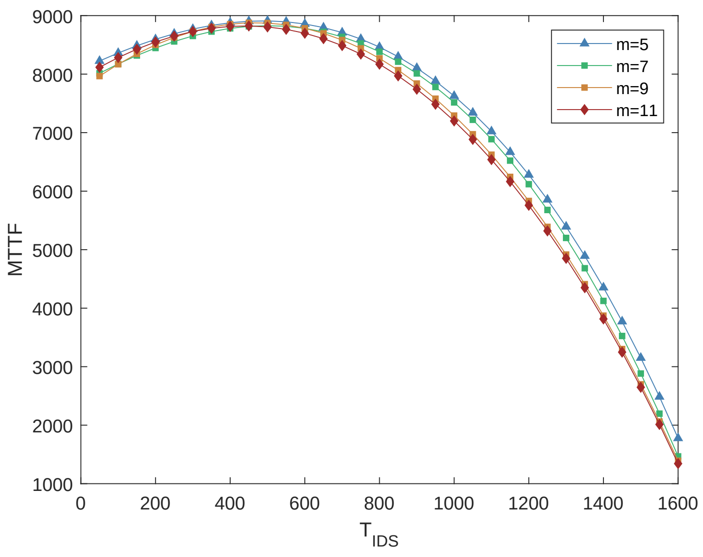


Figure 7. Impact of m on MTTF.

5.3.3. Energy Consumption Rate

The energy consumption rate of a single intrusion detection (z) is the ratio of the energy consumed by performing one intrusion detection to the total available energy. A higher value indicates that the system can detect more intrusions with the available energy. Equation (12) provides a formal definition of z.

$$z = \frac{E_0}{E_{in} - E_{min}} \tag{12}$$

Figure 8 presents the impact of z on the MTTF of the IoD system when P_a = 0.5. By evaluating the impact of different z values on MTTF, it is found that as z increases, MTTF gradually decreases. It is also noted that the optimal T_{IDS} value decreases with the increase of z. The reason is that with the increase in energy consumption rate, the less intrusion detection times can be borne by the disposable energy in the system.

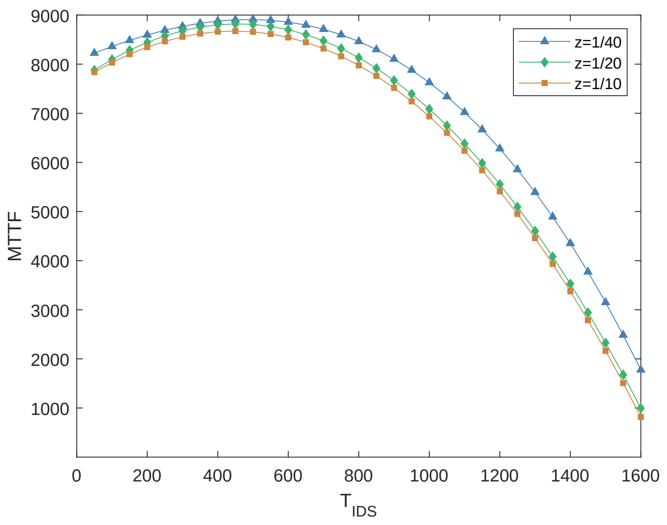


Figure 8. Impact of the energy consumption rate of a single intrusion detection (z) on MTTF.

5.3.4. Per-Node Capture Rate

Figure 9 shows the influence of per-node capture rate P_{cap} on MTTF under different attack probabilities of malicious nodes. A larger P_{cap} means that the good nodes are more likely to be captured maliciously, and the proportion of malicious nodes in the system is greater. Therefore, when P_{cap} increases, the MTTF of the system decreases. Moreover, as P_{cap} rises, the optimal T_{IDS} value that maximizes MTTF also increases. This is because when the per-node capture rate is higher, there is a higher possibility that there will be more malicious nodes in each IDS voting cycle that need to be correctly judged and evicted from the system. If the IDS voting interval is too long, it raises the risk of Byzantine failure, which occurs when at least one third of the nodes are malicious.

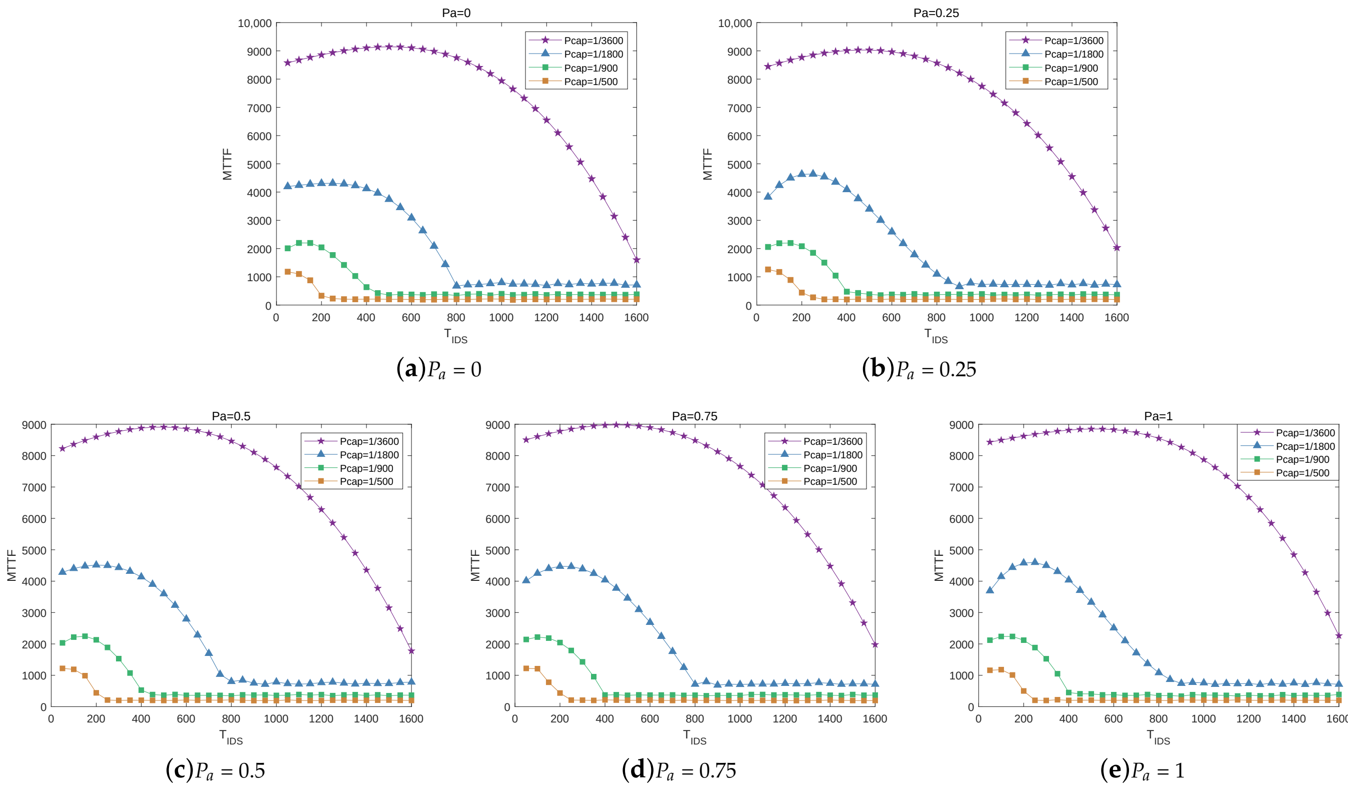


Figure 9. Impact of per-node capture rate on MTTF.

6. Conclusions

This study proposes a two-layer cooperative intrusion detection approach, called Q-TCID, to address the problem of high false positive and false negative rates in intrusion detection. The Q-TCID algorithm combines host-level and system-level strategies to optimize energy consumption and intrusion detection frequency, enabling accurate identification of malicious nodes in a timely manner. Simulation results demonstrate the effectiveness of the proposed algorithm in achieving cooperation and significantly improving the reliability of IoD systems. Overall, the Q-TCID algorithm presents a promising solution for enhancing the security and performance of IoD systems, which find increasing applications across diverse domains.

In the future, we plan to consider the practical situation regarding the Internet of Energy in detail, apply the proposed method to energy in the Internet of Energy equipment, and study the intrusion detection of energy in the Internet of Energy.

Author Contributions: Conceptualization, M.W. and Z.Z.; methodology, M.W. and Y.X.; software, M.W.; validation, Y.X. and Z.Y.; formal analysis, M.W. and X.Z.; investigation, Z.Z. and X.Z.; resources, Z.Z., Y.X., and N.Y.; writing—original draft preparation, M.W.; writing—review and editing, Z.Z., Z.Y., and Y.X.; visualization, M.W. and Y.X.; supervision, Z.Z. and X.Z.; project administration, M.W., Y.X., and Z.Y.; funding acquisition, Z.Z. and N.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the research project of Zhejiang Province under Grant ZF2022003.

Data Availability Statement: Data sharing does not apply to this article as no datasets were generated during the current study.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Hoque, M.A.; Hossain, M.; Noor, S.; Islam, S.R.; Hasan, R. IoTaaS: Drone-based Internet of Things as a service framework for smart cities. *IEEE Internet Things J.* **2021**, *9*, 12425–12439. [CrossRef]
2. Zhang, Z.; Yu, T.; Ma, X.; Guan, Y.; Moll, P.; Zhang, L. Sovereign: Self-contained smart home with data-centric network and security. *IEEE Internet Things J.* **2022**, *9*, 13808–13822. [CrossRef]
3. Mahrez, Z.; Sabir, E.; Badidi, E.; Saad, W.; Sadik, M. Smart urban mobility: When mobility systems meet smart data. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 6222–6239. [CrossRef]
4. Yan, F.; Huang, H.; Yu, X. A Multiwatermarking Scheme for Verifying Medical Image Integrity and Authenticity in the Internet of Medical Things. *IEEE Trans. Ind. Inform.* **2022**, *18*, 8885–8894. [CrossRef]
5. Zhang, Z.; Zhang, Y.; Niu, J.; Guo, D. Unknown network attack detection based on open-set recognition and active learning in drone network. *Trans. Emerg. Telecommun. Technol.* **2022**, *33*, e4212. [CrossRef]
6. Ramadan, R.A.; Emara, A.H.; Al-Sarem, M.; Elhamahmy, M. Internet of drones intrusion detection using deep learning. *Electronics* **2021**, *10*, 2633. [CrossRef]
7. Ahmad, Z.; Shahid Khan, A.; Wai Shiang, C.; Abdullah, J.; Ahmad, F. Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Trans. Emerg. Telecommun. Technol.* **2021**, *32*, e4150. [CrossRef]
8. Cai, L.; Chen, N.; Wei, Y.; Chen, H.; Li, Y. Cluster-based Federated Learning Framework for Intrusion Detection. In Proceedings of the 2022 IEEE 13th International Symposium on Parallel Architectures, Algorithms and Programming (PAAP), Beijing, China, 4–6 November 2022; pp. 1–6. [CrossRef]
9. Wahab, O.A. Intrusion Detection in the IoT Under Data and Concept Drifts: Online Deep Learning Approach. *IEEE Internet Things J.* **2022**, *9*, 19706–19716. [CrossRef]
10. Aldaej, A.; Ahanger, T.A.; Atiquzzaman, M.; Ullah, I.; Yousufudin, M. Smart Cybersecurity Framework for IoT-Empowered Drones: Machine Learning Perspective. *Sensors* **2022**, *22*, 2630. . [CrossRef]
11. Wang, D.C.; Chen, I.R.; Al-Hamadi, H. Reliability of Autonomous Internet of Things Systems With Intrusion Detection Attack-Defense Game Design. *IEEE Trans. Reliab.* **2021**, *70*, 188–199. [CrossRef]
12. Benaddi, H.; Ibrahimi, K.; Benslimane, A.; Jouhari, M.; Qadir, J. Robust Enhancement of Intrusion Detection Systems Using Deep Reinforcement Learning and Stochastic Game. *IEEE Trans. Veh. Technol.* **2022**, *71*, 11089–11102. [CrossRef]
13. Kacem, T.; Wijesekera, D.; Costa, P.; Barreto, A. An ADS-B Intrusion Detection System. In Proceedings of the 2016 IEEE Trustcom/BigDataSE/ISPA, Tianjin, China, 23–26 August 2016; pp. 544–551. [CrossRef]
14. Condomines, J.P.; Zhang, R.; Larrieu, N. Network intrusion detection system for UAV ad-hoc communication: From methodology design to real test validation. *Ad Hoc Netw.* **2019**, *90*, 101759.

15. Dina, A.S.; Manivannan, D. Intrusion detection based on Machine Learning techniques in computer networks. *Internet Things* **2021**, *16*, 100462. [CrossRef]
16. Hadi, H.J.; Cao, Y.; Nisa, K.U.; Jamil, A.M.; Ni, Q. A comprehensive survey on security, privacy issues and emerging defence technologies for UAVs. *J. Netw. Comput. Appl.* **2023**, *213*, 103607. .: 10.1016/j.jnca.2023.103607. [CrossRef]
17. Shrestha, R.; Omidkar, A.; Roudi, S.A.; Abbas, R.; Kim, S. Machine-Learning-Enabled Intrusion Detection System for Cellular Connected UAV Networks. *Electronics* **2021**, *10*, 1549. [CrossRef]
18. Sun, S.; Fan, X.; Xia, Y.; Zhu, C.; Liu, S.; Yi, L. Coverage Reliability of IoT Intrusion Detection System based on Attack-Defense Game Design. In Proceedings of the 2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), Wuhan, China, 9–11 December 2022; pp. 74–82. [CrossRef]
19. Liang, J.; Ma, M.; Tan, X. GaDQN-IDS: A Novel Self-Adaptive IDS for VANETs Based on Bayesian Game Theory and Deep Reinforcement Learning. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 12724–12737. [CrossRef]
20. Praveena, V.; Vijayaraj, A.; Chinnasamy, P.; Ali, I.; Alroobaea, R.; Alyahyan, S.Y.; Raza, M.A. Optimal Deep Reinforcement Learning for Intrusion Detection in UAVs. *Comput. Mater. Contin.* **2022**, *70*, 2639–2653. [CrossRef]
21. Abu Al-Haija, Q.; Al Badawi, A. High-performance intrusion detection system for networked UAVs via deep learning. *Neural Comput. Appl.* **2022**, *34*, 10885–10900. [CrossRef]
22. Tao, J.; Han, T.; Li, R. Deep-Reinforcement-Learning-Based Intrusion Detection in Aerial Computing Networks. *IEEE Netw.* **2021**, *35*, 66–72. [CrossRef]
23. Heidari, A.; Navimipour, N.J.; Unal, M. A Secure Intrusion Detection Platform Using Blockchain and Radial Basis Function Neural Networks for Internet of Drones. *IEEE Internet Things J.* **2023**, *10*, 8445–8454. [CrossRef]
24. Fu, R.; Ren, X.; Li, Y.; Wu, Y.; Sun, H.; Al-Absi, M.A. Machine Learning-Based UAV Assisted Agricultural Information Security Architecture and Intrusion Detection. *IEEE Internet Things J.* **2023**, 1. [CrossRef]
25. Zhang, Y.; Lin, B.; Hu, X.; Wang, Z. Deployment and optimization of multi-UAV-assisted maritime Internet of Things for waterway data collection. In Proceedings of the 2021 International Conference on Security, Pattern Analysis, and Cybernetics (SPAC), Chengdu, China, 18–20 June 2021; pp. 577–580. [CrossRef]
26. Shi, Y.; Xie, L.; Hou, Y.T.; Sherali, H.D. On renewable sensor networks with wireless energy transfer. In Proceedings of the 2011 Proceedings IEEE INFOCOM, Shanghai, China, 10–15 April 2011; pp. 1350–1358. [CrossRef]
27. Lamport, L.; Shostak, R.; Pease, M. The Byzantine generals problem. In *Concurrency: The Works of Leslie Lamport*; Association for Computing Machinery: New York, NY, USA, 2019; pp. 203–226.
28. Ciardo, G.; Muppala, J.K.; Trivedi, K.S. SPNP: Stochastic Petri Net Package. In Proceedings of the Third International Workshop on Petri Nets and Performance Models (PNPM), Kyoto, Japan, 11–13 December 1989; Citeseer: Forest Grove, OR, USA, 1989; Volume 89, pp. 142–151.
29. Ing-Ray, C.; Ding-Chau, W. Analyzing dynamic voting using Petri nets. In Proceedings of the 15th Symposium on Reliable Distributed Systems, Niagara-on-the-Lake, ON, Canada, 23–25 October 1996; pp. 44–53. [CrossRef]