Improving Intrusion Detection Systems with Multi-Agent Deep Reinforcement Learning: Enhanced Centralized and Decentralized Approaches

Amani Bacha[1], Farah Barika Ktata[1] and Faten Louati[2]
[1]*MIRACL Laboratory, ISSATSo, Sousse University, Sousse, Tunisia*
[2]*MIRACL Laboratory, FSEGS, Sfax University, Sfax, Tunisia*

Keywords: Multi-Agent Deep Reinforcement Learning (MADRL), Intrusion Detection System (IDS), Deep Q-Network (DQN), NSL-KDD, MADQN, COCA-MADQN, MADQN-GTN.

Abstract: Intrusion detection is a crucial task in the field of computer security as it helps protect these systems against malicious attacks. New techniques have been developed to cope with the increasing complexity of computer systems and the constantly evolving threats. Multi-agent reinforcement learning (MARL), is an extension of Reinforcement Learning (RL) in which agents can learn to detect and respond to intrusions while considering the actions and decisions of the other agents. In this study, we evaluate MARL's performance in detecting network intrusions using the NSL-KDD dataset. We propose two approaches, centralized and decentralized, namely COCA-MADQN and MADQN-GTN. Our approaches show good results in terms of Accuracy, Precision, Recall, and F1-score.

1 INTRODUCTION

Machine learning (ML) offers several methods for intrusion detection, such as supervised, unsupervised, semi-supervised, and Reinforcement learning (RL) algorithms (Molina-Coronado et al., 2020). RL was first formalized by (Sutton et al., 1998) and subsequently extended to MARL. MADRL is an extension of Deep Reinforcement Learning (DRL) that allows multiple agents to interact and learn from each other in a shared environment. Hence, the MADRL-IDS approach possesses the capability to acquire knowledge from past experiences and assimilate novel attack patterns, thereby enhancing its efficacy in detecting and thwarting cyber intrusions (Sethi et al., 2021). However, MARL approaches have limitations including scalability, non-stationarity, partial observability issues. Non-stationarity is a major challenge in distributed MARL as it can lead to degraded performance or even failure of the learning algorithm. The latter is caused by the changes in an agent's policy during learning and the delay incurred in information exchange between agents (Ibrahim et al., 2021). As for scalability, which is a hurdle in centralized MARL caused by the huge number of agents or the complexity of the state and action spaces, the computational complexity of the centralized approach can become prohibitive, making it difficult to scale up to larger problems. Moreover, the centralized approach requires the sharing of all observations with the central agent. Communication overhead can also become a bottleneck in large-scale MARL problems, and the central agent must process and integrate all the observations from each agent to make decisions. This causes a high communication bandwidth (Zhu et al., 2022). We tackled the mentioned problems by employing two MADQN- IDS-based approaches: centralized and decentralized, namely COCA-MADQN and MADQN-GTN. We evaluated their effectiveness using the NSL-KDD dataset.

COCA-MADQN (Common observation Common action-MADQN): This centralized approach eliminates the need for a single central agent to coordinate all actions. All agents have equal access to observations and can communicate with each other to determine the best course of action. The decision-making process is shared among the agents, and the final action is determined by majority voting. The shared ReplayBuffer further allows agents to learn from each other's experiences, leading to better coordination and overall performance improvement in cooperative MARL problems.

Bacha, A., Ktata, F. and Louati, F.
Improving Intrusion Detection Systems with Multi-Agent Deep Reinforcement Learning: Enhanced Centralized and Decentralized Approaches.
DOI: 10.5220/0012124600003555
In *Proceedings of the 20th International Conference on Security and Cryptography (SECRYPT 2023)*, pages 772-777
ISBN: 978-989-758-666-8; ISSN: 2184-7711

MADQN-GTN (MADQN with Global Target Network): Our decentralized approach is inspired by the concept of Vertical Federated Reinforcement Learning (VFRL)(Qi et al., 2021). MADQN-GTN utilizes a global target network (GTN) instead of individual local target networks for each DQN agent. This proved to be more effective since the weights of the GTN are averaged across all agents, resulting in a more consistent and stable learning process. This approach leads to faster convergence and better performance in cooperative MARL problems.
The remainder of the paper is organized as follows: Section 2 discusses the essential concepts and background. Section 3 provides an overview of the related work on RL and MARL for intrusion detection. Section 4 describes the proposed approaches in detail. Section 5 presents the experimental results and analysis. Finally, Section 6 concludes the paper and outlines possible directions for future research.

2 BACKGROUND

2.1 Markov Decision Process (MDP) and Markov Game (MG)

MDP and Markov Game are both important concepts in the field of RL and have numerous applications in real-world problems including cybersecurity (Nguyen et al., 2020). According to (Canese et al., 2021), MDP is a discrete-time stochastic control process involving a single agent. It consists of a set of states, actions, transition probabilities between states, and rewards associated with each state-action space. MG extends the concept of MDP to include multiple interacting agents and the environment. However, agents collaborate or compete to achieve shared or conflicting objectives.

2.2 Reinforcement Learning (RL)

RL algorithms can be broadly categorized into value-based algorithms and policy-based algorithms. Value-based methods learn the value function to indirectly determine the optimal policies (Mnih et al., 2015). Algorithms like Q-Learning, and DQN are well-known examples of value-based methods. Policy-based methods directly optimize the policies themselves without the need for a separate value function. Algorithms like REINFORCE and Proximal Policy Optimization fall into this category. In addition, value-based approaches excel in off-policy learning and discrete action spaces, while policy-based approaches can handle both discrete and continuous control and often offer strong performance guarantees (Lee et al., 2022). Actor-Critic (AC) Methods combine value-based and policy-based approaches. Notably, the widely used Deep Deterministic Policy Gradient (DDPG) algorithm employs the Actor-Critic approach for continuous action-space tasks (Canese et al., 2021).

3 RELATED WORK

(Lopez-Martin et al., 2022) suggested several extensions and improvements to DQN including double DQN (DDQN) in Adversarial RL for an intrusion detection field. Likewise, a study in (Nguyen and Reddi, 2021) applied DRL methods such as DQN, Double DQN, and actor-critic models for network intrusion detection. It showed that the DRL policy networks are efficient and responsive, making them suitable for online learning and quick adaptation in dynamic data networks. As an extension of DRL, MADRL has become increasingly popular in recent years as it can solve complex real-world problems that traditional RL struggled with (Ibrahim et al., 2021). Many surveys, examined MADRL from different perspectives due to its rising popularity. From a mathematical perspective, some literature provided theoretical analyses of MADRL, namely (Zhang et al., 2021), (Nguyen et al., 2020). MADRL finds applications in various domains such as the medical field. Researchers in (Vlontzos et al., 2019) has introduced an innovative approach that utilizes cooperative MADRL for detecting multiple landmarks in medical images. However, few studies explored the use of MADRL in the context of intrusion detection systems. In reference to the collaborative MARL topic, we consider (Servin and Kudenko, 2008) paper as a valuable one. It proposes a distributed model where network sensor agents learn to communicate signals within a hierarchical structure. Higher-level agents in the hierarchy interpret local information from these signals and notify the network operator of abnormal states when required. This approach addresses the issue of scalability. (Zhu et al., 2014) put forth a new approach for adaptive IDS logic using iterative reinforcement learning and Multi-armed Bandits (MAB). (Caminero et al., 2019) and (Suwannalai and Polprasert, 2020) implemented Adversarial Reinforcement Learning with Deep Q-network (AR-DQN) which is a technique that combines adversarial learning and DRL to improve the robustness of an RL agent against adversarial attacks. The authors (Sethi et al., 2021) proposed a DRL-based IDS that utilized DQN logic across various distributed network nodes, providing a multiview representation, then they designed and implemented a MADQN equipped with an attention mechanism. In this paper, we carefully designed and implemented two MADQN-IDS-based centralized and decentralized approaches using NSL-KDD. Our approaches demonstrated their effectiveness in achieving high-performance metrics such as accuracy, precision, recall, and F1-score, and they overcome some of the common challenges associated with IDS, such as dealing with large volumes of network data and detecting a wide range of attack types. On the whole, our approaches represent a promising solution for enhancing the security of networks and protecting them against malicious attacks.

4 PROPOSED APPROACH

Since we work in a discrete action-space, we have opted for multi-agent Deep Q-Network (MADQN) method. In the context of IDS, MADQN can be used to train multiple DQN-agents to detect and respond to different types of attacks to improve the overall security of the system. By implementing COCA-MADQN and MADQN-GTN, we can explore their trade-offs and benefits in IDS.

4.1 COCA-MADQN

This approach eliminates the need for a central agent and instead, it distributes the decision-making process among the agents. All agents have equal access to observations and can communicate with each other to determine the best course of action. The final action is determined by majority voting. Indeed, during each episode, agents interact with the environment and collect transitions (state, action, reward, next state). These transitions are stored in a shared replay buffer. A batch of transitions (b_s) is randomly sampled from the replay buffer and used to update the agents' neural network parameters (Figure 1).

The use of a shared replay buffer reduces the communication costs and the loss function. For each agent, the target network is updated every episode to stabilize action policy formation and provide an estimation of the expected future value for each action. This estimation is used to update the Q-values associated with actions in the prediction network. The main neural network parameters are optimized using a cost function that measures the difference between the main neural network's predictions and the target network's produced values. The use of a target neural network reduces fluctuations in the main neural network's predictions and improves learning stability

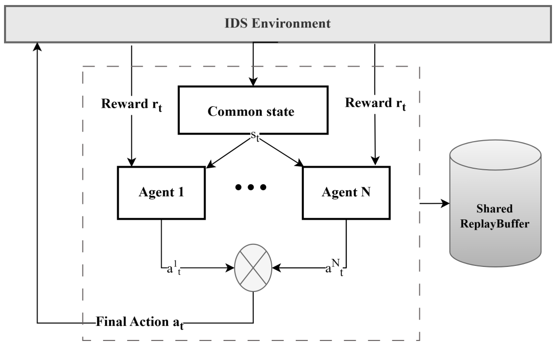

Figure 1: COCA-MADQN: Centralized Approach Model.

and performance (Nguyen et al., 2022). The ε-greedy policy is used to select actions for the agents based on the main neural network's prediction. This means that with probability ε, a random action is selected to encourage the exploration of the action space, while with a probability of 1-ε, the optimal action is chosen based on the main neural network's prediction. Our approach, COCA-MADQN, involved choosing a final action through a majority vote process by selecting the most commonly predicted action by all agents and then assigning an average of all rewards. After that, the state would move to the next state and the process would be repeated until the training was completed. We present a detailed algorithm (Algorithm 1) that outlines the key steps of our approach.

Algorithm 1: COCA-MADQN Algorithm.

```
Initialize:
Replay buffer $D$ to capacity $N$
Step counter $T = 0$
Action-Value $Q$ with random weight $\theta$
Target Action-Value $Q'$ with weight $\theta^- \leftarrow \theta$
repeat
    for each agent i do
        Receive state $s_i$, get $Q(s_i, a, \theta)$
        Choose action $a_i$ according to ε-greedy policy
        Choose final action $a$ by majority voting among all agents
        Execute $a$, get $s'$ and reward $r$
    end
    for each agent i do
        Receive new state $s'_i$
        Store $(s, a_{\text{final}}, r)$ in $D$
        Randomly sample minibatch transitions from $D$
            $Y = \begin{cases} r_i + \gamma \max_{a'_i} Q'(s'_i, a'_i, \theta^-) & \text{if } s' \text{ is not terminal} \\ r_i & \text{if } s' \text{ is terminal} \end{cases}$
        Update parameters $\theta$ using minibatch gradient descent on
          $(Y - Q(s_i, a_i, \theta))^2$
        $s_i \leftarrow s'_i$
    end
    $T \leftarrow T + 1$
    if $T$ mod max_timestep $== 0$ then
        Update target network weight $\theta^- \leftarrow \theta$
    end
until $T > T_{max}$;
```

4.2 MADQN-GTN

First, we started to implement MADQN with a target network for each agent, but the results didn't lead to satisfactory convergence because of the non-stationarity. That's why, we proposed a new approach MADQN-GTN to overcome this limit. To further develop our approach, we used a MADQN with a Global Target Network. This means that we will use a shared target network for all agents in our system, which will help stabilize the training process and improve the overall performance of our model. This approach was influenced by the VFRL (Vertical Federated Reinforcement Learning) described in (Qi et al., 2021), where each agent updates its model based on local observations and periodically sends the aggregated local models to the central server. The updated global model is generated by the central server and distributed to the agents after being combined with the local models. Back to our approach, as explained above, we opted for a shared GTN across all agents to effectively transfer knowledge and experience gained by one agent to other agents in the system, leading to more efficient and effective learning. Our approach will involve training multiple DQN-agents, each agent has its own set of inputs and outputs. The MADQN will learn to predict the Q-values for each agent's actions based on the current state of the environment. The agents will then use these Q-values to select actions that maximize their expected rewards (Figure 2).

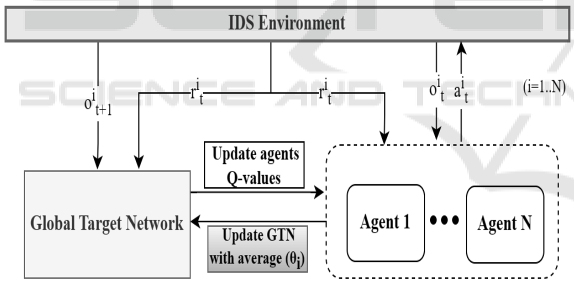

Figure 2: MADQN-GTN: Decentralized Approach Model.

Overall, the use of a MADQN with a Global Target Network is a powerful and flexible approach for learning in multi-agent systems to achieve optimal performance in our particular problem domain. Global Target Network is regularly updated based on the weights of individual agents. In fact, in our implementation, the weights of the Global Target Network are obtained by averaging the weights of all local agents. Our approach is outlined in Algorithm 2, which provides a detailed breakdown of the crucial steps involved. Finally, we created a custom IDS environment with OpenAIGym, a library that enables the creation of simulation environments for reinforcement learning. We opted for a preprocessed and oversampled NSL-KDD which is a benchmark dataset that

Algorithm 2: MADQN-GTN Algorithm.

```
for each agent $i \in \{1,\dots,n\}$ do
    Initialize:
    Replay buffer $D_i$ with capacity $N$
    Action-Value $Q_i$ with random weights $\theta_i$
    Global Action-Value $\hat{Q}_i$ with weights $\theta_i^- = \theta_i$
end
Initialize step counter $T \leftarrow 0$
while $T < T_{max}$ do
    for each agent $i \in \{1,\dots,n\}$ do
        Receive observation $o_i$
        Choose action $a_i$ according to $\varepsilon$-greedy policy based on $Q_i$
        Execute action $a_i$
        Receive next observation $o_i'$
        Receive reward $r_i$
        Store transition $(o_i, a_i, r_i, o_i')$ in $D_i$
    end
    for each agent $i \in \{1,\dots,n\}$ do
        Sample minibatch of transitions $(o_i, a_i, r_i, o_i')$ from $D_i$
            $y_i = \begin{cases} r_i + \gamma \max_{a'} \hat{Q}_i(o_i', a', \theta_i^-) & \text{if } o_i' \text{ is not terminal} \\ r_i & \text{if } o_i' \text{ is terminal} \end{cases}$
        Update parameters $\theta_i$
        $o_i \leftarrow o_i'$
    end
    $T \leftarrow T + 1$
    if $T$ mod $max\_timestep = 0$ then
        Update Global Target Network parameters: $\theta_i^- \leftarrow \text{Average}(\theta_i)$
    end
end
```

enables the evaluation and comparison of IDS (allows researchers to obtain consistent results.)(Ahsan et al., 2023), to be defined as an IDS environment. Agents' observations are the unlabeled rows of the dataset (Network traffic samples), and actions correspond to the categories associated with the labels (indicating the class of attack.). In multi-class classification, the objective is to train agents to classify inputs into five different categories (actions emitted as 0, 1, 2, 3, or 4 depending on the type of intrusion) based on their observations, where 0: Dos, 1: Probe, 2: R2L, 3: U2R and 4: Normal. In the IDS environment, the agent is rewarded for correctly classifying each input. In our case, the reward is 1 if the label corresponding to the current observation matches the action taken by the agent, and 0 otherwise. When all inputs have been processed, the environment is reset to a random state to restart the classification process.

5 EXPERIMENTAL RESULTS AND ANALYSIS

To evaluate the performance of our proposed MADQN approaches for network intrusion detection, we utilized multiple metrics such as Accuracy, Precision, Recall, and F1-score. It is worth noting that solely relying on Accuracy values to evaluate the model's performance may not provide a complete assessment. To conduct a more thorough evaluation, we analyzed the results based on the other performance metrics. We implemented our approaches COCA-MADQN and MADQN-GTN with NSL-KDD with binary classification and multi-class classification and it showed the high-performance metrics below with 3 DQN-agents (Table 1) and (Table 2).

Table 1: Performance metrics' of COCA-MADQN.

Metrics for 3 agents	Multi-class	Binary
Accuracy	0.778	0.985
Precision	0.803	1.0
Recall	0.958	0.985
F1-score	0.874	0.992

Table 2: Performance metrics' of MADQN-GTN.

Metrics for 3 agents	Multi-class	Binary
Accuracy	0.767	0.976
Precision	0.799	0.975
Recall	0.945	0.973
F1-score	0.866	0.974

The results confirm the effectiveness and robustness of these two approaches in intrusion detection. In addition, the plotted curves (Figure 3, Figure 4) shows the convergence of the sum of rewards during the episodes for both test and training. As the number of episodes increases from 0 to 100, the sum of rewards for both the test and training moves towards a common point, indicating that the model is learning and becoming more effective in intrusion detection.

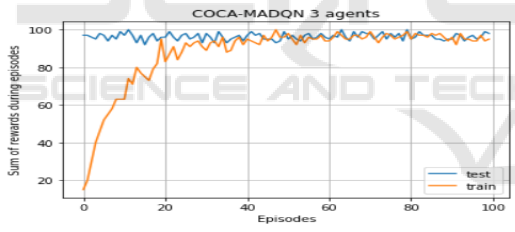

Figure 3: COCA-MADQN convergence curve.

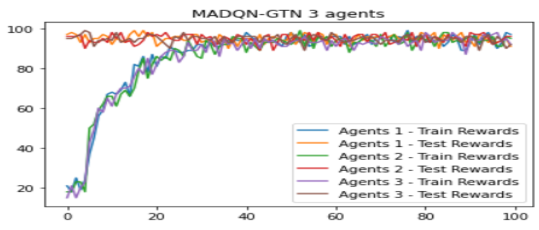

Figure 4: MADQN-GTN convergence curve.

This decentralized approach has overcome the limitations of a local target network for each agent, which often results in poor learning performance and failure of the learning curve to converge during the episodes for both test and training for all agents (Figure 5), primarily due to non-stationarity issues.

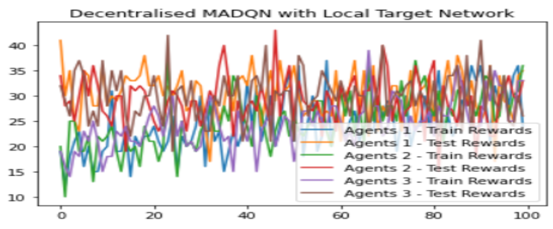

Figure 5: MADQN with Local Target Network convergence curve.

Compared to the state-of-the-art works (Table 3), our two approaches show significant improvement in performance on the NSL-KDD dataset in terms of Accuracy, precision, recall, and F1-score in binary classification as well as multi-class classification. These approaches were able to converge in centralized MADRL. On the one hand, the proposed COCA-MADQN approach was able to converge and eliminate the need for a central agent to coordinate all actions. By providing all agents equal access to observations and allowing them to communicate with each other, decisions were distributed and determined by majority voting. Furthermore, the shared Replay-Buffer allowed agents to learn from each other's experiences, leading to better coordination, scalability, and improved performance in cooperative MARL problems. On the other hand, the proposed MADQN-GTN approach, inspired by Vertical Federated Reinforcement Learning (VFRL), indicated that the use of GTN leads to a more consistent and stable learning process in cooperative MARL problems. Besides, we have overcome the challenges of MARL, such as non-stationarity and confidentiality .

6 CONCLUSION AND PERSPECTIVES

Our experimental results demonstrated that the proposed centralized and decentralised approaches are suitable for intrusion detection systems, hence, their capability to detect network intrusion attacks with high accuracy, precision, recall, and F1-score. Our presented approaches: COCA-MADQN and MADQN-GTN, demonstrated robustness in addressing the challenges of non-stationarity and scalability. However, other factors such as computational complexity should also be considered when deciding which approach to use. In our future work, we intend to implement our proposed methods in a real-world cloud-based environment. This will enable our DQN-based multi-agent to enhance self-learning abilities and accurately detect threats in real-time.

Table 3: Comparison of performance metrics' for intrusion detection with NSL-KDD dataset.

Reference	Approach	Accuracy	Precision	Recall	F1-score
(Caminero et al., 2019)	AE-RL	0.8016	0.7974	0.8000	0.7940
(Suwannalai and Polprasert, 2020)	AE-RL	0.8000	X	X	0.7900
(Sethi et al., 2021)	A-DQN	0.9720	0.9650	0.9910	0.9780
Our COCA-MADQN	MADQN	0.9850	1.000	0.9850	0.9920
Our MADQN-GTN	MADQN	0.9760	0.9750	0.9730	0.9740

REFERENCES

Ahsan, M. M., Raman, S., and Siddique, Z. (2023). Bsgan: A novel oversampling technique for imbalanced pattern recognitions. *arXiv preprint arXiv:2305.09777*.

Caminero, G., Lopez-Martin, M., and Carro, B. (2019). Adversarial environment reinforcement learning algorithm for intrusion detection. *Computer Networks*, 159:96–109.

Canese, L., Cardarilli, G. C., Di Nunzio, L., Fazzolari, R., Giardino, D., Re, M., and Spanò, S. (2021). Multi-agent reinforcement learning: A review of challenges and applications. *Applied Sciences*, 11(11):4948.

Ibrahim, A. M., Yau, K.-L. A., Chong, Y.-W., and Wu, C. (2021). Applications of multi-agent deep reinforcement learning: Models and algorithms. *Applied Sciences*, 11(22):10870.

Lee, H., Hong, J., and Jeong, J. (2022). Marl-based dual reward model on segmented actions for multiple mobile robots in automated warehouse environment. *Applied Sciences*, 12(9):4703.

Lopez-Martin, M., Sanchez-Esguevillas, A., Arribas, J. I., and Carro, B. (2022). Supervised contrastive learning over prototype-label embeddings for network intrusion detection. *Information Fusion*, 79:200–228.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533.

Molina-Coronado, B., Mori, U., Mendiburu, A., and Miguel-Alonso, J. (2020). Survey of network intrusion detection methods from the perspective of the knowledge discovery in databases process. *IEEE Transactions on Network and Service Management*, 17(4):2451–2479.

Nguyen, N. D., Nguyen, T. T., Pham, N. T., Nguyen, H., Nguyen, D. T., Nguyen, T. D., Lim, C. P., Johnstone, M., Bhatti, A., Creighton, D., et al. (2022). Towards designing a generic and comprehensive deep reinforcement learning framework. *Applied Intelligence*, pages 1–22.

Nguyen, T. T., Nguyen, N. D., and Nahavandi, S. (2020). Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications. *IEEE transactions on cybernetics*, 50(9):3826–3839.

Nguyen, T. T. and Reddi, V. J. (2021). Deep reinforcement learning for cyber security. *IEEE Transactions on Neural Networks and Learning Systems*.

Qi, J., Zhou, Q., Lei, L., and Zheng, K. (2021). Federated reinforcement learning: Techniques, applications, and open challenges. *arXiv preprint arXiv:2108.11887*.

Servin, A. and Kudenko, D. (2008). Multi-agent reinforcement learning for intrusion detection. In *Adaptive Agents and Multi-Agent Systems III. Adaptation and Multi-Agent Learning: 5th, 6th, and 7th European Symposium, ALAMAS 2005-2007 on Adaptive and Learning Agents and Multi-Agent Systems, Revised Selected Papers*, pages 211–223. Springer.

Sethi, K., Madhav, Y. V., Kumar, R., and Bera, P. (2021). Attention based multi-agent intrusion detection systems using reinforcement learning. *Journal of Information Security and Applications*, 61:102923.

Sutton, R. S., Barto, A. G., et al. (1998). *Introduction to reinforcement learning*, volume 135. MIT press Cambridge.

Suwannalai, E. and Polprasert, C. (2020). Network intrusion detection systems using adversarial reinforcement learning with deep q-network. In *2020 18th International Conference on ICT and Knowledge Engineering (ICT&KE)*, pages 1–7. IEEE.

Vlontzos, A., Alansary, A., Kamnitsas, K., Rueckert, D., and Kainz, B. (2019). Multiple landmark detection using multi-agent reinforcement learning. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2019: 22nd International Conference, Shenzhen, China, October 13–17, 2019, Proceedings, Part IV 22*, pages 262–270. Springer.

Zhang, K., Yang, Z., and Başar, T. (2021). Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of reinforcement learning and control*, pages 321–384.

Zhu, C., Dastani, M., and Wang, S. (2022). A survey of multi-agent reinforcement learning with communication. *arXiv preprint arXiv:2203.08975*.

Zhu, M., Hu, Z., and Liu, P. (2014). Reinforcement learning algorithms for adaptive cyber defense against heartbleed. In *Proceedings of the first ACM workshop on moving target defense*, pages 51–58.